CORA Cork Open Research Archive
Cartlann Taighde Oscailte Chorcaí

Title	Personal Voice Assistant security - Denial of Service
Authors	Sagi, Prathyusha
Publication date	2025-04-23
Original Citation	Sagi, P. 2025. Personal Voice Assistant security - Denial of Service. PhD Thesis, University College Cork.
Type of publication	Doctoral thesis
Rights	© 2025, Prathyusha Sagi. - https://creativecommons.org/licenses/by/4.0/
Download date	2026-10-30 07:35:59
Item downloaded from	https://hdl.handle.net/10468/19262

UCC
University College Cork, Ireland
Coláiste na hOllscoile Corcaigh

Personal Voice Assistant Security Denial of Service

Prathyusha Sagi

Thesis submitted for the degree of Doctor of Philosophy



NATIONAL UNIVERSITY OF IRELAND, CORK

COLLEGE OF SCIENCE, ENGINEERING AND FOOD SCIENCE

SCHOOL OF COMPUTER SCIENCE & INFORMATION TECHNOLOGY
INSIGHT CENTRE FOR DATA ANALYTICS

April 2025

Head of School: Prof. Dirk Pesch

Supervisors: Prof. Utz Roedig

Contents

List of Figures iv
List of Tables vi
List of Acronyms vii
Acknowledgements xii
Abstract xiii

1 Introduction **1**
1.1 Motivation 2
1.2 Contributions 5
1.3 Publications 7
1.4 Outline of the Dissertation 9

2 Background **12**
2.1 Personal Voice Assistants (PVAs) 12
2.2 Audio Signal Processing 16
2.3 Automatic Speech Recognition (ASR) 18
2.4 Wake Word Detection 20
2.4.1 Wake Word Detection using HMMs 21
2.4.2 Wake Word Detection using DNNs 23

3 Internet of Things (IoT) **26**
3.1 Internet of Targets (IoTs): Security and Privacy of IoT devices . 27
3.2 Security Attacks on PVAs 29
3.3 Denial of Service (DoS) 32
3.4 Jamming Attacks on PVAs 36

4 Wake Word Jamming **38**
4.1 Related Work on Wake Word Jamming 39
4.2 Wake Word Detection 41
4.2.1 Jamming Process 44
4.2.2 Signal Combination 47
4.2.3 Jamming Energy 48
4.3 Evaluation Results 48
4.3.1 Experiment Set 1 — Superimposing Noise and Constant Energy 50
4.3.2 Experiment Set 2 — Superimposing Noise and Variable Energy 53
4.3.3 Experiment Set 3 — Replacing Wake Word and Constant Energy 55
4.3.4 Experiment Set 4 — Replacing Wake Word and Variable Energy 57
4.3.5 Evaluation of Jamming in an Over-The-Air Setup 57
4.3.6 Zebra Effect 60
4.3.7 Noise Types 62
4.3.8 Jamming Energy 63

4.3.9 Result of Required Jamming Energy for Succesful Jamming 66
4.3.10 Wake Word Jamming on Different Wake Words and Algorithms . . . 67
4.3.10.1 Jamming using "Alexa" as a Wake Word 67
4.3.10.2 Jamming using "Marvin" as a Wake Word . . . 69
4.3.10.3 MHAtt-RNN Wake Word Detection Model . . . 70
4.3.10.4 Experiment on Echo Device 71
4.3.11 Findings 72

5 Robust Wake Word Detection 74
5.1 Related Work on Robust Wake Word Detection 75
5.2 Building a Jamming-Resistant Wake Word Detection Model . . . 78
5.2.1 Sensitive Regions and Jamming Coverage 79
5.3 Jamming Signals used for Adversarial Training 81
5.3.1 Adversarial Training Data 82
5.3.2 Adversarial Training Process 83
5.3.3 Resistance to Noise Variation 84
5.3.4 Results of Adversarial Training in Practical Experimental Setup 84
5.3.5 Generalization 87
5.4 Findings 91

6 Detection of Jamming Attack 94
6.1 Related Work on Detection of Wake Word Jamming 95
6.2 Attack Detection 100
6.2.1 Training Data 101
6.2.2 Training Process 103
6.2.3 Direction of Arrival (DOA) 104
6.2.4 Short Time Energy (STE) 105
6.3 Evaluation 106
6.3.1 Experiment Set 1 108
6.3.2 Experiment Set 2 108
6.3.3 Experiment Set 3 111
6.3.4 Experiment Set 4 112
6.3.5 Experiment Set 5 113
6.3.6 Experiment Set 6 114
6.3.7 Experiment Set 7 115
6.3.8 Experiment Set 8 115
6.4 Findings 116

7 Identifying Sensitive Areas 118
7.1 Understanding Sensitive Areas 119
7.2 Role of Jamming Coverage 122
7.3 Role of Signal Processing and Wake Word Model 123
7.3.1 Signal Processing 124

7.3.2 Model Behaviour 124
7.4 Findings 125

8 Conclusion **128**
8.1 Research Overview 128
8.2 Wake Word Jamming 129
8.3 Robustness via Adversarial Training 129
8.4 Detecting Ongoing Attacks 130
8.5 Identifying Sensitive Areas 131
8.6 Limitations and Future Work 132
8.7 Conclusion 133

List of Figures

2.1 PVA system. The front end carries out wake word detection. Speech recognition is carried out by the back end cloud 15
2.2 System design of speech processing showing key stages: preprocessing, feature extraction, and model-based inference 20

4.1 LSTM-WD Wake Word Detection Steps (Bottom to Top). Wake Word (10s audio clip sampled at 44.1 kHz) $\rightarrow$ spectrogram windows of 10ms $\rightarrow$ Deep Neural Network (DNN) model $\rightarrow$ probability scores (1375 scores for 10s input). Wake word is detected if any probability score is above 0.5 [SDSV20]. 43
4.2 Waveform and spectrogram representations of the considered jamming signal types. 46
4.3 Wake Word 'Activate' 47
4.4 Wake word detection success superimposing white noise with constant energy. 51
4.5 Wake word detection success superimposing white noise with variable energy. 54
4.6 Wake word detection success replacing the audio signal white noise with constant energy. Heat maps show results for (a) $\tau = 2$ms and (b) $\tau = 256$ms durations 56
4.8 Experimental Setup for the evaluation of real-time jamming . . 57
4.7 Wake word detection success replacing the audio signal white noise with variable jamming signal energy. Heat maps show results for (a) $\tau = 2$ms and (b) $\tau = 256$ms durations. 58
4.9 Wake Word Detection Success in Real Time Experimental Setup using Ping Noise of 128ms. 60
4.10 Wake Word Detection Pattern under 32ms White Noise Jamming with 10ms Step Size 62
4.11 White noise attenuation through USB speaker playback 64
4.12 Wake Word Detection Success 65
4.13 Wake Word Detection Success 68
4.14 Jamming using Ping noise on "Alexa" 69
4.15 Jamming using Ping noise on "Marvin" 70

5.1 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. 85
5.2 Impact of jamming using white and ping noise on the performance of wake word detection using 'Activate' after adversarial training with babel noise. WaveForm of Activate Superimposed with Jamming Signal on the Top. 86
5.3 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. WaveForm of Activate Superimposed with Jamming Signal on the Top 88

5.4 Jamming using ping noise on the wake word "Alexa" after adversarial training. WaveForm of Alexa Superimposed with Jamming Signal on the Top. The attack is considered unsuccessful ($C = 10.0\%$). 89
5.5 Jamming using ping noise on the wake word 'Marvin' after adversarial training. WaveForm of Marvin Superimposed with Jamming Signal on the Top. The attack is considered unsuccessful ($C = 7.4\%$). 90
5.6 *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. WaveForm of Marvin Superimposed with Jamming signal on the Top 92

6.1 Features of signals in high-frequency range showing microphone non-linearity effects. Figure adapted from He et al. [HBT+19a]. 98
6.2 The wake word detection process of LSTM_MCC. The audio signal (wake word mixed with Babel noise of 128ms) is divided into frames of 10 ms and features from each spectrogram are extracted and fed into an Recurrent Neural Network (RNN). The softmax function is applied to the outputs and the maximum probability of the output is considered to make the categorization decision. 103
6.3 Comparison of Short Time Energy (STE) graphs. 109
6.4 Direction of arrival when wake word audio, was mixed with ping noise of 128ms long, displaying a change in the angle of direction in the sensitive area if the wake word. 113

7.1 Comparison of the impact of jamming using ping noise on the performance of wake word detection of the word 'House' using the original LSTM_WD and MHAtt-RNN models without adversarial training. WaveForm of House Superimposed with Jamming Signal on the Top 121
7.2 Waveforms of the four wake words used in this study. Each subplot shows the amplitude over time (ms) for *Alexa*, *Marvin*, *Activate*, and *House*, illustrating the distinct acoustic signatures and duration differences across wake words 122

List of Tables

4.1 Jamming signal parameters as explained in 4.2.1 and their tested ranges for parameter definitions). The timing range [0, ..., 631] ms covers the entire duration of the 631ms wake word sample, testing jamming placement at 1ms intervals. 49

5.1 Jamming Coverage Before and After Adversarial Training 82

6.1 Performace Validation of LSTM_MCC Model 110

7.1 Summary of key findings on sensitive regions in wake words. . 125

List of Acronyms

ASR Automatic Speech Recognition

CNN Convolutional Neural Network

DNN Deep Neural Network

HMM Hidden Markov Model

IoT Internet of Things

LSTM Long Short-Term Memory

PJD Protection Jamming Device

VAD Voice Activity Detection

PVA Personal Voice Assistant

RNN Recurrent Neural Network

SNR Signal-to-noise Ratio

SR Speech Recognition

FFT Fast Fourier Transform

DFT Discrete Fourier Transform

TTS Text to Speech

SNR Signal to Noise Ratio

DoS Denial of Service

DDoS Distributed Denial of Service

MFCC Mel-Frequency Cepstrum Coefficient

PVA Personal Voice Assistant

ADC Analog-to-digital converter

RNN Recurrent Neural Network

SVM Supported Vector Machine

NLP Natural Language Processing

TTS Text-to-Speech

GRU Gated Recurrent Unit

RMS Root Mean Square

CRNN Convolutional - Recurrent neural network

USB Universal Serial Bus

STE Short Time Energy

DOA Direction of Arrival

AI Artificial Intelligence

ML Machine Learning

ICU Intensive Care Unit

DSR Discrete Speech Recognition

CSR Continuous Speech Recognition

LPC Linear Predictive Coding

IoTs Internet of Targets

RFID Radio Frequency Identification

RF Radio Frequency

DSP Digital Signal Processing

ASP Audio Signal Processing

CiDAT Cross-Informed Domain Adversarial Training

DAT Domain Adversarial Training

MLP Multi Layer Perceptron

LTE Long Term Evolution

TDOA Time Difference of Arrival

MitM Man-in-the-Middle

DARPA The Defense Advanced Research Projects Agenc

FT Fourier Transform

MEMS Micro-Electro-Mechanical Systems

MQTT Message Queuing Telemetry Transport

CoAP Constrained Application Protocol

AWS Amazon Web Services

SPL Sound Pressure Level

I, Prathyusha Sagi, certify that this thesis is my own work and has not been submitted for another degree at University College Cork or elsewhere.

Prathyusha Sagi

Acknowledgements

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.

Prathyusha Sagi
Cork, April 2025

Abstract

Personal Voice Assistant (PVA) have gained significant popularity across domains ranging from smartphones and smart speakers to critical setups like Intensive Care Units (ICUs) in hospitals, educational environments, and military operations. These PVAs offer hands-free convenience, allowing users to perform tasks such as setting reminders, controlling smart home devices, and managing complex processes. Despite their utility, they present notable security risks. One key vulnerability lies in the wake word detection process. PVAs continuously listen for a specific trigger word or phrase called a *Wake Word*, such as 'Alexa', 'Hey Siri', or 'Ok Google'. This wake word detection system can be vulnerable to attacks where an attacker masks the wake word using noise, preventing the device from waking up and processing user requests, effectively causing a Denial of Service (DoS) attack. Our study demonstrates how an attacker can strategically time a carefully designed noise burst, known as a jamming signal, to interfere with wake word detection in a PVA. By superimposing the jamming signal over specific *sensitive regions* of the wake word, the PVA can be rendered unresponsive. This is particularly concerning in high-stakes environments such as Intensive Care Units (ICUs) or military operations, where voice commands are crucial.

To address this vulnerability, we explore adversarial training, where the wake word detection model is retrained on wake word samples with jamming signals superimposed on sensitive regions, making the model resilient to targeted interference. We further show that using a wider range of noise types during adversarial training improves robustness to previously unseen jamming signals, while focusing on sensitive regions reduces training time and effort. Signal processing factors such as signal energies and phonetic characteristics, along with the model's structure, play a key role in identifying these sensitive regions.

Beyond robustness, alerting users to ongoing attacks is equally essential. We extend our wake word detection model from a binary classifier to a three-class classifier, labeling audio as *non-wake word*, *wake word*, or *wake word + jamming signal*. To reduce false alarms, we incorporate Direction of Arrival (DOA) and Short Time Energy (STE). DOA isolates jamming signals by identifying their direction relative to the wake word, while STE detects sudden energy spikes during sensitive regions, indicating deliberate interference. Combining either with the three-class classifier reduces unnecessary alarms, making the system robust to jamming and capable of detecting actual attacks.

Chapter 1

Introduction

Personal Voice Assistants (PVAs), with the infusion of Artificial Intelligence (AI) capabilities, have changed the way users interact with digital environments. These devices utilize Natural Language Processing (NLP) and Machine Learning (ML) to understand and respond to user requests from simplifying daily tasks to being used in healthcare diagnostics. We are becoming increasingly accustomed to interacting with devices and digital environments, such as smart homes and cars, through voice commands. PVAs are now commonly found as standalone devices such as Amazon Echo or Google Home. They are also integrated into almost every phone, tablet, and computer, like Siri on Apple devices or Cortana on Windows. PVAs are used in appliances like TVs and set-top boxes. They are found in wearables such as smartwatches that provide assistance to the users on the go. They are found in home automation systems [RSAP21] and in certain cases, they are also used for detecting and mitigating fire and smoke [BC21]. Additionally, they are also found in cars that offer hands-free navigation and entertainment features [Lug17, Sch20]. Due to their ease of use and effectiveness, the number of these devices being used is rapidly increasing. With the advancements in NLP and ML techniques, PVAs can now recognize and interpret complex speech patterns, accents, and dialects, making it easier for users to interact with them using their voice [S.p]. Research says that about 71% of consumers prefer to query using voice commands rather than typing [S.p]. An overwhelming 97% of Americans now own some type of cellphone, with nine in ten owning a smartphone [Cen24]. With the rise in the usage and popularity of these devices, a significant concern has emerged regarding their security. Research indicates that most individuals are in close proximity to at least one of these devices, whether they are aware of

it or not [CR22]. Given that these devices are designed to understand and process speech, it is crucial to thoroughly examine the associated security risks. Our work focuses on analyzing how an attacker might exploit the acoustic channel of these devices to execute a Denial of Service (DoS). Understanding and mitigating these potential vulnerabilities is essential to ensuring the safe and secure operation of voice enabled technology in our increasingly connected environments.

1.1 Motivation

PVAs have evolved from being simple day-to-day helpers to becoming essential tools in a wide range of industries including smart homes, retail, banking systems as well as in more complex settings like education, healthcare, and the military [ST20, HGRM23, SHRL20, DDS+19]. For example, PVAs such as Amazon's Echo, Google's Nest are central to home automation, enabling users to control lighting, thermostats, security systems through voice commands [Pau23]. In the retail sector, they can help order groceries hands-free where dynamic shopping lists can be built via voice [Sta25]. In banking and finance, PVAs like Bank of America's *Erica*, allow users to check balances, transfer funds and pay bills using voice comands [oA25]. In education, teachers use PVAs to streamline their classes by tracking attendance, monitoring student performance, and providing quick, accurate answers to questions [TS20]. In healthcare, AI enabled PVAs are valuable for both patients and practitioners. They assist patients with daily activities, manage mood disorders, support disease treatments, and provide consultations [MLMT20]. As per the study conducted in [KRG22], the Boston Children's Hospital initiated a pilot program using PVAs in patient rooms. These applications were tested to help hospital staff streamline the pre operative organ validation and checklist process, as well as manage administrative information in the ICU. [Med19] shows that since 2017, the U.S. Army has utilized voice enabled devices, allowing soldiers to easily access crucial information, consult maps for navigating around landmines, and transmit vital messages. As PVAs become more embedded in critical areas like healthcare, education, and the military, their reliability and security become a more important necessity. These systems often perform time sensitive tasks, and their unavailability can directly lead to operational failures.it is important that in many scenarios, where resilient PVA operations are necessary, the device must always be up and running for the users to interact. If the PVA becomes un-

available to use for it's intended users, it results in a DoS attack. An attacker might attempt to render the PVA unavailable for use potentially leading to a DoS attack thereby disrupting its availability when it is most needed.

PVAs deployed in high-stakes environments are particularly vulnerable to availability based threats. For instance, a compromised PVA in a hospital setting, assisting with medication administration or accessing clinical records may become unresponsive during a DoS attack, delaying urgent medical care and endangering patient safety. Similarly, a military-grade voice-enabled device could be rendered unusable at a critical moment when a soldier requires access to mission-critical information, leading to life-threatening consequences.

In addition to availability failures, compromised PVAs may be manipulated to perform harmful actions once an attacker gains control. Malicious voice commands could trigger unsafe operations such as visiting phishing websites, forwarding private emails, making unauthorized financial transactions, or unlocking connected smart devices and vehicles [YJW+22].

Beyond misuse, PVAs present significant privacy risks due to continuous audio capture. If improperly secured, they may store or transmit sensitive speech data without explicit consent. A recent lawsuit involving Siri demonstrated how unintended recordings were retained and reviewed, raising concerns about surveillance, identity theft, and data exploitation [Bhu25]. These risks highlight the need for robust security mechanisms to protect PVAs from unauthorized access, data breaches, and cyber exploitation.

The motivation for this study is to address these threats. Attackers may attempt to exploit a PVA's ability to process audio input, particularly through the acoustic channel, to disrupt normal operations. By examining how adversarial audio can impair wake word detection and futher command processing, we identify specific vulnerabilities and develop countermeasures that ensure continuous and reliable functionality. The goal is to maintain secure operation without compromising responsiveness. Ensuring resilience against these disruptions is critical in scenarios where the failure of a PVA could result in safety-critical consequences.

A PVA is a networked computer system, and as such it is subject to general threats [KD23]. A PVA can be hacked and, for example, be used as a node in a botnet [ZCL+18]. The PVA cloud infrastructure can be breached and user specific data can be stolen [LKS21]. Such classical security challenges must

obviously also be addressed within PVAs and associated service infrastructures. A DoS attack can be performed on the PVA in different ways; for example, generating a request rate that exceeds the PVA's normal operating load and leads to significant packet loss or increased response latency, effectively making the service unavailable such as in syn-flood attack [OWH19a, OWH19b].

Another form of DoS attack can be on the acoustic channel of the device such as injecting noises to alter the input audio signal and making it unrecognizable by the device [CCK+19]. One common method for carrying out a DoS attack is using acoustic jamming. By introducing noise into the acoustic channel, the attack can prevent the ASR of the PVA from functioning correctly. Jamming signals can either be applied either continuously or more selectively targeting specific parts of an audio signal. For instance, an attacker might focus jamming efforts on the wake word recognition process, thereby preventing the PVA from recognizing the wake words and processing the following user commands. This can effectively render the PVA unresponsive, blocking users from accessing its services when needed. This kind of attack poses a significant threat, especially in scenarios where reliable PVA functionality is critical.

Wake word recognition is the first and most crucial step for a PVA. The wake word, also known as a trigger word or hot word, is the specific word that activates the PVA. The device continuously listens to its surroundings, and as soon as it detects the wake word, it activates and forwards the user's request to the backend cloud, where a more advanced ASR system processes the request. If this wake word detection process is disrupted, the device cannot wake up or process the user's request, resulting in a DoS attack. This disruption can occur due to acoustic interference, which might come from random background noise, conversations, music, or a deliberate acoustic jamming signal.

In a real scenario, an attacker might be able to introduce sound near the PVA, which could interfere with the wake word detection. This could be done by taking control of a device with audio capabilities, such as a smart TV, home alarm system, or room thermostat, all of which are often programmable and equipped with speakers. A practical attack configuration might involve placing the compromised device within 1–3 meters of the PVA, ensuring that the emitted audio falls within the microphone's pickup range. By employing a wake word detection model similar to that of the target PVA, the attacker can continuously monitor ambient speech. When the model detects the onset of the wake word, it can immediately emit a short jamming signal millisecond-level timing.

This interference masks the wake word features before the target PVA can fully capture and classify them.

Jamming can be strategically used to control the usage of PVAs in certain environments. For instance, existing research has shown that jamming can be employed for consent management. Users who do not consent to being recorded can use a device, like a phone, to block recording [SCZ20] [CBYR18]. On the other hand, an attacker could use jamming to completely prevent the use of a PVA. In this context, jamming serves as a form of DoS attack.

Jamming signals can be applied continuously or targeted more selectively to interfere with specific parts of an audio signal. Previous research has mainly focused on jamming using continuous sound sources, such as guitar sounds or white noise [LQL+19]. However, this type of jamming is very noticeable. As a result, other studies have explored the use of inaudible sound sources for jamming [ZYJ+17].

In this research, we focus on more targeted jamming of the wake word and explore techniques to detect and mitigate crafted jamming attacks on the wake word detection algorithms of a PVA.

1.2 Contributions

This research investigates acoustic DoS methods, focuses on the selective jamming approach rather than continuous jamming. A very detailed study is conducted on specifics of crafting a jamming signal. A good jamming signal should be difficult to detect so that it should not be noticeable by users, yet powerful enough to cause interruption to the wake word detection. Thus, signal strength, duration, timing and frequency must be considered. Broadly speaking, the jamming signal energy should be minimized in its strength, duration, and bandwidth to reduce detection and also to improve jamming efficiency. Another important parameter to consider for a more successful jamming is the position of the jamming signal in the duration of the wake word. To understand this we used a jamming signal of varying power and duration 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 observed that if placed precisely in those areas of the wake word, an attacker can indeed interrupt the wake word detection by using a jamming signal with

minimal detectability.

To counter such deliberate jamming efforts, the wake word model should be made more powerful, to achieve this, we used adversarial training which involves training a wake word model with data that includes wake word samples superimposed with jamming signals only in the identified sensitive areas of the word. By subjecting the model to these adversarial conditions during training, it becomes robust to jamming attacks. The idea behind adversarial training is to familiarize the algorithm with jamming signals that might be encountered. 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 all in training. To reduce the number of adversarial examples we only generate examples for sensitive regions. The important factor to consider is the jamming signal used for adversarial training. It is observed that if the jamming signal can contain different kinds of noises such as background chatter, cutlery clinking, music, etc., that cover multitude of noises it has more ability to defend against jamming using unknown noise types as well. This approach was tested with different wake words and different wake word detection algorithms and was proved that adversarial training can defend against jamming by placing the noise only in the identified sensitive areas of the word.

The work then focuses on detecting the jamming signals during wake word detection process. We enhanced the existing wake word model which underwent adversarial training into a multi class classifier. The objective is that, it is not only important to detect the wake word amidst jamming signals but also to detect the presence of jamming signals within the wake word and to alert the user. By simultaneously focusing on detecting the wake word and spotting jamming signals, we provide a comprehensive solution to combat wake word jamming. This approach enables users to recognize potential threats, allowing for timely intervention and mitigation strategies. Here, the wake word model is adapted to act as a three class classifier using transfer learning that enables it to distinguish among wake words, non-wake words and wake words mixed with jamming signals, i.e. attacks. Upon completing the training process, the wake word detection model can classify clean wake words and wake words mixed with jamming signals as wake words, and any other words as non-wake words. Therefore, the model's ability to correctly categorize wake words amidst jamming signals determines improved resilience against DoS attacks. However, false alarms can still occur. To reduce these, we explored two techniques: *Direc-*

tion of Arrival (DOA) is employed to identify the angle of direction of arrival of wake word and jamming signals and if there is a significant difference in their directions, the possibility of a DoS attack is higher. *Short Time Energy (STE)* is used to detect abrupt changes in the energy pattern of an audio signal. Identifying such changes in a wake word audio suggests the presence of short, high intensity jamming signal, indicating the possibility of a DoS attack. We evaluated the effective combination to use alongside the multi class classifier's predictions and observed that STE is a better choice to use in combination with the wake word model for more efficient detection of jamming signals.

As the work revolves around jamming by targeting sensitive areas within the wake word, it is crucial to identify them without having to conduct an extensive set of experiments. We analyzed factors that could make a certain area sensitive to jamming signals, such as acoustic signal properties and wake word model's behaviour, and understood that sensitive areas are present in every wake word, but there are no set of universal rules that can help identify them. The identification process is highly dependent on the specific audio signal features and the wake word detection algorithm employed

1.3 Publications

This thesis adds several contributions to the current research field. These contributions have been published and submitted to leading international conferences and workshops. Detailed descriptions of the specific contributions in each submitted work are provided in the respective chapters. Below is a list of these publications:

- Sagi, Prathyusha, Arun Sankar, and Utz Roedig. "Personal PVA wake word jamming." 2024 IEEE International Conference on Pervasive Computing and Communications Workshops and other Affiliated Events (PerCom Workshops). IEEE, 2024.

  In this thesis, we demonstrate DoS attack by compromising the wake word detection algorithm of the PVA using jamming signals. We designed a study to analyze how effective jamming signals should be constructed, considering factors like signal strength, timing, duration, frequency, and bandwidth for successful wake word jamming. We demonstrated that even minimal energy jamming signals, if timed correctly, could disrupt

wake word detection, especially in the sensitive areas of the wake word.

- Sagi, Prathyusha, Arun Sankar, and Utz Roedig."Adversarial training to prevent wake word jamming in Personal PVAs". 2024 20th International Conference on Distributed Computing in Smart Systems and the Internet of Things (DCOSS-IoT).

  In this work, we proposed a method to counteract the jamming attacks through adversarial training. By incorporating jammed wake word samples (adversarial samples) during the training phase, the resilience of wake word detection algorithms against jamming attacks is improved. It is found that by carefully selecting the type of jamming signal used during training, wake word recognition can become robust even against jamming signals that were not known during the training phase. The identification of sensitive areas within the wake word that are highly susceptible to acoustic interference helps optimize the generation of these adversarial training samples.

- Sagi, Prathyusha, Arun Sankar, and Utz Roedig. "Detection of Wake Word Jamming". 6th Workshop on CPS and IoT Security (co-located with ACM CCS 2024) (CPSIoTSec 2024).

  In this work, we propose methods to detect the wake word jamming attacks and alert users about them. We emphasize that recognizing the presence of an attack is more critical than just trying to counteract it. To achieve this, we transform the wake word detection model into a multi-class classifier. This enhanced model is designed to not only distinguish between valid wake words and non-wake words but also to detect wake words that are masked by a jamming signal. This additional classification allows the system to detect jamming attempts as they happen. However, this kind of adaptation might introduce some false positives during the detection process, to avoid this, we introduce another layer of verification to make sure that the model's predictions about the attack are valid. We use two techniques to achieve this. The first is Short Time Energy (STE) to monitor sudden changes in the audio signal energy. These changes are often a clear indicator about the presence of jamming signals, as they disrupt the natural flow of sound. The second technique is determining the Direction of Arrival (DOA) of sound sources, where changes in the direction of sound during wake word detection can indicate an attack. We

compare these two techniques and demonstrate that STE proves to be a more reliable method for detecting jamming attacks. This is because DOA information can be misleading in noisy environments, where unplanned background sounds may cause changes in the direction of sound, leading to false positives. Therefore, STE offers a more consistent and effective measure for identifying jamming signals and ensuring the reliability of wake word detection.

- Sagi, Prathyusha. Live demonstration titled "Wake Word Jamming and Mitigation" at the 2024 Cyber RCI Conference, held at SETU Carlow on November 25, 2024.

  The demonstration had a two part presentation, *Wake Word Jamming*: this part showcased how an attacker can precisely emit a jamming signal into the sensitive regions of a wake word, effectively disrupting the detection process. *Robust Wake Word Detection*: this part demonstrated how an adversarially trained model could withstand such jamming signals in real-time.

1.4 Outline of the Dissertation

The remainder of this dissertation is organised as follows.

Chapter 2 provides the foundational background required to understand the remainder of this dissertation. It begins with an overview of PVAs, including their historical evolution, architecture, and front-end, back-end processing pipeline. The chapter then examines the principles of ASR, outlining how PVAs transform acoustic signals into interpretable commands. A detailed discussion of wake word detection follows, covering both Hidden Markov Model (HMM)-based and Deep Neural Network (DNN)-based approaches, as well as commercial systems such as Alexa, Siri, and Google Assistant. Finally, the chapter reviews existing literature on wake-word jamming, highlighting prior approaches, known limitations, and the key differences that motivate the contributions of this dissertation. Chapter 3 situates PVAs within the broader IoT ecosystem. It outlines the structure and characteristics of IoT devices, with particular emphasis on their security and privacy challenges including weak authentication, outdated firmware, protocol vulnerabilities, and large-scale attacks such as the Mirai botnet [KKSV17]. The chapter then examines general IoT attack classes, illustrating how these vulnerabilities extend to PVAs. Building on this founda-

tion, the chapter provides a comprehensive review of security attacks specific to PVAs, including adversarial audio, command injection, voice spoofing, and malicious skill manipulation. The chapter concludes by presenting the concept of acoustic jamming in PVAs, distinguishing between privacy driven jamming and malicious DoSoriented jamming, and identifying the acoustic threat model that frames the research in subsequent chapters. Chapter 4, discusses the DoS attack in the form of wake word jamming, explaining in detail about how a jamming signal can be constructed for successful jamming. We provide an in-depth analysis of the impact of jamming signal strength, timing, duration, frequency and bandwidth on wake word recognition. We then investigate the impact of different noise types such as White noise, Ping noise and Keyboard noise on wake word recognition. We show that the specifically selected jamming noise type is secondary and that rather the correct selection of jamming signal parameters such as timing decide on jamming success.
Chapter 5, shows the impact of adversarial training on a wake word model to defend against wake word jamming. We first perform a detailed analysis of the jamming signal itself to determine the minimum signal energy required to disrupt wake word detection. 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, because this frequency range covers the dominant spectral content of typical speech signals and the bandwidth processed by most commercial PVA front-ends. Wake word detection models rely primarily on energy in this range, and frequencies above 8 kHz contribute minimally to recognition on standard microphones, which are often low-pass filtered around 8 kHz. 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. This provides a neutral and well defined reference point from which to systematically vary the jamming energy. Starting at 0 dB acts as a baseline, the jamming energy is incrementally increased to determine the minimum level required to cause interruption. The jamming signal energy is increased by a constant scaling factor which is the square root of the ratio between the energy of the longest jamming signal and the energy of the shorter signal 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 which is approximately 85-90 db). This gives us insight into the level of energy needed to cause wake word jamming and the areas of the wake word that are very susceptible to jamming. We then choose this area and created training samples for

adversarial training by superimposing jamming signals in that area. We identify a suitable signal (i.e. 'babel noise') for adversarial sample generation such that attacks with a priori unknown signals are detected.
In Chapter 6, we show that the detection of jamming signals during wake word recognition is equally important as mitigation. Rather than just detecting the wake words even in the presence of jamming signals, identifying the jamming and alerting the users is even more crucial. We transform the wake word detection model that underwent adversarial training to act as a three-class classifier to classify the input signal as wake word, non wake word, or a wake word with the jamming signal in it. However adversarial training can introduce a risk of false positives(where the wake word model might detect random noises as wake words). To address this problem, we use another two signal processing techniques such as DOA and STE to analyse the input signal further and to confirm that the model's predictions about the attack are indeed true. Chapter 7 explores the methods for identifying the sensitive regions within a wake word. It examines the characteristics of the audio signal, as well as how the wake word detection model processes these signals, to understand why certain parts of the wake word are more vulnerable to jamming. The analysis reveals that both signal processing techniques and the wake word detection algorithm play crucial roles in determining which areas of the word are particularly sensitive to disruption. By pinpointing these sensitive regions, we can focus our training efforts on these critical segments, which reduces the number of required jamming configurations from 16,848,000 to 24,000. This leads to a correspondingly large reduction in computational cost and training time.

Following this, Chapter 8 concludes the dissertation by summarizing the key findings, discussing the limitations of the current approach, and outlining potential directions for future work.

Chapter 2

Background

This section provides a short overall description of PVA systems and their functionality, with details describing ASR algorithms, followed by a description of the wake word detection used in commercial PVAs. Thereafter we describe existing work on wake word jamming and outline differences to our work presented in this paper.

2.1 Personal Voice Assistants (PVAs)

PVAs are intelligent software applications that let users communicate with it using natural voice commands. The journey of these PVAs began with IBM's groundbreaking invention called *Shoebox* [IBMa] in the year 1961. This device was revolutionary as it could convert spoken words into digital commands, laying the foundation for future advancements in voice activated technology.

Shoebox could recognize commonly used English words and the digits 0 through 9 allowing users to perform basic calculations simply by speaking [IBMa]. This was a significant leap forward in how people interacted with computers at the time. However, it had its limitations. Despite these early challenges, Shoebox sparked a wave of innovation that would eventually make PVAs a crucial part of daily life. In the following decades, major research milestones significantly advanced speech technology. Systems such as *Harpy* in the 1970s [Low76] introduced efficient graph-search methods for continuous speech recognition, and the The Defense Advanced Research Projects Agencs (DARPAs) Speech Understanding Program in the 1980s [DTI72] established statistical modelling as the dominant approach. The adoption of Hidden

Markov Models, formalized in Rabiner's tutorial [Rab89], enabled robust large vocabulary speech recognition and laid the groundwork for modern voice interfaces.

These developments eventually led to practical consumer products in the 1990s and early 2000s, such as *Dragon NaturallySpeaking*, is speech recognition software for Windows PCs. It transcribes spoken words and inserts the text directly into the currently focused input field or document window [dra25].

The landscape of digital assistants transformed further with the introduction of smart speakers. These devices, equipped with voice recognition capabilities, quickly became central to many homes, responding to voice commands to play music, set alarms, or even order groceries. This innovation soon extended to mobile technology, where PVAs became widely used features on smartphones, further embedding them into everyday life.

Siri, launched by Apple in 2011, was a pioneer in integrating voice technology into the smartphone experience. With Siri, users could send messages, make calls, and schedule meetings using simple voice commands. This convenience set a new standard for how people interacted with their devices. Not long after Siri's invention, Google introduced *Google Now* as part of the Android ecosystem. Google Now transformed user interaction by processing queries and delivering personalized information cards relevant to users' interests and daily routines. Amazon's *Alexa*, delivered via Echo smart speakers, expanded the use of voice interfaces by supporting interactions with various smart-home devices and online services. With Alexa, users could control smart home devices, shop online using just their voice, and access a wide range of third-party applications known as "skills." Advancements in NLP and ML have significantly improved the ability of PVAs to interpret and respond to human speech. When a user speaks to a PVA, the speech is converted into text, analyzed, and a response is generated. ML further enhances this process by enabling PVAs to learn from their interactions, adapt to new speech patterns, and improve over time without needing human intervention. PVAs are now capable of performing a wide range of tasks, including setting reminders, answering questions, and controlling smart home devices. The adoption of these devices is increasing, and they are used in a growing number of digital interactions. The PVA market has seen a significant rise in usage and adoption in recent years. The number of digital PVAs in use has increased from 4.4 billion in 2022 to an expected 8.4 billion by 2024 [Ana24]. This growth is driven by advancements in voice recogni-

tion technology, increased consumer awareness, and the rapid proliferation of connected devices. The number of connected IoT devices reached 18.5 billion in 2024, representing a 12% growth over 2023, and is expected to grow 14% year-over-year to 21.1 billion by the end of 2025. Looking further ahead, the number is estimated to reach 39 billion in 2030 [Ana25]. Popular PVAs globally include Amazon Alexa, Google Assistant, Apple Siri, and Microsoft Cortana, each offering unique features and capabilities such as smart home integration (Alexa), contextual search and multilingual support (Google Assistant), device continuity and calendar integration (Siri), and productivity tools through Microsoft 365 services (Cortana) to meet a wide range of consumer preferences and needs. Research suggests that PVAs are particularly appealing in situations where hands-free operation is necessary or more convenient, such as while driving or cooking. According to a report by NPR and Edison Research, 24% of US adults own a smart speaker, with 74% of those owners using their devices at least once per day [Ana24]. This growing adoption of smart speakers, which primarily rely on voice interaction, indicates a rising preference for hands-free technology in the PVA market. In addition to convenience, PVAs also support multitasking. A study found that 57% of PVA users leverage the technology to multitask [Ana24]. This is particularly evident in tasks like playing music, setting alarms, or getting directions while engaged in other activities.

Today, PVAs are being integrated into various devices and platforms, expanding their reach and applications. For example, the global market for automotive voice recognition is expected to grow at a Compound Annual Growth Rate of 19.4% and reach a market valuation of $4.2 billion by 2030. Market data suggests that the proportion of internet users in the U.S. interacting with PVAs will rise from 36.6% in 2022 to 46.9% by 2025 [Ana24].

A PVA system consists of a front end and a back end as depicted in Figure 2.1. The front end generally comprises of a microphone and wake word detection. The front end of a PVA can either be a stand-alone hardware device, often referred to as a smart speaker (e.g., Google Home, Echo Dot), or it can be an application on a smart device (e.g., Apple's Siri, Microsoft Cortana, Bixby) [For21]. The microphone continuously captures surrounding sound to identify the wake word by using a detection algorithm. When a user speaks to the PVA, the microphone serves as a transducer, converting sound waves into electrical signals. These signals must undergo several processing stages to achieve high quality audio. Initially, the electrical signals are amplified by an amplifier. Next, a low-pass filter is applied to eliminate background noise and

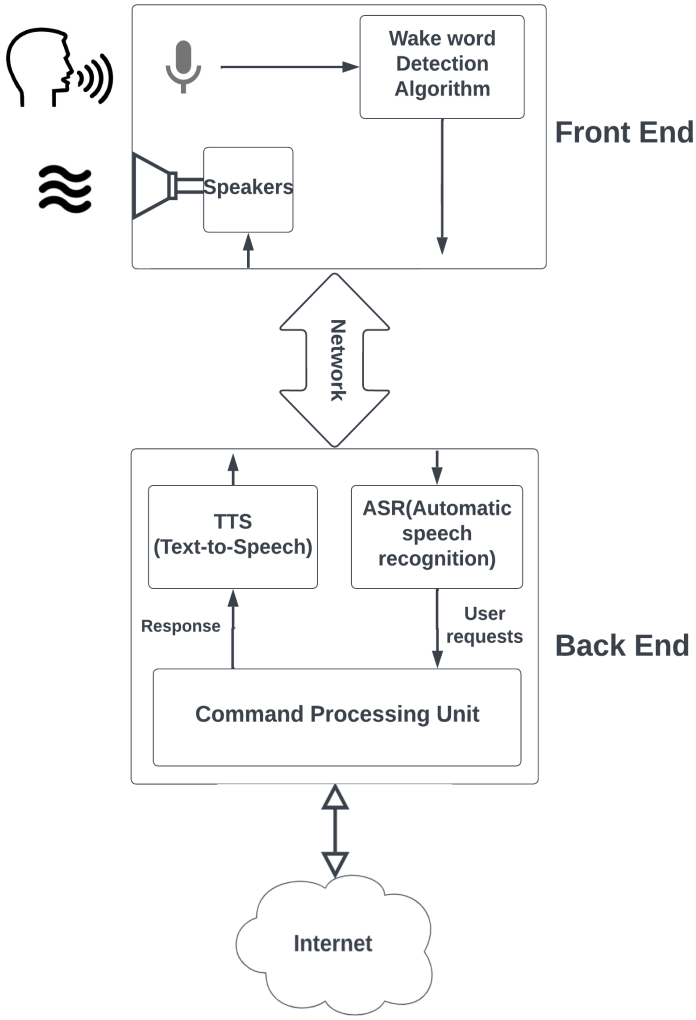


Figure 2.1: PVA system. The front end carries out wake word detection. Speech recognition is carried out by the back end cloud

filter out unwanted frequencies from the signal. After this, the clean signal is passed through an analogue-to-digital converter, which converts the analogue signal into a digital format, making it suitable for further processing by the PVA [YJW+22].

Upon detecting the wake word, the PVA enters the *Activation Phase* and records speech following the wake word. The recorded speech is assumed to be a user command, and it is transported to the back end for the *Recognition Phase*. The back end consists of a sophisticated ASR to transcribe the user command into text. A typical ASR system includes pre processing the audio signal which contains voice activity detection, feature extraction where the audio signal is split into short segments or frames from which the features are extracted using techniques such as Mel-Frequency Cepstrum Coefficient (MFCC),Discrete Fourier

Transform (DFT), Linear Predictive Coding (LPC) [RS07]. Finally the extracted features are fed to a ML model that decodes the features into a sequence of words, phonemes or characters. Typically these decoders are Deep Neural Networks (DNNs). After performing the ASR of the input audio signal, the command processing unit is used to interpret and execute the user command. The command processing unit is connected to other Internet services to obtain information from third-party servers or sources and to implement user commands. The spoken audio is analyzed and the requested user commands are executed and a response will be generated. The response is then converted from Text-to-Speech (TTS) and sent back to the terminal or the PVA which will then be played through the device's speakers as an acoustic response. Recordings are stored in the backend and can be used for ongoing improvements to ASR algorithms and other services.

2.2 Audio Signal Processing

Signal processing is a field of engineering, and it is the study and manipulation of signals to extract, enhance, or optimize the data they contain. A signal is an entity representing the variation of a physical quantity, such as sound, light, or temperature, over time or space. In practical systems, this is performed as Digital Signal Processing (DSP), where real-world analogue signals are first converted into digital samples via an Analog-to-digital converter (ADC) and then processed algorithmically. As a result, signal processing is foundational to the modern world, influencing sectors like entertainment, communication and biotechnology [Soc]. Signal processing allows for analyzing, modifying, and synthesizing signals to achieve various goals, such as enhancing signal quality [20003], detecting patterns [SDJ09] or transmitting data efficiently [Dev24]. The primary objective of signal processing is to extract meaningful insights while reducing noise, distortion, or other irrelevant aspects of the signal. Signal processing has become indispensable in fields like communications, healthcare, audio engineering, and image processing. Signals can be broadly classified into two categories: analogue signals and digital signals. Analogue signals are continuous in both time and amplitude. They represent physical quantities as smooth waveforms that can take any value within a specified range. Examples of analogue signals include sound waves and temperature variations. Analogue signals are more prone to degradation due to noise and distortion during transmission and storage. Unlike analogue signals, which

are continuous in both time and amplitude, digital signals are discrete. They are represented as sequences of numerical values that approximate the original analogue waveform. Digital signals are generally more resilient to noise and interference, which makes them easier to process, transmit, and store without significant degradation in quality. The conversion from analog to digital is carried out by an analogue-to-digital converter, which samples the continuous signal at fixed intervals. Each sample is then quantized into a numerical value, enabling the signal to be handled efficiently by digital systems [TJ18].

Audio Signal Processing (ASP) is a specialized area of DSP that deals with the analysis and manipulation of sound signals. Sound is analog in nature and is first converted into an analog electrical signal using microphones or other transducers. In audio signal processing, this electrical signal is then converted into digital form through sampling and quantization. This digital representation allows for the application of various algorithms both at the input and output stages of audio processing. Input-side processing may include microphone array beamforming and Voice Activity Detection (VAD), which enhance the capture of relevant speech while suppressing background noise and non-speech segments. These techniques help improve the quality of the signal. Once digitized, additional methods are applied, such as noise reduction to eliminate unwanted background noise, audio compression to reduce data size without significantly affecting quality, and equalization to balance different frequency components for clearer sound reproduction. Echo cancellation is also critical, particularly in telecommunication systems, to suppress reflected sounds that can interfere with real-time communication.

In the context of PVAs, audio signal processing plays a key role in the system's interaction with users. PVAs, such as Amazon Alexa, Google Assistant, and Apple Siri, rely on the continuous monitoring of acoustic signals from their environment to detect specific wake words, which activate the device. This process begins with the acquisition of audio signals through built-in microphones. The audio signals are processed to identify the presence of the wake word. Audio signal processing techniques, such as filtering and pattern recognition, are essential in accurately detecting these words in noisy environments. While these methods enhance robustness in noisy and unpredictable environments, they can also lead to a loss of user specific information such as speaking style, accent, or context. This trade-off, though necessary for scalability and real-time performance, may limit the assistant's ability to adapt to individual users' nuances. This still remains an open challenge in PVA development.

Once the wake word is detected, the device processes the subsequent voice commands issued by the user. The audio input is further processed to extract relevant features, such as the pitch, tone, and phonetic content, enabling the system to interpret the spoken language. Digital Signal Processing (DSP) algorithms are used to recognize the patterns in the speech signal and translate them into actionable commands. In addition to wake word detection, noise reduction, and beamforming techniques are also employed to enhance the clarity of the user's voice, especially in environments with multiple sound sources or background noise. Beamforming is a spatial filtering technique that uses multiple microphones to isolate sound coming from a specific direction while suppressing noise from other directions [ZXZ+15]

For PVAs, the use of audio signal processing is fundamental not only for improving the accuracy of wake word detection but also for ensuring that the system responds efficiently and correctly to user inputs. The complexity of these systems requires efficient DSP techniques, allowing them to perform consistently in real-time and under challenging acoustic conditions.

2.3 Automatic Speech Recognition (ASR)

ASR refers to the process of transcribing spoken language into text by identifying and mapping acoustic patterns to linguistic units. ASR started with basic systems that could only recognize a limited range of sounds and has since evolved into advanced systems capable of fluently understanding natural language [AAA+21]. ASR is an interdisciplinary research field that includes linguistics, computer science, and electrical engineering. The primary objective of speech recognition is to automatically convert spoken words into text and then analyze the intent behind the speech from the transcribed text. An ASR system simulates human speech processing by converting the analogue acoustic signal into digital formats, extracting features, and applying machine learning and statistical analysis to identify phonemes (units of sound in speech) and ultimately generate text. The visual overview of the process is shown in the Figure. 2.2. ASR systems are broadly classified into discrete and continuous systems, and further into speaker-dependent, speaker-independent, or speaker-adaptive systems. Discrete Speech Recognition (DSR) systems require the user to pause between each word, simplifying the recognition task as the start and end points of each word are easily detectable. An example of this is voice command systems where each word is spoken separately. Continuous Speech Recognition (CSR)

systems, on the other hand, handle natural speech where words are spoken in a continuous flow without pauses, making it more complex due to difficulties in detecting word boundaries and variations in speech speed and pronunciation. These systems are used in applications such as transcribing speech. Speaker-dependent systems necessitate users to train the system by recording samples of words, sentences, or phrases, resulting in high accuracy for the specific user who trained the system. An example is PVAs that learn the user's voice (Apple's Siri). Speaker-independent systems do not require prior training from the user and are designed to recognize speech from any speaker within a specific language or accent group (PVAs such as Amazon's Echo). Speaker-adaptive systems start as speaker-independent systems and gradually adapt to the speaker's voice characteristics over time, improving their accuracy through continuous learning from the user's speech.

Techniques used in speech recognition include the Fourier Transform (FT) and HMM. Fast Fourier Transform (FFT) is a fast, practical implementation of DFT which is a mathematical method used to transform a signal from its time domain to its frequency domain, aiding in the analysis of the frequency components of the speech signal. This transformation is essential in the initial stages of speech processing to convert audio signals into spectral features such as MFCCs, which represent the short-term power spectrum of the audio. These spectral features are then used to identify the sequence of speech sounds (phonemes) by comparing them to known patterns in speech data. HMM are statistical models used to represent the sequence of phonemes in speech. They estimate the probability of a given sequence of observed acoustic features corresponding to a particular sequence of phonemes, accommodating variability in pronunciation, accent, and speed.

The working process of speech recognition begins with capturing the speech input through a microphone, followed by converting the analogue audio signal into a digital format using an analogue-to-digital converter, typically preceded by an anti-aliasing low-pass filter to prevent aliasing during sampling. This digital audio signal undergoes preprocessing steps such as noise reduction, normalization, and segmentation to enhance its quality. The preprocessed signal is then transformed to extract relevant features that represent the characteristics of the speech. This often involves applying techniques like the FFT followed by feature extraction methods such as MFCC. Acoustic models, often based on HMM or DNN are trained using large datasets of speech recordings to learn the statistical properties of different phonemes. During the decoding phase, the

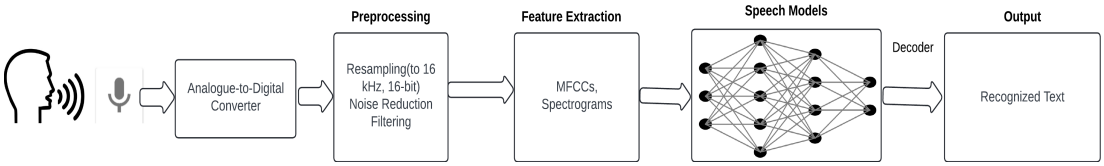


Figure 2.2: System design of speech processing showing key stages: preprocessing, feature extraction, and model-based inference

feature vectors derived from the input speech are compared with the acoustic models to determine the most likely sequence of phonemes or words. A decoder uses algorithms to find the best match between the observed features and the trained models, producing a sequence of recognized words. Language models are employed to improve accuracy by considering the context and probability of word sequences, capturing the likelihood of certain word combinations occurring in a given language. The final stage may include post-processing steps applied to the recognized text, and formatting the output. While some advanced systems incorporate user specific adaptation over time, direct user feedback is typically used in broader system improvement rather than immediate refinement. The recognized speech is then presented in a usable form, such as converting it to text or executing a command [CS09]. An example of the working process is as follows: the user speaks into a microphone, and the spoken words are converted into a digital signal. This signal is analyzed using FFT to extract frequency components and key features are extracted such as MFCC. The features are matched against trained acoustic models and the decoder transcribes the speech into text. Language models refine the text by using context, and the final recognized text is outputted for use in various applications.

2.4 Wake Word Detection

Wake word detection is the entry point to gain access to PVA services and they are detected locally at the front end of PVA. Some examples for wake words are *Alexa* for Amazon's Echo, *Ok Google* or *Hey Google* for Google devices, *Hey Siri* for Apple devices and so on. The wake word can be speaker dependent (eg. Hey Siri) or speaker independent (eg. Alexa). For Apple devices, the PVA is trained on the *Hey Siri* utterances of a user and the PVA is activated only when the wake word is spoken by that specific user. For most other PVAs the wake word recognition is trained using utterances of many users and the PVA is activated by any user speaking the wake word.

Wake word detection systems typically operate on local devices with limited computational resources and power supplies. Consequently, these systems must be designed to consume minimal power to preserve battery life for subsequent user interactions. Additionally, the memory available on voice controlled devices is restricted to reduce both cost and power usage, necessitating low peak memory consumption during wake word detection. Furthermore, to enhance user experience in human device interactions, it is crucial for the wake word detection to trigger immediately after the user finishes speaking, ensuring minimal latency in response times [W+21].

Wake word detection algorithms differ significantly from Speech Recognition (SR) algorithms used in the backend. While backend SR is designed to transcribe speech to text which is a computationally expensive process, wake word detection is a simpler task. It functions as a binary classification algorithm, determining whether a specific wake word is present in an audio stream. This process does not require transcription and operates continuously in an online streaming manner, meaning it does not need to wait for the entire audio stream to finish before detecting the wake word. Wake word detection algorithms assess probability scores in real-time [Pic25, SDSV20]. When the model's predicted scores exceed a predetermined threshold, a positive wake word detection is reported.

Similar to the ASR algorithms, wake word detection systems can be built using HMM, DNN or *HMM-DNN* hybrid models.

2.4.1 Wake Word Detection using HMMs

Hidden Markov Models (HMMs) have been traditionally used for wake word detection because they can model speech as a sequence of hidden states often phonemes or sub-phonetic states evolving over time. In this method, the continuous speech signal is first converted from the time domain to the frequency domain to extract features. The features capture important characteristics of the audio, such as energy distribution and formant structures over time. When the wake word is spoken, the audio is recorded and processed to extract features like frequency and pitch. These features are then compared to the sequence of sounds that the model expects for the wake word. The HMM calculates the likelihood that the sequence of sounds matches the wake word it was initially trained to recognise. If this likelihood is high, the system determines that the wake word has been spoken, and the PVA activates. HMMs are effective because

they can handle variations in how the wake word is spoken, such as different pronunciations, speeds, or accents, by allowing some flexibility in matching the sounds. This ensures that the wake word is accurately detected, even if it's not spoken exactly the same way every time.

Building on this fundamental approach, the detection process involves a more advanced decoding techniques similar to that used in ASR, utilizing the Viterbi search algorithm [LGKK19]. This algorithm keeps track of multiple high scoring paths at each audio frame and synchronously extends them using dynamic programming techniques. To enhance efficiency and reduce computational costs, beam search is employed. Beam search focuses on paths whose scores are within a predefined beam width of the best path, meaning only paths that are close in score to the leading candidate are further processed. Each path's score is incrementally updated using both the frame wise acoustic score and a language model score, which guides the search based on the likely phonetic sequences associated with the wake word. Once all audio frames have been processed, the system traces back through a decoding graph to identify the most likely sequence of phonemes. In wake word detection systems that use a two-pass or dual-graph approach, there are typically two separate decoding graphs, one models the wake word and the other graph models background speech or non-wake word content. Both graphs compute likelihood scores based on how well the incoming audio matches their respective models. The system then compares these scores, if the wake word graph yields a significantly higher score than the non-wake word graph, the system concludes that the wake word has been detected [W$^+$21].

Although HMMs are capable of modeling temporal sequences, their ability to capture long-range dependencies is limited. In theory, feeding longer sequences into the model would require adding more states to represent the additional temporal structure. However, increasing the number of states leads to greater model complexity, which demands significantly more training data and computation. This becomes especially impractical for wake word detection models, which are typically implemented on the front end of devices with limited processing and memory resources. DNNs and recurrent models, which can better capture long-term context, have largely replaced HMMs in modern wake word detection systems. Nevertheless, hybrid HMM-DNN architectures are sometimes used to balance temporal modeling with efficient acoustic feature learning.

2.4.2 Wake Word Detection using DNNs

DNNs have become a widely adopted approach in wake word detection systems due to their capability to learn and recognize complex patterns in speech signals [WPS+18, MZS+17, Sir17]. The process begins by converting the raw audio input into features that the DNN can interpret. Commonly used features include MFCCs and spectrograms. MFCCs represent the short-term power spectrum of sound and are designed to model the human ear's perception of frequency, while spectrograms provide a visual representation of the signal's frequency content over time, this helps models capture how speech characteristics evolve. These features capture essential characteristics such as pitch, tone, and timing. Once extracted, these features are fed into a DNN trained to recognize patterns associated with the target wake word. Popular architectures used for this task include Convolutional Neural Networks (CNNs) for capturing spatial patterns in the spectrogram, Recurrent Neural Networks (RNNs) for modeling temporal dependencies, or combined *C-RNN* models that leverage both. These networks progressively transform the input features into higher-level representations, enabling the system to detect the presence of a wake word in real-time speech. During training, The DNN is trained with large amounts of labeled audio data, including both examples of the wake word and other non relevant sounds (non wake word or background noise). Through this training process, the network learns to distinguish the wake word from other audio inputs by adjusting the weights of the connections between neurons. When the system is in use, the DNN processes the incoming audio features in real time. As it processes these features, the network computes the probability that the current audio corresponds to the wake word. If the probability exceeds a certain threshold, the system recognizes that the wake word has been spoken and activates the PVA.

DNNs are particularly effective for wake word detection because they can handle a wide variety of acoustic conditions, such as different speakers, accents, and background noises, when given sufficient training data. Additionally, DNNs can be fine tuned to improve performance further, making them highly adaptable to different environments and usage scenarios. The result is a highly accurate and reliable wake word detection system that responds correctly even in challenging conditions. Commercial ASR systems such as Alexa, Google and Siri use DNN based models for wake word detection. While the exact implementation details of the algorithms used by commercial systems are not publicly dis-

closed, several research papers describe similar DNN-based architectures that underpin these systems [WPS+18, MZS+17, Sir17]

Based on the work published by Amazon, CNNs are the commonly used type of DNNs for wake word detection in Alexa. Recent work from Amazon presents two new techniques for wake word detection. The first technique is referred to as 'monophone based background modelling for two stage on device wake word detection'. A monophone refers to a single, context-independent phoneme unit used to model speech sounds. As the name suggests the model includes two stages where a DNN-HMM decoder is used in the first stage and the wake word is identified based on a threshold. The second stage classifier is trained using feature vectors derived from wake word candidates from the first stage to make a final decision [WPS+18]. The second technique is named 'Time Delayed Bottleneck Highway Networks'. Highway networks are a type of feed forward neural networks which are inspired by Long Short-Term Memorys (LSTMs) and hence uses a gating function. A gating function allows the network to adaptively combine transformed inputs : those that have been passed through a nonlinear function such as a neural network layer and untransformed inputs : those passed directly without modification, helping to maintain important features and ease training in deep architectures. In this approach, audio features are extracted using a DFT, even though DFT and FFT produce same output, the authors in [GKS+18] chose DFT because it allows linear normalization of the raw audio signal, which improves convergence during training and avoids the suboptimal local minima often encountered with nonlinear pre-processing. Additionally, DFT can be implemented efficiently as a linear transformation at the input layer, keeping computational cost low during inference. Following this, a bottleneck layer is used to reduce the dimension of the complex DFT input features and can also reduce the network size. After the bottleneck layer a time delayed window to splice the bottleneck features is used to obtain the temporal information of the phoneme classification which decides the presence of wake word [GKS+18].

Google's wake word detection uses LSTM which is a type of RNN and an additional language model (N-gram model) for wake word classification [MZS+17]. This work uses both on device and server side wake word detection for accurate detection.

Apple Siri's wake word detection uses a CNN-HMM based hybrid model, which begins by extracting frame-wise audio features from short, fixed-duration,

typically 20–25 ms of audio using a Mel-filter bank [Med20]. These features are fed into a DNN composed of convolutional layers, integrated with an HMM decoder that provides likelihood estimates for each audio frame. A decision is made based on whether the output exceeds a predefined threshold. Notably, this system is designed to support personalized wake word detection, where the model is fine-tuned using the individual user's voice during short training or enrollment phase requiring a few repetitions of the wake word typically a few seconds of audio. This personalization improves accuracy by adapting the model to the speaker's unique vocal characteristics [Sir17].

As shown, different approaches are used for commercial wake word detection, but they all use some form of DNN along with classifiers to predict the presence of wake word with the help of a threshold. Since, the specific implementations of these commercial systems are not publicly available. In this thesis, we use two publicly available algorithms for experimentation. The first is the wake word detection algorithm by Supriya et al. [SDSV20], referred to as *LSTM-WD*, which is a Convolutional - Recurrent neural network (CRNN)-based method resembling commercial architectures and is explained in detail in Chapter 4. The second is a keyword spotting algorithm proposed by Oleg et al., which leverages a multi-head attention mechanism to detect target keywords and is described in Chapter 4.

Chapter 3

Internet of Things (IoT)

The IoT refers to a network of physical devices embedded with sensors, software, and other technologies, such as actuator and connectivity modules like WiFi and Bluetooth, enabling them to connect to and exchange data with other devices and systems over the internet. The primary objective of IoT is to create a seamless network of devices that can interact autonomously, thereby enhancing efficiency, convenience, and decision making in various domains such as healthcare, transportation, and home automation. While the techniques discussed apply broadly, IoT devices are emphasized due to their widespread use in voice interfaces and their vulnerability to acoustic attacks. This context underscores the practical relevance of the work.

IoT devices are diverse and can range from simple household items like smart thermostats and light bulbs to more complex systems like industrial machinery and autonomous vehicles. These devices are typically equipped with unique identifiers such as Radio Frequency Identification (RFID) tags that allow them to be recognized and tracked within the network [AIM10]. The core idea behind IoT is to integrate the physical world with the digital world, allowing objects to sense their environment, process data, and take actions based on that data without human intervention. This creates opportunities for automation, real-time monitoring, and enhanced efficiency in various domains. For instance, in smart homes, IoT devices like thermostats, lighting systems, and security cameras can be interconnected to provide personalized and energy efficient living environments. Some of these devices also incorporate voice interfaces (built-in microphones/speakers) or integrate with external Personal Voice Assistant (PVA), enabling voice-based control and interaction [eco25]. In indus-

trial settings, IoT devices can monitor machinery, predict maintenance needs, and optimize production processes, leading to significant cost savings and increased productivity.

One of the critical aspects of IoT is the ability to provide real-time insights and facilitate data-driven decision making [Kan22, Mal24]. The data collected by IoT devices can be analyzed to uncover patterns, improve operations, and enhance user experiences. For example, in healthcare, wearable IoT devices can continuously monitor a patient's vital signs and send alerts to medical professionals if any abnormalities are detected, enabling timely interventions and improved patient care. However, the rapid proliferation of IoT devices also brings several challenges, particularly in the areas of security, privacy, and interoperability. As these devices often operate in sensitive environments and collect personal or critical data, ensuring their security is paramount to prevent unauthorized access and data breaches. Moreover, the lack of standardized communication protocols can hinder the interoperability of IoT devices, making it difficult to integrate them into a cohesive system.

3.1 Internet of Targets (IoTs): Security and Privacy of IoT devices

With the rise of smart devices like smartphones, smart TVs, and smartwatches, concerns about the security of these devices have grown. Many of these devices ask users to share personal information to provide a more personalized experience. However, most users are not fully aware of the extent to which their IoT devices and associated services collect personal data, which raises significant privacy concerns. Unfortunately, many IoT devices are not designed with strong security and privacy measures in mind. Although this issue spans various digital platforms, it is especially critical in the IoT ecosystem particularly in PVAs due to their always-on nature and continuous listening capabilities and their connectivity to other home applications in a smart home environments.

Securing these IoT devices is crucial to prevent unauthorized access, data breaches, and other malicious activities. Research has been conducted to improve security through the use of privacy nudges, which are designed to raise user awareness about data privacy. This research highlights the importance of making users more conscious of how their data is being used and suggests that well crafted privacy nudges can effectively help users take more control over

their personal information and protect their privacy [ASS+15].

Many IoT devices suffer from inadequate authentication mechanisms. Manufacturers, aiming for ease of use, often deploy devices with weak or default passwords, which are rarely changed by end-users. This creates an entry point for attackers, allowing them to gain unauthorized access to the device. For instance, the Mirai botnet, which leveraged default credentials to compromise thousands of IoT devices, is a stark reminder of the consequences of poor authentication practices. Once compromised, these devices were used to launch a massive distributed denial of service (DDoS) attack that disrupted major internet services across the globe [KKSV17].

The lack of regular software updates presents significant security risks. Many IoT devices operate on outdated firmware, which may contain known vulnerabilities [Sol24]. Unlike traditional computing devices, where users are prompted to update their software, IoT devices often lack such mechanisms. Even when updates are available, users might not be aware of them, or the process to apply updates might be too complex [Dat23]. This creates a persistent risk, as attackers can exploit known vulnerabilities in outdated firmware to compromise devices [SWW15]. The diversity of IoT devices and the lack of standardized security protocols further exacerbate the problem. The IoT ecosystem is highly fragmented, with devices from different manufacturers using different communication protocols, security mechanisms, and data formats. This heterogeneity makes it challenging to implement uniform security measures across all devices, creating a patchwork of security practices that attackers can exploit [ARHAU17].

General security attacks on IoT devices can be broadly categorized into several types. One of the most common is the DoS attack, where compromised IoT devices are used to overwhelm a target system with traffic, rendering it unavailable to legitimate users. The aforementioned Mirai botnet is a prime example of this type of attack, where attackers used compromised IoT devices to launch DDoS attacks of unprecedented scale. Another prevalent attack is the Man-in-the-Middle (MitM) attack. In this scenario, an attacker intercepts and potentially alters the communication between two IoT devices or between a device and its controller. Due to the often unencrypted nature of IoT communication, these attacks can be relatively easy to execute, allowing attackers to eavesdrop on sensitive information or inject malicious commands into the system [CDL16]. In addition to MitM attacks, IoT devices are also vulnerable to

firmware tampering. Attackers can inject malicious code into the firmware of an IoT device, altering its functionality or creating a backdoor for future exploitation. This is particularly concerning for devices used in critical infrastructure, where tampering with the operation of a single device could have cascading effects on the entire system [BNT19]. While these general security threats affect the broader IoT ecosystem, PVAs face additional unique vulnerabilities due to their always-listening design and audio-based interaction model. They may be compromised through network attacks,and acoustic-based attacks, which are discussed in detail in section 3.2

Given the integration of IoT devices into smart home environments, PVAs have become a major target for attackers due to their widespread adoption and central role in managing smart devices. Millions of households worldwide now rely on PVAs such as Amazon Alexa, Google Assistant, and Apple Siri to control lighting, climate, and home security systems. This popularity is driven by the convenience of hands-free interaction and the ability to manage many devices through a single, unified interface. While traditional IoT security issues such as weak authentication or outdated software already present risks, these vulnerabilities become even more serious in the context of PVAs. This is because PVAs allow remote and often inaudible control over devices through voice commands, which can be exploited if the system is compromised. Research has demonstrated attacks including hidden voice commands, ultrasonic attacks, voice squatting, and acoustic injections [YJW+22], all of which exploit weaknesses in voice input mechanisms. Once a PVA is compromised, an attacker could issue unauthorized commands to control other connected devices, such as unlocking smart doors, turning off alarms, or disabling cameras. This interconnected control magnifies the potential damage. Additionally, the always-listening nature of PVAs and their reliance on wake word detection introduce unique vulnerabilities, such as the risk of acoustic denial-of-service attacks, which are the focus of this thesis.

3.2 Security Attacks on PVAs

PVAs operate by continuously listening for a wake word and then processing voice commands to execute specific tasks. This always on nature, combined with the complexity of natural language processing, makes PVAs susceptible to a range of security attacks. One of the most prominent categories of attacks against PVAs involves adversarial attacks, where malicious actors manipulate

audio inputs to trigger unauthorized actions.

Adversarial attacks exploit the way PVAs interpret sound. By embedding inaudible or nearly imperceptible commands within benign audio, attackers can deceive PVAs into executing unintended actions [SKZ+18]. These attacks, often referred to as hidden voice commands, that leverage the fact that PVAs may interpret frequencies outside the range of human hearing, thus making the commands inaudible to the user but intelligible to the device. This can lead to unauthorized actions such as making purchases, controlling smart home devices, or even exfiltrating sensitive information [CMV+16]. As the thesis focuses on acoustic level vulnerabilities like jamming, understanding such input based threats strengthens the motivation for securing the audio pipeline.

Another significant threat to PVAs is the Man-in-the-Middle (MitM) attack. In this scenario, an attacker intercepts the communication between the PVA and the backend servers or other connected devices. By eavesdropping on the communication, the attacker can access sensitive information or alter the transmitted data. For instance, in a smart home environment with voice-controlled smart locks such as August or Yale Assure [Hom24], an attacker could potentially exploit unencrypted local network traffic between IoT devices or use replay attacks to trigger unauthorized commands. The lack of encryption in some communication protocols used by PVAs increases the risk of such attacks [FFZ25]. Voice spoofing is another prevalent attack vector targeting PVAs. In a voice spoofing attack, an attacker mimics the voice of the legitimate user to trick the PVA into executing commands. This can be accomplished through various techniques, such as voice synthesis or using pre recorded audio samples or replay attacks. Voice recognition systems are vulnerable to spoofing attacks, with research demonstrating False Acceptance Rates increasing from near 0% to 70-86% when subjected to synthetic speech or voice conversion attacks [WEK+15]. These attacks exploit current limitations in distinguishing legitimate from spoofed voices, particularly when advanced synthesis techniques closely replicate the target speaker's characteristics. This vulnerability can be exploited to bypass authentication mechanisms and perform unauthorized actions [NRM24, MSPLDLR22]. Another area of concern is the exploitation of PVAs for data leakage. PVAs are integrated with multiple services and applications, with capabilities to access emails, calendars, contacts, billing information, and connected smart home devices [Fed24, Sec24]. These integrations enable PVAs to collect and process personal data, including voice recordings, location data, and behavioural patterns [GDP25]. Attackers can exploit vulnerabilities

in these integrations to extract data from the PVA. For example, if a PVA is linked to a user's email or calendar, an attacker could potentially retrieve sensitive information through unauthorized commands. Furthermore, PVAs may inadvertently record and store private conversations in the cloud by default, which could be accessed by attackers if proper security measures are not in place [Bhu25, LCR+23].

In addition to these direct attacks, PVAs are also vulnerable to social engineering attacks. Attackers can exploit the trust users place in their PVAs by crafting phishing attacks that prompt users to reveal sensitive information or perform actions that compromise their security. For example, Amazon and Google enable customization and the use of third-party applications, referred to as 'skills' or 'actions'. These skills and actions, provide a range of services that Alexa or Google Home do not natively offer [ZMF+19]. A recent study demonstrated how malicious third-party skills can covertly rewrite news headlines before speaking them to users, subtly altering key words to manipulate perception [SJT+21]. For example, changing 'slow his campaign pace' to 'keep his campaign pace' fundamentally alters the meaning while remaining plausible. The study with 220 participants showed this attack successfully induced misperceptions about reported events, with users unable to detect the manipulation due to the lack of visual confirmation in voice-only interfaces [SJT+21]. There are two types of such attacks: Voice Squatting and Voice Masquerading. In a Voice Squatting attack, an attacker creates a malicious skill with a name that is phonetically similar to a legitimate one, exploiting ambiguities in speech recognition. For example, a skill named "Capital Won" could hijack commands intended for "Capital One" because these phrases sound nearly identical when spoken, making it difficult for the PVA to distinguish between them [ZMF+19]. Once invoked, the malicious skill can collect sensitive information or perform phishing attacks on unsuspecting users. In a Voice Masquerading attack, a malicious skill mimics the PVA's responses to deceive users into believing they are interacting directly with Alexa rather than with a third-party skill. For instance, the malicious skill might say 'Please provide your account number to continue,' using Alexa's voice and tone [ZMF+19]. This deception is possible because PVAs provide no clear auditory indication when transitioning between the PVA and a skill, or when a skill session ends. Furthermore, attackers can employ silent audio to keep the microphone active without the user's knowledge, enabling eavesdropping on private conversations [ZMF+19].

There are command injection attacks that involve creating unclear voice com-

mands that are unrecognizable to human listeners but understandable by the ASR system. The attacker injects these commands into the system, often using techniques like hidden voice commands or ultrasonic frequencies [CMV+16, YLZ+20, ZYJ+17]. There are attacks that use light-based signal injection where a laser beam is modulated with the audio signal, which is then directed at the microphone of a PVA. When the laser light hits Micro-Electro-Mechanical Systems (MEMS) microphones, the photoacoustic effect converts the light into an electrical signal, tricking the device into interpreting the light as sound. This vulnerability is specific to MEMS microphones. This allows the attacker to inject malicious voice commands remotely from significant distances, researchers demonstrated the attack at 110 meters as it was the maximum distance safely available to them, though the actual range is limited primarily by the attacker's ability to aim and focus the laser beam [SCR+20]. These examples highlight broader threats to PVAs however, the work in this thesis addresses a different class of risk which is acoustic denial-of-service through wake-word jamming, which prevents legitimate voice interaction.

The integration of PVAs with other IoT devices creates dependencies that amplify the impact of availability attacks. PVAs often serve as a central control interface for various IoT devices, from lights and thermostats to security systems. While post-activation attacks that compromise a PVA can lead to unauthorized control of connected devices, denial of service attacks that prevent legitimate PVA activation create a different threat: rendering the entire smart home control system unavailable to authorized users. In time-critical scenarios such as voice-activated emergency calls, medical alert systems, or security camera activation, the inability to activate the PVA through wake word jamming could have serious consequences.

3.3 Denial of Service (DoS)

DoS attacks represent one of the most significant threats to the integrity and availability of IoT devices and the services they support [KABM24, NTS+25]. Given the pervasive nature of IoT devices in critical infrastructure, smart homes, and industrial applications, DoS attacks can have severe consequences, ranging from operational disruptions to significant financial losses. This chapter explores the general nature of DoS attacks on IoT devices, examining the methodologies, vulnerabilities, and impacts, and then transitions into a discussion on DoS attacks targeting the network components of PVAs. IoT devices are inher-

ently vulnerable to DoS attacks due to their limited computational resources, memory, and often inadequate security measures. DoS attacks typically aim to overwhelm the target device or network, rendering it unable to perform its intended functions. In the context of IoT devices, these attacks can be executed in several ways.

One common method is the exploitation of the device's limited processing capabilities. Attackers can flood the IoT device with a high volume of requests, overwhelming its ability to process legitimate ones. Simple IoT sensors and actuators, which typically have only a few kilobytes of RAM and process 10-20 packets per second under normal operation [Gil21, Boa25], can be overwhelmed with floods of hundreds to thousands of packets per second [Sup24]. This is particularly effective against IoT devices that have low processing power, which are unable to handle the sudden surge in traffic. For instance, DoS attacks can target smart cameras, sensors, or thermostats, causing them to become unresponsive or behave unpredictably [KE21]. Another method involves targeting the network infrastructure that supports IoT devices. Attackers can flood the network with traffic, saturating the available bandwidth and preventing legitimate data from reaching its destination. This type of attack is known as a Distributed Denial of Service (DDoS) attack when executed from multiple compromised devices, often referred to as a botnet. The infamous Mirai botnet, which compromised thousands of IoT devices, is a prime example. Mirai was used to launch massive DDoS attacks, overwhelming network infrastructures and causing widespread service outages [KKSV17]. In addition to direct flooding attacks, attackers can exploit specific vulnerabilities in IoT devices to trigger a DoS attack. For example, by sending malformed packets or exploiting protocol weaknesses, an attacker can cause the IoT device to crash or reboot, effectively taking it offline. Such attacks are particularly dangerous because they can be executed with minimal resources and may not require a large volume of traffic to be effective. For example, work by Hussein et al. [HN22] exploits vulnerabilities in the Constrained Application Protocol (CoAP), a lightweight application protocol for resource-constrained devices, and ZigBee, a low-power wireless communication protocol commonly used in smart home devices. Using a Raspberry Pi running an Message Queuing Telemetry Transport (MQTT) broker which is a publish-subscribe messaging protocol widely deployed in IoT systems to emulate an end user, the attackers gain unauthorized access to the ZigBee network. By injecting the ZigBee End Device with high frequency signals that trigger the IKEA smart bulb every 250 milliseconds, the attacker floods the

device with false boolean payloads, preventing legitimate user interactions and exhausting the device's resources. Similarly, [HMKA19] highlights how DoS attacks exploit default configurations of popular MQTT brokers such as unlimited message queues or high connection limits.

Another form of DoS attack involves battery depletion, particularly relevant for battery powered IoT devices. Attackers can continuously trigger energy intensive operations on the device, such as turning on sensors or activating communication modules, leading to rapid battery drain. Once the device's battery is depleted, it becomes inoperable until recharged, which could be difficult in remote or inaccessible locations [IV19]. Work by Vlad et al. [IR22] explains how battery depletion attacks can reduce the battery life of a Narrow-band-IoT device from 17 years to just four months, making such deployments commercially infeasible. The impact of DoS attacks on IoT devices can be profound. In smart home environments, attacks could disable security systems, leaving premises vulnerable. In industrial settings, the interconnected nature of IoT systems means that DoS attacks disrupting a single device could trigger cascading failures across the entire network, leading to production halts or safety hazards [ARHAU17].

Transitioning to the network aspects of PVAs, these systems are not immune to DoS attacks. PVAs have become increasingly integrated into consumer IoT ecosystems, particularly in smart home environments where they serve as central control hubs. The number of connected IoT devices reached 18.5 billion in 2024, representing a 12% growth over 2023, and is expected to grow 14% year-over-year to 21.1 billion by the end of 2025. Looking further ahead, the number is estimated to reach 39 billion in 2030 [Ana25]. The network components of PVAs, including the underlying communication protocols, cloud services, and internet connectivity, present multiple attack surfaces for DoS attacks. One of the primary vectors for DoS attacks on PVAs is the exploitation of their reliance on cloud based services. PVAs depend heavily on cloud infrastructure to process voice commands, manage user data, and coordinate interactions with other IoT devices. By launching a DDoS attack against the cloud service provider, attackers can disrupt the connectivity between the PVA and the cloud, rendering the assistant unable to respond to user commands or control connected devices. This was demonstrated in several high profile DDoS attacks where major cloud providers experienced outages, affecting millions of users [SGS+15, CBG18, RMF23].

Another potential target for DoS attacks on PVAs is the local network infrastructure. PVAs typically operate over Wi-Fi or other local networks, which can be overwhelmed by a targeted DDoS attack. By flooding the network with traffic, attackers can saturate the available bandwidth, preventing the PVA from communicating with the cloud or other IoT devices. This not only disrupts the PVA but can also affect all devices connected to the same network, amplifying the impact of the attack [OWH19b].

The consequences of DoS attacks on the network components of PVAs can be significant. For individual users, such attacks can result in the loss of control over smart home devices, leading to inconvenience or, in more severe cases, security risks. While large scale DDoS attacks targeting PVA backend infrastructure remain theoretical, cloud service disruptions have demonstrated the potential impact on PVA availability for instance, Amazon Web Services (AWS) outages have rendered Alexa services temporarily unavailable, affecting millions of users and connected devices [Dat25]. The interconnected nature of PVAs and IoT devices means that service disruptions, whether from DoS attacks or infrastructure failures, can extend beyond the immediate target, affecting the broader IoT ecosystem and potentially compromising critical services. Alepis et al.[AP17] shows how malicious apps can secretly activate PVAs on a user's device without their permission and execute voice commands. The goal of these attacks is to steal personal data, make unauthorized calls or purchases, or even target nearby devices. The paper also explores how attackers can use tools like text-to-speech technology, cloud services, and techniques such as detecting nearby devices through WiFi signals and vibrations to carry out these attacks.

This chapter provides essential background on the broader threat landscape facing PVAs to establish context for our focused contribution. PVAs present multiple attack surfaces: network infrastructure, cloud services, application layer, and the acoustic interface. While network-level and application-level attacks have received substantial attention in prior research, the acoustic interface, specifically DoS through the acoustic channel, remains critically under explored despite being the primary entry point for all voice interactions. Unlike network attacks that can be mitigated through firewalls and encryption, or application attacks addressable through authentication, acoustic DoS attacks exploit the fundamental physics of sound propagation and audio signal processing, making them both harder to defend against and more universally applicable across all PVA deployments.

3.4 Jamming Attacks on PVAs

Jamming, in the context of communications and signal processing, refers to the deliberate interference with the transmission or reception of signals, rendering them unintelligible or unusable [Enc24]. This technique is commonly associated with Radio Frequency (RF) communication, where it is used to block or disrupt wireless communication channels [Dep20]. However, the concept of jamming can be extended to the acoustic domain, where it involves the intentional introduction of noise or interference into the sound environment to disrupt the input audio signals. PVAs rely on their microphone arrays to capture audio from the surrounding environment, which is then processed to recognize and respond to voice commands. Acoustic jamming can be executed either for the purpose of privacy protection of the users or to disrupt the functionality of the PVA with a malicious intent leading to a DoS attack.

For privacy protection, jamming is used to prevent PVAs from eavesdropping on sensitive conversations [CBYR18, SCZ20] Privacy protection in PVAs can be achieved through selective jamming mechanisms that prevent unintended recording of private conversations while allowing legitimate voice commands to pass through. The core idea revolves around detecting the wake word faster than the PVA itself and then applying a jamming signal to the latter part of the wake word, ensuring the PVA fails to recognize it and thus does not record subsequent audio. This type of jamming can be relatively simpler to implement yet highly effective. An attacker could deploy a simple device that generates noise whether it be white noise, where signals distributed across all audible frequencies, directional sound waves which are focused acoustic beams targeting specific locations, or ultrasonic signals with frequencies above 20 kHz, inaudible to humans but detectable by microphones in proximity to the target PVA.

For DoS attacks, acoustic jamming aims to render the PVA unresponsive or unreliable, with potentially significant consequences. Beyond the basic noise generation techniques, DoS attacks can exploit the resonance effect in MEMS microphone, where attackers broadcast frequencies that match the natural resonance frequency of the microphone's diaphragm, causing it to vibrate excessively and distort or block incoming audio signals [OAK18]. These acoustic jamming methods are highly effective in disrupting the PVA's ability to process legitimate voice commands [ZYJ+17]. Investigating these jamming methods as a specific research challenge would be worthwhile, particularly in understanding detection mechanisms and developing robust countermeasures against such

attacks.

The acoustic channel makes it an attractive target for attackers, though voice-based attacks require physical proximity to devices. Unlike network-based DoS attacks, which often require technical expertise and resources, acoustic jamming can be executed with relatively simple equipment, though careful calibration may be necessary depending on environmental conditions and target distance. A noise generator or ultrasonic emitter can launch an effective DoS attack on a PVA. Depending on the sophistication of the jamming device, the attack could be sustained indefinitely, continuously disrupting the PVA's ability to function. Alternatively, the jamming could be applied selectively, targeting specific times when the PVA is expected to be in use, thus maximizing the impact of the disruption. An example of such jamming is "Wake Word Jamming".

Chapter 4

Wake Word Jamming

xThe wake word is an essential component of PVA functionality, as it signals the device to wake up from its passive listening state and actively process voice commands from the user. This feature is designed to make the interaction with the PVA seamless, enabling hands-free operation without the need for physical input. However, the wake word detection mechanism is susceptible to acoustic jamming attacks. If the acoustic interference occurs during the detection phase, the PVA may fail to recognize the wake word and remain in its passive state, unable to perform any tasks or respond to commands. This type of disruption can lead to what is referred to as a DoS attack.

In this chapter, we explore how a DoS attack can be carried out by disrupting the wake word detection with jamming signals. We describe how a good wake word jamming signal should be constructed. A good jamming signal should be difficult to detect and should not be noticeable by users. Thus, signal strength, duration, timing and frequency need to be considered. Broadly speaking, the jamming signal energy should be minimised in strength, duration, bandwidth to reduce detection and also to improve jamming efficiency, that is, to maximize the probability of preventing wake word detection while minimizing the energy required to do so. We provide an in-depth analysis of the impact of jamming signal strength, timing, duration, frequency and bandwidth on wake word recognition. We investigate the impact of different noise types such as White noise, Ping noise and Keyboard noise on wake word recognition. We show that the specifically selected jamming signal type is secondary and that rather the correct selection of jamming signal parameters such as timing determines jamming success.

4.1 Related Work on Wake Word Jamming

Since the wake word is the entry point to a PVA, as described in section 2, it is crucial to ensure the reliability of wake word detection. The accuracy of this process depends on two main factors: the effectiveness of the detection algorithms and the acoustic conditions under which the wake word is captured by the PVA's microphone. The acoustic environment includes everything from the quality of the air transmission, distance from the microphone, and any surrounding sounds. When the PVA is moved to a different location, this environment changes, potentially introducing new background noises or altering how the wake word reaches the microphone. These changes can disrupt wake word detection, making it harder for the device to recognize commands. Background noise can come from everyday household sounds like doors closing or the TV, or from public noises like street traffic and chatter, depending on where the PVA is placed. While these interferences are unintentional, they are unavoidable. However, the impact of such noises can be minimized by using noise-robust wake word detection algorithms that are designed to work even in noisy environments [LKK20]. Similar to these approaches, the algorithms investigated in this thesis also incorporate background noise in their training to ensure robustness in real-world acoustic conditions.

In addition to accidental interference, there is also the possibility of intentional interference from external sources, which is known as wake word jamming. In this scenario, a jamming signal masks the wake word, preventing the PVA from recognizing it and entering its active state. This type of interference can have different motives. On the positive side, it might be used to protect a user's privacy in public by preventing the device from activating unintentionally. However, it could also be used maliciously, such as in a DoS attack, where an attacker jams the wake word to block legitimate access to the PVA.

In literature, the wake word jamming is done by continuously interrupting the wake word using various sound sources that are audible (guitar, white noise etc) and creates a noisy environment [SCZ20, LQL+19, CBYR18].

Work by Cheng et al. [CBYR18] has explored the use of a jamming signal to selectively block PVA activation, allowing users to control when their device can listen to voice commands. A Protection Jamming Device (PJD) continuously observes the channel for the wake word, and as soon as the wake word is spoken, the PJD emits a jamming signal. The jamming signal is emitted before comple-

tion of the wake word detection by the PVA. The work examines how much signal overlap between jamming signal and wake word is required to perform successful jamming. Also, the use of different sounds is explored. However, details on how to construct the jamming signal efficiently are not analyzed (e.g. duration, bandwidth, signal strength). In this work, we systematically analyze these jamming signal parameters and their impact on wake word detection to enable the construction of efficient jamming signals with minimal energy expenditure.

Work by Ke Sun et al. [SCZ20] looks at jamming the acoustic channel to protect speech privacy against always-on microphones. A jamming device called MicShield continuously emits a jamming signal to deafen the PVA while MicShield monitors for the wake word phoneme pattern. Emission is stopped as soon as the wake word phoneme pattern is recognized. The jamming is performed using white noise of 0-4kHz but the audio signal is made inaudible by using a higher frequency carrier of 40kHz. Due to microphone nonlinearities and low-pass filtering, the 40 kHz ultrasonic carrier is stripped away, leaving only the low-frequency jamming noise (0-4 kHz) that affects the PVA's microphone. This approach exploits nonlinearities common to commercial microphones, making it broadly applicable across different PVA devices.

Work by Juncheng et al. [LQL+19] describes a DoS attack on PVAs by jamming the wake word. An adversarial music synthesizer is used to create synthesized guitar-timbre sounds to perform the jamming. It is observed that the jamming success rate increases when the jamming signal has a volume above 40dBA as measured by a decibel meter in the testing environment. The work does not explore other types of noises and does not investigate how to create an efficient jamming signal. The jamming follows a continuous jamming approach which could alert the user.

Work by Gong et al.[GGCPRB+19] also describes a DoS attack referred to as *AudiDoS*. In this method, small perturbation is continuously played near the target PVA to cause interruption, the perturbations are unnoticeable by the users in the vicinity. These perturbations cause the models to misclassify user inputs, making the system unable to correctly respond to spoken commands. This method is designed to be a universal adversarial perturbation, aiming to be effective regardless of the user's command. The attack is not performed on a wake word detection algorithm specifically but rather targets the general speech recognition model with continuously played perturbations.

Long-duration and continuous jamming is very noticeable, and therefore the use of inaudible sound sources has also been explored in [LKK20]. The idea is that by exploiting the properties of microphones, which have nonlinearities, an attacker can inject inaudible voice commands that cannot be heard by humans but can still be interpreted by these systems. The authors introduce a technique called *DolphinAttack*, where they modulate low-frequency voice commands onto ultrasonic carriers (above 20 kHz), making the signals inaudible to humans. These inaudible commands are then demodulated by the microphones of voice-controlled systems, allowing the devices to process and execute the commands as if they were regular audible instructions. The goal is not to disrupt the wake word detection but rather to exploit the PVA for unauthorized actions.

While the aforementioned works demonstrate various wake word jamming approaches, they primarily focus on continuous jamming without investigating the optimal timing and placement of jamming signals within the wake word. Continuous jamming is more detectable. In contrast, this thesis investigates selective jamming by systematically analyzing when and where to place jamming signals within the wake word for maximum effectiveness. We examine how jamming signal parameters such as timing, duration, bandwidth, and signal strength affect wake word detection. This enables the design of efficient jamming signals that target specific parts of the wake word, achieving high disruption with minimal energy consumption.

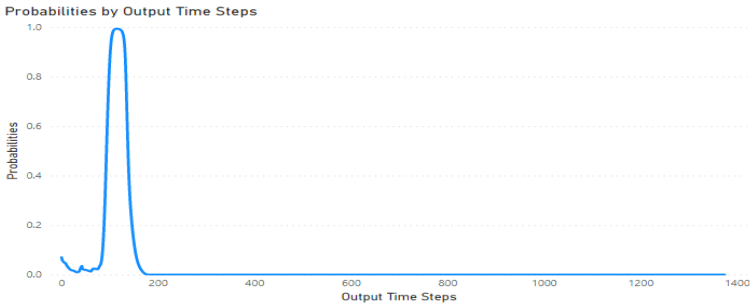
4.2 Wake Word Detection

The wake word detection system is at the core of how people interact with PVAs. Wake word detection algorithms work differently compared to SR algorithms used in the back end. Wake word detection only checks for the presence of the wake word rather than transcribing full speech, and the process is executed continuously. DNNs are widely employed for wake word detection in both commercial and research systems. The implementation details of these commercial devices are not available for public, however, we use for our study the public available wake word detection algorithm by Supriya et al. [SDSV20] which is similar to the commercial wake word detection algorithms. We refer to this algorithm as *LSTM-WD*. This algorithm, like most commercial wake word systems, implements speaker-independent detection trained to recognize the wake word from any speaker rather than being personalized to a specific user's

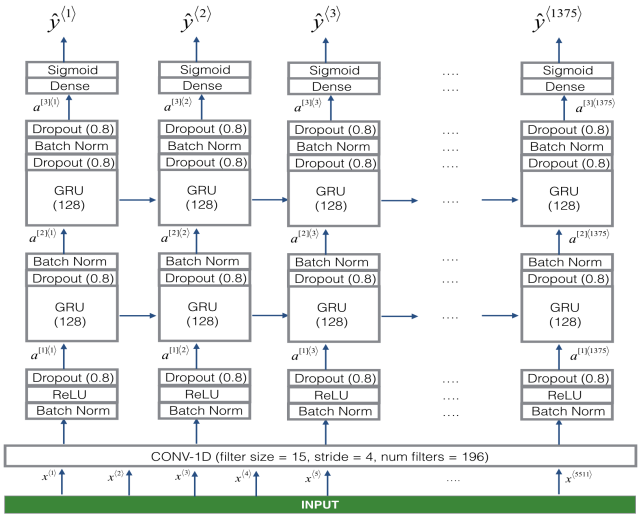
voice. Since wake word detection operates continuously on-device, maintaining a lightweight model is critical to meet strict computational, memory, and power constraints. For this reason, commercial PVAs predominantly employ speaker-independent wake word detection. The LSTM-WD uses a one-dimensional convolutional layer connected to two Gated Recurrent Unit (GRU) layers and a final dense+sigmoid layer to predict the wake word. The details of the architecture are as shown in 4.1.

The model works with a 10 second audio input sampled at 44.1 kHz with 16-bit depth and mono channel. The audio is processed into a spectrogram using overlapping windows of length 10ms, producing 5,511 time frames per 10 second input. The spectrogram is then fed to a 1D convolutional layer as shown in Fig. 4.1. The features extracted by the convolutional layer for each window are then processed by two GRU layers. GRUs are a type of RNN designed to handle sequential data and capture temporal patterns in audio. They are similar to LSTM units but use a simpler architecture with fewer parameters. GRUs address the vanishing gradient problem of standard RNNs through two gating mechanisms: a reset gate that controls how much past information to retain, and an update gate that determines how much new information to incorporate. This allows the model to effectively learn long-term dependencies in the sequential spectrogram features and identify patterns characteristic of the wake word.

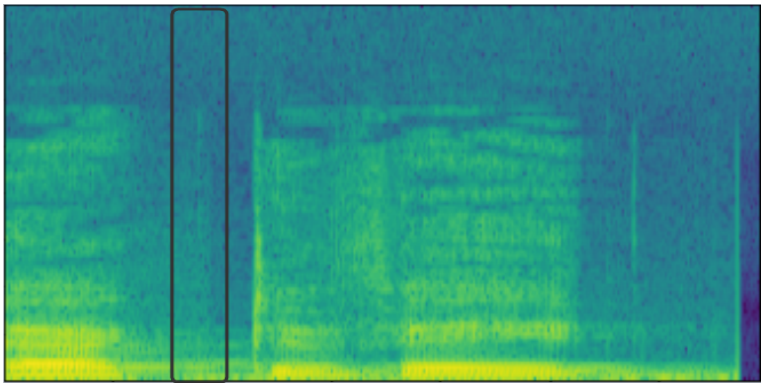
To stabilize and accelerate the training process, batch normalization is applied to the features extracted by the convolutional layer or between the GRU layers. Batch normalization normalizes the activations of each layer, which helps stabilize training and improve convergence speed [STIM18, IS15]. Additionally, dropout is used as a regularization technique to prevent overfitting. During training, dropout randomly sets a fraction of neurons to zero, forcing the network to learn more robust features that are not reliant on any single neuron [SHK+14]. This ensures that the model generalizes well to unseen data. The outputs from the GRU layers are fed into a dense layer with a sigmoid activation function, which produces a probability score for each time step. The 1D convolutional layer reduces the 5,511 input time frames to 1,375 time steps through convolution. Therefore, for a 10-second audio input, the dense layer outputs 1,375 probability scores, corresponding to each of these time steps, indicating the likelihood that the wake word is present at that specific moment. The threshold for wake word detection is set to 0.5, which is a standard threshold for binary classification with sigmoid activation, representing equal



(a) Probability Graph



(b) Model



(c) Spectrogram



(d) Speech Signal (Wake Word)

Figure 4.1: LSTM-WD Wake Word Detection Steps (Bottom to Top). Wake Word (10s audio clip sampled at 44.1 kHz) → spectrogram windows of 10ms → DNN model → probability scores (1375 scores for 10s input). Wake word is detected if any probability score is above 0.5 [SDSV20].

probability between the presence and absence of the wake word. If any one of the 1,375 probability scores crosses this threshold, the given audio input is recognized as containing the wake word.

4.2.1 Jamming Process

The key to successful jamming is a properly designed jamming signal with appropriate timing, duration, bandwidth, and signal strength. One of the main objectives of this thesis is to systematically analyze these jamming signal parameters and identify the optimal characteristics for effective wake word disruption with minimal energy expenditure. Ideally, the jamming signal should be difficult to detect while still being strong enough to disrupt wake word detection. To determine which parameter combinations are most effective, we create jamming signals from the jamming signal design space as shown in Table 4.1. A wake word sample is then combined with(superimposed or replaced) with each jamming signal and fed to the LSTM-WD model. The model generates 1,375 probability scores, and if all of these scores fall below the threshold of 0.5, the jamming signal is considered successful in preventing the wake word from being detected. To analyze how the position of the jamming signal affects its effectiveness, we systematically shift the jamming signal across the wake word duration. The jamming signal is initially aligned with the start of the wake word, and then shifted forward in 1ms increments, testing each position to determine jamming effectiveness. This small step size allows for a detailed examination of how the jamming signal affects the wake word at various points in time.

To begin our experiments with wake word jamming, we investigate different jamming signal types and their parameters. While we initially focus on white noise as it contains equal frequencies and energies across the entire spectrum, providing a good baseline, we also examine other signal types such as Ping noise and Keyboard noise for our experiments. The waveform and spectrogram representations of these noise signals are illustrated in 4.2. The jamming signal is restricted to a particular duration and frequency band, and is described by the following five parameters:

1. *Type* T: The type of base signal used; for example, White noise, Ping noise and Keyboard noise.
2. *Strength* S: S describes how loud the jamming signal is in relation to the

wake word. S is calculated as:

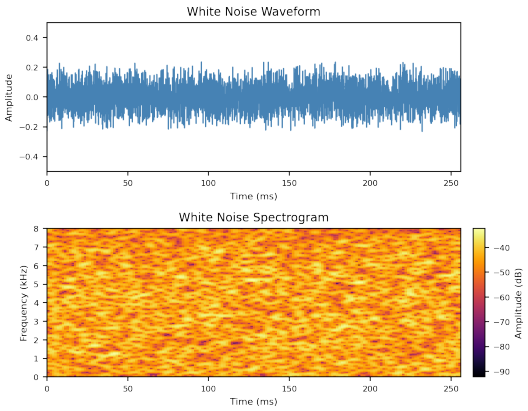
$$S = 20 \times \log\left(\frac{RMS^2_{signal}}{RMS^2_{noise}}\right) \tag{4.1}$$

Where RMS_{signal} and RMS_{noise} are respectively the root mean square values of signal and noise.

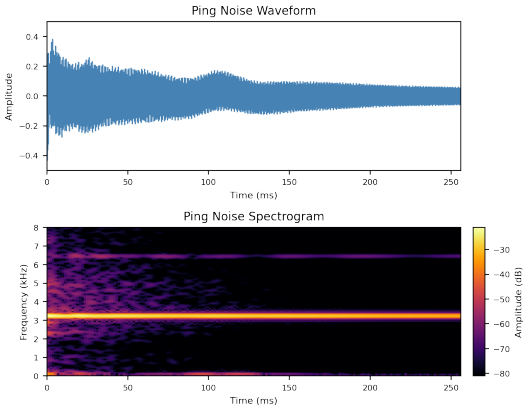
3. *Timing* t: t describes where in relation to the wake word the noise occurs. For $t = 0ms$ the noise starts at the beginning of the wake word.
4. *Duration* τ: τ is the noise window length placed at t.
5. *Frequency* f: f describes in the frequency domain where the noise starts.
6. *Bandwidth* B: B describes the length of the noise in the frequency domain starting at f.

For example, a jamming signal with parameters, $\{T = white, S = 0dB, t = 50ms, \tau = 10ms, f = 0kHz, B = 2kHz\}$ refers to white noise that has the same loudness as the wake word (S = 0dB indicates equal signal levels), starts 50ms after the wake word begins, lasts for 10ms, and covers a frequency range from 0kHz to 2kHz. In our experiments, we maintain the jamming signal at the same amplitude level as the wake word to provide a controlled baseline for analyzing the effects of other parameters such as timing, duration, and frequency content. The audio samples are normalized to -16.5 dBFS in the digital domain, corresponding to the measured Root Mean Square (RMS) level of the wake word signals used in our experiments. When played back at typical volume levels, this digital level corresponds to normal conversation loudness of approximately 60-70 dB in the acoustic domain. The evaluation presented in this thesis uses the wake word 'Activate' from the publicly available trigger word detection dataset, which includes a development set of 25 10-second audio clips from multiple speakers. The dataset is provided as part of the trigger word detection algorithm implementation and contains samples with varying accents [SDSV20]. We use these samples to test the jamming signal effectiveness across different voice patterns and ensure the generalizability of our findings.

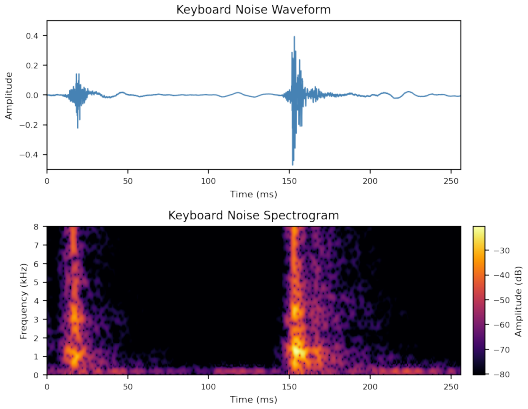
For the detailed jamming analysis presented in this chapter, we use one specific speaker's recording of the wake word 'Activate' as our reference sample. This sample has a duration of 631ms and is used consistently across all jamming experiments. Using a single speaker's sample allows us to clearly represent the effect of jamming parameters without the confounding factor of duration vari-



(a) White Noise



(b) Ping Noise



(c) Keyboard Noise

Figure 4.2: Waveform and spectrogram representations of the considered jamming signal types.

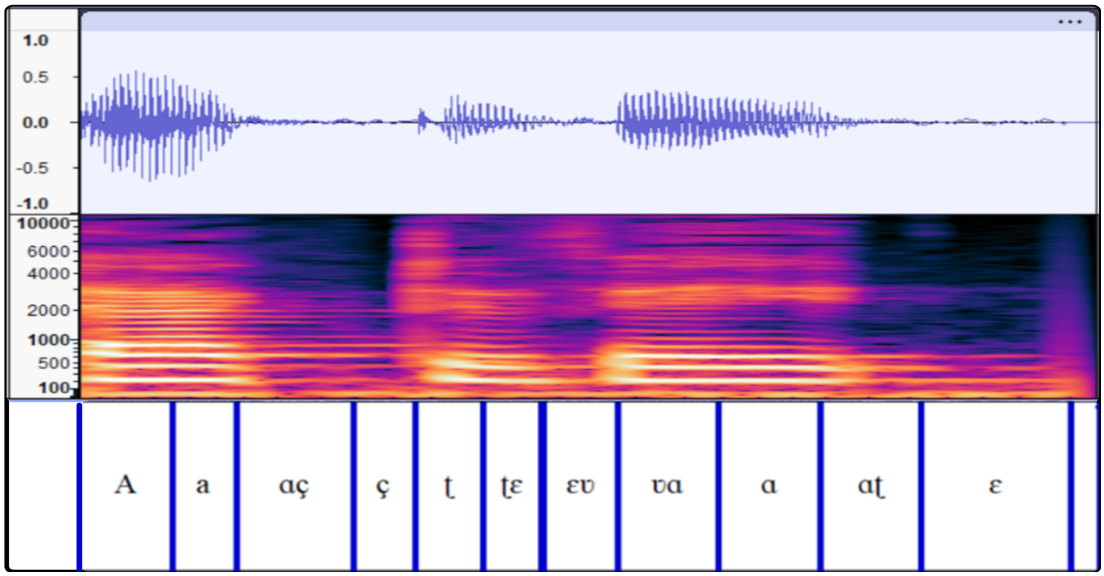


Figure 4.3: Wake Word 'Activate'

ability, as different speaker's utterances of 'Activate' have different durations. To validate the generalizability of our findings across different voices, we also tested the jamming techniques with samples from all available speakers in the dataset, and observed consistent jamming effects.

The chosen wake word segment is 631ms long, as shown in Figure 4.3 with its spectrogram and time domain phoneme level representation. However, the LSTM-WD model requires a 10-second input window, so we construct an input signal by placing the wake word at 0ms and padding the remaining with trailing silence to provide a total signal duration of 10 seconds. The figure only shows the wake word duration which is 631ms out of 10-seconds.

4.2.2 Signal Combination

We consider in our experiment two methods of combining wake word and jamming signal: *superimposition* and *replacement*. In case of superimposition, the jamming signal is superimposed over the wake word before the resulting combination is fed to the LSTM-WD detector. In case of replacement, the jamming signal completely replaces portions of the wake word signal. This scenario was tested to understand the extreme case where the wake word is entirely cut off. While replacement is not a practical jamming approach in real-world scenarios with live speech, it provides useful insights into the wake word detection algorithm's behaviour when portions of the wake word are missing.

4.2.3 Jamming Energy

A good jamming signal should be difficult for human users to perceive and should not be noticeable. The jamming signal energy should be minimized (strength, duration, bandwidth) which reduces the risk of detection and also improves jamming efficiency. An attacker may therefore like to know how best to invest a given amount of jamming energy. For example, should energy be invested in a wider jamming bandwidth B or a longer jamming duration τ?

To evaluate this aspect we consider two experimental scenarios: *constant energy* and *variable energy*.

In the case of constant energy we use the jamming signal of the experiment set with the longest jamming duration τ and largest bandwidth B as reference. For this reference case we calculate the wake word audio(631ms) signal energy E_{ref} as:

$$E = N \cdot A^2 \tag{4.2}$$

where N is the number of samples and A is the RMS amplitude of the signal. We then calculate the jamming signal energy E_i for experiment i in the set with lower duration τ_i and bandwidth B_i. We then calculate a compensation factor $c = \sqrt{(E_{ref}/E_i)}$ which is used to adjust the jamming signal amplitude. Thus, each jamming signal in the experiment set uses the same signal energy. We chose this approach as an attacker may like to know how best to invest a given amount of jamming energy.

In the case of variable energy the amplitude of the jamming signal is not corrected. Thus, if jamming bandwidth B or a jamming duration τ is reduced, the signal energy is also reduced.

4.3 Evaluation Results

We present four experiment sets. In each experiment set the parameter space listed in Table 4.1 is used to construct jamming signals which are then combined with the wake word sample described in Section 4.2 and shown in Figure. 4.3. Thus, each experiment set evaluates the impact of 151,440 different jamming signals (631 positions of wake word, 4 durations, 4 strength levels and 15 bandwidth combinations) on the wake word. For each jamming signal

Table 4.1: Jamming signal parameters as explained in 4.2.1 and their tested ranges for parameter definitions). The timing range [0, ..., 631] ms covers the entire duration of the 631ms wake word sample, testing jamming placement at 1ms intervals.

Parameter	**Range**	**Unit**
Type (T)	[White noise,Ping noise, Keyboard noise]	-
Strength (S)	[-10, -5, 0, 5]	dB
Timing (t)	[0, 1, 2, 3, ..., 631]	ms
Duration (τ)	[2, 32, 128, 256]	ms
Frequency configurations (center frequency f and bandwidth B):		
$B = 1$ kHz	$f \in$ [0, 1, 2, 3, 4, 5, 6, 7]	kHz
$B = 2$ kHz	$f \in$ [0, 2, 4, 6]	kHz
$B = 4$ kHz	$f \in$ [0, 4]	kHz
$B = 8$ kHz	$f = 0$	kHz

tested we note the result of the wake word detection. Due to the large number of parameter combinations and the multi-dimensional nature of the parameter space, we visualize the results using heat maps. Heat maps are particularly suitable for this analysis because:

- They allow simultaneous visualization of multiple parameters while using colour intensity to represent the binary detection outcome
- They reveal spatial and temporal patterns in jamming effectiveness, showing precisely when and where in the wake word jamming is most effective
- They compress 151,440 data points into interpretable figures, making comprehensive analysis tractable.

The four experiment sets are:

1. Experiment Set 1 - superimposition and constant energy: The jamming signals are superimposed with the wake word and jamming energy is constant when modifying jamming duration τ or bandwidth B. All jamming signals are normalized to have equal total energy by adjusting their amplitude using the compensation factor.
2. Experiment Set 2 - superimposition and variable energy: The jamming signals are superimposed with the wake word and the amplitude of the jamming signal remains constant when modifying jamming duration τ or bandwidth B, no energy normalization is applied here.
3. Experiment Set 3 - replacement and constant energy: The jamming signal replaces the wake word and jamming energy remains constant.

4. Experiment Set 4 - replacement and variable energy: The jamming signal replaces the wake word and the jamming signal amplitude remains constant.

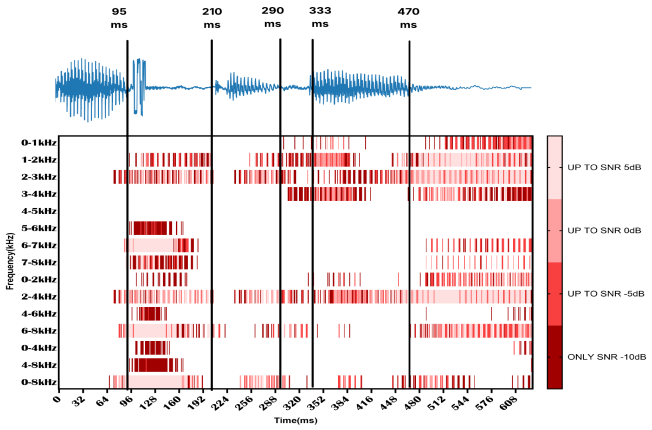
5. Experiment Set 5 - Evaluation of Jamming in an Over-The-Air-Setup

4.3.1 Experiment Set 1 — Superimposing Noise and Constant Energy

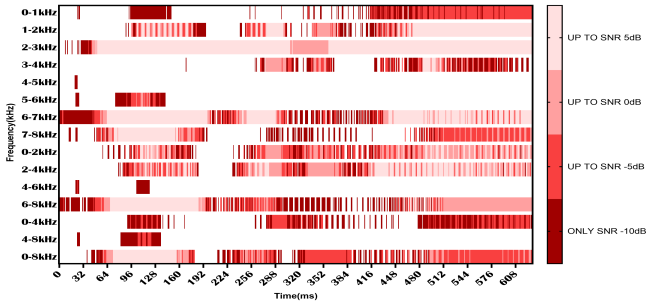
The results are shown as heat maps in Figure 4.4, which represents the wake word detection success superimposing white noise with constant energy. Each sub-figure shows the effect of different jamming signal duration τ. The x-axis(independent) represents time and covers the duration of the wake word of 631ms. The y-axis represents the jamming signal frequency f and bandwidth B combinations considered. The colour represents the initial signal strength S calculated for $\tau = 256ms$ and $B = 8kHz$.

If a jamming signal is successful in interrupting wake word detection we select the figure of corresponding jamming duration τ and the row corresponding to the used frequency f and bandwidth B. We then select in this row the time t when the jamming signal started in relation to the wake word and mark this position with the colour corresponding to the initial jamming signal strength S. We use the colour shade for the weakest signal that is able to prevent wake word detection; note that if a weak signal is successful, a stronger signal with the same configuration (t, τ f, B) will also succeed. Darker red indicates positions requiring stronger jamming signals meaning these portions of the wake word are harder to disrupt. Lighter shades indicate positions where weaker jamming signals were sufficient, representing more vulnerable portions that are easier to jam with less energy. White areas show areas in which jamming was unsuccessful.

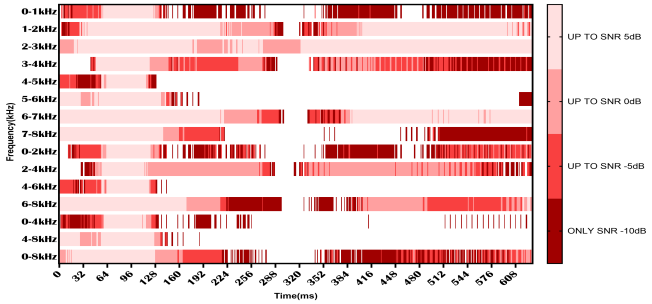
The heat maps show where jamming in the time and frequency domain can be effective. The waveform displayed above the heatmaps shows an example of the wake word audio signal with a jamming signal superimposed at one specific position, illustrating what a jammed wake word looks like in the time domain. The vertical lines mark the boundaries between different phonetic regions of the wake word, specifically, the transitions between voiced and unvoiced segments. Note that the x-axis time intervals and labels are automatically generated by GraphPad Prism software based on the data range [Gra].



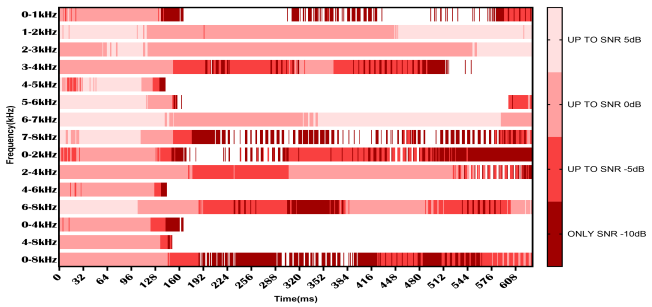
(a) 2ms jamming duration



(b) 32ms jamming duration



(c) 128ms jamming duration



(d) 256ms jamming duration

Figure 4.4: Wake word detection success superimposing white noise with constant energy.

Using Figure. 4.4 it can be seen that the more energy the jamming signal has (defined via initial strength S) the more effective the jamming is (all red areas of any shade are effective for the strongest signal). This observation aligns with intuition.

However, a longer duration τ of the jamming signal is not necessarily more useful than a shorter duration. For example, as shown in Figure. 4.4d in the band 0-8kHz a jamming signal of duration $\tau = 256ms$ placed between $t = 0$ and $t = 150$ is only successful for an initial strength of $S = 0dB$. As shown in Figure. 4.4c a signal of a shorter duration $\tau = 128ms$ is successful at an initial strength of $S = 5dB$. This pattern is observed consistently across multiple timing positions and frequency bands in our experiment set, indicating that for a given jamming energy budget, intermediate durations can be more effective than the longest duration. In summary, for a given jamming energy budget the longest jamming duration τ is not necessarily the best choice.

Figure. 4.4a depicts the results for a jamming window of 2ms duration. Spending the available jamming energy budget in such short jamming duration τ is not as successful as using a longer duration. However, it can be seen that such short intense burst of interference can effectively disrupt wake word detection if placed correctly. For example, a 2ms jamming signal is particularly effective between 100ms and 150ms. This is counter-intuitive as when looking at the wake word in the time domain (See Figure. 4.3), this is a relative silent area within the wake word. This jamming sensitive area is also visible when considering longer jamming duration, e.g. 128ms as shown in Figure. 4.5c. It is visible that a jamming signal with low strength is effective in this area between 100ms and 150ms. The jamming signal is effective in areas where the wake word has a low energy density. The selection of jamming frequency f and bandwidth B has also some impact on jamming success. While the use of the entire considered spectrum from 0-8kHz is effective it is notable that the bands 2-3kHz and 6-7kHz are specifically useful. One would assume the usefulness of the 2-3kHz band as speech is present in this range. The effectiveness of the 6-7kHz band is initially unexpected, as fundamental frequencies of human speech do not typically extend to this range. However, this band may be effective due to microphone sensitivity characteristics at higher frequencies, or sibilant sounds which produce high-frequency spectral content that can extend into the 6-7kHz range. The band from 4-5kHz is least useful for a jamming signal. The analysis presented above is based on a single wake word sample from one speaker to provide detailed insights into jamming signal characteristics. To validate

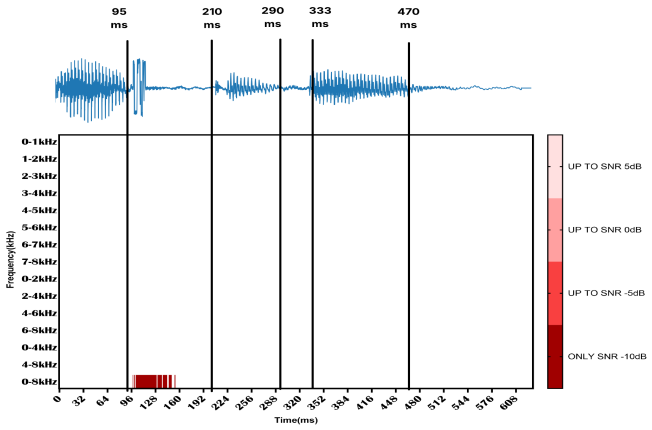
the generalizability of these findings, we also evaluated jamming effectiveness across all available wake word samples in the dataset. The key observations are consistent across all tested samples, though with some speaker dependent variation in the precise timing and amplitude requirements.

Note that in this experiment set, the jamming energy is constant for a selected initial S; if the duration τ or bandwidth B is reduced the amplitude of the jamming signal is increased. Thus, Figure. 4.4 allows us to determine how an attacker may best invest an available energy budget to disrupt the detection of this particular wake word utterance. If the attacker is not able to precisely time the attack signal it would be the best option to use a frequency band of 2-3kHz. In this frequency band a jamming signal of 32ms, 128ms or 256ms duration is always successful, irrespective of timing and in most cases with the lowest initial signal strength. Also signals with 2ms duration are often effective irrespective of timing in this frequency band; however, a stronger signal strength is often required. If the attacker has better control on timing of the jamming signal more options exist. For example, a short 2ms jamming signal placed at 96ms to 162ms is successful. The attacker can allow for some variability in timing and still be successful. A short duration of the jamming signal may be beneficial as it may be less noticeable by users than a long duration, even if emitted with lower amplitude.

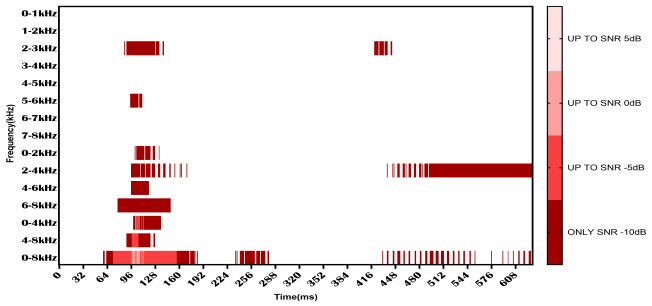
4.3.2 Experiment Set 2 — Superimposing Noise and Variable Energy

In this experiment, the amplitude of noise signal is kept constant across all jamming duration resulting in the longest window having the strongest jamming signal energy. It is important to note that the 0-8kHz, 256ms configuration serves as the reference baseline for calculating signal strength values S in both experimental scenarios. Consequently, the 0-8kHz row in Figure 4.5d shows the same signal strength values as the 0-8kHz row in Figure 4.4d. However, they are not duplicates since they represent two fundamentally different jamming strategies. The key distinction between the two experiments lies in how other duration and bandwidth combinations are treated. In the constant energy case, shorter windows have increased amplitude to maintain equal energy, whereas in the variable energy case, all jamming windows use the same amplitude, resulting in lower total energy for shorter windows.

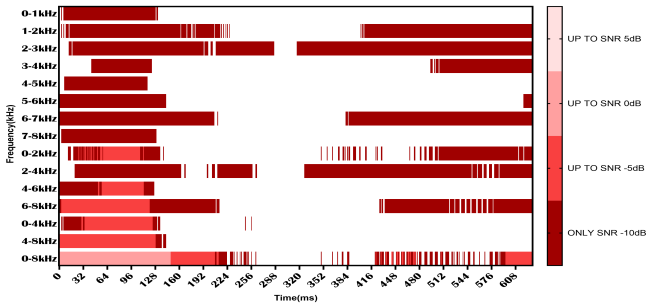
As we can see, the jamming signal becomes quickly ineffective when reducing



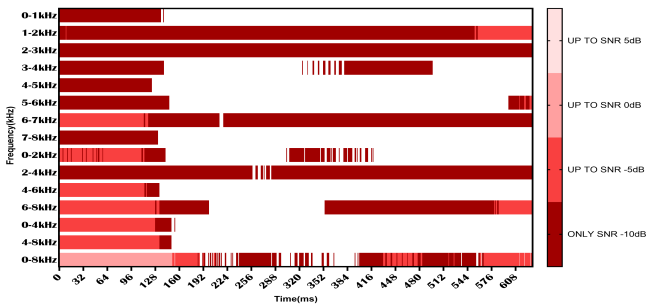
(a) 2ms jamming duration



(b) 32ms jamming duration



(c) 128ms jamming duration



(d) 256ms jamming duration

Figure 4.5: Wake word detection success superimposing white noise with variable energy.

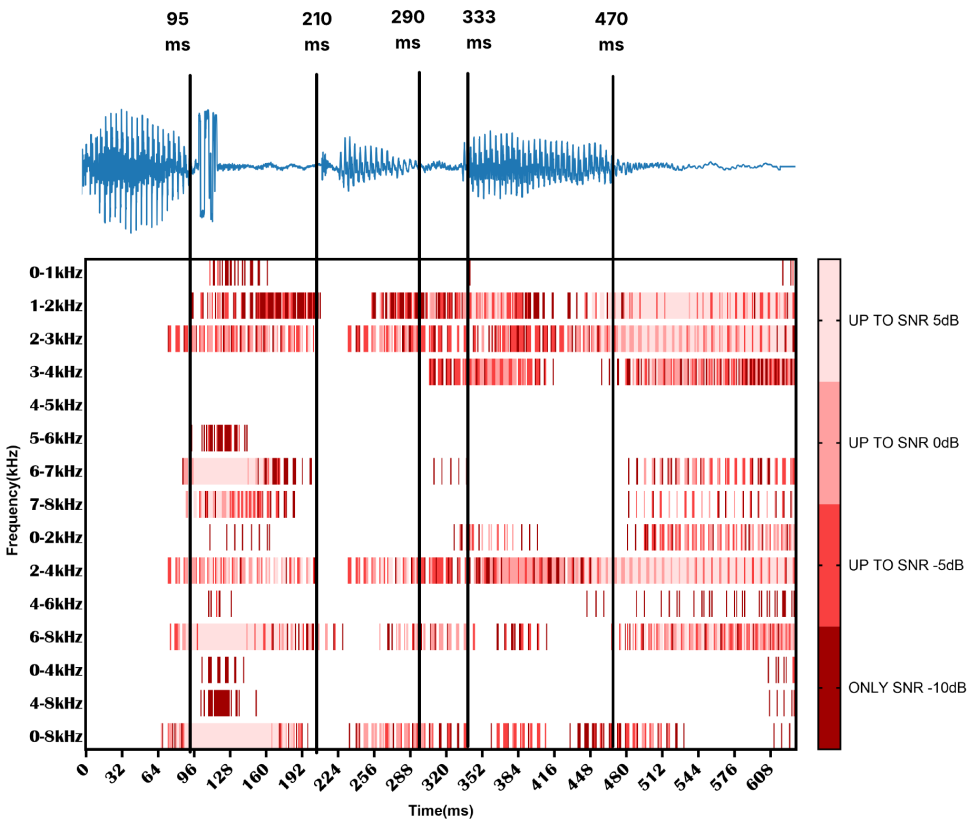
the jamming duration as the signal energy reduces. The bands 2-3kHz and 6-7kHz are most effective, this observation corresponds with the previous results from Set 1.

For a short jamming duration of 2ms the jamming signal is entirely ineffective except when using it with the full band of 0-8kHz between 100 and 150ms. As the amplitude is kept constant when reducing bandwidth and duration this is the only situation where a 2ms jamming signal has sufficient energy to cause disruption. For 32ms duration the signal is effective between a 100ms to 150ms. For 128ms and 256ms duration jamming in all frequency bands from 0ms to around 150ms. It has to be noted that a signal placed at $t = 0$ with duration of 128ms or 256ms reaches well into the period between 100ms and 150ms with sufficient energy in this area. The results indicate a particular sensitive region from 100 to 150ms; this corresponds with results from Set 1.

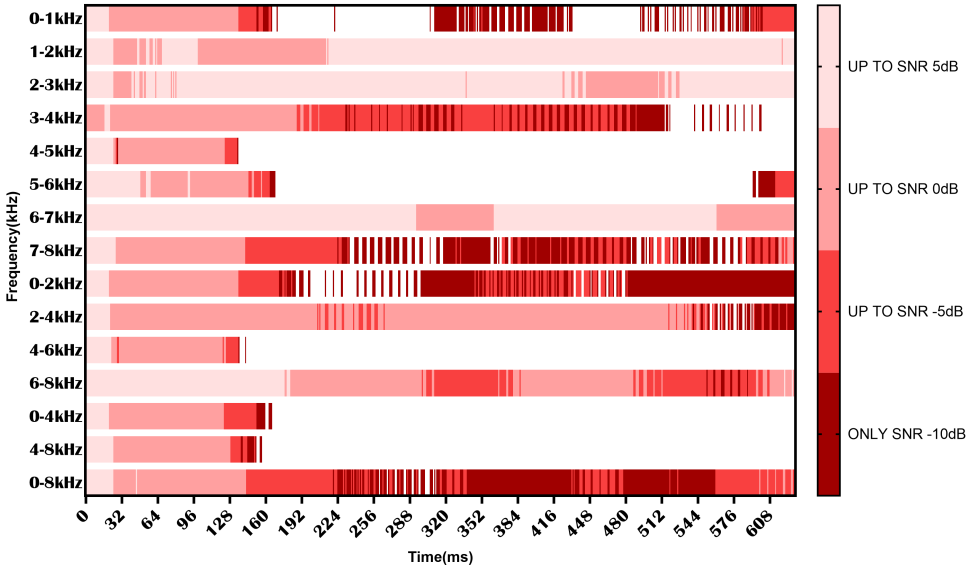
4.3.3 Experiment Set 3 — Replacing Wake Word and Constant Energy

In this experiment set the wake word is replaced with the jamming signal instead of superimposing it. This can be seen as an ideal jamming case. In areas where the jamming signal is placed, only white noise is remaining. For example, an attacker could craft a signal such that after superimposition only white noise remains. Such approach may be feasible with accurate timing and the knowledge of the audio signal; we know that the audio contains the spoken word 'Activate'. The signal energy of the replaced jamming signal is kept constant when varying duration τ and bandwidth B; the initial signal strength S is calculated for $\tau = 256ms$ and $B = 8kHz$. The results of the experiment are shown in Figure. 4.6. We only show the results for $\tau = 2ms$ and $\tau = 128ms$ as the results are similar to the results of Set 1. In some regions, replacement jamming is more effective. For example, in the 6-7kHz band around 320ms for $\tau = 2$ms, replacement successfully prevents wake word detection. However, in other regions replacement is less effective. The 0-1kHz band starting at $t = 450$ms for $\tau = 2$ms shows reduced jamming success with replacement compared to superimposition

It can be concluded from this experiment that effort in crafting a jamming signal that creates an optimal white noise in the jamming region is not worth the effort. Superimposed white noise is sufficient.



(a) 2ms jamming duration



(b) 256ms jamming duration

Figure 4.6: Wake word detection success replacing the audio signal white noise with constant energy. Heat maps show results for (a) $\tau = 2$ms and (b) $\tau = 256$ms durations

4.3.4 Experiment Set 4 — Replacing Wake Word and Variable Energy

In this experiment set the wake word is again replaced with the jamming signal. The energy is variable as the amplitude remains constant. The results are shown in Figure. 4.7.

When comparing the two scenarios for varying noise energy given in Figure. 4.5 (superimposing wake word with noise) and Figure. 4.7 (replacing the wake word with noise), it can be seen that both sets exhibit very similar patterns. Replacing does not improve jamming compared to superimposition.

4.3.5 Evaluation of Jamming in an Over-The-Air Setup

We assessed our jamming evaluation in a practical experimental setup, where both wake word and jamming signal are played through Universal Serial Bus (USB) speakers, and the mixed signal is captured over the air by a microphone array and passed to the wake word detection model that runs on a backend laptop. The experimental setup for real-time jamming is shown in Figure 4.8.

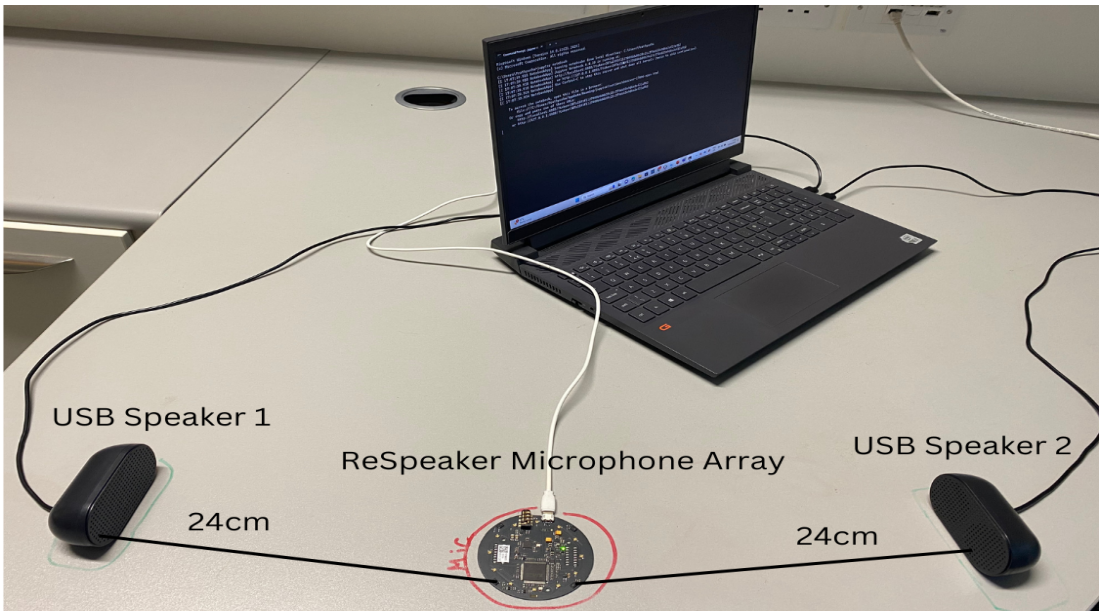
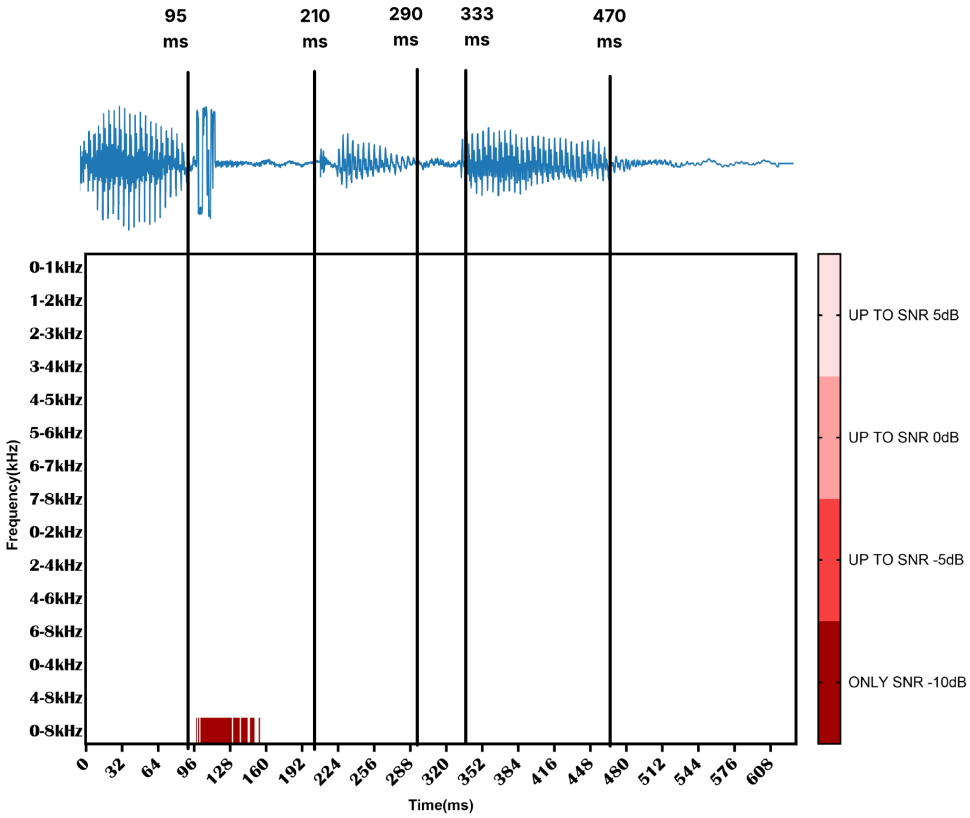
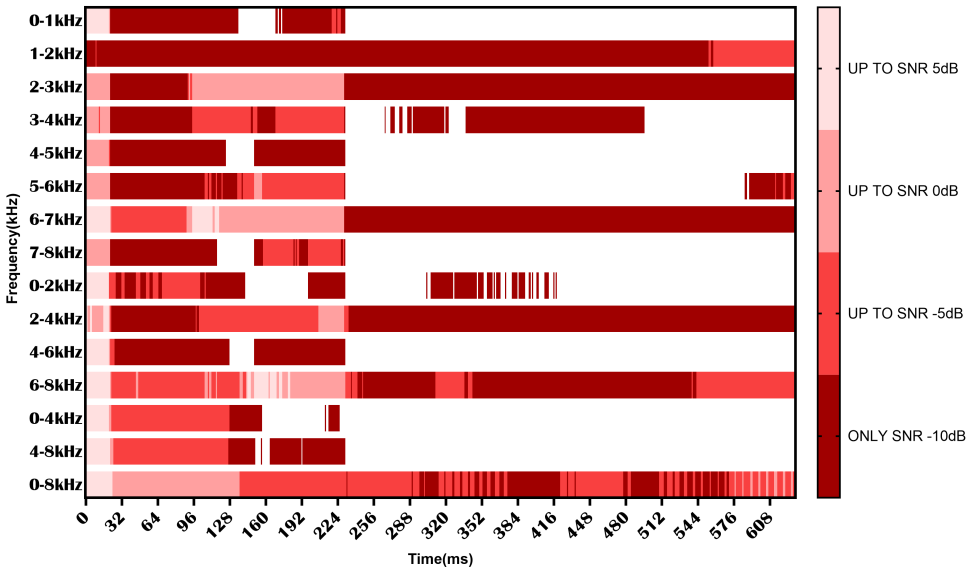


Figure 4.8: Experimental Setup for the evaluation of real-time jamming

The mini-USB speaker on the left-hand side (USB speaker 1) of Figure 4.8 plays the wake word and that on the right-hand side (USB speaker 2) plays the jamming signals. The speakers support a 48 kHz sampling rate for audio playback



(a) 2ms jamming duration



(b) 256ms jamming duration

Figure 4.7: Wake word detection success replacing the audio signal white noise with variable jamming signal energy. Heat maps show results for (a) $\tau = 2$ms and (b) $\tau = 256$ms durations.

and are positioned equidistantly at 24cm from the microphone array that captures the mixed signal. For the purposes of this experimental evaluation, we chose the 24cm distance to represent typical near-field voice interaction scenarios with personal voice assistants, and the symmetric placement ensures controlled and balanced acoustic conditions for systematic testing. While this controlled laboratory setup allows us to validate the jamming effectiveness identified in our offline analysis, we acknowledge that a practical attack scenario would involve different configurations, including varying distances, angles, and environmental conditions. Future work will include evaluation of more realistic attack setups. We used a *ReSpeaker Mic Array v2.0* microphone array equipped with four microphones [MG20]. The experimental setup is placed in a room measuring 3.8m in length and 3.4m in width to simulate realistic acoustic conditions. The ReSpeaker Mic Array v2.0 microphone array captures the audio using all four microphones and combines the audio into a single channel. This mono audio signal is then provided as input to the wake word detection model which provides decisions about the presence of wake words. We did not use timing differences in this work, as the focus is on evaluating the robustness of wake word detection under acoustic interference.

The practical experiment is conducted using the same wake word 'Activate' that has been used in the previous sections for evaluation. The jamming is done using 'Ping noise' with different variations as described in Table 4.1 instead of white noise. This choice reflects short, sudden background sounds that often occur in real-world environments, such as beeps or notifications. Using a different type of interference helps test how well the wake word detector performs under a variety of noise conditions.

Initially, both the wake word and the jamming signal are played simultaneously. The volume level of the USB speaker 1 that played wake word is maintained higher than the USB speaker 2 that plays noise signals, to ensure that the noise is not overpowering. In order to study the impact of noise at each position of the wake word, we introduced a progressive delay of 1 ms in playing the jamming signal during each iteration. This allows us to recreate the superimposition of noise over the full length of the wake word scenario as described in section 4.3.1.

As the evaluation iteration proceeds, the progressive delay increases. When the jamming signal is delayed by 100 ms from the start of the wake word, the jamming signal interrupts wake word detection continuously between 100 and

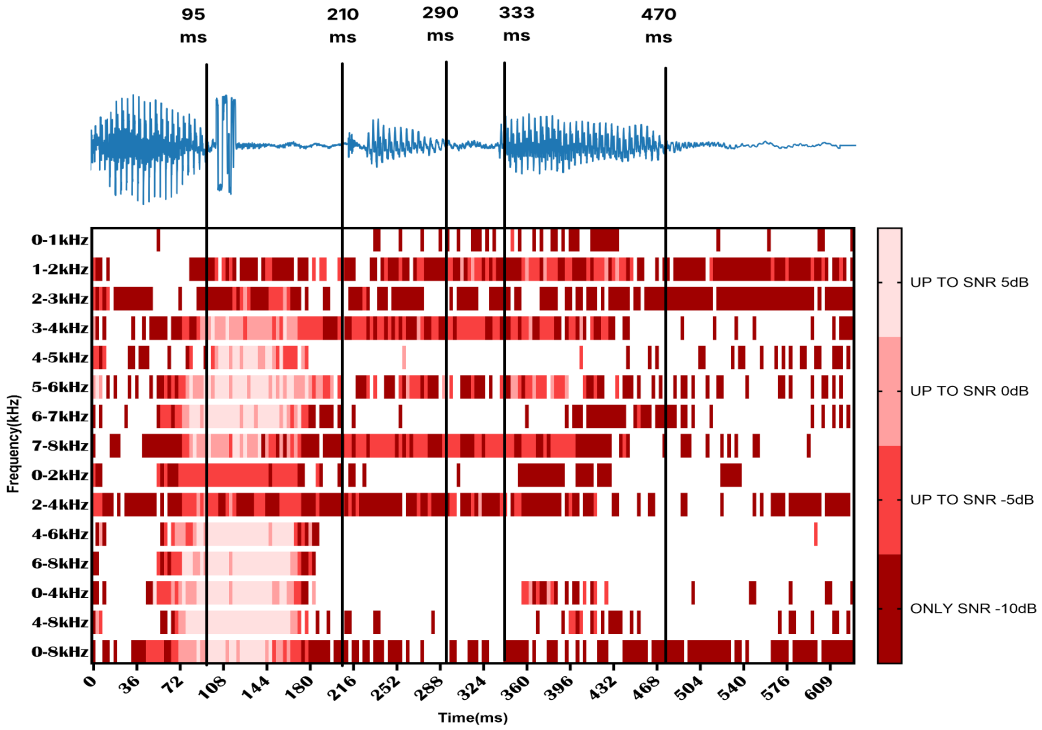


Figure 4.9: Wake Word Detection Success in Real Time Experimental Setup using Ping Noise of 128ms.

150ms after the wake word began, referred to as the sensitive region. The noise began to impact the wake word detection when precisely timed within this sensitive area. To visualize the effect of jamming in the real-time setup, the results of the experiment using ping noise of 128ms long duration are shown in Figure 4.9. It can be observed that the noise impacted the wake word detection successfully in the sensitive region that closely mirrors the phenomenon illustrated in section 4.3.1, where the evaluation is done for jamming using readily available processed data. Although the figure presents results from a single speaker for consistency across experiments, we tested the setup with other speakers as well and observed a similar jamming effect, indicating that the phenomenon is not user-specific.

4.3.6 Zebra Effect

Many areas in the heat maps in the above figures exhibit a “zebra” pattern, meaning that in places successful and unsuccessful wake word detection alternate especially for the lower energy jamming signals. For example, placing the jamming signal at a position t ms is successful, at $t + 1$ms it is unsuccessful and at $t + 2$ms successful again.

This effect is due to the mismatch in the time domain step size used for providing the incoming audio spectrogram to the model and the time domain step size chosen for sliding the jamming window over the wake word. As explained in section 4.2, the wake word detection model used in this study processes audio through spectrogram feature extraction. The spectrogram is fed to the model frame-wise, where each frame is of size 10ms long approximately. In our experimentation, the noise is placed over the wake word incrementally in small time steps of 1ms, which is only 10% of the spectrogram's frame size (10ms). When the noise which is placed over this frame of audio input, moves right by one step in the time axis let's say $t + 1ms$, it may span across two adjacent spectrogram frames, especially if the noise duration is longer than 10 ms. In such cases, the noise may partially mask spectral features such as the frequency values in one frame but leave features in the adjacent frame unmasked, leading to inconsistent detection results. Whereas for the noise of a shorter duration (2ms, 4ms and 8ms), the impact entirely depends on the position of the noise window over the audio as the audio signals are dynamic and vary over time.

The inconsistencies in the model's output shown as *zebra* pattern due to the spreading of noise over adjacent frames of the spectrogram could be avoided by matching the length of the noise window with the length of the spectrogram frame. To verify this assumption, We have also conducted an experiment by changing the step size of placing white noise over the wake word to 10ms and the end results of the wake word detection pattern are different from the previous scenarios, as shown in Figure. 4.10. When the noise is shifted over the wake word with a 10ms step size, the "zebra" pattern vanishes because the step size matches the frame size of the spectrogram (10 ms). This alignment ensures that the noise is applied consistently within each spectrogram frame, avoiding partial masking.

However, using a 10ms step size significantly reduces the granularity of the analysis. Since the step size is equal to the frame size, the noise placement becomes less precise, and the experiment loses its ability to capture fine-grained variations in the impact of the noise on wake word detection. This coarse step size may overlook critical details about how the position and duration of the noise influence the wake word model's behaviour. To acquire a better understanding of the system's sensitivity to noise placement, a smaller step size such as 1ms was chosen for the rest of the experiments.

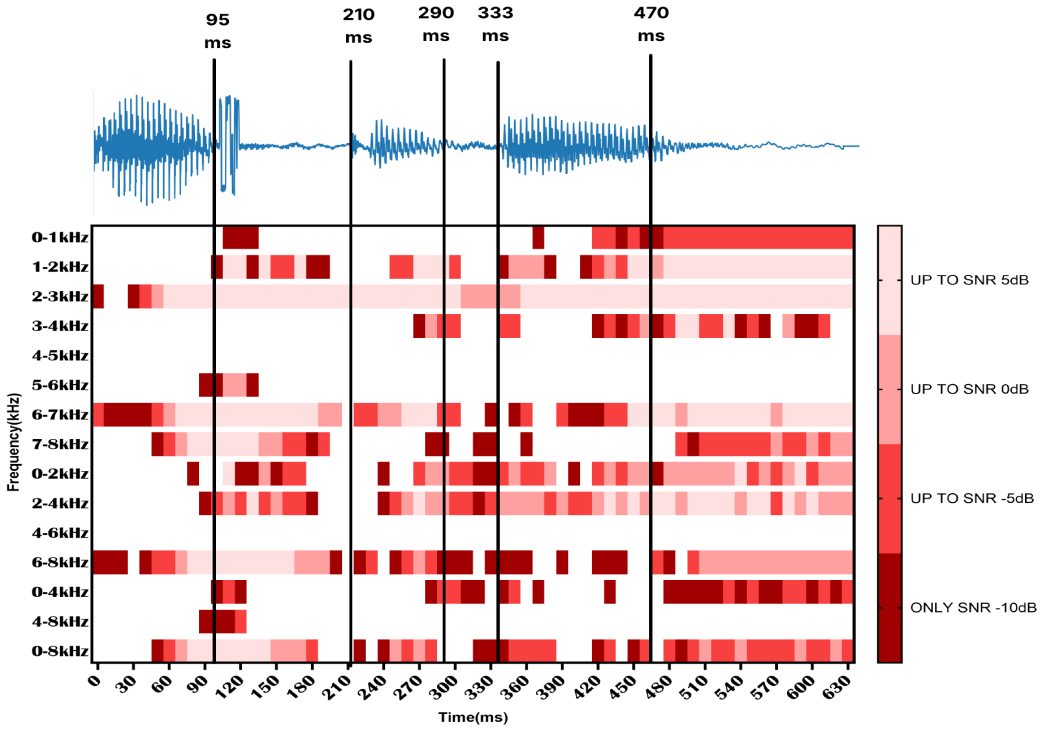


Figure 4.10: Wake Word Detection Pattern under 32ms White Noise Jamming with 10ms Step Size

4.3.7 Noise Types

As described in Section 4.3.3 there is little benefit in crafting a jamming signal that creates an optimal white noise in the jamming region. Our experiments apply white noise through superimposition onto the wake word audio. While we tested both superimposition and replacement, both methods yield similar jamming effectiveness within the target region, as the noise sufficiently corrupts the spectral features regardless of the local amplitude of the wake word. However, that does not mean that other types of signals could be more effective than white noise.

We carried out experiments using a *ping* noise and *keyboard* noise as specified in the table 4.1. White noise is very noticeable as jamming noise and it might be possible to use sounds that are perceived less intrusive as jamming signal. Ping noise contains periodic tones as shown in the figure 4.2b, while keyboard noise consists of irregular typing sounds creating intermittent broadband bursts 4.2c.

The results for jamming with a duration of $\tau = 8ms$ using superimposition with constant energy (similar to Set 1) are shown in Figure 4.12. This duration was chosen as a representative mid-range value to enable fair comparison between

noise types.

The figures show that both noises, ping and keyboard have a noticeable effect on the wake word detection process. Both noise types have a very similar effect and are better than white noise. The ping noise has a slightly higher impact than keyboard, covering a larger area in the heat map. However, we can conclude that for effective jamming the specific type of jamming signal is irrelevant; signal strength, bandwidth and timing determine the success. For example, if using the frequency band from 2-3kHz and a time between 100ms and 150ms any of the three noises is an effective jamming signal.

In the real-time experimental scenario, we conducted jamming using white noise and ping noise and observed that the white noise was not very effective in interrupting the wake word detection as compared to ping noise. It seems to be because of the USB speaker's limitations in playing the white noise. We verified this by comparing the spectral characteristics of white noise played through the USB speaker and recorded versus the original white noise file. Spectral analysis revealed that while the frequency characteristics were preserved, the played-back signal exhibited significantly reduced amplitude (approximately 80-90% attenuation), which can be observed in the figure 4.11.

4.3.8 Jamming Energy

An attacker aiming to maximize disruption while minimizing detection would strategically allocate their energy budget, focusing on the sensitive areas of the wake word. This approach would allow the attacker to use less energy for the jamming signal, thus staying less noticeable. Human auditory perception plays a crucial role in determining detection risk. Lower amplitude jamming signals are harder for humans to perceive, particularly when they overlap temporally with the wake word itself.

To gain a deeper understanding of the energy required for a jamming signal at different positions within the wake word, we conducted an additional experiment. In this experiment, we used a jamming signal with a duration of $\tau = 2ms$ and a bandwidth of $B = 8kHz$. This jamming signal was superimposed on the wake word at various time points t, and the wake word detection success was evaluated. The initial power was set at an SNR of 0 dB, meaning the wake word and jamming signal were of equal strength. The energy of the jamming signal was then incrementally increased until jamming was successful at a spe-

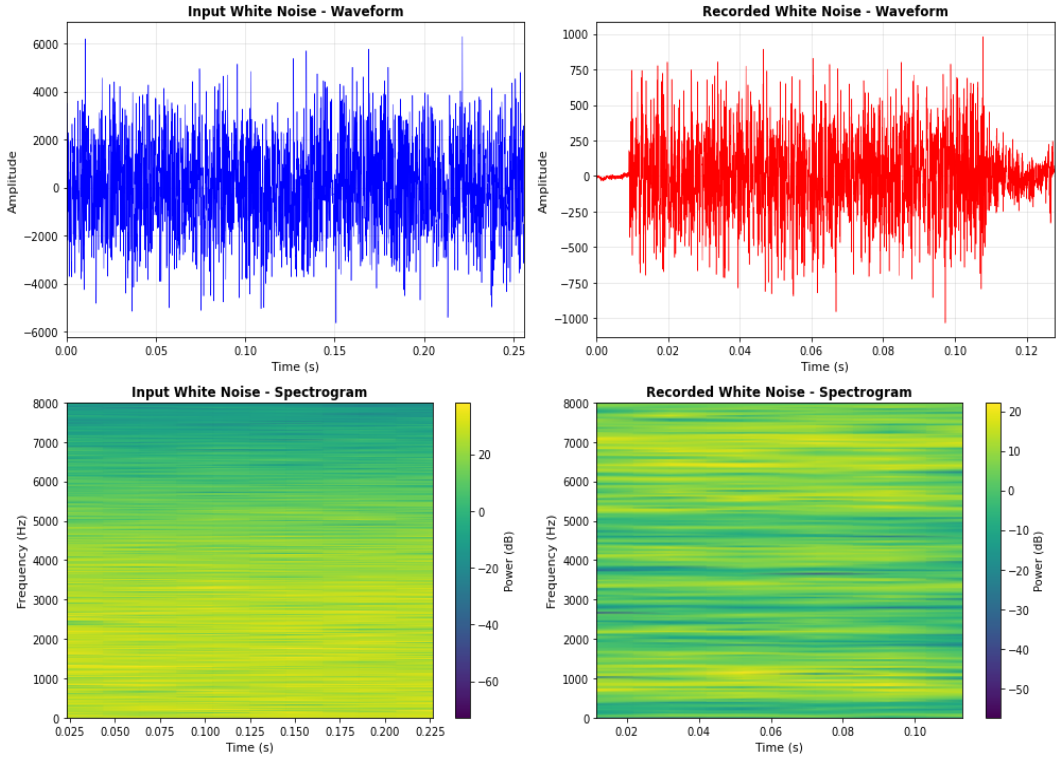
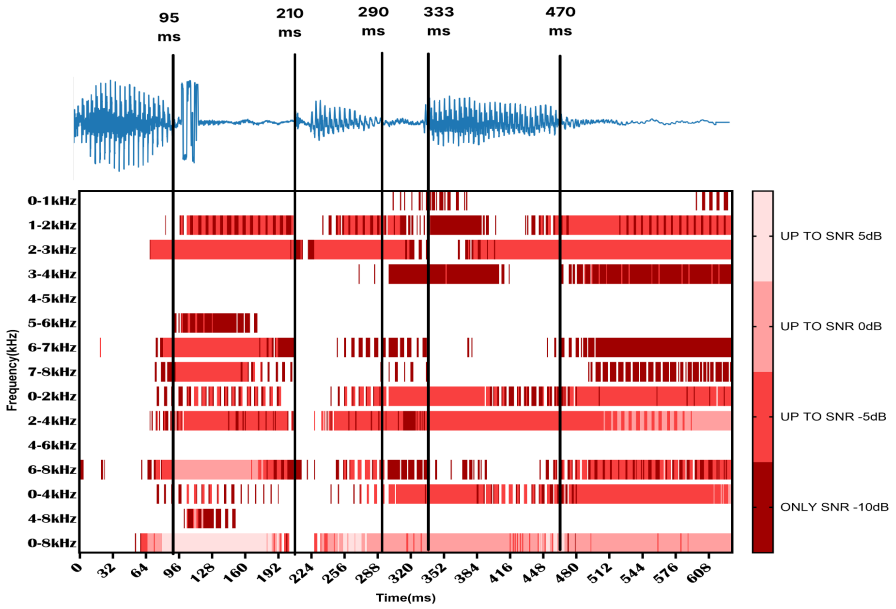


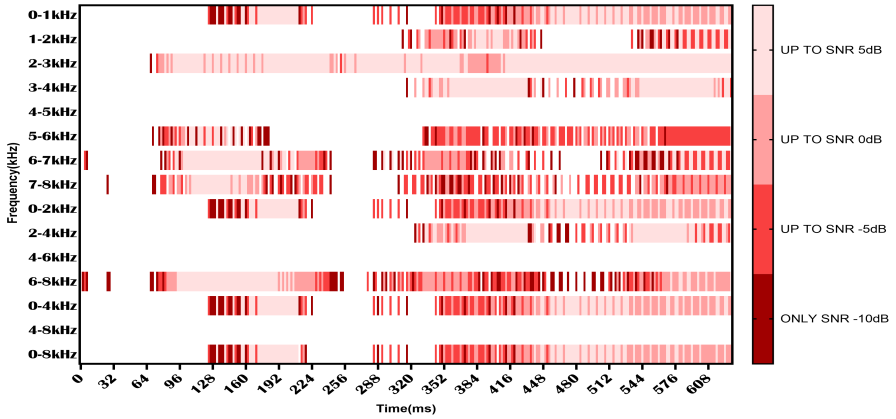
Figure 4.11: Comparison of input white noise (left) versus recorded white noise after USB speaker playback (right). Top: Waveforms showing significant amplitude reduction. Bottom: Spectrograms demonstrating preserved frequency characteristics (0-8 kHz) but reduced power levels.

cific wake word position or the jamming energy reached a limit of 500 times the energy of the wake word at that region. To maintain practicality, we did not allow the energy to increase infinitely due to real-world constraints, such as output limitations of typical consumer USB speakers which are typically around 85-90 dB Sound Pressure Level (SPL) and to avoid speaker nonlinear distortion at high output levels and the threshold where sound becomes uncomfortably loud or potentially harmful to human hearing.

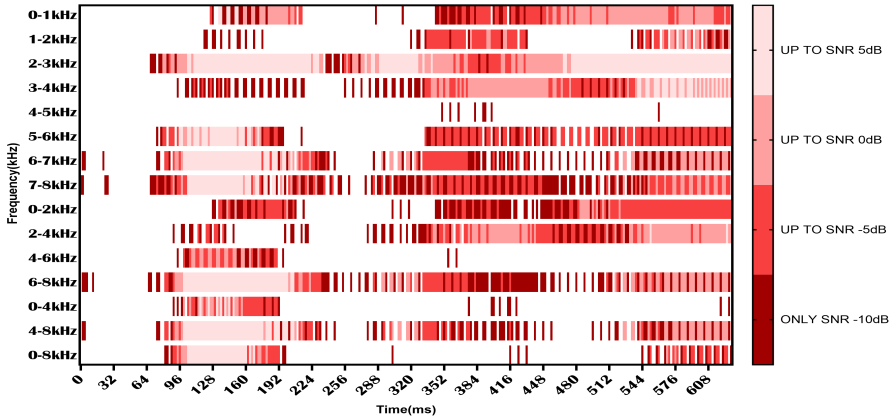
The experimental setup is as illustrated in 4.3.5, with two USB speakers and a microphone array. The two USB speakers played the wake word and the jamming signal simultaneously, with one of the speakers emitting the jamming signal with a gradual delay of 1 ms, allowing the jamming signal to move across the wake word. This configuration was designed to replicate the experimental conditions of superimposing a jamming signal over the wake word, as detailed in 4.3.2.



(a) 8ms jamming duration, white noise



(b) 8ms jamming duration, key noise



(c) 8ms jamming duration, ping noise

Figure 4.12: Wake Word Detection Success while Superimposing Different Noise Types Using a Jamming Duration of 8ms with constant energy.

Refined Heatmap Approach

We employed a refined heat map approach to capture the results. The previous heatmaps used colour coding to represent the strength (S dB as shown in 4.1) of the jamming signal across various frequency bands. The colour in these heatmaps now represents the minimum energy required for a jamming signal to successfully disrupt wake word detection at a specific point in time(t ms as shown in Table 4.1) during the wake word detection. These new heatmap representations also clearly highlight the sensitive regions within the wake word. By focusing on these regions, an attacker can achieve successful jamming with a jamming signal of minimal energy. We continue to use this heatmap approach throughout the remaining experiments, as it offers more insights compared to previous representations.

4.3.9 Result of Required Jamming Energy for Succesful Jamming

Fig. 4.13 shows the result of jamming the LSTM-WD model using white noise of varying duration, placement, and power in the form of a heat map. The X-axis represents time and covers the duration of the wake word (in ms) and the Y-axis represents the different jamming signal duration (in ms). 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. It can be seen in Fig. 4.13 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 (jamming signal power equals wake word power) 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 effectiveness varies depending on the duration, strength, and placement of the signal. In some regions, a slight adjustment in signal timing determines jamming success or failure. However, there are also regions where

precise signal positioning and timing are less critical for successful jamming. Notably, all evaluated jamming signal durations are effective in the area from 85ms to 150ms, which is the identified *sensitive regions* in previous experiments. The sensitive areas also seem to be unvoiced areas of the wake word, with low energy. From this experiment, one could suggest that it could be beneficial for a jammer to place the jamming signal in the unvoiced region of a wake word. It is possible that an unvoiced region of a long duration can help an attacker to place a jamming signal as strict timing is then not required.Analysis of the wake word 'Activate' audio waveform and spectrogram reveals that this sensitive region (85-150ms) corresponds to an unvoiced segment with low energy, specifically the /k/ and /t/ phonemes.However, this entirely depends on the wake word as some wake words can have sensitive areas even in the voiced regions which is explored in detail in chapter 7.

4.3.10 Wake Word Jamming on Different Wake Words and Algorithms

We used the similar wake word jamming approach on different wake words and the result of jamming is similar to our findings on different wake words and wake word detection algorithms. Obviously, the positions of sensitive areas within the wake word are different as they are specific to a word. The other wake words chosen for our evaluation are 'Alexa' and 'Marvin'. We also evaluated the effect of jamming with a different wake word detection model by Oleg et al called as *MHAtt-RNN* [RKS+20], and the model is effected by jamming in a similar way as *LSTM-WD* model.

4.3.10.1 Jamming using "Alexa" as a Wake Word

To further generalize our findings on jamming, we retrained the LSTM_WD model specifically for the wake word, "Alexa" For the dataset preparation, we used an "Alexa" dataset sourced from [Anh18] containing 369 samples of people saying "Alexa". We used these samples as positive examples (wake words). To gather negative examples (non-wake words), we used spoken words dataset sourced from [MCB+21] containing words other than "Alexa". Following the same methodology as previous experiments, we created 10-second audio samples containing silence and background noise, with equal numbers of wake word and non-wake word instances randomly placed within each sample. Wake word occurrences were labeled as 1, and non-wake word occurrences as 0.

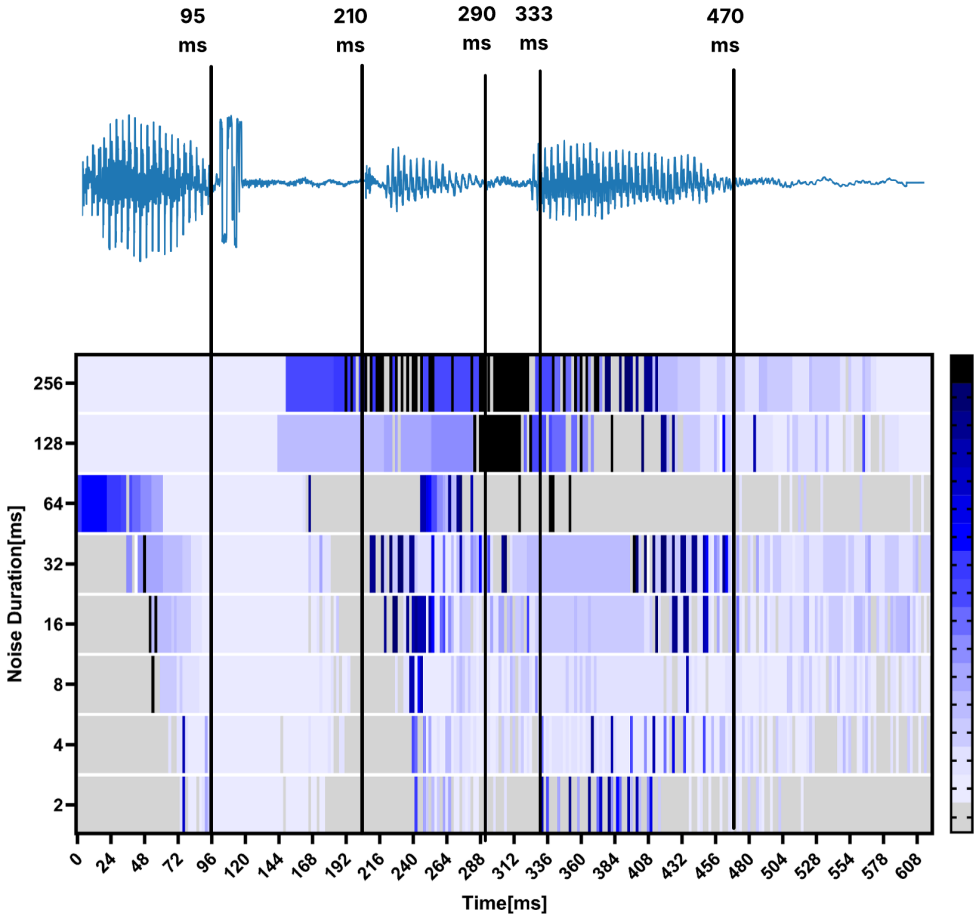


Figure 4.13: Wake word detection performance when superimposing white noise over the wake word 'Activate'. The colour intensity represents the noise energy level required for successful jamming. Grey represents areas where jamming was not possible. Waveform of Activate superimposed with a Jamming Signal on the Top.

The complete dataset consisted of 70,000 such 10-second audio samples, split into 80% (56,000 samples) for training and 20% (14,000 samples) for testing. We trained this new "Alexa" detection model for 100 epochs with a batch size of 50, then evaluated it on previously unseen test samples to verify detection accuracy before conducting jamming experiments.

The jamming was successful on the new wake word, "Alexa," as shown in Figure 4.14. While the sensitive areas differed in position compared to previous wake words, they were still present.

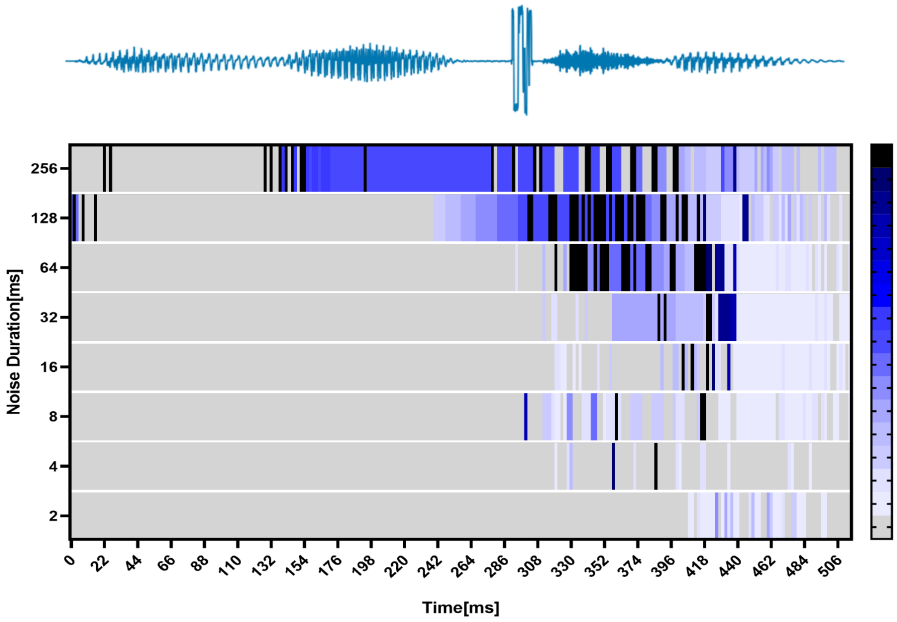


Figure 4.14: Jamming using ping noise on the wake word "Alexa". WaveForm of Alexa Superimposed with a Jamming signal on the Top

4.3.10.2 Jamming using "Marvin" as a Wake Word

We retrained the LSTM_WD wake word model with another wake word called "Marvin" using the same method as explained in previous section 4.3.10. The dataset used here is the speech commands, from which we chose "Marvin" as our wake word to train the model. The dataset contains 1,746 utterances of the word "Marvin". The negative samples for non wake words are also sourced from the same speech commands dataset and we chose the rest of the words other than "Marvin" as non wake words. We labelled the data such that wake words are marked as 1s and rest of the words are marked as 0s. We created a total of 1746 data samples by superimposing wake words and non wake words onto a 10sec audio sample of background noise. The data samples are split into 80% for training and 20% for testing. We trained the model for 200 epochs with a batch size of 50. The increased number of epochs compared to the "Alexa" model was necessary due to the substantially smaller dataset size, requiring more training iterations to achieve comparable model performance. This should be viewed as a practical training adjustment rather than a guaranteed way to offset limited data. After training, we tested the model on previously unseen audio samples.

We executed jamming with ping noise on the wake word "Marvin" of 420ms long and the results are shown in the figure 4.15. Because the wake word

and jamming signal amplitudes are closer, the jamming signal is not visually separable in the time domain plot. The jamming was also successful on the new wake word, "Marvin" with sensitive areas that are very susceptible to jamming.

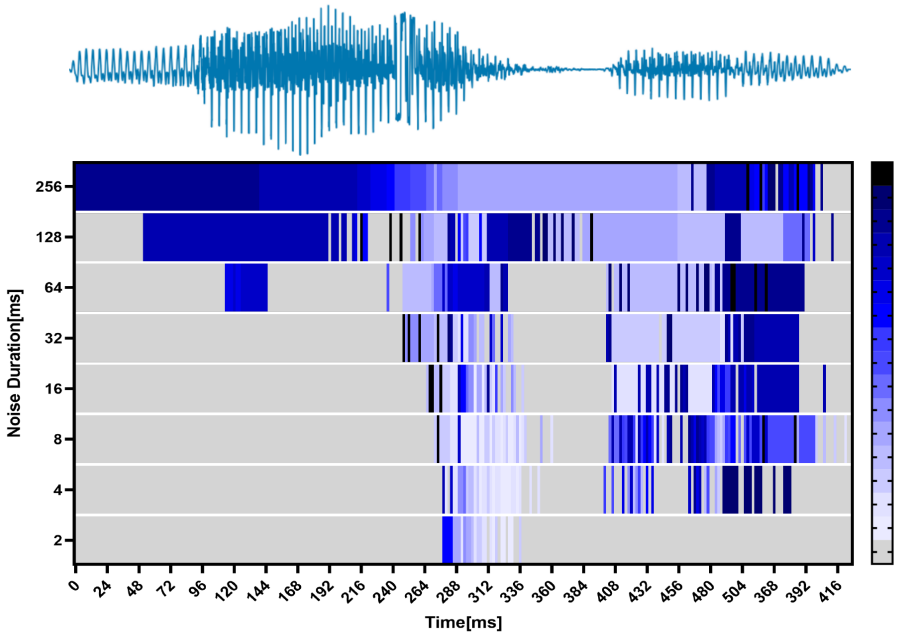


Figure 4.15: Jamming using ping noise on the wake word "Marvin". WaveForm of Marvin Superimposed with a Jamming signal on the Top

4.3.10.3 MHAtt-RNN Wake Word Detection Model

We employed the *MHAtt-RNN* algorithm introduced by Oleg et al.[RKS+20], which was originally developed for keyword spotting. Keyword spotting is the task of detecting specific words or phrases in continuous audio streams, commonly used in voice activated systems to identify commands or wake words [CPH14, ZSLC17]. This model architecture consists of a speech feature extractor and a neural network-based classifier. The architecture works as follows, the input audio is first divided into overlapping frames of 40ms duration with 20ms overlap. Each frame is processed through a CNN to extract MFCC features. These MFCC features are then fed into the multi-head attention mechanism, which consists of four parallel attention heads. Each attention head independently learns to focus on different temporal patterns in the audio features, for example, one head focuses on phoneme boundaries while another focuses on energy transitions. The outputs from all four attention heads are concatenated and passed through GRU layers, which capture the temporal dependencies in the audio sequence. Finally, a dense layer with softmax activation classifies

whether the input contains the target keyword. This multi-head attention approach allows the model to simultaneously attend to multiple aspects of the audio signal, improving its ability to distinguish the keyword from background noise and similar-sounding words [RJ22]. The architecture is similar to the standard keyword spotting model proposed by Sercan et al. [AKC+17].

We adopted this model architecture and retrained it to predict the presence of wake words in incoming audio signals, focusing on the wake word “Marvin”. The dataset used for training is divided into 80% training data, consisting of 1,293 samples, and 20% test data with 320 samples, each having an average duration of approximately 700 milliseconds. This dataset used, is sourced from the Speech Commands dataset referenced in [War18a].

After training the model to recognize the wake word “Marvin”, we implemented jamming using the over-the-air setup. The results of jamming on the wake word “Marvin” with this newly trained MHAtt_RNN model were found to be very similar to those for the word “Marvin” using the LSTM_WD model. This demonstrates that the presence of sensitive regions is consistent across different model architectures. However, when comparing “Marvin” to "Activate" (using LSTM_WD for both) from figures 4.13 and 4.15, the specific temporal positions of sensitive regions differed, reflecting the distinct phonetic structures of these two wake words.

4.3.10.4 Experiment on Echo Device

We extended our testing to a commercial device, the Amazon Echo, to explore how well it can resist wake word jamming. Since the Echo’s algorithm is proprietary and internal processing details are not publicly available, we could not automate the testing process and instead recorded each result manually. Our experiments showed that the Echo is indeed vulnerable to jamming. Specifically, when noise was added in the first 40–50 milliseconds of the wake word “Alexa,” the device failed to respond. This suggests that the Echo also has sensitive regions, but it required longer or more intense jamming signals to be disrupted compared to our test models.

We believe this difference is due to the Echo’s features, including built-in noise cancellation, acoustic echo cancellation which were not present in our experimental setup. Overall, the Echo device displayed a similar pattern of sensitivity to jamming as our tested algorithms. Since we lacked access to the Echo’s internal detection confidence scores and could only observe its response whether

detected or not detected from the device's activations, we could not generate the detailed energy-versus-position heatmaps produced for our other models.

4.3.11 Findings

To jam the wake word 'Activate' of the LSTM-WD algorithm, a jamming signal is to be constructed which is described by signal type, strength, timing, duration, frequency and bandwidth.

Signal type: Our experiments have shown that the signal type used to construct the jamming signal is not particularly relevant. White noise, ping and keyboard noises are all useful signals providing similar success rates. Also, a signal designed to produce perfect white noise in the jamming region is not better than other signals. Thus, a signal type should be used that is least noticeable by users such as signals that blend into the ambient environment to conceal the jammer *Signal strength:* As intuition suggests, a strong signal is usually better for disruption of the wake word recognition than a weak one. However, the signal strength is limited by the available loudspeaker to emit the jamming signal and a strong signal is again noticeable by users.

Signal timing: The timing of the jamming signal in relation to the wake word is key to successful jamming. As we have shown, the jamming signal should be placed in the unvoiced region of a wake word as the wake word has in these areas the smallest energy and the jamming signal can therefore be of low energy as well. For the used wake word 'Activate' the regions to consider are from $100ms$ to $150ms$ and from $470ms$ on. As the jammer may first have to recognize the wake word itself before placing a jamming signal the region from $470ms$ might be best as it gives sufficient time to synchronize jamming with the wake word. However, even the area from $100ms$ to $150ms$ should be feasible for a practical system. To relax timing constraints of the jamming device the jamming signal should be placed at a time where jitter has little impact. As we have shown, the areas from $100ms$ to $150ms$ and from $470ms$ on fulfill this requirement. Variations of up to 50ms in placing the signal have no impact on jamming success depending on the wake word and the algorithm used. However, it also should be noted that placing a disturbance in the unvoiced speech area would be more noticeable by users than placing it in the voiced region of the wake word. In a practical attack scenario, an attacker must be aware of several constraints: the need for precise timing synchronization, hardware capable of real-time wake word detection and playback, and the need for prior

knowledge of the target wake word's phonetic structure and the wake word detection algorithm.

Signal duration: A long jamming signal duration is more useful than a short duration. However, if the energy budget for a jamming device is limited (e.g. a battery powered jammer) it is not useful to use the longest possible jamming duration as then the signal energy becomes too weak. Also, a jamming signal with long duration might be more noticeable by users than a short pulse. A short signal of only $2ms$ is sufficient if timed correctly.

Signal frequency and bandwidth: The jamming signal does not have to cover the entire frequency band. Jamming is most effective in the area between 1-2kHz, 2-3kHz and 6-7kHz. A bandwidth of 1kHz is sufficient to achieve a disruption. As speech components are not present in the 6-7kHz band a PVA could filter this frequency range to prevent jamming. Thus, the area of 1-3kHz is most useful for jamming but is also the most noticeable by users.

It should be noted that the zebra effect should also be taken into account when deciding on jamming signal placement. An attacker should not place a jamming signal in an area with a strong zebra pattern as slight timing variations will decide on the success of the jamming process. Placing the signal in an area that shows a long duration of jamming success is preferred.

Real Time jamming Experiment Findings: In real-time, the noise signal can indeed disrupt a wake word detection model and if the interruption is in the sensitive regions, it can effectively interfere with wake word detection. However, it is noteworthy that the noise should be of a longer duration and contain higher frequencies such as ping noise are much more effective in doing the interruption.

Chapter 5

Robust Wake Word Detection

Wake word detection algorithms in PVAs are not designed to handle acoustic DoS attacks as explained in Chapter 4. Commercial PVAs employ sophisticated noise cancellation methods. These methods can deal with background noise to some extent, however, they are insufficient to suppress deliberate jamming efforts. To address this, we investigate the mitigation methods to make the wake word detection model robust against such crafted deliberate jamming attacks. Specifically, we propose using adversarial training, a technique that involves training the wake word model on samples of wake words along with jamming signals. This approach allows the model to get familiarized with the jamming signals during the training process to learn and adapt to their presence thereby effectively improving its resilience.

The process of adversarial training involves superimposing jamming signals over the wake word only in the sensitive areas identified during the jamming experiment. By training with these samples, the model gains the ability to recognize wake words when faced with similar forms of interference during real-world operation. Another crucial element of our adversarial training process is the careful selection of jamming signal type used for training. We observe that using a jamming signal that can sound or cover a diverse variety of different types of noises such as ping noises, household noises, etc., can greatly improve its ability to generalize. This enables the wake word detection model to be resilient not only to known jamming signals but also to new and previously unknown jamming signals it may encounter during the wake word detection process.

We optimize the adversarial training effort by identifying areas of the wake

word that are highly susceptible to acoustic interference. By identifying these sensitive areas, we generate training samples that make the model more resilient to jamming. We put this approach to the test with different wake words, and two types of wake word detection algorithms, and we found that the adversarial training consistently strengthened the model's ability to detect the wake word accurately despite the presence of jamming signals. This strategy improves the model's ability to recognize wake words despite interference and enables it to detect wake words even when exposed to unfamiliar jamming signals not encountered during training. These findings suggest that our method could make wake word detection more dependable even during a jamming attack.

5.1 Related Work on Robust Wake Word Detection

Existing research focused on making a wake word detection model robust enough to detect wake words in noisy environments. Work by Lim et al [LKK20], introduces a technique called Cross-Informed Domain Adversarial Training (CiDAT) to improve the reliability of wake word detection systems in noisy environments. Detecting wake words becomes challenging when background noise or interference is present. The CiDAT method addresses this issue by training the model to identify wake word features regardless of the type of background noise present. Traditionally, wake word detection systems struggle because the features they use to identify specific words are easily disrupted by noise. While existing techniques, such as Domain Adversarial Training (DAT), aim to reduce the impact of noise by creating "domain-invariant" features, features that are consistent whether or not noise is present, DAT alone is often insufficient for handling a wide variety of complex background sounds. To overcome this, CiDAT builds on DAT but introduces an innovative approach where two parts of the model, called the "word encoder" and the "domain encoder," interact closely. The word encoder is responsible for identifying features related to the wake word itself, while the domain encoder focuses on distinguishing between different noise types. By introducing "cross-informed" connections between these two encoders, the model is able to improve its focus on wake word features while reducing the impact of background noise.

This interaction between the encoders happens through pathways that allow the word encoder to use information from the domain encoder, and vice versa. For example, as the domain encoder identifies elements that are specific to the

noise, it shares this information with the word encoder. This exchange helps the word encoder isolate the wake word features from the noise, enabling the model to "ignore" background sounds more effectively. This bidirectional sharing of information strengthens the wake word detection capability, making the model better equipped to handle noisy environments. During training, CiDAT uses both clean and noisy audio samples to help the model learn in a range of environments. This training includes the addition of simulated noisy conditions, allowing the model to become familiar with common noise types it might encounter once deployed. The model optimizes its detection by training on "quadruplet" groups of data that include both similar and distinct word and noise combinations. This diverse training set reinforces the system's ability to distinguish the wake word even when unexpected sounds are present. The approach demonstrated 83-89% recall rates across various unseen background noise types (restaurant, street, airport, train) at SNR levels of 10-20dB, representing 18-28% relative improvement over conventional domain adversarial training methods. However, this approach focuses on robustness to natural background noise rather than targeted defenses against deliberate jamming attacks, which is the focus of our work.

Work by [GMS+20] Yixin et al explores a method for robust wake word detection in PVAs, particularly targeting the challenging scenario of far-field settings, where speech signals are often distorted by various types of environmental noise. To enhance wake word detection robustness without relying on large amounts of transcribed, real-world data, the paper presents a combined approach using multi-condition training and semi-supervised learning. In multi-condition training, the researchers start with a small, clean set of close talk microphone recordings of the wake word. To mimic far field environments, they create diverse training examples by adding reverberation, which simulates room echo effects, and overlaying various types of noise for example, household sounds or media playback. This approach effectively transforms limited clean audio into an extensive, noise diverse dataset, allowing the model to adapt to different conditions without requiring a large amount of real world noisy data. In this way, the model learns to handle audio variations typical in household settings, making it more adaptable to new environments and noise levels that it encounters in practice.

To further enhance robustness, the authors apply semi-supervised learning. This involves using additional wake word examples from large, untranscribed audio corpora. An ASR model generates transcripts for these untranscribed

recordings, identifying instances where the wake word or similar-sounding words occur. These samples are then used to expand the training set, including both confirmed wake word examples and "confusable" words: phrases that sound similar to the wake word but are not intended to trigger the PVA. By training with these confusable examples, the model develops the ability to avoid false positives, becoming both sensitive to the wake word and resistant to misidentification.

Though this research covers and focuses on wake word detection amid household or background noises and also develops an ability to avoid false positives, it does not address targeted deliberate jamming attacks.

Work by [YZ22] Queichen et al, enhances the wake word detection model by using Res2Net's multi-scale feature extraction that processes mel-spectrogram features at multiple temporal scales and dual-stage classification strategy. Res2Net, which serves as the model's backbone, is structured to capture details at multiple scales within each layer. This multi-scale approach allows the model to adapt to variations in wake word length and pronunciation by learning fine and broad patterns simultaneously, making it more resilient to inconsistencies introduced by different speakers or environments. To further improve robustness, the model processes each audio signal into two types of spectrograms: one with resizing and the other with slicing. Resizing adjusts each spectrogram to a uniform length, enabling the model to detect global wake word patterns. Slicing, on the other hand, divides the spectrogram into smaller sections, allowing the model to focus on local patterns within parts of the wake word. This dual approach strengthens the model's ability to recognize wake words amid background noise or overlapping speech by analyzing both the overall and detailed structure of the audio.

Additionally, the model employs SpecAugment: spectrogram masking as a data augmentation technique during training to simulate noisy or interrupted conditions. SpecAugment involves randomly selecting time and frequency segments of the spectrogram and masking them, forcing the model to learn wake word detection under partial or obscured audio. Such training makes the model less reliant on clean signals, preparing it to function reliably even in challenging real-world conditions. Finally, the model uses two classifiers, each trained on one type of spectrogram. These classifiers then combine their results to confirm a wake word, improving detection accuracy by incorporating both global and local perspectives. This architecture, paired with strategic data augmentation,

makes the model robust against common real-world noise, ensuring efficient and reliable wake word detection.

Different from our work, robust wake word detection amid jamming interference is not explored and addressed here.

Work by [SYO+18] generates adversarial perturbations using Fast Gradient Sign Method [GSS14], a technique that computes small mathematical changes to the input spectrogram based on the model's loss gradients.

This approach is evaluated using two speech recognition benchmarks: the Aurora-4 and CHiME-4 corpora. Adversarial data augmentation leads to significant improvements in word error rate, particularly under challenging conditions with high noise levels or channel distortions. The method is further enhanced by combining it with teacher-student training, where the teacher model is trained on clean data and the student model learns from noisy data and the teacher's guidance, resulting in even greater word error rate reductions. While this adversarial training approach improves general robustness, it differs from our work, which uses realistic acoustic jamming signals specifically designed for acoustic denial-of-service attacks on wake word detection.

5.2 Building a Jamming-Resistant Wake Word Detection Model

The experiments conducted on wake word jamming reveal that most wake words contain specific sensitive regions that are highly susceptible to interference from jamming signals. These signals can effectively disrupt the detection process if they target these vulnerable areas. For example, in the case of the wake word 'Activate', the sensitive regions were found to fall roughly between 100ms and 150ms from the start of the word, as identified in Section 6.3. Given that these sensitive areas exist, training the wake word model with samples in which jamming signals are superimposed precisely on these regions can enhance the model's resilience. By introducing wake word samples that contain both short and long, low- and high-energy jamming signals focused only on these sensitive areas, the model becomes familiarized with various jamming scenarios. This targeted exposure helps the model learn to recognize wake words even when these specific regions are subjected to interference. For this experimentation, we used the same wake word detection algorithm by Supriya

et al. [SDSV20] trained on the wake word 'Activate' as it was used for jamming experiments. We then performed adversarial training of the model by training it with sample of wake word 'Activate' superimposed with jamming signals to prevent wake word jamming. We observed that the model became robust in wake word detection and was able to detect a data sample as wake word even with the presence of jamming signals in it. To ensure that this approach worked beyond a single wake word, we tested it with additional wake words like 'Alexa' and 'Marvin', as well as with the MHAtt-RNN model introduced in Chapter 4. The results showed that adversarial training consistently strengthened each model's ability to detect wake words despite jamming. This suggests that this training approach can be widely applied across different wake words and detection models, making wake word recognition systems more robust against interference. Its possible that adversarial training could increase false positives, we address false positive detection and mitigation strategies in detail in Chapter 6. To reduce the amount of training required, we focused only on the sensitive regions of the wake word rather than superimposing jamming signal at each position covering the entire duration of the wake word. By focusing only on these sensitive regions, we effectively minimized the training effort while still allowing the model to learn to recognize wake words under jamming conditions. This selective approach ensures that the model gains resilience against interference where it matters most, without the need for exhaustive training on every part of the wake word. The identification of these sensitive regions is addressed in detail in Chapter 7. The only additional cost of adversarial training is the one-time generation of training samples with superimposed jamming signals in sensitive regions. Adversarial training does not increase the model size or computational requirements. The model architecture remains identical to the baseline.

5.2.1 Sensitive Regions and Jamming Coverage

An attacker trying to disrupt the wake word detection would likely focus only on the sensitive regions of the wake word. By targeting only these sensitive regions rather than the entire wake word, the attacker could achieve maximum disruption using minimal energy to produce a jamming signal that has maximum impact on the wake word detection. To identify these regions within the wake word, we propose a new metric called *Jamming Coverage* C. This metric helps us identify and measure these "sensitive regions".

A *Sensitive region* R is defined as a specific time window within the wake word marked by a start time t_S and an end time t_E with duration $T = t_E - t_S$. Within this time window, the jamming attack is successful for $\tau\%$ of all test cases which include variations in placement, duration and power levels of jamming signal. For a region to be classified as sensitive, it must exceed a minimum duration threshold T_{min} ($T > T_{min}$) to be considered a valid region. For example, Fig. 4.13 exhibits one sensitive region R_1 with $T = 164ms - 88ms = 76ms$ for a threshold of $\tau = 99\%$. In our study, we set $\tau = 99\%$ and $T_{min} = 20ms$ to consistently define sensitive regions by avoiding strictness, because this threshold aggregates success across multiple jamming configurations. In practice, an experiment may reveal multiple sensitive regions or none, depending on the characteristics of the wake word and the conditions of the test. Each instance of a spoken wake word, such as "Activate," may show different sensitive regions based on variations between speakers, including speaking rate, accent, pronunciation, and voice characteristics. Sensitive regions reflect vulnerabilities in how this specific trained detector processes acoustic input rather than universal properties of the wake word itself. The sensitive region is important because it shows an area the wake word is particularly vulnerable to interference. 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. Essentially, jamming within a sensitive region requires less precise effort to successfully interfere with wake word detection.

Jamming coverage C is defined as the area of a heat map where jamming is successful, relative to the total heat map area, with respect to the duration of the wake word. It is the percentage of blue regions (indicating successful jamming) compared to gray regions (indicating no jamming effect) on the heat maps.

For example, Fig. 4.13 displays a heat map for the wake word 'Activate', where jamming coverage is $C = 69.2\%$. This high percentage means that a significant portion of the wake word duration is vulnerable to jamming interference, as shown by the extensive blue regions. A robust wake word detection system, which is resistant to jamming, should ideally exhibit a jamming coverage of zero percent. The value of C is signal dependent, affected by the acoustic properties of the specific wake word, its phonetic structure, speaker variations, and the model's internal representation. By aggregating C across multiple speakers and utterances, we can generalize the metric and obtain a more speaker independent estimate of jamming vulnerability.

5.3 Jamming Signals used for Adversarial Training

As described in section 4.2.1, we initially consider "White Noise" as the jamming signal because of its flat spectrum and can impact all frequency regions of the wake word equally. This allowed us to observe the general impact of jamming. For adversarial training, we superimposed white noise of varying duration specifically over the sensitive areas of the wake word to help the model learn to detect the wake word despite interference. After training, the model successfully resisted white noise jamming, reducing the jamming coverage C value reduced from 69.2% to 6.8%. However, when the model was tested with different type of noise such as ping noise as the jamming signal, it failed to recognize the wake word and was vulnerable to interference. To investigate this further, we used another type of jamming signal that can cover a variety of noises such as "babel noise". The babel noise is generated manually in which we included background chatter, keyboard typing and cutlery clinking. We used a web-based source [Pig20] for the generation of this babel noise. The noise is then segmented into desired durations and used as a jamming signal for adversarial training purposes.

For the experimentation, we start by choosing the jamming signal type first and then limit it to the frequency band from 0kHz to 8kHz with durations varying between 2ms and 256ms. We also varied the jamming signal's power, starting with an initial SNR of 0dB (where the wake word and jamming signal have equal strength as defined in chapter 4). The jamming signal energy is gradually 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 as described in Section 4.2.3).

While earlier evaluations, explained in Chapter 4, demonstrated that different types of jamming signals, such as white noise, ping noise, or keyboard typing noise can all effectively disrupt wake word detection. In the training context, the choice of jamming signal becomes crucial. Although many signals can cause interference during evaluation, diverse and representative jamming signals are essential during training to generalize robustness across different interference types.

We propose a simple and efficient method for improving the jamming resistance of the wake word detection algorithm by using adversarial training. The idea behind adversarial training is to train the algorithm with jamming signals that

Table 5.1: Jamming Coverage C (%) Before and After Adversarial Training LSTM-WD with Different Noises and Wake Words. Jamming coverage represents the percentage of wake word parameter space where jamming successfully prevents detection.

Wake Words	**Adversarial Training**	**Jamming Signal**	
		White Noise C (%)	**Ping Noise** C (%)
Activate	None	69.2	72.6
Activate	White Noise	6.8	66.0
Activate	Babel Noise	7.0	9.8
Alexa	None	30.6	51.3
Alexa	White Noise	3.8	27.8
Alexa	Babel Noise	4.1	10.0
Marvin	None	43.5	50.8
Marvin	White Noise	14.3	14.3
Marvin	Babel Noise	5.2	7.4

might be encountered.

5.3.1 Adversarial Training Data

To create adversarial examples, we combined wake words with jamming signals. However, given the vast number of potential jamming signals each with varying positions, durations, and power levels, generating all possible examples would be time consuming and costly. To reduce the number of adversarial examples, we only generate examples for sensitive regions. Through our analysis, we observe that a sensitive region consistently observed between $t_{S'} = 80ms$ and $t_{E'} = 200ms$ after the start of the wake word for different speakers and we chose this area R' to focus our training effort (where $t_{S'}$, $t_{E'}$ denote the start and end times of this particular region R'). Considering the different positions, duration, and power of noise signals in R' we consider 24,000 jamming signals (rather than the 16,848,000 combinations required if we covered the entire duration of the wake word). These noise signals are superimposed on the existing samples of thirteen different users speaking the word 'Activate'. Thus, a total of 312,000 adversarial samples were created. Each time the noise of a particular duration is superimposed over the wake word at a given position p of the sensitive region, an adversarial audio sample is created. All users data were combined during training to build a single, speaker-independent wake word detector. Personalization of the model per user is not explored in this work.

Before training, each sample needed to be labeled. Since the goal is to distin-

guish between wake words and non-wake words, We labelled the words where 1 denotes the presence of the wake word (positive class), including wake word samples mixed with jamming signals, and 0 denotes non wake word audio (negative class). This labeling allowed the model to learn how to differentiate between true wake words and other sounds, even in the presence of jamming signals. Importantly, the model does not learn to recognize jamming patterns as indicators of wake words; rather, it learns wake word features despite the presence of interference.

5.3.2 Adversarial Training Process

We began adversarial training with a pre-trained LSTM_WD model that is designed to detect the wake word “Activate”. All layers of this model are retained. In the initial phase, we conducted adversarial training with data samples of the wake word "Activate," each overlaid with white noise as a jamming signal in the identified sensitive regions within the word. To prepare for training, the adversarial data samples are split into 20% for validation and 80% for training. We used a binary cross-entropy loss function for training since the goal is to classify each input as one of the two possible classes(either wake word or non wake word). The model was trained over 100 epochs with a batch size of 10 in each epoch. We selected the final model from the last epoch as our adversarially trained LSTM_WD model.

This model is then tested against jamming with white noise and the resulting heat map is given in Fig. 5.1a for one speaker uttering ‘Activate’. Only a few regions in the heat map are coloured blue and the jamming coverage is reduced from $C = 69.2\%$ (Fig. 4.13) to $C = 6.8\%$. This reduction demonstrates the success of adversarial training, as it shows the model has become largely resistant to white noise. Additionally, the once-sensitive regions no longer appear on the heat map, indicating that these vulnerable areas are no longer susceptible to interference.

In the next phase, we tested the adversarial trained model’s resilience using ping noise instead of white noise as an interference signal. This noise type was not included in the adversarial training set and was therefore unfamiliar to the LSTM-WD model. The result given in Fig. 5.1b shows that most regions in the heat map are coloured light blue indicating that even minimal energy from the ping noise was enough to disrupt the model’s performance in these regions. This outcome demonstrates that while adversarial training with white

noise improved the model's resistance to similar noise types, it does not fully generalise all kinds of interference signals particularly high-energy ones such as ping noise. The model's vulnerability to this new jamming signal highlights the need for a broader and more varied set of adversarial examples to enhance robustness against a wider range of noise types.

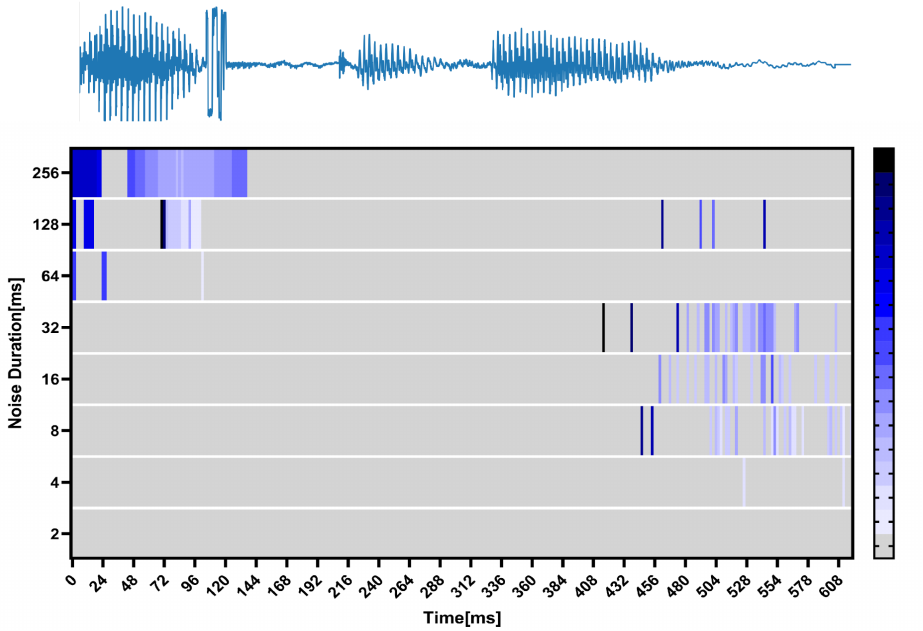
5.3.3 Resistance to Noise Variation

As explained in the previous Section 5.3, to build a more universally resilient model that can withstand a wide range of jamming signals, we needed a noise type that could effectively represent various types of interference. For this purpose, we selected babel noise as a replacement for white noise in adversarial training. Using babel noise as the new jamming signal, we generated adversarial samples and adapted the model in the same way as previously described. The experimental results using white noise and ping noise as jamming signals are shown in Fig. 5.2. In both heat maps, most of the regions are coloured grey indicating that training with babel noise effectively made the model more robust to an unseen and high-energy noise signal like ping noise. The model also became robust to white noise when trained with babel noise. For ping noise, the adversarial training using babel noise decreased the jamming coverage C to 9.8% from 66.0% (see Table 5.1).

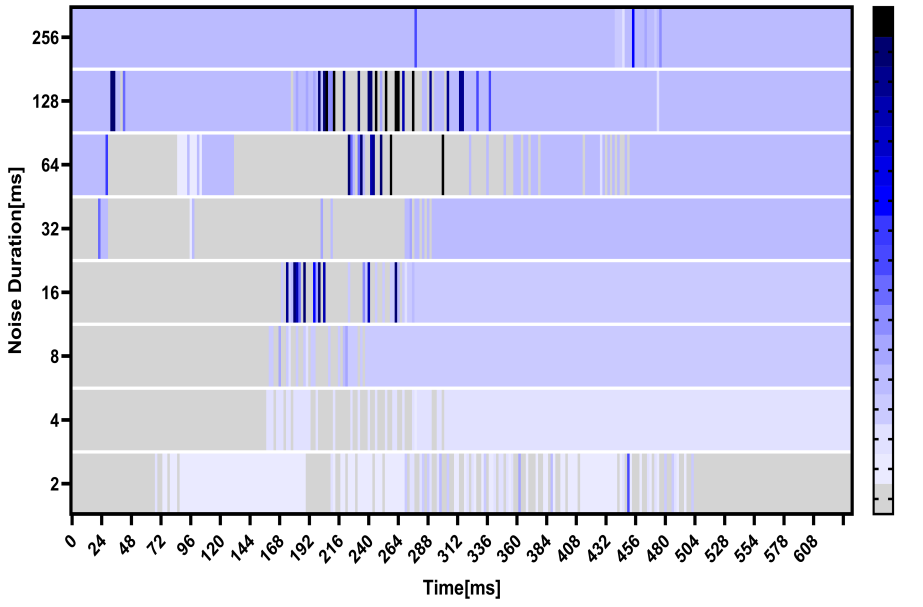
This reduction demonstrates that training with babel noise significantly improved the model's resilience to various jamming types, not only defending against white noise but also against ping noise. This improved resilience is likely due to the varied composition of babel noise, which includes a wide range of sound intensities and types. Importantly, the sensitive regions previously vulnerable to attack are now absent across all tested noise types, suggesting a robust, noise-resilient wake word detection model that can handle diverse interference conditions.

5.3.4 Results of Adversarial Training in Practical Experimental Setup

We replicated the same set of experiments in a practical setting using the experimental setup explained in 4.3.5, where the wake word and the jamming signals of varying duration are played through two separate USB speakers. One of the USB speakers played the wake word, while the other played the jam-

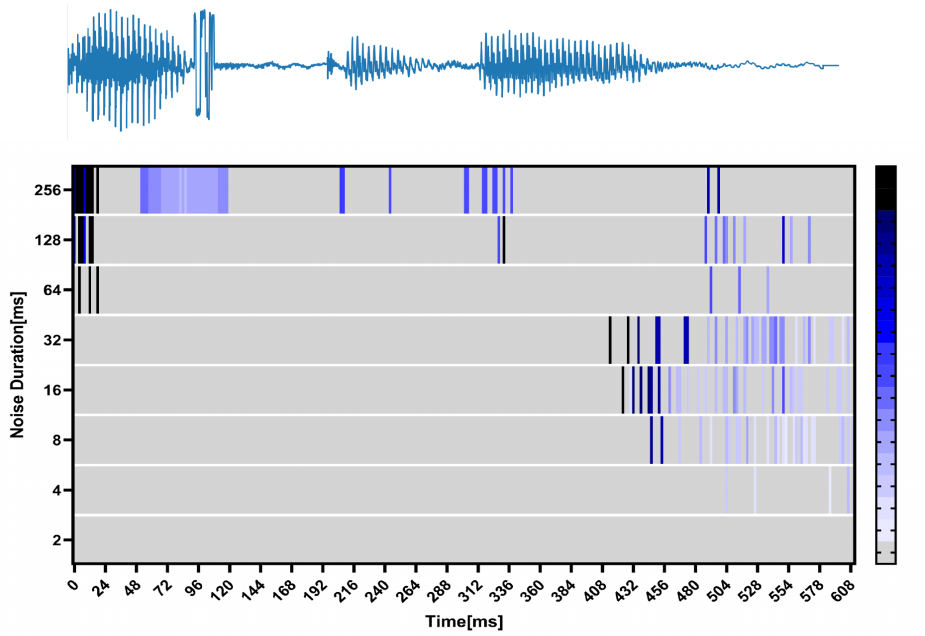


(a) Jamming using white noise. The attack is considered unsuccessful ($C = 6.8\%$).

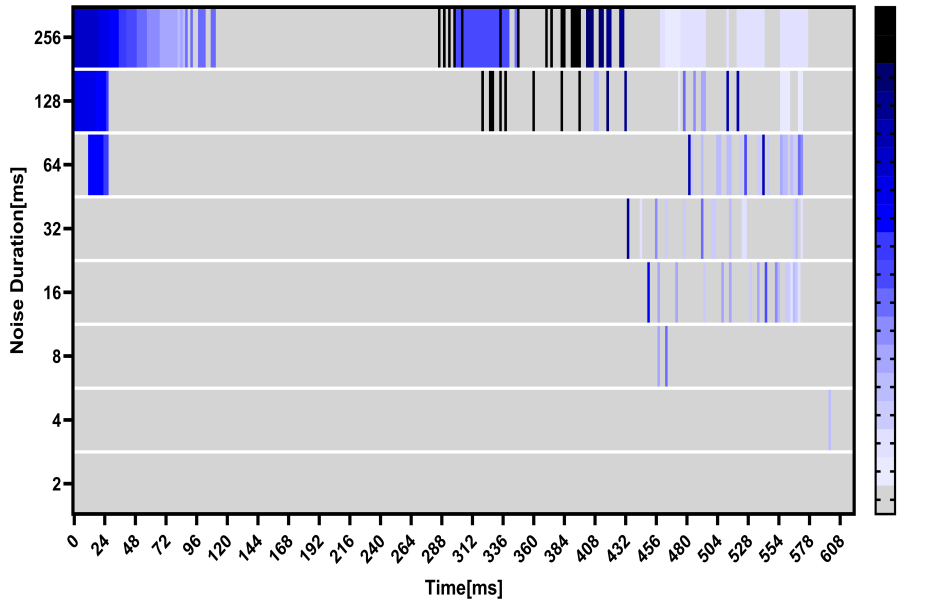


(b) Jamming using ping noise. The attack is considered successful ($C = 66.0\%$).

Figure 5.1: 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.



(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 ($C = 9.80\%$).

Figure 5.2: Impact of jamming using white and ping noise on the performance of wake word detection using 'Activate' after adversarial training with babel noise. WaveForm of Activate Superimposed with Jamming Signal on the Top.

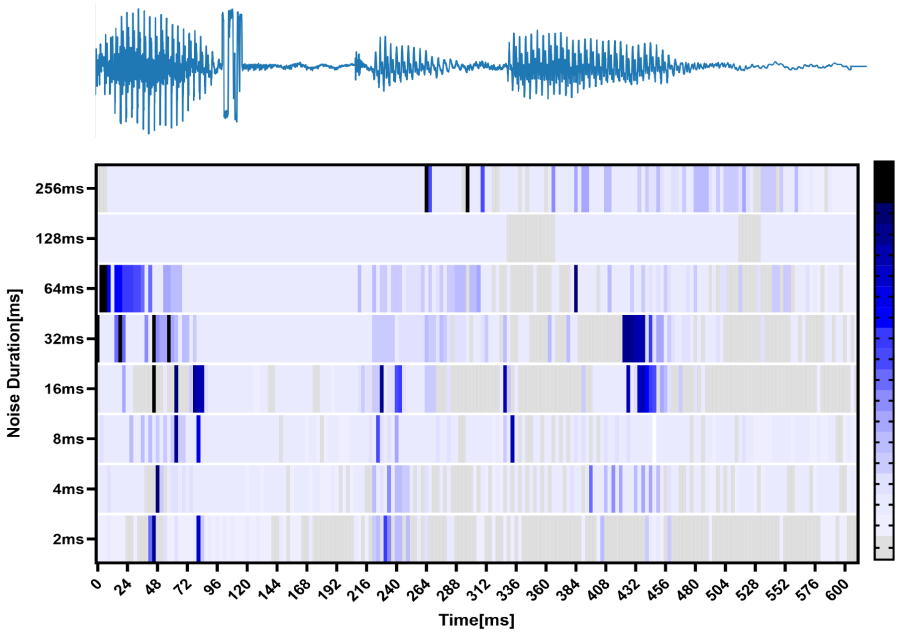
ming signals. In the beginning, both the wake word and jamming signal started playing simultaneously. However, with each iteration, we introduced a pro-

gressive delay to the jamming signal, allowing us to simulate a scenario where noise overlaps different parts of the wake word. As the jamming noise was played through USB Speaker 2 the delay shifted the interference signal incrementally across the wake word. For cases where the initial jamming signal did not disrupt detection, we incrementally increased the noise intensity until it reached a practical threshold set at 70 times the original jamming energy, with respect to the baseline levels described in Section 4.2.1. This corresponds to approximately 83-84dB SPL, an 18dB increase over the baseline, representing the practical maximum for undetected jamming. This approach allowed us to closely examine the effectiveness of the model under challenging interference conditions.

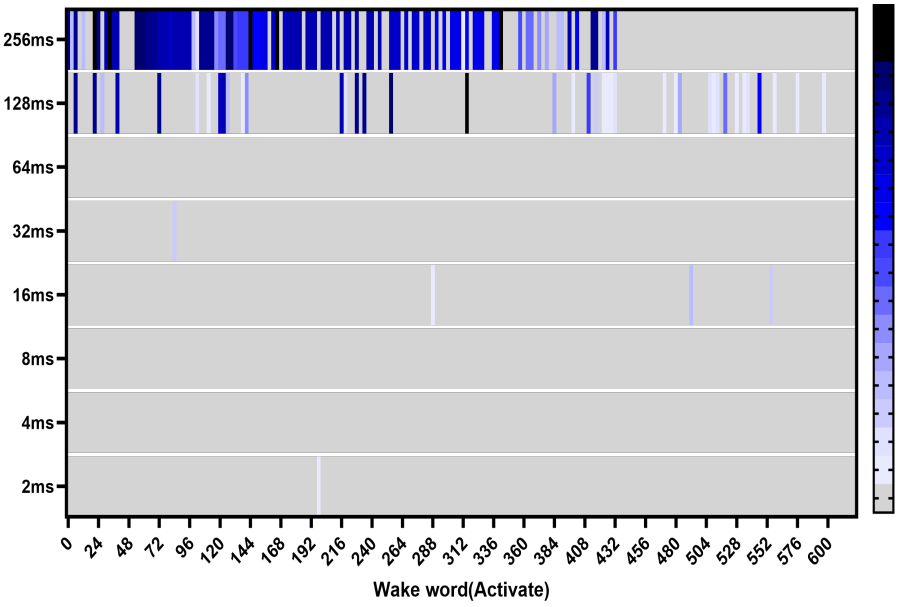
We conducted experiments to assess the wake word model's performance using ping noise as an interference signal, comparing its response before and after adversarial training with babel noise. In these tests conducted in the practical setting, ping noise served as the jamming signal, providing a challenging, high-pitched interference that the model had not encountered during standard training. By evaluating the model's detection capability in both conditions, we aimed to determine how effectively adversarial training with babel noise improved its resilience to such previously unseen noise types. The detailed results of this comparison are presented in Fig. 5.3, illustrating the model's strengthened resistance to ping noise in a real-world experimental setting after undergoing adversarial training.

5.3.5 Generalization

We also extended our evaluation of adversarial training to other wake words, such as 'Alexa' and 'Marvin', to see if the enhanced resistance to jamming observed with the wake word 'Activate' could be replicated with different words. The adversarial training process for 'Alexa' and 'Marvin' was similar to the approach we used for 'Activate'. We began by identifying the sensitive regions of each wake word through initial jamming, as explained in Section 4.3.10. Once these vulnerable areas were identified, we superimposed babel noise over these sensitive regions, using it as the jamming signal for adversarial training. This evaluation allowed us to determine whether the adversarial training with babel noise would consistently strengthen the model's resilience to jamming across various wake words. Our findings revealed that while the model's behaviour under jamming was similar across different wake words, the specific "sensi-



(a) Jamming using Ping Noise on the Wake Word Activate in a practical experimental setup. The attack is considered successful ($C = 74.3\%$).



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

Figure 5.3: 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. WaveForm of Activate Superimposed with Jamming Signal on the Top

tive regions" vulnerable to interference were slightly different depending on the unique sound patterns of each wake word. These differences in sensitive regions are due to each wake word's acoustic properties and the wake word model being used, which will be explored and explained in detail in Chapter 7. For example, comparing Fig. 4.14 from Chapter 4 with Fig. 5.4 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. 4.14 with jamming coverage of 51.3% and was clearly unsuccessful after the training as shown in Fig. 5.4 with jamming coverage of 10.0%.

A similar trend was observed for the wake word 'Marvin'. Fig. 4.15 and Fig. 5.5 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. 4.15 and Fig. 5.5, that the jamming coverage has reduced from 50.8% to 7.4% after adversarial training.

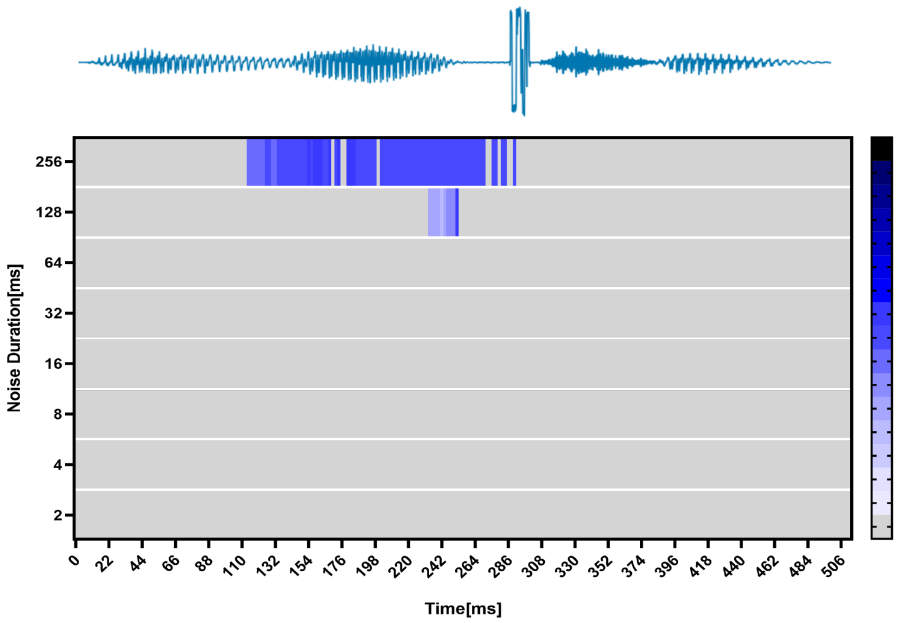


Figure 5.4: Jamming using ping noise on the wake word "Alexa" after adversarial training. WaveForm of Alexa Superimposed with Jamming Signal on the Top. The attack is considered unsuccessful ($C = 10.0\%$).

The jamming coverage C for different wake words is displayed in Table 5.1 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 behaviour towards adversarial training, that is

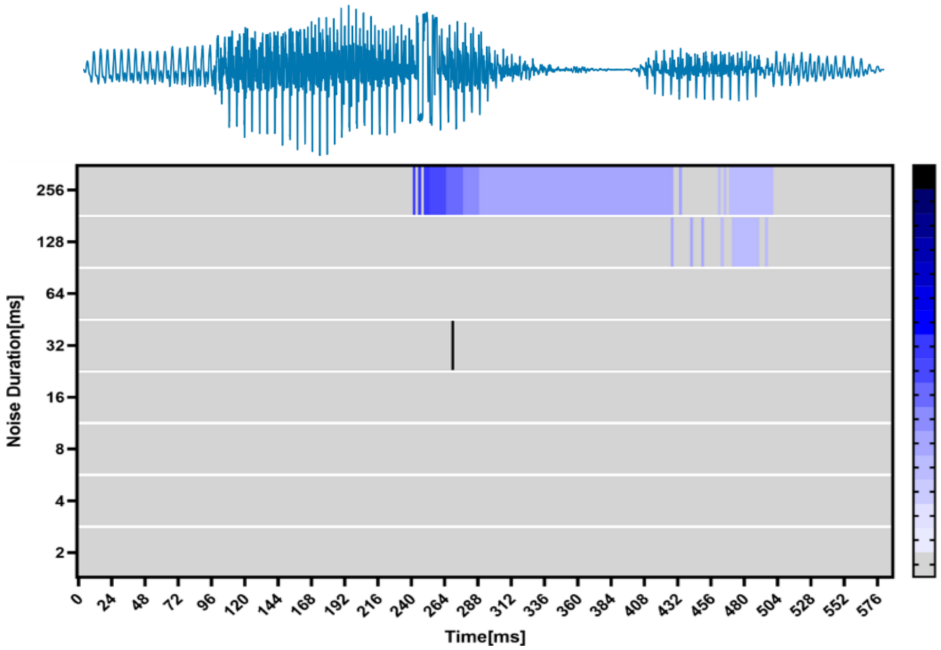


Figure 5.5: Jamming using ping noise on the wake word 'Marvin' after adversarial training. WaveForm of Marvin Superimposed with Jamming Signal on the Top. The attack is considered unsuccessful ($C = 7.4\%$).

babel noise training consistently reduces jamming coverage for both white and ping noise across all wake words.

These findings suggest that adversarial training can be a powerful method for improving a wake word model's resilience to jamming attacks. By testing this approach across different wake words including 'Activate,' 'Alexa,' and 'Marvin', we demonstrated that adversarial training successfully reduced jamming vulnerability, even though each word had distinct acoustic characteristics and sensitive regions. This consistency of achieving similarly low jamming coverage across all tested wake words despite their different phonetic structures suggests that adversarial training can be generalized to enhance the robustness of wake word models against jamming for a wide range of wake words, making it a versatile defense method.

In addition to testing the adversarial training method on the LSTM-WD wake word detection model, we evaluated its effectiveness on a different wake word detection algorithm called MHatt-RNN. We found that the adversarial training approach produced similar improvements in resilience to jamming attacks for this model as well. Specifically, although the sensitive regions within the wake word differed slightly in position compared to those observed in the LSTM-WD model, they were still clearly identifiable.

By training the *MHatt-RNN* model with samples of the wake word 'Marvin' combined with babel noise at the sensitive regions, we were able to enhance its robustness to jamming attacks. The results of jamming tests, conducted both before and after adversarial training, are presented in Fig. 5.6. As shown, the model's vulnerability to jamming significantly decreased after training, similar to the improvements seen with the *LSTM-WD* model. This indicates that our adversarial training approach can be adapted across different wake word detection algorithms, helping to generalize its applicability as an effective defense against interference.

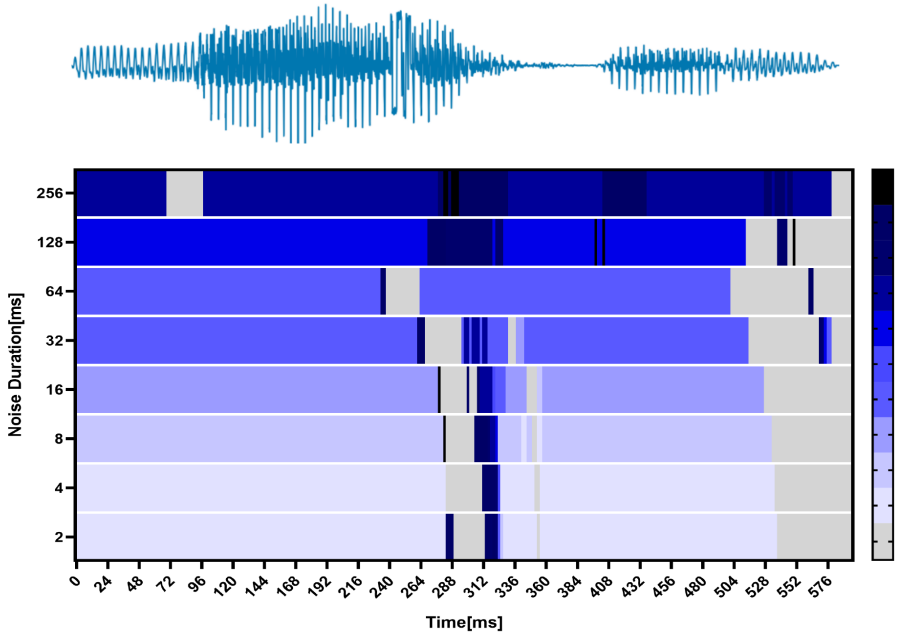
By comparing Fig. 4.15 and Fig. 5.6a we observe that before adversarial training, the *LSTM-WD* model demonstrates stronger resistance to jamming compared to the *MHatt-RNN* model. However, after adversarial training, *MHatt-RNN* surpasses *LSTM-WD* in jamming resistance, as shown in Fig. 5.5 and Fig. 5.6b. Importantly, in both models, adversarial training effectively eliminates the presence of sensitive regions, resulting in minimal jamming coverage and a more robust performance against interference.

5.4 Findings

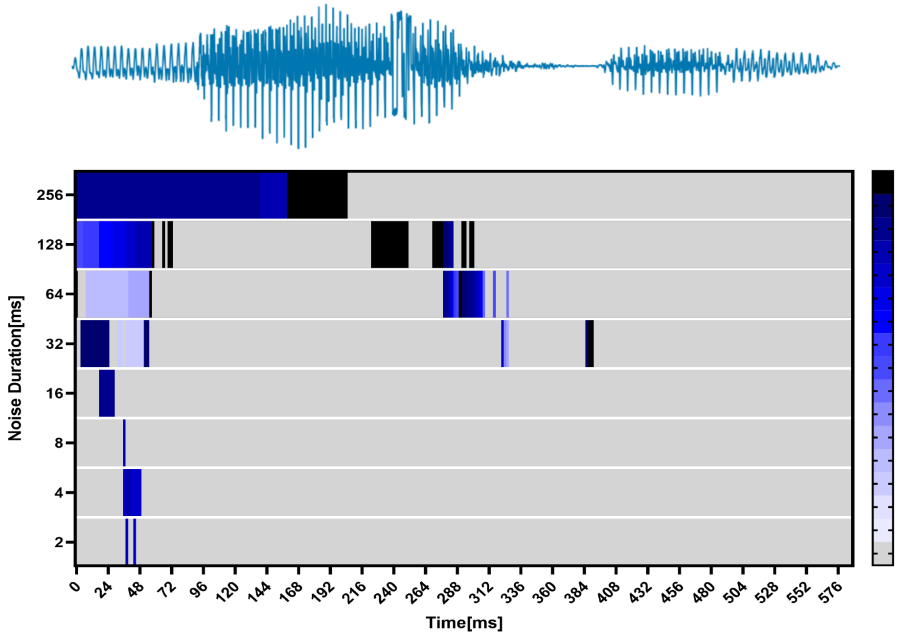
Our experiments show that targeted adversarial training significantly enhances a wake word detection model's resilience to jamming across various noise types and wake words.

Initial tests with the pre-trained *LSTM-WD* model revealed a high susceptibility to jamming, particularly within sensitive regions of the wake word. For instance, when jamming with white noise, the model exhibited a jamming coverage of 69.2% on the wake word "Activate". Adversarial training with white noise significantly improved the model's resistance to this noise type, reducing the jamming coverage to 6.8%. However, when exposed to a different jamming signal, such as ping noise, the model's performance degraded, indicating a need for a more generalized training approach.

To create a better defense against a wider range of jamming signals, we replaced white noise with babel noise for adversarial training. Babel noise, which includes diverse, realistic background sounds, allowed the model to generalize better against various interference types. This training approach resulted in a significant reduction in jamming coverage for both white noise and ping noise, reducing ping noise coverage to 9.8% from 66.0%, as demonstrated in



(a) Jamming using ping noise on the wake word 'Marvin' before adversarial training. The attack is considered successful ($C = 85.3\%$).



(b) Jamming using ping noise on the wake word 'Marvin' after adversarial training. The attack is considered unsuccessful ($C = 10.5\%$).

Figure 5.6: *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. WaveForm of Marvin Superimposed with Jamming signal on the Top

the heatmaps. Notably, no sensitive regions were present after training, suggesting that babel noise equips the model to defend against a broader spectrum of jamming signals.

To assess whether the adversarial training results are applicable across different wake words, we tested the approach on additional wake words, "Alexa" and "Marvin." Before adversarial training, both words showed unique sensitive regions susceptible to jamming. Adversarial training using babel noise, however, effectively eliminated these sensitive regions, reducing jamming coverage for example, 10.0% for "Alexa" and 7.4% for "Marvin". These findings suggest that the adversarial training method can be generalized to defend multiple wake words against jamming.

We evaluated the robustness of two different detection algorithms, *LSTM-WD* and *MHatt-RNN*, both before and after adversarial training with babel noise. Initially, *LSTM-WD* showed greater jamming resistance than *MHatt-RNN*. However, after adversarial training, *MHatt-RNN* outperformed *LSTM-WD* in resisting jamming attacks, with both models exhibiting low jamming coverage and the absence of sensitive regions. This outcome indicates that the adversarial training approach is beneficial across different model architectures, enhancing robustness to interference.

To validate these results in a practical setting, we replicated the experiments using USB speakers and a microphone array. We incrementally introduced delays and adjusted the intensity of the jamming signal. The findings from these practical experiments aligned with those from the simulated ones, confirming that adversarial training can reliably improve a model's resilience to jamming even in real world environments. Another significant finding from the practical experiments is that the wake word detection algorithm became robust to not just a spectrum of jamming signals but also maintained its accuracy in the presence of loud and constant background noises, such as those found in a busy college environment as the setup was demonstrated during a college event.

While adversarial training effectively reduces jamming vulnerability, we acknowledge that a comprehensive evaluation of false positive rates on acoustically similar words was not performed in this study. This represents an important direction for future work, as robust systems must maintain low false alarm rates on similar sounding non-wake words while achieving jamming resistance.

Chapter 6

Detection of Jamming Attack

As discovered in the previous chapters, a PVA can undergo DoS attack by exploiting the wake word detection algorithms [SSR24]. Attackers carefully design these jamming signals to interfere with wake word recognition [CBYR18, GGCPRB+19]. They often target specific, sensitive parts of the wake word to maximize disruption, making the PVA unreliable or unusable during the attack. Adversarial training could be used to address such attacks. This approach involves training the wake word model using examples where jamming signals are superimposed over the sensitive regions of the wake word. By exposing the model to these challenging scenarios during training, it becomes more robust and better equipped to handle jamming attacks.

However, making the model resistant to jamming is only part of the solution. While robust wake word detection is essential, it is equally important to detect the presence of jamming attacks so that the system can take appropriate action, such as alerting the users. In this chapter, we extend the work discussed earlier by enhancing the wake word detection model to not only reliably recognize wake words in the presence of interference but also identify jamming attacks as they occur. By addressing both wake word recognition and jamming attack detection, we aim to provide a more comprehensive and practical solution to counter wake word jamming in real-world scenarios.

This work extends the adversarially trained wake word detection model proposed in chapter 5, transforming it into a multi-class classifier through retraining to detect jamming attacks. This enhancement shifts the model from merely tolerating interference to proactively identifying the presence of a jamming attack.

Adversarial training of any neural network model can introduce a notable challenge: an increased risk of false positives. Addressing this issue is crucial to maintaining the model's reliability. In the context of a wake word detection model, false positives may occur when legitimate wake words are misclassified as jamming attacks (wake words combined with jamming signals). To mitigate this problem, we enhance the detection framework by integrating two additional techniques:

- Direction of Arrival (DOA): By analyzing the angle of arrival of audio signals, we can distinguish between wake words and jamming signals based on their source direction. Jamming signals often originate from a different location than the wake word, making DOA a useful tool for identifying potential interference.
- Short Time Energy (STE): STE is employed to detect abrupt changes in the energy pattern of an audio signal. Such changes are indicative of high-intensity, short-duration signals typical of jamming attempts. By flagging these anomalies, the model can better differentiate between natural variations in audio and deliberate interference.

These techniques, along with the adversarial training process, can reduce false positives while maintaining robust wake word recognition and enabling reliable detection of jamming attacks. This integrated approach ensures a practical and scalable defense mechanism for wake word detection in personal voice assistants.

6.1 Related Work on Detection of Wake Word Jamming

The detection of jamming attacks has been extensively explored in wireless networks and IoT devices [USL20, TGCEKK20, MGKP09, PIK11, GLY14]. However, there is a notable gap in research specifically targeting jamming attacks in the audio domain, particularly in the context of wake word jamming attacks affecting PVAs. Work by [USL20] explores jamming signal detection and classification in wireless communication systems using a Multi Layer Perceptron (MLP) neural network. The approach distinguishes between legitimate signals and various jamming types, including Continuous Wave Interference, Multi Continuous Wave Interference, and Chirp Interference [Poi11, Tor18], in digital video

broadcasting scenarios. Real-time datasets are generated using a Universal Software Radio Peripheral system, with feature extraction techniques including statistical measures like mean, standard deviation, skewness, kurtosis [TK09] and wavelet transforms [Mal99] to characterize the signals. Principal Component Analysis [Jol02] is applied for dimensionality reduction, which improves computational efficiency by eliminating less important features while retaining discriminative information. The MLP classifier, optimized using gradient-based learning and evaluated against Supported Vector Machine (SVM) models, demonstrates high precision in detecting and classifying jamming signals under varying noise levels. Though the work mainly addresses jamming detection in wireless networks, the principles can be extended to audio jamming as well. However, the work directly doesn't address how an audio signal jamming can be detected especially in terms of a PVA device.

Research done in [TGCEKK20] investigates a methodology to detect and classify smart jamming attacks in Long Term Evolution (LTE) networks, particularly addressing the vulnerabilities of synchronization and reference signals. The approach integrates a wavelet-based preprocessing step with machine learning classifiers to identify three types of jamming: barrage jamming, synchronization signal jamming, and reference signal jamming. Preprocessing involves transforming the received signals into a time-frequency representation using wavelet transforms, which allow for adaptive resolution and improved detection of short-duration or frequency-specific jamming attacks. This shows the advanced detection capabilities developed for wireless systems, capabilities that remain largely unexplored in the acoustic domain. Unlike wireless jamming, which benefits from decades of detection research, acoustic jamming of wake word detection systems has received minimal attention, leaving PVAs vulnerable to targeted interference attacks.

Work by [HLYH21], explores the vulnerabilities of acoustic self-jamming channels to overshadowing attacks, where an attacker transmits a signal simultaneously with the legitimate sender to modify the message received. It evaluates two prominent acoustic self-jamming schemes, *Dhwani* [NCPV13] and *PriWhisper* [ZLC+19], and demonstrates that such attacks can go undetected by human participants due to auditory masking thresholds. Using experimental setups, the authors show how attackers can synchronize with the receiver and transmit signals 5–11 dB stronger than legitimate transmissions at the receiver to successfully overshadow them without exceeding human perceptibility thresholds.

The paper proposes a physical layer integrity mechanism as a countermeasure, incorporating random security signals interspersed with data symbols. These signals are designed to be distinguishable from regular data by the receiver while being resistant to overshadowing. The mechanism involves detecting the security signals by measuring symbol energy [PS08] and identifying signals exceeding a threshold above average power, then discarding overlapping attack signals, and verifying message integrity using hash-based authentication. This approach enhances the integrity of self-jamming systems without requiring prior security relationships between devices. While the work discusses jamming and its countermeasures in the context of acoustic systems, its methods and goals differ significantly from the problem we are addressing, which focuses on detecting jamming signals that can specifically disrupt wake word detection algorithms in PVAs.

There is some research done on the detection and cancellation of inaudible voice commands that target PVAs. Inaudible attacks typically exploit the microphone's inherent non-linearity where the microphone's response is not proportional to the input signal, causing ultrasonic frequencies to be demodulated into the audible range, to inject ultrasonic-modulated commands that are imperceptible to humans but recordable and executable by the PVAs, an example of such attacks is a *Dolphin-Attack*.

The detection of inaudible commands described in the research done by [HBT+19a] relies on identifying artifacts introduced by the non-linear behaviour of microphones. This non-linearity affects both audible and ultrasonic frequency ranges, though it is particularly exploited when processing ultrasound signals for attack purposes. Attackers exploit this non-linearity to inject inaudible commands by modulating human speech onto ultrasonic carriers, typically requiring the attack signal to reach the microphone with loudness around 65 dB to produce detectable distortion artifacts [HBT+19a]. While these signals are imperceptible to humans, they are shifted into the audible range by the microphone's hardware through frequency mixing and intermodulation, creating telltale artifacts as illustrated in Figure 6.1.

The detection system, called LipRead, takes advantage of these imperfections by analyzing three key artifacts. First, it detects energy in very low frequencies (below 50 Hz), which does not typically appear in normal human speech but emerges when ultrasonic signals undergo non-linear distortion. Second, LipRead examines the correlation between high and low-frequency energy com-

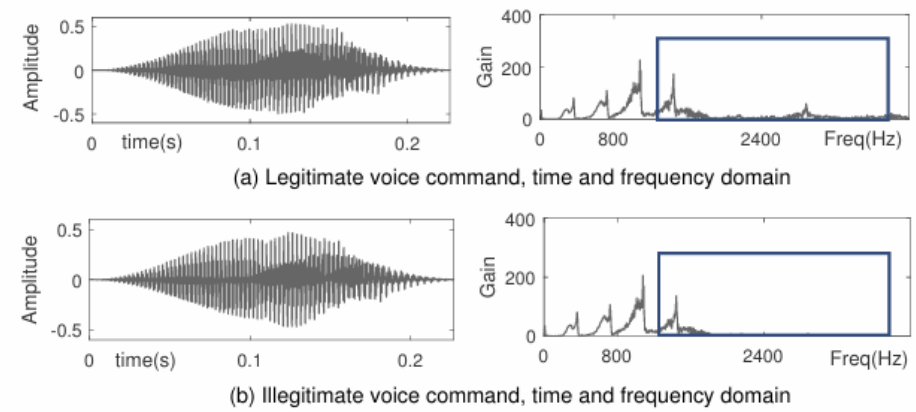


Figure 6.1: Features of signals in high-frequency range showing microphone non-linearity effects. Figure adapted from He et al. [HBT+19a].

ponents, normally predictable in human speech but disrupted during ultrasonic attacks due to the microphone's transformation of the signal. Third, it analyzes the symmetry of the audio signal's amplitude distribution, as legitimate human speech has balanced amplitudes while ultrasonic signal recording introduces asymmetry and skewness. The system employs a machine learning model trained on examples of normal audio and audio containing inaudible commands to identify these attack patterns with high accuracy. However, this work addresses only inaudible ultrasonic attacks and does not detect the audible, short, high-intensity jamming signals that can compromise the PVA's wake word detection algorithms.

Work by [JGL+24], presents a detection system called EarArray designed to identify inaudible voice commands like those in DolphinAttack injected into voice assistants like Amazon Echo. These attacks exploit the non-linearities of microphones to modulate malicious commands onto ultrasonic signals, which microphones demodulate into audible frequencies for the device, while remaining inaudible to humans. EarArray relies on a fundamental physical principle: acoustic attenuation, which affects sound waves of different frequencies differently as they propagate. Ultrasound signals (e.g., 25 kHz carrier frequency) attenuate significantly faster than audible signals due to absorption attenuation being proportional to the square of the frequency [JGL+24]. Specifically, the attenuation coefficient can be expressed as $a(\omega) = a_0\omega^n$ where $n \in [0, 2]$, meaning that a 25 kHz ultrasonic signal experiences approximately 2,500 times greater attenuation than an audible signal over the same distance. By analyzing how the sound's intensity decays across multiple microphones, EarArray can distinguish between normal voice commands and ultrasonic attacks.

The system works without requiring additional hardware modifications. It uses the microphones already present in modern devices and incorporates software to process the audio signals. EarArray measures the sound intensities received at each microphone, computes attenuation patterns, and identifies the distinct propagation characteristics of ultrasonic signals, specifically, that higher frequency signals experience greater attenuation with distance and produce uneven intensity distributions across the microphone array compared to normal audible voice commands. This is achieved through feature extraction methods such as band power, spectral analysis, and machine learning classifiers like SVM and CNN. Additionally, EarArray can locate the attacker's direction based on the energy distribution across the microphones. Experiments validate EarArray's high effectiveness, achieving over 99% detection accuracy for inaudible commands and robust performance across varying distances, angles, and ambient noise conditions. It also detects other attack types, such as laser-based attacks, with 100% accuracy. The system's reliance on sound propagation physics makes it resistant to evasion techniques and suitable for deployment on existing smart devices.

These aforementioned works do not directly address the challenge of detecting jamming signals that compromise wake word detection systems in PVAs, rendering devices unavailable for operation. Although there has been some research showing that wake word detection can be robust in noisy environments, no method currently exists that can detect the jamming signals so that users can be alerted about deliberate jamming attacks on their devices. We propose a novel approach that converts the wake word detection algorithm into a three-class classifier to detect wake words in the presence of jamming signals so that the users can be aware of the attack. Our work also makes use of Direction of Arrival (DOA) and Short Time Energy (STE) to validate the predictions of the wake word model.

DOA is a technique used to determine the direction in which a sound signal arrives at a microphone array for sound source tracking [JJSL21]. Existing work has used microphone arrays and DNNs to estimate the direction of arrival of an audio signal [JJSL21, MWH22]. We use this technique to track the DOA of the wake word audio to identify any changes indicating a potential attacker. STE analyzes changes in the energy of an audio signal over time. Previous work has utilized STE in telephone conversations to differentiate speech from background noise, optimizing resource usage in communication systems [EGYY02]. Here, we use STE to detect the high-intensity, short-duration noise pulses that

disrupt wake word detection as described in Chapter 4.

By combining DOA and STE with the predictions from our multi-class wake word detection classifier, we offer a robust system for detecting and alerting users to jamming attacks, providing a novel and effective solution to this pressing issue in PVA security. However, DOA based detection requires a device with multiple microphones; however, single-microphone devices can still benefit from the STE analysis and multi-class classifier components of our system.

6.2 Attack Detection

A jamming signal, regardless of its duration, can significantly disrupt wake word detection if it is placed within the sensitive regions of the wake word. Addressing such disruptions requires more than just adversarial training techniques, as these signals are often subtle, short, and difficult to identify.

For instance, while *Amazon's Echo* devices include built-in capabilities to detect predefined sounds such as a baby crying, a dog barking, snoring, or coughing [Smi, Dav25], these features are tailored to a specific set of recognizable and relatively prolonged sound patterns. Such systems are not designed to identify or respond to noises outside of these predefined categories. Jamming signals, which can be exceptionally brief, high-pitched, and strategically targeted, fall outside the scope of these predefined sound effects. This limitation underscores the need for advanced detection techniques capable of identifying and classifying unexpected or anomalous sounds, particularly those designed to interfere with wake word detection.

To detect the presence of jamming signals in incoming audio input, we adapt the wake word detection model into a three-class classifier. This enhanced model is designed to classify incoming audio into one of three categories: *Class 0: Non Wake Word*, *Class 1: Wake Word*, or *Class 2: Attack* (indicating that a jamming signal interferes with the sensitive regions of the wake word). As with any detection system, minimizing false positives is critical for reliability. Achieving this requires not only improving the classifier's accuracy but also integrating additional validation mechanisms to reinforce confidence in its predictions. When the classifier identifies a potential attack, further validation is performed to confirm the accuracy of the detection.

One effective validation method involves analyzing the DOA of the incoming

wake word audio signal. PVAs are commonly equipped with a microphone array, which enables the estimation of the direction from which the sound originates. This directional analysis is particularly useful because an attacker and the legitimate user are unlikely to be located in the same position. If a jamming signal is introduced during the wake word utterance, the DOA will typically exhibit a brief and noticeable shift, temporarily indicating the direction of the jamming signal. This directional change, especially when it occurs within the sensitive regions of the wake word, serves as a strong indicator that the classifier's detection of an attack is correct.

Another complementary approach for validation involves analyzing the energy profile of the audio signal using STE analysis. STE measures the variations in energy over short time intervals and is particularly effective at identifying sudden abrupt changes in signal energy. By focusing on the sensitive regions of the wake word, STE can detect abrupt changes that often accompany jamming signals. Such energy fluctuations provide an additional layer of evidence, supporting the classifier's prediction of an attack. It should be noted that STE is most effective against superimposition of jamming signal scenario, replacement of jamming signals, though is not an ideal jamming approach as explained in chapter 4 would require detection through other means, such as the multi-class classifier itself or DOA analysis.

In the event that an attack is confirmed, the PVA can notify users through its built-in alert mechanisms. These notifications may include visual indicators, audio alerts, or messages through companion apps, ensuring that users are made aware of the jamming attempt in real-time. Although the specifics of how alerts are generated and communicated are beyond the scope of this research, incorporating such mechanisms is vital for user awareness and prompt response to security threats.

6.2.1 Training Data

We utilized the training samples generated for adversarial training as described in Chapter 5. This dataset comprised 312,000 audio samples, which were created by superimposing Babel noise of varying durations onto the sensitive regions of the wake word. The original model was trained on recordings of the wake word "Activate" spoken by multiple speakers, alongside non-wake word samples. This training data from the previous chapter was used as it is without modification. Since the model is designed to classify three distinct types

of input, the labeling scheme for the data was adapted accordingly. The training data is labeled to enable the model to effectively differentiate between the following categories:

- Legitimate wake words are labeled as *Class-1 (Wake Word)*.
- Audio that does not contain a wake word is labeled as *Class-0 (Non-Wake Word)*.
- Wake words that have been superimposed with jamming signals are labeled as *Class-2 (Wake Word + Jamming Signal)*.

This comprehensive labeling scheme ensures that the model can robustly distinguish between these categories, enabling accurate classification. While the model is trained on wake words superimposed with jamming signals to build robustness, the Class-0 (Non-Wake Word) category encompasses all audio not containing a wake word, including those with background noise. The model learns to discriminate based on the presence of the wake word pattern itself, rather than the noise characteristics. Therefore, a separate category for non-wake words with jamming is not included.

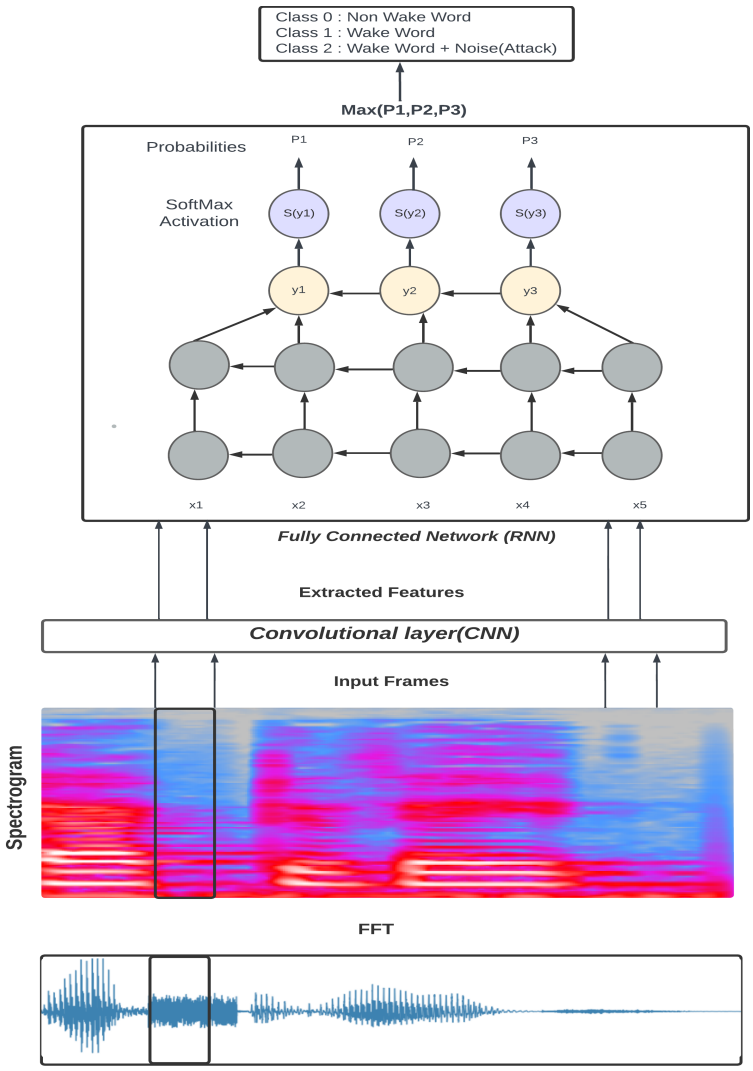


Figure 6.2: The wake word detection process of LSTM_MCC. The audio signal (wake word mixed with Babel noise of 128ms) is divided into frames of 10 ms and features from each spectrogram are extracted and fed into an RNN. The softmax function is applied to the outputs and the maximum probability of the output is considered to make the categorization decision.

6.2.2 Training Process

The development of the multiclass classifier begins with the pre-trained LSTM_WD model, originally trained to classify audio inputs as either containing a wake word or not. This pre-trained model serves as a foundational starting point, and all its layers are retained. In its original configuration, the last layer of the LSTM_WD model employs a sigmoid activation function, suitable for binary classification tasks. For this multiclass classification task, the activation function in the last layer is replaced with a softmax activation function. The softmax function is chosen because it outputs a probability distribution across

all three classes that sums to 1, enabling the model to indicate the relative likelihood of each class and select the most probable one. This adjustment allows the model to output probabilities across the three target classes. The architecture of the modified model, referred to as LSTM_MCC, is illustrated in Figure 6.2. The training samples are divided into two subsets: 80% of the data is used for training, ensuring the model has ample exposure to diverse input scenarios, and the remaining 20% is reserved for validation. The model is trained using the categorical cross-entropy loss function. Categorical cross-entropy is chosen as the loss function in this context because it is particularly well-suited for multiclass classification tasks, where the goal is to assign an input to one of several mutually exclusive classes. Training is conducted over 500 epochs with a batch size of 50. After training, the model generated at the final epoch is selected for multi-class classification. The LSTM_MCC model is an improved version of the original binary classifier, designed to handle the more challenging task of classifying the incoming audio signal.

6.2.3 Direction of Arrival (DOA)

To verify any directional changes in the incoming audio samples, we employed the DOA, technique, which is designed to estimate the angle from which an audio signal originates. This method is particularly useful for determining the direction of sound sources, allowing us to detect the presence of jamming signals. For the experimentation, we used the ReSpeaker Mic Array to estimate the angle of arrival of incoming audio signals. This device runs a DOA algorithm to identify the direction of sound sources. While the exact implementation details of the algorithm used by the ReSpeaker Mic Array are proprietary and not publicly disclosed, the general principles behind DOA estimation in signal processing are well understood and widely applied [JJSL21, MWH22].

DOA is a critical concept in signal processing that determines the angle or direction from which an audio signal originates relative to a microphone array. This technique however, requires a multi-microphone configuration, single microphone devices cannot perform DOA estimation. The process begins with the placement of multiple microphones in a specific geometric configuration on the array. When a sound wave reaches the array, it arrives at each microphone at slightly different times, depending on the relative distances of the microphones from the sound source. These differences in arrival times, referred to as Time Difference of Arrival (TDOA), form the basis of DOA calculations.

To estimate TDOA, techniques such as generalized cross-correlation and least-squares methods are commonly employed. These methods analyze the audio signals captured by the microphones, measuring the time delays between the signals to identify the TDOA with precision. Once the TDOA values are established for pairs of microphones, the DOA of the incoming sound wave can be computed. This involves leveraging the known geometric configuration of the microphone array and applying mathematical models to determine the angle from which the sound originated [JJSL21]. While we do not independently verify or implement these algorithms, we leverage the ReSpeaker's built-in DOA capabilities as a reliable hardware component for validating our jamming detection approach.

The spatial information provided by DOA estimation, particularly when the jamming signal and wake word originate from different directions, could theoretically enable sound source separation techniques [DVG10], potentially allowing recovery of the wake word signal even when partially overlapped with jamming noise. However, our current work focuses on the detection of the jamming attacks rather than signal recovery.

6.2.4 Short Time Energy (STE)

STE is a technique used to analyze how the energy of an audio signal varies over time. Rather than computing energy across the entire signal duration, STE focuses on energy fluctuations within smaller time segments, or frames. This enables detection of rapid amplitude changes in the signal. In the context of our threat model (Chapter 4), the jamming signals we consider are designed to disrupt specific sensitive regions of the wake word. When such jamming noise is superimposed onto the wake word, it typically manifests as sudden energy spike that exceeds the amplitude of natural speech. This spike is then captured by the STE analysis and can serve as a key indicator of a potential disruption to the wake word detection process [ÖZA19]. To apply this method, we divide the incoming audio signal into short frames, each with a duration of 20 milliseconds. For each frame, we calculate the STE, which measures the amount of energy present in that specific time window. This allows us to track how the energy fluctuates over time, helping to pinpoint any sudden bursts or irregularities. We then set a detection threshold for identifying these significant energy changes. In our case, the threshold for detecting a peak in STE is fixed at 0.1, where the STE values are scaled to the range [0, 1] based on the maximum

energy observed in a sliding window. This value was determined empirically through evaluation where we tested threshold values ranging from 0.05 to 0.5 in increments of 0.05.

When the STE analysis reveals a spike in the energy within the sensitive regions of the wake word, we can conclude that there has been an attempt to interfere with the wake word detection, likely due to a jamming signal. However, it is important to note that energy spikes alone are not sufficient to conclusively identify jamming attacks, as coincidental environmental noise (e.g., a door slam, or sudden background sound) could also produce similar energy anomalies. This is why STE serves as one component of our multi-layered validation approach rather than a standalone detection mechanism.

When anomalies are detected in sensitive regions, particularly with validation from multiple sources such as the wake word detection model, and STE or DOA, we alert the user rather than dismissing the event as potential environmental noise. The impact of a false positive (unnecessary alert) is minimal compared to a false negative (undetected attack enabling unauthorized access).

6.3 Evaluation

The LSTM_MCC model, designed to detect jamming attacks, was evaluated in multiple scenarios through eight experiments conducted 50 times under various conditions including clean wake words, wake words with background café noise [STE22], wake words with jamming signals, and non-wake word samples. The experimental methodology followed the over-the-air setup described in Section 4.3.5. The results were compared against the baseline LSTM_WD model, which can distinguish between wake words and non-wake words but cannot identify jamming attacks. These experiments covered different types of noise, jamming signals, and positions of the wake word relative to the noise or jamming signals. A summary of all the evaluated scenarios is provided in Table 6.1. Below, we outline the experimental setup and discuss the results of eight distinct experiments. The experiments involved feeding the models with either wake words or non-wake words under various noisy conditions and the relative timing of the noise with respect to the wake word. Two types of background noises were used: café noises obtained from [STE22] and ambient coffee shop sounds from the same source. These were selected to represent common deployment scenarios containing speech, music, and ambient sounds,

enabling evaluation of whether the LSTM_MCC model can distinguish environmental sounds from deliberate jamming attacks. The reason for including background noise testing is to demonstrate that the model maintains low false positive rates in noisy environments while still detecting actual attacks which is an important requirement for practical deployment where users frequently interact with voice assistants in cafés, offices, and other noisy settings. Additionally, the experiments used several types of jamming signals, including manually generated ping noises with a frequency range of 0-8 kHz and an SNR of 0dB. These ping noises were tested at different length such as 2ms, 16ms, 64ms, and 128ms. Other jamming signals, such as keyboard noise and white noise, were also tested, but the results for these signals were excluded because ping noise proved to be the most effective for evaluation purposes. The non-wake words were sourced from the publicly available dataset [War18b].

The experimental results in table 6.1 highlights each model's predictions. It also shows the number of times jamming attacks successfully disrupted wake word detection for both models. The performance of the models was measured based on their classification accuracy and the number of successful attack detections.

In addition to these metrics, the experiments analyzed the DOA position, which determines the direction from which an audio signal arrives. If a jamming signal originated from a different direction than the legitimate wake word, the DOA position shifted accordingly. This directional information provided valuable insights into whether the detected audio was a legitimate wake word or an attack. The experiments also measured the STE peak position, which identifies the point in time where the energy in the audio signal is highest. This analysis helped pinpoint the location of the jamming signal within the audio sequence, further verifying the presence of an attack.

The LSTM_WD model results were categorized into two classes: clean wake words (WW) and non-wake words (NWW). In contrast, the LSTM_MCC model, enhanced for multi-class classification, categorized audio inputs into three classes: clean wake words (WW), non-wake words (NWW), and wake words with jamming signals (WW+Jammer). The table reflects the counts for each classification type made by the models across the 50 different experimental runs, providing a detailed comparison of their performance. This analysis highlights the robustness of the LSTM_MCC model in detecting and mitigating jamming attacks while maintaining accurate classification of legitimate audio inputs. It is important to not that, when LSTM_MCC correctly identifies

"WW+Jammer", the attack is effectively neutralized and the system recognizes the legitimate wake word allowing access while simultaneously alerting the user to the jamming attempt. Thus, the attacker fails to achieve their goal of blocking access or preventing detection. The user maintains normal functionality but gains security awareness, enabling investigation of the interference source or implementation of protective measures

6.3.1 Experiment Set 1

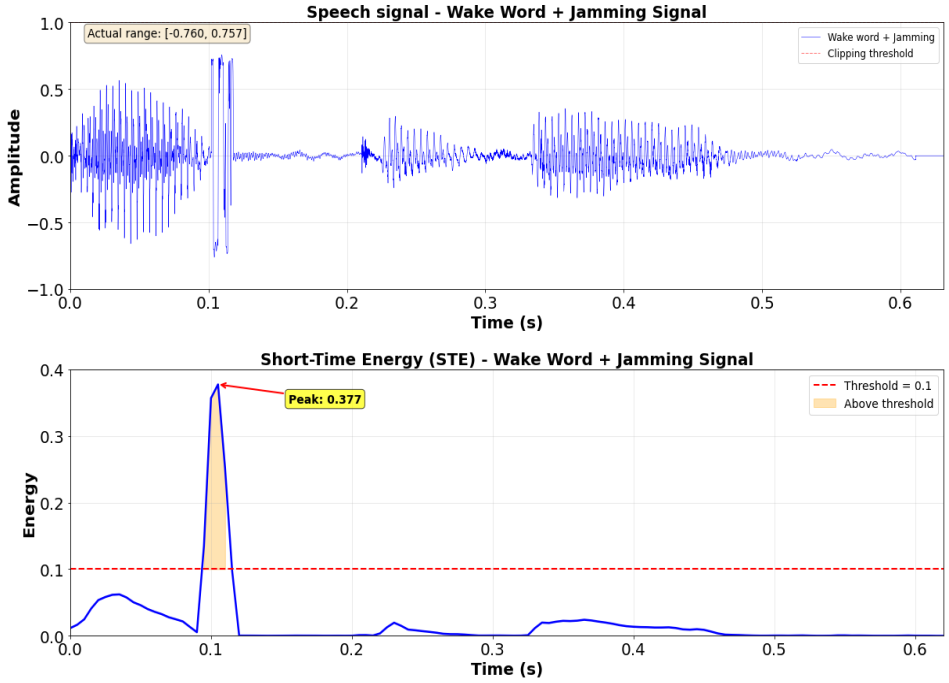
Experiment Set 1 is designed to establish a baseline for comparison. In this setup, no jamming attacks are introduced, meaning the environment is free from interference or additional noise that could disrupt the system. The purpose of this experiment is to observe how both models perform under ideal conditions.

A wake word is provided to each model 50 times, ensuring consistency across the tests. Both the LSTM_WD model and the enhanced LSTM_MCC model correctly classify the input as a wake word in every instance. Since there are no attacks or jamming signals present, additional metrics such as STE and DOA remain inactive and do not register any anomalies.

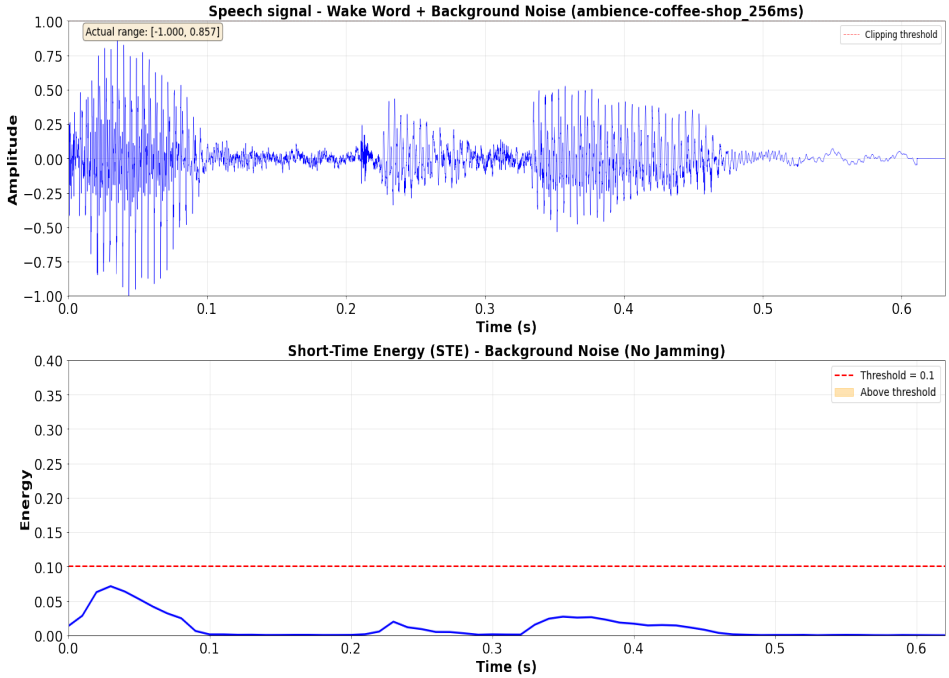
6.3.2 Experiment Set 2

In Experiment Set 2, we test how well the models perform in a noisy environment. To simulate this, we mix café background noise with the wake word. However, there are no attacks present in this scenario, the goal is to observe how both models handle real world background noise.

The LSTM_WD model, which was not trained to handle noise or attacks, failed to correctly detect the wake word in 10 out of 50 trials. This indicates that it struggles to identify the wake word when background noise is present. In contrast, the LSTM_MCC model, which was trained to deal with more challenging conditions, successfully identified the wake word in all 50 trials. The LSTM_WD model reflects the current state-of-the-art approach used in commercial PVAs, where the wake word detector fundamentally lacks the capability to detect a jamming attack and alert users when attacks occur. The LSTM_MCC model represents an architectural enhancement of the same LSTM_WD model. Therefore, the comparison between LSTM_WD and LSTM_MCC demonstrates the value of our proposed enhancement. However, the LSTM_MCC model mis-



(a) Short time energy graph of wake word in the presence of 16ms ping noise showing a peak in the sensitive area



(b) Short time energy graph of the wake word in the presence of cafe noise with no peaks

Figure 6.3: Comparison of STE graphs.

Table 6.1: Performance validation of the LSTM_MCC model using wake word (WW), non-wake word (NWW), and wake word with noise (WW+Jammer) samples. Additionally, the model is compared with the LSTM_WD model to demonstrate its superior performance.

Experiment	Word	Noise	Position of Noise Over Wake Word(ms)	Expected Output	LSTM_WD Outputs		LSTM_MCC Outputs			DOA Positions	STE Peak Position	Number of Successful Attacks Considering DOA and STE	
					WW	NWW	WW	NWW	WW+Jammer			LSTM_WD	LSTM_MCC
1	Wake Word	No Noise	NA	WW	50	0	50	0	0	No Change in Direction	No Energy Peak	0	0
2	Wake Word	50 Different Cafe Background Noises	0 (At the Start of Wake Word)	WW	40	10	42	0	8	∼at 240 ms (Changes Direction in the Silent Regions of the Wake Word)	No Energy Peak	10	0
3	Wake Word	PN SNR0 BW 0-8000 64ms	0 (At the Start of Wake Word)	WW	50	0	50	0	0	∼at 0 ms (At the Start of the Wake Word)	0 (At the Start of Wake Word)	50	0
4	Wake Word	PN SNR0 BW 0-8000 128ms	100-150	WW+Jammer	0	50	0	0	50	∼around 100ms-150ms (Sensitive Area)	100ms-150ms (Sensitive Area)	50	0
5	Wake Word	PN SNR0 BW 0-8000 16ms	100-150	WW+Jammer	6	44	8	0	42	∼around 100ms-150ms Sensitive Area	100ms-150ms (Sensitive Area)	44	0
6	Wake Word	PN SNR0 BW 0-8000 2ms	100-150	WW+Jammer	41	9	43	0	7	∼around 100ms-150ms Sensitive Area	100ms-150ms (Sensitive Area)	9	0
7	Non Wake Word	Ambience-coffee-shop(2sec)	NA	NWW	0	50	0	50	0	No Change in Direction	No Energy Peak	Not an Attack	Not an Attack
8	Non Wake Word	50 Different Non-Wake Words	NA	NWW	0	50	0	50	0	No Change in Direction	No Energy Peak	Not an Attack	Not an Attack

takenly classified the café background noise as a potential attack (wake word with jamming noise) in 8 out of the 50 trials. While this demonstrates the model's sensitivity to anomalies, it also highlights the issue of false positives. False positives occur when the model incorrectly flags a benign situation as an attack, which can lead to unnecessary alarms. To reduce these false positives and avoid triggering false alarms, we analyze additional metrics such as DOA and STE.

The DOA metric, which detects changes in the direction of the incoming sound, showed that a directional change occurred after 240 milliseconds into the wake word in this specific experiment set. This timing does not fall within the sensitive region of the wake word. The direction change likely occurred because the background noise was more prominent to the microphones than the wake word at that moment, in the silent regions of the wake word. The STE metric, which measures energy spikes in the audio signal, did not show any peaks within the sensitive region of the wake word in this scenario. This indicates that the observed fluctuations are purely due to background noise and not caused by a jamming event. An example of this analysis is shown in Figure 6.3b.

By applying either DOA or STE analysis, 8 of the LSTM_MCC model's false attack classifications are corrected, reducing the false-positive count to zero. In other words, these checks ensure that a normal background noise scenario is not mistakenly labeled as an attack. This improvement is further supported by the data presented in the "successful attacks" column in the Table 6.1, showing the effectiveness of these additional validation steps in noisy environments.

6.3.3 Experiment Set 3

From Experiments Sets 3 to 6, the models are evaluated by introducing jamming signals to disrupt wake word recognition. The jamming signals used are short "ping" noises within a 0-8 kHz frequency range. These signals are designed to have the same power level as the wake word signal (an SNR of 0 dB as discussed in Chapter 4.2.1) and are tested at different durations: 2ms, 16ms, 64ms, and 128ms, which are then inserted at each position relative to the wake word duration.

In Experiment 3, a 64ms jamming signal is played at the beginning of the wake word. Since this signal does not overlap with the sensitive regions of the wake word, it has no effect on the performance of either model. Both the LSTM_WD

and the LSTM_MCC model correctly classify all 50 samples as wake words. As summarized in Table 6.1, the results show that all 50 wake word inputs are classified correctly without any errors. The LSTM_MCC model does not label this scenario as an attack, which is the expected behaviour. This could be because, though a jamming signal is technically present, it does not overlap with the sensitive areas of the wake word. Since the model was specifically trained to detect attacks only when jamming signals interfere with these sensitive regions, this instance is not identified as an attack. The LSTM_MCC model's output reflects this, confirming that it functions as intended in this scenario.

6.3.4 Experiment Set 4

In Experiment Set 4, a 128ms jamming signal was used to test the models' ability to detect and respond to attacks targeting the sensitive areas of the wake word. The jamming signal was introduced with a delay of 100ms after the wake word started, ensuring it overlapped with the sensitive regions that are crucial for detection. This delay was incrementally increased by 1ms for 50 iterations to thoroughly evaluate the models' performance under varying conditions.

The LSTM_MCC model successfully detected the presence of the jamming signal and correctly classified the input as an attack. Despite the interference, it was still able to recognize the wake word, allowing the user's request to be processed without disruption. In contrast, the LSTM_WD model, which cannot identify attacks, failed to recognize the wake word. This failure could expose the PVA to a DoS attack, as the device would not respond to the user's command.

To further validate the LSTM_MCC model's predictions, additional metrics were analyzed. The DOA position revealed a consistent change in direction between approximately 100ms and 150ms after the start of the wake word in all 50 test cases. This change aligns with the timing of the jamming signal, as shown in Figure 6.4, which illustrates how the direction shifted when the ping noise of 128ms was inserted with the sensitive areas of the wake word. The STE analysis also supported the findings. It showed a clear energy peak in the sensitive regions of the wake word, confirming that the jamming signal was deliberately inserted to disrupt detection. This consistency between the DOA and STE metrics strengthens the conclusion that the LSTM_MCC model correctly identified the attack.

By combining the LSTM_MCC model's attack classification with the results from

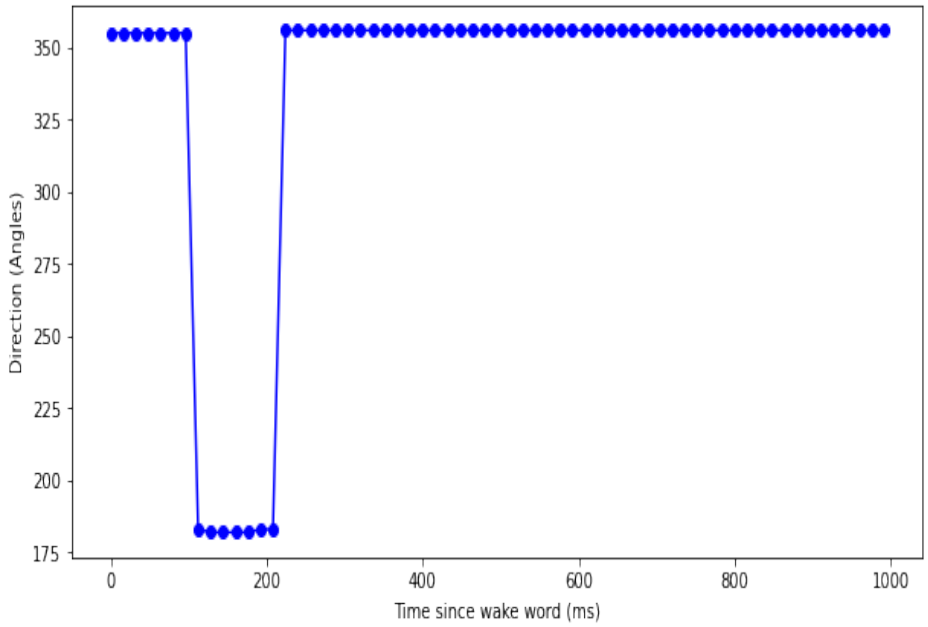


Figure 6.4: Direction of arrival when wake word audio, was mixed with ping noise of 128ms long, displaying a change in the angle of direction in the sensitive area if the wake word.

DOA and STE analyses, the experiment confirms that the model effectively detects and mitigates the attack. However, increasing robustness to jamming can introduce a trade-off where, the model becomes more sensitive to anomalies, which may result in occasional false positives for audio that resembles the wake word but is not an attack. Understanding and mitigating this trade-off represents an important direction for future work,

6.3.5 Experiment Set 5

In Experiment Set 5, a 16ms jamming signal was used to evaluate the model's behaviour. The LSTM_MCC model performed well, correctly identifying the input as a wake word 8 times and as an attack signal 42 times out of 50 trials. This indicates that while the model occasionally classified the input as a legitimate wake word, it mostly detected the presence of the jamming signal and flagged it as an attack. In contrast, the LSTM_WD model struggled in the same scenario. It identified the input as a wake word only 6 times and misclassified it as a non-wake word in the remaining 44 cases. These misclassifications effectively resulted in an attack being successful in 44 out of 50 trials because the device would fail to process the user's command, potentially leading to a DoS attack.

Additional metrics further validated the performance of the LSTM_MCC model. The DOA analysis showed a consistent change in direction between approximately 100ms and 150ms after the wake word started. This timing coincided with the insertion of the jamming signal, indicating that the signal likely came from a different direction than the legitimate wake word. The STE analysis also supported this conclusion. It detected a sudden spike in the audio signal's energy between 100ms and 150ms after the start of the wake word in all 50 trials. Figure 6.3a shows an example of the STE graph when a 16ms jamming signal was introduced at 110ms. The spike in energy aligns perfectly with the timing of the jamming signal, confirming its presence.

The DOA and STE results align with the LSTM_MCC model's predictions, and strongly supports the conclusion that the input audio contained a jamming signal. As a result, the number of successful attacks when using the LSTM_MCC model is zero.

6.3.6 Experiment Set 6

In Experiment Set 6, the duration of the jamming signal is reduced to 2ms. The LSTM_MCC model, designed to detect attacks, classified the input as a wake word 43 times and identified it as an attack signal in 7 cases out of 50 trials. On the other hand, the LSTM_WD model model showed similarly limited impact, correctly detecting the wake word in 41 trials and failing in 9.

This result aligns with the expectation established in Chapter 4 where very short jamming signals are often too brief to meaningfully disrupt wake word detection, especially when they do not sufficiently overlap with the sensitive regions of the wake word. Therefore, these unsuccessful attack attempts do not meaningfully degrade system performance and, in practice, would not warrant raising an alarm. The wake word is still recognized in most cases, and the attack does not significantly interfere with the device's functionality.

For the few cases where the LSTM_MCC model identified the input as an attack, further validation can be performed using additional metrics such as DOA and STE. The DOA results, as noted in the table, indicate a change in the direction of the audio signal, suggesting a possible attack. Similarly, the STE results show a slight energy peak, supporting the possibility of interference. These additional validations can help confirm whether the detected anomaly truly represents an attack.

Overall, while the LSTM_MCC model's attack detection capability is effective even for very short jamming signals, the minimal impact of such attacks suggests that the urgency of raising an alarm may be lower in these cases. However, from a security standpoint, logging or flagging such attack attempts is valuable, as awareness of an attempted attack can inform system monitoring. In cases where the LSTM_MCC model identifies an attack, either the DOA position or the STE peak position can be considered, as both are displaying a possibility of attack as per the results noted in the table.

6.3.7 Experiment Set 7

In Experiment Set 7, the models were tested using only café background noise to evaluate how well they could distinguish between harmless environmental sounds and deliberate jamming signals. This experiment was particularly important for the LSTM_MCC model, as it was trained using various types of noises, including adversarial examples. The goal was to ensure that this training did not cause the model to mistakenly classify random background noises as attacks. The results show that the LSTM_MCC model successfully identified all 50 samples of café background noise as non-wake words. This demonstrates that the model did not confuse benign background noise with jamming signals, even though it had been exposed to noise during training. The absence of false positives in this scenario highlights the model's ability to accurately distinguish between genuine attacks and everyday environmental sounds.

These findings confirm that the adversarial training process effectively prepared the LSTM_MCC model to recognize and respond to jamming signals without compromising its ability to ignore non-threatening noises. This makes the model robust and reliable in real-world environments, where background noise is common and should not trigger unnecessary alarms. However, this experiment does not rule out the possibility that certain naturally occurring noises, especially if impulsive or high-energy, could coincidentally resemble the timing or spectral characteristics of a jamming signal. Investigating whether such "accidental attacks" can arise from everyday background sounds is important.

6.3.8 Experiment Set 8

In Experiment Set 8, the models were tested with non-wake words to evaluate their ability to correctly distinguish these inputs from actual wake words. This

experiment serves as a simple verification to ensure that both models can accurately classify non-wake words under normal conditions. The results show that both the LSTM_WD and LSTM_MCC models successfully identified all 50 samples of non-wake words, correctly classifying them as such.

Continuous jamming attacks were not tested in this set of experiments. Continuous jamming is a straightforward form of interference that can easily disrupt wake word detection and render the system unusable. Such attacks are highly noticeable to users because of their persistent nature and the obvious lack of device responsiveness. For this reason, they are not a focus of this experiment, as they do not offer meaningful insights into the detection capabilities of the models.

6.4 Findings

Our experiments demonstrate that wake word jamming attacks can be effectively detected and mitigated by transforming a wake word detection model through adversarial training into a multi-class classifier. This enhanced model, combined with validation techniques such as DOA and STE, provides a robust framework for identifying and addressing jamming attacks. Some of the key findings in the experiment are:

Transforming the wake word detection model into a multi-class classifier enables it to distinguish between three types of input: wake words, non-wake words, and jamming signals. This transformation makes the model recognize jamming signals that interfere with sensitive regions of the wake word, significantly enhancing its ability to detect and mitigate jamming attacks.

While the multi-class classifier performs well in detecting jamming attacks, it occasionally misclassifies random background noise as a jamming signal. This limitation highlights the need for additional validation techniques to minimize false positives and ensure that alarms are raised only in the presence of genuine threats. Without such validation, unnecessary alerts could undermine the system's reliability.

DOA is a simple and computationally efficient method for verifying jamming attacks, especially for PVAs equipped with microphone arrays. By determining the angle from which sound originates, DOA can detect directional changes that may indicate interference. However, since DOA is sensitive to general

background noise, relying solely on this technique can lead to false positives, particularly in environments with fluctuating sound sources. Although many modern PVAs physically contain multiple microphones, access to the raw multi-channel audio needed for DOA estimation is not always available to developers, which limits its practical use.

STE, which measures changes in signal energy over time, is particularly useful for PVAs without microphone arrays. STE reliably identifies energy peaks caused by abrupt changes in sound, such as jamming signals, and does not trigger false positives from random background noise. This makes it a valuable tool for distinguishing between harmless noise and actual attacks.

Combining STE analysis with the predictions of the multi-class classifier greatly improves the accuracy of attack detection. When STE shows energy peaks in the sensitive regions of the wake word, it confirms the presence of a jamming attack, validating the classifier's output. This layered approach ensures both precision and robustness in detecting adversarial interference. In our experiments, these peaks did not happen by accident, where normal background noises never produced a peak in those regions, and the results matched very well with when jamming was actually present. However, it is still possible, that a real-world sound could sometimes look similar to a jamming pulse. It is an open question, and exploring this remains an important direction for future work.

Through comparative evaluation, STE was shown to be more effective and reliable than DOA for complementing the multi-class classifier. While DOA is helpful in scenarios involving directional analysis, STE is better suited for reducing false positives and ensuring accurate detection in real-world settings. This makes STE a preferred technique for improving detection performance across various devices and environments. This does not mean STE is universally better, but rather that in our specific experimental setup it produced fewer false indications than DOA

Chapter 7

Identifying Sensitive Areas

It is crucial to identify sensitive areas within a wake word that are more vulnerable to jamming attacks for two key reasons. For attackers, this knowledge helps plan and execute a DoS attack, and for developers to design more robust wake word detection algorithms against targeted DoS attacks. the location and characteristics of these sensitive areas depend on factors like audio pre-processing and feature extraction methods, temporal analysis parameters and model architecture. For an attacker who intends to cause DoS attack on the model must have an understanding of what makes a certain region of a wake word more vulnerable to jamming. Identification of sensitive areas is also important to pinpoint the training efforts only in those regions rather than the whole wake word as described in Chapter 5. This will reduce the costs and time involved in adversarial training. However, pinpointing these sensitive areas is not straightforward. For commercial wake words like "Alexa" or "Hey Google", attackers can obtain devices for offline analysis or place recording equipment in environments where these devices operate, making experimentation relatively accessible. However, for custom or proprietary wake words in enterprise or security-critical systems, direct access is not available. Neither attackers nor developers can afford to run extensive experiments to identify them, as this would require significant time, resources, and access to the system. This study aims to bridge that gap by exploring how the structure of a wake word, its energy patterns, and the type of detection model used influence its vulnerability to jamming. Through a series of experiments, we investigated where these sensitive areas are located, how they behave under different jamming conditions, and how the design of the wake word detection algorithm affects its ability to handle interference. Sensitive areas vary depending on factors such as the wake word's

phonetic structure, energy distribution, and the wake word detection model's architecture. Through experiments using the LSTM_WD model, we identified consistent vulnerability patterns. Sensitive areas align with voiced phonetic segments containing phonetic transition boundaries. While specific boundaries vary with model architecture, LSTM_WD model shows sensitivity at significant hidden state transitions. The general correlation between energy concentration, phonetic voicing, and vulnerability appears in implementations. This enables both defensive design strategies and efficient attack planning based on energy and phonetic structure analysis.

While we could not find a mechanism to identify sensitive areas for our tested set of wake words, we were able to develop a set of guidelines that can help predict where these regions likely appear.

7.1 Understanding Sensitive Areas

Our study confirms that sensitive areas are specific regions within a wake word that are more vulnerable to jamming attacks. These areas exist in almost every wake word, but their position within the word can vary depending on multiple factors, including the word itself, the speaker's pronunciation, and the wake word detection model being used.

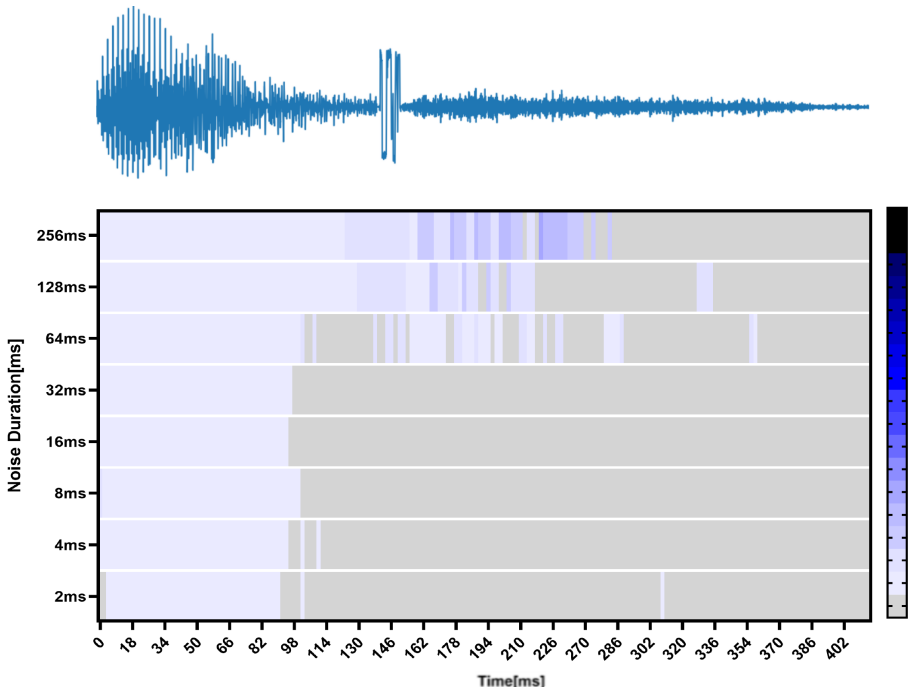
To further analyze these sensitive areas, we examined the results from wake word jamming experiments conducted on multiple wake words. Since most of the wake words tested in our study were multi-syllabic, we selected a monosyllabic wake word, *"House"* to observe how jamming affects words with a simpler structure. For this experiment, we trained the LSTM_WD model using a dataset of 1,690 utterances of the word "House". The original model was trained on non wake words already. After training, the model was tested with 10 previously unseen words that were not included in the training phase to evaluate its ability to differentiate between wake words and non-wake words under jamming conditions. We also trained the MHAtt-RNN model with the same number of samples and tested it with 10 previously unseen words. The word "House" is sourced from the speech commands dataset [War18a].

To systematically analyze the impact of jamming, we used ping noise with a frequency range of 0–8 kHz and tested how it affected wake word recognition across various jamming durations. The results of these tests on two wake word detection algorithms were visualized as heatmaps, as shown in Figure 7.1.

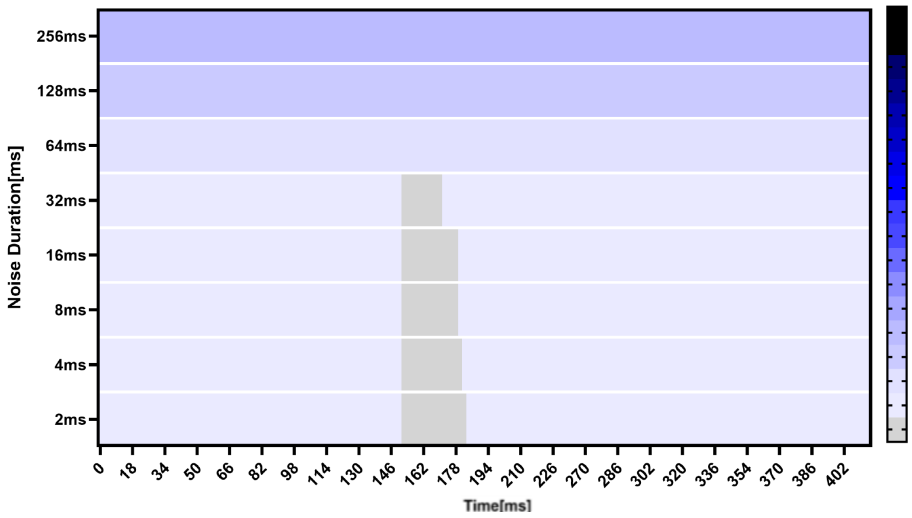
From the heatmap for the LSTM_WD algorithm shown in 7.1a, it is evident that the sensitive areas of the word "House" are concentrated at the beginning of the word. These sensitive regions persist across all tested jamming durations, suggesting that the initial part of the word plays a crucial role in wake word recognition. In contrast, the heatmap for the MHAtt-RNN model shown in 7.1b presents a different pattern, with the effect of the ping noise distributed more evenly across the entire duration of the wake word, regardless of the position of the noise. This could be because, the attention mechanism in the MHAtt-RNN enables the model to focus on multiple parts of the wake word simultaneously. The model dynamically weighs the importance of different parts of the wake word during wake word detection.

Each wake word analyzed so far: *Alexa*, *Marvin*, *Activate*, and *House* has a unique acoustic signature shaped by its syllables, phonemes, and energy distribution. These wake words also vary in duration: *Alexa* (510ms) and *House* (420ms) are relatively short, while *Activate* (630ms) and *Marvin* (580ms) are longer. Duration affects jamming dynamics, for example, shorter wake words provide a narrower attack window, while longer wake words offer attackers more flexibility to target multiple vulnerable regions or sustain jamming across broader temporal spans. The waveforms of these wake words can be observed from 7.2.

For monosyllabic words like *House*, the sensitive areas tend to be present in the early portion of the word. This could be because most of the energy is distributed in the beginning part of the word and the wake word detection model relies on these initial high-energy phonetic features to recognize the word accurately. Once this high-energy region has passed, the model becomes significantly more resistant to jamming, meaning that if a jamming signal is introduced later in the word, it does not disrupt detection as effectively. On the other hand, for multi-syllabic words like *Alexa*, *Marvin* and *Activate*, the sensitive regions are more spread out throughout the word. These words contain multiple phoneme transitions, meaning that acoustic features such as formant frequencies, spectral energy peaks are distributed across different sections. The sensitive regions are often located at syllable transitions or other parts of the word where distinct phonetic changes occur, making those areas more vulnerable to interference. In these cases, an attacker aiming to jam wake word detection would have a wider range of target points compared to a monosyllabic word. This distinction between monosyllabic and multi-syllabic wake words highlights how word structure directly affects jamming vulnerability. The posi-



(a) Jamming using ping noise on the wake word 'House' using the LSTM_WD model



(b) Jamming using ping noise on the wake word 'House' using the MHAtt-RNN model

Figure 7.1: Comparison of the impact of jamming using ping noise on the performance of wake word detection of the word 'House' using the original LSTM_WD and MHAtt-RNN models without adversarial training. WaveForm of House Superimposed with Jamming Signal on the Top

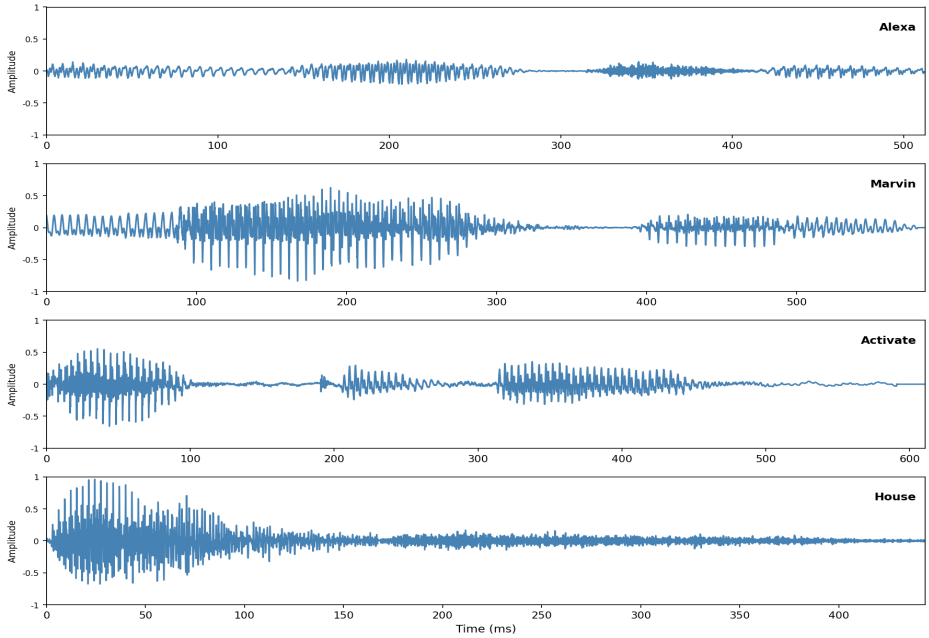


Figure 7.2: Waveforms of the four wake words used in this study. Each subplot shows the amplitude over time (ms) for *Alexa*, *Marvin*, *Activate*, and *House*, illustrating the distinct acoustic signatures and duration differences across wake words

tioning of sensitive regions depends not only on the energy distribution within the word but also on how the wake word detection model processes and prioritizes different phonetic elements for recognition.

7.2 Role of Jamming Coverage

Another important factor to consider while confirming a specific region as sensitive is the jamming coverage. Sensitive regions do exist across all wake words, but the jamming coverage that confirms them might vary. As per our definition of sensitive region explained in Section 5.2.1, it is a region where the jamming is considered successful across all test cases, including different durations of jamming signal and varying power with a jamming coverage threshold of 99%. Our experiments indicate that the identification of these sensitive areas is influenced by two primary factors: the structure of the wake word and the wake word detection model employed. For example, words like *Activate, Alexa, Marvin, and House*, a sensitive area might have jamming coverage close to 99% across different noise durations. However, this strict threshold does not apply to all wake words, as their phonetic structure, energy distribution, and the wake word detection model's processing capabilities can influence jamming vulnera-

bility. For certain wake words, the 99% jamming coverage threshold may be too high, and a lower threshold must be considered, particularly when their structure or energy distribution affects how they respond to jamming. For example, in our experiments, wake word 'Activate' exhibited a peak jamming coverage of 99% compared to 46.78% for 'House'. The 99% threshold was chosen as a conservative starting point to establish high confidence sensitive regions. However, applying it rigidly across all wake words would fail to capture genuine vulnerabilities in words with different acoustic properties. Additionally, the type of jamming signal used influences the jamming coverage percentage. For example, low-intensity noise like drums, animal roar, etc. [Nie04] may not have the necessary characteristics to effectively interfere with wake word detection. If such weak jamming signals are included in the jamming coverage calculation, they may lower the percentage of observed jamming coverage, making it seem as though no sensitive areas exist when, in reality, they do. If the 99% threshold is applied universally, it may result in some wake words appearing to have no sensitive areas at all, which would be misleading. Instead, the threshold should be adjusted dynamically based on factors such as word structure, phonetic transitions, and energy distribution to accurately identify sensitive regions.

As observed in our experiments from chapter 5, multi-syllabic wake words such as Activate, Alexa, and Marvin achieve jamming coverage close to 90%, regardless of their structure or the detection model used. However, for certain monosyllabic words in our corpus, such as House, the sensitive areas tend to be much smaller, sometimes with jamming coverage lower than 50%.In such cases, the decision on what threshold to apply must be carefully considered, ensuring that the identification of sensitive areas remains accurate and meaningful.

7.3 Role of Signal Processing and Wake Word Model

The process of identifying sensitive areas in a wake word is influenced by the interplay between acoustic signal processing and the behavior of the wake word detection model. Signal processing lays the groundwork by extracting key acoustic features, such as energy valleys, phoneme transitions, and temporal boundaries, which naturally create weak spots in the wake word. However, the final detection performance is determined by how the wake word model interprets these features. The model's architecture, whether a Convolutional -

Recurrent neural network (CRNN) or an attention-based model, plays a critical role in shaping its sensitivity to specific regions of the input signal. For example, as shown in Figures 4.15 and 5.6a, the LSTM_WD model, a CRNN-based one processes the input sequentially, resulting in distributed sensitivity across the temporal span of the wake word. In contrast, the MHatt-RNN model uses multi-head attention to selectively weight specific regions, effectively grouping acoustically salient segments and concentrating sensitivity around key phonetic transitions. This difference is reflected in the jamming coverage: the MHatt-RNN model exhibits a lower coverage of 50.78% compared to 85.3% for the baseline, indicating that attention mechanisms consolidate vulnerable regions rather than spreading them across the entire utterance.

7.3.1 Signal Processing

Signal processing techniques are responsible for transforming raw audio into a format suitable for machine learning models. This involves extracting features such as MFCC, spectrograms, or log-mel energies, which capture the acoustic characteristics of the wake word. Although the mel scale is derived from human auditory perception, it remains widely adopted in machine learning based speech processing because it provides an effective compression of the frequency spectrum that emphasizes distinctions relevant for speech recognition tasks [SKMR13]. These features could reveal potential sensitve areas, for example, *Energy Valleys* which are the regions where the signal's energy drops, often corresponding to low-energy consonants like /s/, /f/, /t/, *Phoneme Transitions*, between phonemes, where the acoustic properties of the signal change abruptly and *Temporal Boundaries*. These features depend on the structure of the wake word and can provide a starting point for identifying sensitive areas within a wake word.

7.3.2 Model Behaviour

The architecture of the wake word detection model significantly influences how it processes the acoustic features and where it focuses its attention. Different architectures exhibit distinct behaviors. For example, sequential models such as Recurrent Neural Network (RNN) or CRNN, rely heavily on temporal dependencies and are particularly sensitive to specific time steps in the input signal. This might make the models focus more on the areas where the energy is highest, and those regions are the identification criteria for the models. This would

also make the models more vulnerable to jamming signals if present in those critical regions.

On the other hand, models with attention mechanisms such as MHatt-RNN distribute their focus across multiple regions of the input signal, allowing them to weigh different parts of the wake word dynamically [SZWX18]. While this can make them more robust to localized noise, it also means that their vulnerabilities may be less predictable, as they depend on the learned attention patterns rather than fixed temporal cues.

For an attacker aiming to identify the most vulnerable areas in a wake word without conducting a series of experiments, or for developers seeking to train the model for robustness against jamming attacks, they would need to focus on two key factors. First, they would need to analyze the energy profile of the word, identifying natural dips and transitions where phoneme boundaries or low-energy consonants occur. Second, they would need to understand the model architecture being used, whether it emphasizes sequential processing like RNNs or has a more distributed focus like Mhatt-RNN. By combining insights from both the acoustic signal and the expected model behavior, an attacker could reasonably predict where noise would be most disruptive, even without full access to the system.

Table 7.1: Summary of key findings on sensitive regions in wake words.

Factor	**Finding**
Word structure	Multi-syllabic words have distributed sensitive regions; monosyllabic words have concentrated sensitive regions
Jamming signal	Effectiveness varies depending on the type of jamming signal used
Signal characteristics	Signals with high acoustic energy and spectral overlap with the wake word are more disruptive
Model behaviour	Detection model architecture determines how weak spots manifest as sensitive regions

7.4 Findings

Our study highlights the importance of identifying sensitive areas within wake words, both for understanding how jamming attacks can be executed and for

developing strategies to mitigate such attacks. Through a series of experiments, we explored the factors that influence the location and behavior of these sensitive regions. The findings reveal that the identification of sensitive areas depends on several key factors, including the wake word's energy distribution, syllable structure, and other phonetic properties.

Our experiments conducted in this chapter confirm that sensitive regions exist in all four wake words evaluated. The results demonstrate that for multi-syllabic words, these sensitive areas are more spread out across the word whereas for words that are monosyllabic in nature, the sensitive areas are concentrated wherever the energy distribution is highest. This suggests that the structure of the wake word directly impacts its susceptibility to jamming.

The experiments also reveal that jamming coverage plays a significant role in identifying sensitive regions. The results indicate that while most wake words exhibit jamming coverage close to 90%, monosyllabic words tend to have smaller vulnerable regions, sometimes with jamming coverage below 50%. This variation highlights the need for a dynamic threshold when defining sensitive areas rather than applying a universal 99% coverage threshold.

Then the focus moves to what kind of word can actually be a wake word, most of the commercial Personal Voice Assistant (PVA) use multi-syllabic words as wake words, and monosyllabic words are rarely used as wake words, which could be because of their frequent occurrence in daily conversations, that could lead to unintended activations of the PVA.

The effectiveness of jamming attacks varies significantly depending on the type of jamming signal used. Signals with higher acoustic energy and spectral overlap with the wake word's frequency bands are more disruptive, as they more effectively mask the phonetic features the detection model relies on. In our experiments, ping noise achieved higher jamming success than other types of noises, due to its transient, high energy bursts. Keyboard typing noise, while more naturalistic, showed lower effectiveness due to its irregular temporal patterns and limited spectral coverage. This suggests that the choice of jamming signal is critical for an attacker and should be tailored to the specific wake word and its acoustic properties.

Another key finding is the interplay between signal processing and wake word model behavior in determining jamming vulnerability. The signal processing step reveals natural weak spots within the wake word, such as energy dips and

phoneme transitions, which make certain regions more susceptible to jamming. However, the wake word detection model determines how these weak spots manifest into sensitive regions that make a wake word detection algorithm susceptible to jamming. As future work, we aim to develop a diagnostic tool that automatically identifies sensitive regions for a given wake word, parameterized by the detection model and jamming signal type.

While our experiments provided valuable insights, we must acknowledge that we couldn't pinpoint a universal mechanism to identify sensitive areas across all wake words. However, we were able to establish a set of rules and patterns that can guide both attackers and developers as summarised in Table 7.1. For instance, analyzing energy profiles and understanding model behavior can help predict where sensitive regions are likely to occur. These findings not only help attackers plan more effective jamming strategies but also help developers to focus the training efforts only in the identified sensitive regions of the wake word to make the model robust against a targeted jamming attack.

Chapter 8

Conclusion

8.1 Research Overview

The research explores the security challenges facing Personal Voice Assistants (PVAs), focusing specifically on the vulnerability of wake word detection systems to jamming attacks. PVAs have become very popular in everyday life appearing in smartphones, smart speakers, wearable devices, and even in cars and hospitals due to their hands-free convenience. These devices streamline tasks in diverse fields, such as education, healthcare, and the military. For instance, teachers use voice assistants to automate classroom activities, medical staff rely on them for patient management in hospitals [MLMT20], and the U.S. Army has utilized voice-enabled systems to access critical information on the battlefield [Med19]. Given this broad scope of usage, any disruption to a PVA becomes more than just an inconvenience as it can interrupt essential operations and potentially endanger users.

A DoS attack against a PVA compromises its primary functionality by preventing the device from recognizing wake words, which are essential for initiating any user command. Our research specifically concentrates on how such a DoS attack is carried out through "wake word jamming," a targeted form of interference introduced during the crucial detection phase of a wake word. By masking the acoustic signal, attackers can force the device to stay in its inactive state, effectively denying service.

8.2 Wake Word Jamming

Our experimentation begins by examining how a jamming attack can be designed and deployed to disrupt wake word detection. A wake word detection model typically monitors incoming audio to decide if a specific word like "Alexa," "Activate," or "Marvin" has been spoken. Through extensive experiments, we demonstrate how attackers can exploit this mechanism, using carefully timed "jamming signals" to mislead the model. We identify the following critical factors in executing a successful wake word jamming attack.

- **Jamming Signal Parameters:** Attackers can create a jamming signal with varyingD signal power, duration, and timing to cause the maximum disruption using minimal energy. By placing a short but intense burst of noise at precisely the right moment, an attacker can mask critical audio features of the wake word.
- **Types of Noise:** Although white noise is the most straightforward form of jamming signal, more specialized noises such as "ping" noises may be equally or more effective. Our tests using white noise, ping noise, and keyboard typing noise confirm that various noise types can disrupt wake word detection, provided they are well timed and sufficiently strong.
- **Position of Jamming Signals in Wake Words:** We observe that wake words contain few segments that are vulnerable to interference, we call them *Sensitive Regions*. Injecting jamming signals within these time windows yields greater disruption compared to randomly placed signals.

By systematically varying these factors, we identify how an attacker can render the PVA useless, preventing users from issuing voice commands at crucial moments such as in a hospital setting where reliable voice interactions might be life saving or on a military mission where quick access to data is necessary [Med19, MLMT20].

8.3 Robustness via Adversarial Training

Since wake word detection is the access point of a PVA's functionality, our next step involved strengthening the model against these targeted jamming efforts. We introduced an "adversarial training" approach, where the wake word detection model(LSTM_WD/ Mhatt-RNN) is retrained on datasets that include wake words mixed with jamming signals. By superimposing varying jamming signals

precisely over the identified sensitive regions, we effectively train the system to handle targeted interference.

- **Adversarial Samples Generation:** We create adversarial examples by superimposing jamming signals onto identified sensitive regions of the wake word rather than superimposing them across the whole length of the wake word. This allows the model to learn discriminative features that distinguish legitimate wake words from jammed variants during training using lesser training samples.
- **Different Noise Types for Training:** We find that using a diverse noise set, particularly "babel noise" that includes background chatter, keyboard typing and cutlery clinking. It provides sufficient acoustic variability to generalize to real-world previously unseen jamming signals like "ping" noises or café environments containing both background conversations and ambient sounds (keyboard typing, cutlery clinking), as demonstrated in Chapter 6.
- **Reduction of Jamming Coverage:** Experiments show a significant drop in the "jamming coverage": the proportion of a wake word susceptible to jamming, after adversarial training. The sensitive areas that were previously large, get diminished or even eliminated in the wake word set we tested, because the model now learns to handle interference if present in those areas.

This improved resistance ensures that even if an attacker tries to compromise the wake word at specific intervals, the model can still detect and respond to legitimate user commands. However, adversarial training alone does not address detecting ongoing attacks it simply strengthens the system to remain functional.

8.4 Detecting Ongoing Attacks

While adversarial training improves the robustness of the model, it does not alert users when a jamming attack is happening. Recognizing this, we enhance the system to not only withstand interference but also to identify potential attacks in real time. We transform our binary wake word detector into a three-class classifier, adding a category that identifies "wake word + jamming signal" scenarios. This way, even if the device correctly recognizes the wake word, it can simultaneously detect and alert that a jamming attack is underway.

This additional layer can introduce more false alarms. Therefore, we incorporate two complementary techniques to strengthen the detection reliability:

- **Direction of Arrival (DOA):** PVAs typically use microphone arrays, which enable directional analysis. When the model suspects an attack, DOA can verify if there is an unexpected change in the sound's angle. If the attacker emits noise from a different location than the legitimate user, DoS helps confirm that suspicion.
- **Short Time Energy (STE)** STE checks for abrupt spikes in the audio signal's energy. Typically, random background noises do not produce sharp energy peaks in critical segments of the wake word. If STE reveals a sudden jolt in the sensitive region, it strongly indicates an attack. STE is particularly useful for devices with only one microphone where DOA cannot be applied.

By combining DOA or STE checks with our three class classification system, the number of false positives drops significantly, and actual attacks can be flagged or blocked. This comprehensive detection method makes the system both robust against jamming and capable of alerting users to ongoing threats.

8.5 Identifying Sensitive Areas

In our final part, we analyze why certain parts of a wake word are more vulnerable than others and whether these regions can be predicted without exhaustively testing every possible jamming configuration:

- **Sensitive Regions:** Our results show that each wake word has unique weak spots. For monosyllabic words like "House," the highest energy and defining phonetic elements occur early on, so an attacker's best bet is to jam that first portion. For multi-syllabic words like "Alexa" or "Activate," vulnerability is spread over multiple syllable boundaries. Our analysis focused on a limited set of single-word wake words, systematic investigation of how sensitive region patterns generalize across broader phonetic structures, including multi-word wake phrases like"Hey Google", different linguistic contexts, and various speaker accents, remains an important area for future work.
- **Jamming Coverage Threshold:** We note that not all words can reach a 99% coverage threshold. For some words, significant jamming coverage

might be as low as 50%, especially if the word is short and has only a small interval of vulnerability. Hence, it is crucial to set realistic thresholds for designating a region as sensitive.

- **Model Dependent Impact:** Different wake word architectures like CRNNs or multi-head attention based networks process acoustic features differently. Thus, an attacker would need to also consider the detection model's behavior (or example, whether it focuses heavily on certain time frames or phonemes to precisely identify where a jamming signal will do the most damage.
- **Predicting Vulnerabilities:** By analyzing signal energy distributions and phoneme shifts, alongside knowledge of the model's internals, attackers or defenders can predict where an attack is most likely to succeed.

8.6 Limitations and Future Work

Our work focuses on jamming signals in the audible range (up to about 8 kHz). However, attackers may also exploit ultrasounds above the human hearing threshold for example, above 20 kHz to deliver inaudible jamming signals, often leveraging microphone non linearities. Future research should address this less perceptible but potentially powerful form of jamming.

Our defense mechanisms involve practical trade-offs, for example, multi-layered validation adds processing overhead, and evaluation was limited to four wake words using mel-spectrogram features with LSTM detection. Generalization to broader vocabularies, alternative features (MFCC, raw audio), and different architectures requires validation.

Our experiments were conducted in controlled settings like in indoor spaces without explicit characterization of echo, reverberation, or other room acoustic properties. We did not measure or control for these environmental factors in our experimental design. However, real-world deployments encounter these conditions. These phenomena interact with jamming in complex ways. Future work should investigate defense performance across varying acoustic environments.

Another point for future research involves implementing a user alert mechanism using the device's built-in notification system. While our current work focuses on detecting and mitigating jamming attacks, it does not include an

active method of informing the user about ongoing interference. Future efforts could explore how best to integrate timely alerts such as visual indicators, audio cues, or smartphone notifications to ensure users are instantly informed of potential security threats and can take appropriate action.

Our work has provided valuable insights into identifying sensitive areas within wake words that are vulnerable to the specific jamming attacks investigated in this study. However, a significant limitation we encountered is the inability to precisely pinpoint sensitive areas across all wake words using a universal mechanism.

8.7 Conclusion

In essence, our research investigates the acoustic channel vulnerabilities in PVAs and offers a blueprint for both reinforcing wake word detection against jamming and detecting ongoing attacks. By pinpointing sensitive regions within wake words, exploiting adversarial training, and combining classification outputs with techniques like DOA or STE, we advance the resilience of PVAs. This integrated approach ensures not only that devices remain operational despite malicious interference but also that users can be alerted in a timely manner, preserving the promise of effortless, hands-free voice interaction in even the most demanding environments.

References

[20003] Chapter 12 - signals and data. In D.L. Massart, B.G.M. Vandeginste, S.N. Deming, Y. Michotte, and L. Kaufman, editors, *Chemometrics: a textbook*, volume 2 of *Data Handling in Science and Technology*, pages 157–163. Elsevier, 2003.

[AAA+21] Sadeen Alharbi, Muna S. Al-Razgan, Alanoud Alrashed, Turkiayh Alnomasi, Raghad Almojel, Rimah Alharbi, Saja Alharbi, Sahar Alturki, Fatimah Alshehri, and Maha Almojil. Automatic speech recognition: Systematic literature review. *IEEE Access*, 9:131858–131876, 2021.

[AIM10] Luigi Atzori, Antonio Iera, and Giacomo Morabito. The internet of things: A survey. *Computer Networks*, 54(15):2787–2805, 2010.

[AKC+17] Sercan Ömer Arik, Markus Kliegl, Rewon Child, Joel Hestness, Andrew Gibiansky, Christopher Fougner, Ryan Prenger, and Adam Coates. Convolutional recurrent neural networks for small-footprint keyword spotting. In Francisco Lacerda, editor, *Interspeech 2017, 18th Annual Conference of the International Speech Communication Association, Stockholm, Sweden, August 20-24, 2017*, pages 1606–1610. ISCA, 2017.

[Ana24] Astute Analytica. Voice assistant market. `https://www.astuteanalytica.com/industry-report/voice-assistant-market#:~:text=and%20adoption%20in%20recent%20years,of%20consumer%20preferences%20and%20needs`, 2024. [Accessed 02-12-2025].

[Ana25] IoT Analytics. State of iot 2025. `https://iot-analytics.com/number-connected-iot-devices/`, 2025. [Accessed 04-

12-2025].

[Anh18] Amir Anhari. Alexa dataset, 2018.

[AP17] Efthimios Alepis and Constantinos Patsakis. Monkey says, monkey does: Security and privacy on voice assistants. *IEEE Access*, 5:17841–17851, 2017.

[ARHAU17] Mirza Abdur Razzaq, Sajid Habib, Muhammad Ali, and Saleem Ullah. Security issues in the internet of things (iot): A comprehensive study. *International Journal of Advanced Computer Science and Applications*, 8, 01 2017.

[ASS+15] Hazim Almuhimedi, Florian Schaub, Norman Sadeh, Idris Adjerid, Alessandro Acquisti, Joshua Gluck, Lorrie Cranor, and Yuvraj Agarwal. Your location has been shared 5,398 times! a field study on mobile app privacy nudging. 04 2015.

[BC21] Gideon Buniel and Monalee Dela Cerna. i-detect: An internet of things voice-activated home automation with smoke and fire detection and mitigation system. In *2021 IEEE 13th International Conference on Humanoid, Nanotechnology, Information Technology, Communication and Control, Environment, and Management (HNICEM)*, pages 1–6, 2021.

[Bhu25] Johana Bhuiyan. Apple, 2025.

[BNT19] Meriem Bettayeb, Qassim Nasir, and Manar Abu Talib. Firmware update attacks and security for iot devices: Survey. In *Proceedings of the ArabWIC 6th Annual International Conference Research Track*, ArabWIC 2019, New York, NY, USA, 2019. Association for Computing Machinery.

[Boa25] Boardor. How many devices can an iot gateway connect? `https://boardor.com/blog/how-many-devices-can-an-iot-gateway-connect`, 2025. IoT devices process 10 packets per hour with local processing.

[CBG18] Deepali Chaudhary, Kriti Bhushan, and Brij B Gupta. Survey on ddos attacks and defense mechanisms in cloud and fog computing. *International Journal of E-Services and Mobile Applications*, 10:61–83, 07 2018.

[CBYR18] Peng Cheng, Ibrahim Ethem Bagci, Jeff Yan, and Utz Roedig. Towards reactive acoustic jamming for personal voice assistants. In *Proceedings of the 2nd International Workshop on Multimedia Privacy and Security, MPS@CCS 2018, Toronto, ON, Canada, October 15, 2018*, pages 12–17. ACM, 2018.

[CCK+19] Geumhwan Cho, Jusop Choi, Hyoungshick Kim, Sangwon Hyun, and Jungwoo Ryoo. Threat modeling and analysis of voice assistant applications. In *Information Security Applications: 19th International Conference, WISA 2018, Jeju Island, Korea, August 23–25, 2018, Revised Selected Papers 19*, pages 197–209. Springer, 2019.

[CDL16] Mauro Conti, Nicola Dragoni, and Viktor Lesyk. A survey of man in the middle attacks. *I E E E Communications Surveys and Tutorials*, 18(3):2027–2051, 2016.

[Cen24] Pew Research Center. Mobile fact sheet. `https://www.pewresearch.org/internet/fact-sheet/mobile/`, 2024. [Accessed 27-08-2024].

[CMV+16] Nicholas Carlini, Pratyush Mishra, Tavish Vaidya, Yuankai Zhang, Micah Sherr, Clay Shields, David Wagner, and Wenchao Zhou. Hidden voice commands. In *25th USENIX Security Symposium (USENIX Security 16)*, pages 513–530, Austin, TX, August 2016. USENIX Association.

[CPH14] 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.

[CR22] Peng Cheng and Utz Roedig. Personal voice assistant security and privacy—a survey. *Proceedings of the IEEE*, 110(4):476–507, 2022.

[CS09] E. Chandra and C. Sunitha. A review on speech and speaker authentication system using voice signal feature selection and extraction. In *2009 IEEE International Advance Computing Conference*, pages 1341–1346, 2009.

[Dat23] Dataprot.net. Internet of things statistics for 2024 – taking things apart. https://dataprot.net/blog/iot-statistics/ , 2023. [Accessed 04-12-2025].

[Dat25] Dataconomy. AWS outage: A complete list of every site and app that went down. https://dataconomy.com/2025/10/20/aws-outage-a-complete-list-of-every-site-and-app-that-went-down/, 2025. Amazon Alexa service disruption due to AWS infrastructure failure.

[Dav25] Adam Davidson. 6 useful alexa routines that put echo sound detection to good use. https://www.howtogeek.com/useful-alexa-routines-that-put-echo-sound-detection-to-good-use/, 2025. [Accessed 24-03-2025].

[DDS+19] Praveen Damacharla, Parashar Dhakal, Sebastian Stumbo, Ahmad Y Javaid, Subhashini Ganapathy, David A Malek, Douglas C Hodge, and Vijay Devabhaktuni. Effects of voice-based synthetic assistant on performance of emergency care provider in training. *International Journal of Artificial Intelligence in Education*, 29(1):122–143, 2019.

[Dep20] Department of Homeland Security. Radio frequency interference best practices guidebook. https://www.cisa.gov/resources-tools/resources/radio-frequency-interference-best-practices-guidebook , 2020.

[Dev24] Analog Devices. A beginner's guide to digital signal processing (dsp). https://www.analog.com/en/lp/001/beginners-guide-to-dsp.html#:~:text=Signals%20may%20be%20compressed%20so,and%20video%20via%20telephone%20lines)., 2024. [Accessed 13-09-2024].

[dra25] dragon. Dragon naturallyspeaking. https://www.universityofgalway.ie/information-solutions-services/servicesforstaff/software/softwarelist/software-accessibility/dragonnaturallyspeaking/, 2025. [Accessed 02-12-2025].

[DTI72] DTIC. Speech understanding research. https://apps.dtic.mil/sti/citations/AD0783507, 1972. [Accessed 02-12-2025].

[DV17] Jyoti Deogirikar and Amarsinh Vidhate. Security attacks in iot: A survey. In *2017 International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC)*, pages 32–37, 2017.

[DVG10] N. Q.K. Duong, E. Vincent, and R. Gribonval. Under-determined reverberant audio source separation using a full-rank spatial covariance model. *Trans. Audio, Speech and Lang. Proc.*, 18(7):1830–1840, September 2010.

[eco25] ecobee. Ecobee thermostat. `https://support.ecobee.com/s/articles/Alexa-FAQs-Setup-Use-and-Basic-Troubleshooting`, 2025. [Accessed 04-03-2026].

[EGYY02] Dong Enqing, Liu Guizhong, Zhou Yatong, and Cai Yu. Voice activity detection based on short-time energy and noise spectrum adaptation. In *6th International Conference on Signal Processing, 2002.*, volume 1, pages 464–467 vol.1, 2002.

[Enc24] Encyclopaedia Britannica. Jamming. `https://www.britannica.com/technology/jamming`, 2024. Broadcasting a strong signal that overrides or obscures a target signal.

[Fed24] Federal Trade Commission. How to secure your voice assistant and protect your privacy, 2024.

[FFZ25] Hamidreza Fereidouni, Olga Fadeitcheva, and Mehdi Zalai. Iot and man-in-the-middle attacks. *SECURITY AND PRIVACY*, 8, 03 2025.

[For21] Fortra. Security awareness and working from home: What about devices with personal voice assistants? `https://www.terranovasecurity.com/blog/security-awareness-personal-voice-assistants#:~:text=Voice%20assistants%20are%20mostly%20known,company%20is%20working%20from%20home`, 2021. [Accessed 02-12-2025].

[GDP25] GDPR Advisor. Gdpr and digital personal assistants: Managing voice and text data, 2025.

[GGCPRB+19] Taesik Gong, Alberto Gil C. P. Ramos, Sourav Bhattacharya, Akhil Mathur, and Fahim Kawsar. Audidos: Real-time denial-of-service adversarial attacks on deep audio models. In *2019 18th*

IEEE International Conference On Machine Learning And Applications (ICMLA), pages 978–985, 2019.

[Gil21] Jordan Gillard. Networking considerations in the internet of things. *Medium*, 2021. IoT devices have limited RAM (a few kilobytes) and processing power.

[GKS+18] Jinxi Guo, Ken'ichi Kumatani, Ming Sun, Minhua Wu, Anirudh Raju, Nikko Strom, and Arindam Mandal. Time-delayed bottleneck highway networks using a DFT feature for keyword spotting. In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2018*, pages 5489–5493, Calgary,Canada, 2018. IEEE.

[GLY14] Kanika Grover, Alvin Lim, and Qing Yang. Jamming and anti-jamming techniques in wireless networks: A survey. *International Journal of Ad Hoc and Ubiquitous Computing*, 17(4):197–215, 2014.

[GMS+20] Yixin Gao, Yuriy Mishchenko, Anish Shah, Spyros Matsoukas, and Shiv Vitaladevuni. Towards data-efficient modeling for wake word spotting. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7479–7483, Barcelona,Spain, May 2020.

[Gra] Graphpad. Graphpad prism. `https://www.graphpad.com/features` . [Accessed 04-03-2026].

[GSS14] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

[HBT+19a] Yitao He, Junyu Bian, Xinyu Tong, Zihui Qian, Wei Zhu, Xiaohua Tian, and Xinbing Wang. Canceling inaudible voice commands against voice control systems. In *The 25th Annual International Conference on Mobile Computing and Networking*, MobiCom '19, New York, NY, USA, 2019. Association for Computing Machinery.

[HBT+19b] Yitao He, Junyu Bian, Xinyu Tong, Zihui Qian, Wei Zhu, Xiaohua Tian, and Xinbing Wang. Canceling inaudible voice commands against voice control systems. In *The 25th Annual Inter-*

national Conference on Mobile Computing and Networking, MobiCom '19, New York, NY, USA, 2019. Association for Computing Machinery.

[HGRM23] Shahedul Hasan, Eshatir Radiat Godhuli, Md Shezanur Rahman, and Md Abdullah Al Mamun. The adoption of conversational assistants in the banking industry: is the perceived risk a moderator? *Heliyon*, 9(9):e20220, 2023.

[HLYH21] Qiao Hu, Yuanzhen Liu, Anjia Yang, and Gerhard P. Hancke. Preventing overshadowing attacks in self-jamming audio channels. *IEEE Transactions on Dependable and Secure Computing*, 18(1):45–57, 2021.

[HMKA19] Ahmed Hintaw, Selvakumar Manickam, Shankar Karuppayah, and Mohammed Abomaali. A brief review on mqtt's security issues within the internet of things (iot). *Journal of Communications*, 14:463–469, 06 2019.

[HN22] Noon Hussein and Armstrong Nhlabatsi. Living in the dark: Mqtt-based exploitation of iot security vulnerabilities in zigbee networks for smart lighting control. *IoT*, 3(4):450–472, 2022.

[Hom24] August Home. August smart locks + alexa, 2024.

[IBMa] IBM. (1961) shoebox - ibm archives (78-013). `https://mediacenter.ibm.com/media/(1961)+Shoebox+-+IBM+Archives+(78-013)/0_4m2ynnkk`. [Accessed 27-08-2024].

[IBMb] IBM. Speech recognition. `https://www.ibm.com/history/voice-recognition`. [Accessed 27-08-2024].

[Int22] Internet Freedom Foundation. Privacy of the people: Alexa, google, siri, we hear for you?, 2022.

[IR22] Vlad Ionescu and Utz Roedig. Battery depletion attacks on nb-iot devices using interference. In Sokratis Katsikas, Costas Lambrinoudakis, Nora Cuppens, John Mylopoulos, Christos Kalloniatis, Weizhi Meng, Steven Furnell, Frank Pallas, Jörg Pohle, M. Angela Sasse, Habtamu Abie, Silvio Ranise, Luca Verderame, Enrico Cambiaso, Jorge Maestre Vidal, and Marco Antonio Sotelo Monge, editors, *Computer Security. ESORICS 2021*

International Workshops, pages 276–295, Cham, 2022. Springer International Publishing.

[IS15] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pages 448–456. pmlr, 2015.

[IV19] Philokypros Ioulianou and Vassilios Vassilakis. Battery drain denial-of-service attacks and defenses in the internet of things. *Journal of Telecommunications and Information Technology*, 2:37–45, 07 2019.

[JGL+24] Xiaoyu Ji, Zhang Guoming, Xinfeng Li, Gang Qu, Xiuzhen Cheng, and Wenyuan Xu. Detecting inaudible voice commands via acoustic attenuation by multi-channel microphones. *IEEE Transactions on Dependable and Secure Computing*, PP:1–17, 01 2024.

[JJSL21] C Junu Jahana, MS Sinith, and PP Lalu. Direction of arrival estimation using microphone array. In *2021 Fourth International Conference on Microelectronics, Signals & Systems (ICMSS)*, pages 1–6, 2021.

[Jol02] Ian T. Jolliffe. *Principal Component Analysis*. Springer, New York, NY, 2nd edition, 2002.

[KABM24] Mohamed Riadh Kadri, Abdelkrim Abdelli, Jalel Ben Othman, and Lynda Mokdad. Survey and classification of dos and ddos attack detection and validation approaches for iot environments. *Internet of Things*, 25:101021, 2024.

[Kan22] Kyoung-Don Kang. A review of efficient real-time decision making in the internet of things. *Technologies*, 10:12, 01 2022.

[KD23] Borna Kalhor and Sanchari Das. Evaluating the security and privacy risk postures of virtual assistants, 2023.

[KE21] Rozan Khader and Derar Eleyan. Survey of dos/ddos attacks in iot. *Sustainable Engineering and Innovation*, 3:23–29, 01 2021.

[KKSV17] Constantinos Kolias, Georgios Kambourakis, Angelos Stavrou, and Jeffrey Voas. Ddos in the iot: Mirai and other botnets. *Computer*, 50(7):80–84, 2017.

[KRG22] Mike Krey and Rubén Ramirez Garcia. Voice assistants in healthcare: The patient's perception. In *2022 8th International Conference on Information Management (ICIM)*, pages 120–130, 2022.

[LCR+23] Jingjin Li, Chao Chen, Mostafa Rahimi Azghadi, Hossein Ghodosi, Lei Pan, and Jun Zhang. Security and privacy problems in voice assistant applications: A survey. *Computers & Security*, 134:103448, 2023.

[LGKK19] Jüri Lember, Dario Gasbarra, Alexey Koloydenko, and Kristi Kuljus. Estimation of viterbi path in bayesian hidden markov models. *Metron*, 77:137–169, 2019.

[LKK20] Hyungjun Lim, Younggwan Kim, and Hoirin Kim. Cross-informed domain adversarial training for noise-robust wake-up word detection. *IEEE Signal Processing Letters*, 27:1769–1773, September 2020.

[LKS21] Yanyan Li, Sara Kim, and Eric Sy. A survey on amazon alexa attack surfaces, 2021.

[LL10] Dr. L. Arockiam Lawrence and Vani L. A survey of denial of service attacks and it's countermeasures on wireless network. *International Journal on Computer Science and Engineering*, 2, 08 2010.

[Low76] B. T. Lowerre. *The Harpy speech recognition system*. PhD thesis, Carnegie Mellon University, Pennsylvania, April 1976.

[LQL+19] 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 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS*, pages 11908–11918, Vancouver, Canada, 2019.

[Lug17] Giuseppe Lugano. Virtual assistants and self-driving cars. In *2017 15th International Conference on ITS Telecommunications (ITST)*, pages 1–5, 2017.

[Mal99] Stéphane Mallat. *A Wavelet Tour of Signal Processing*. Academic Press, San Diego, CA, 2nd edition, 1999.

[Mal24] Saadia Malik. Data-driven decision-making: Leveraging the iot for real-time sustainability in organizational behavior. *Sustainability*, 16:6302, 07 2024.

[MCB+21] Mark Mazumder, Sharad Chitlangia, Colby Banbury, Yiping Kang, Juan Manuel Ciro, Keith Achorn, Daniel Galvez, Mark Sabini, Peter Mattson, David Kanter, et al. Multilingual spoken words corpus. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*, 2021.

[Med19] Jenny Medeiros. Here's how the military is using voice technology. 2019.

[Med20] Medium. Understanding the mel spectrogram. `https://medium.com/analytics-vidhya/understanding-the-mel-spectrogram-fca2afa2ce53`, 2020. [Accessed 04-03-2026].

[MG20] Septimiu Mischie and Gabriel Găşpăresc. Respeaker mic array 2.0 for speech processing algorithms. In *2020 International Symposium on Electronics and Telecommunications (ISETC)*, pages 1–4, 2020.

[MGKP09] Aristides Mpitziopoulos, Damianos Gavalas, Charalampos Konstantopoulos, and Grammati Pantziou. A survey on jamming attacks and countermeasures in wsns. *IEEE Communications Surveys & Tutorials*, 11(4):42–56, 2009.

[MLMT20] Simone Montali, Gianfranco Lombardo, Monica Mordonini, and Michele Tomaiuolo. Voice assistants in hospital triage operations. In Roberta Calegari, Giovanni Ciatto, Enrico Denti, Andrea Omicini, and Giovanni Sartor, editors, *Proceedings of the Workshop on 21st Workshop "From Objects to Agents", Bologna, Italy, September 14-16, 2020*, volume 2706 of *CEUR Workshop Proceedings*, pages 147–159. CEUR-WS.org, 2020.

[MSPLDLR22] Arun Sankar Muttathu Sivasankara Pillai, Phillip L. De Leon, and Utz Roedig. Detection of voice conversion spoofing attacks using voiced speech. In Hans P. Reiser and Marcel Kyas, editors, *Secure IT Systems*, pages 159–175, Cham, 2022. Springer International Publishing.

[MVRC21] Umberto Morelli, Ivan Vaccari, Silvio Ranise, and Enrico Cambiaso. Dos attacks in available mqtt implementations: Investigating the impact on brokers and devices, and supported anti-dos protections. In *Proceedings of the 16th International Conference on Availability, Reliability and Security*, ARES '21, New York, NY, USA, 2021. Association for Computing Machinery.

[MWH22] Wolfgang Mack, Julian Wechsler, and Emanuël A.P. Habets. Signal-aware direction-of-arrival estimation using attention mechanisms. *Computer Speech & Language*, 75:101363, 2022.

[MZS+17] Assaf Hurwitz Michaely, Xuedong Zhang, Gabor Simko, Carolina Parada, and Petar S. Aleksic. Keyword spotting for google assistant using contextual speech recognition. In *2017 IEEE Automatic Speech Recognition and Understanding Workshop, ASRU 2017*, pages 272–278, Okinawa, Japan, 2017. IEEE.

[NCPV13] Rajalakshmi Nandakumar, Krishna Chintalapudi, Venkat Padmanabhan, and Ramarathnam Venkatesan. Dhwani: Secure peer-to-peer acoustic nfc. *ACM SIGCOMM Computer Communication Review*, 43, 09 2013.

[Nie04] Sandy Niemann. Helping children who are deaf. `https://en.hesperian.org/hhg/Helping_Children_Who_Are_Deaf:Understanding_sound#:~:text=Lower%2Dpitch%20sounds%20have%20a,m%2Dm%2Dm%22%20have%20a%20low%20pitch.`, 2004. [Accessed 19-02-2025].

[NRM24] Muhammad Nafees, Abid Rauf, and Rabbia Mahum. Automatic spoofing detection using deep learning. *Global Social Sciences Review*, IX:111–333, 03 2024.

[NTS+25] Mamoona Nawaz, Shireen Tahira, Dilawar Shah, Shujaat Ali, and Muhammad Tahir. Lightweight machine learning frame-

work for efficient ddos attack detection in iot networks. *Scientific Reports*, 15(1):24961, 2025.

[oA25] Bank of America. Erica-virtual financial assistant, 2025.

[OAK18] Taekkyung Oh, William Aiken, and Hyoungshick Kim. Hey siri – are you there?: Jamming of voice commands using the resonance effect (work-in-progress). In *2018 International Conference on Software Security and Assurance (ICSSA)*, pages 73–76, 2018.

[OWH19a] Dain Overstreet, Hayden Wimmer, and Rami J. Haddad. Penetration testing of the amazon echo digital voice assistant using a denial-of-service attack. In *2019 SoutheastCon*, pages 1–6, 2019.

[OWH19b] Dain Overstreet, Hayden Wimmer, and Rami J. Haddad. Penetration testing of the amazon echo digital voice assistant using a denial-of-service attack. *2019 SoutheastCon*, pages 1–6, 2019.

[ÖZA19] Selma ÖZAYDIN. Examination of energy based voice activity detection algorithms for noisy speech signals. *Avrupa Bilim ve Teknoloji Dergisi*, pages 157–163, 2019.

[Pau23] Paul. Let's review my house: This is a google smart home in 2023, 2023.

[Pic25] Picovoice. Porcupine wake word. `https://picovoice.ai/platform/porcupine/#:~:text=Porcupine%20Wake%20Word%20is%20a,Tech%20%2D%20but%20with%20superior%20technology.`, 2025. [Accessed 03-12-2025].

[Pig20] Dr. Ir. S. Pigeon. The Babel Noise Generator, 2020.

[PIK11] Konstantinos Pelechrinis, Marios Iliofotou, and Srikanth V. Krishnamurthy. Denial of service attacks in wireless networks: The case of jammers. *IEEE Communications Surveys & Tutorials*, 13(2):245–257, 2011.

[Poi11] Richard A. Poisel. *Modern Communications Jamming Principles and Techniques*. Artech House, Boston, MA, 2nd edition, 2011.

[PS08] John G. Proakis and Masoud Salehi. *Digital Communications*. McGraw-Hill, 5th edition, 2008.

[Rab89] L.R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.

[Ran25] Viral Rang. Smart home security bandwidth requirements: Internet speed guide. `https://viralrang.com/smart-security-internet-requirements`, 2025. IoT sensors: 20-250 kbps; 1080p cameras: 2-3.5 Mbps; 4K cameras: 5-10 Mbps.

[RJ22] Ludan Ruan and Qin Jin. Survey: Transformer based video-language pre-training. *AI Open*, 3:1–13, 2022.

[RKS+20] Oleg Rybakov, Natasha Kononenko, Niranjan Subrahmanya, Mirkó Visontai, and Stella Laurenzo. Streaming keyword spotting on mobile devices. In Helen Meng, Bo Xu, and Thomas Fang Zheng, editors, *INTERSPEECH 2020, 21st Annual Conference of the International Speech Communication Association, Virtual Event*, pages 2277–2281, Shanghai, China, October 2020.

[RMF23] Shafiq Rehman, Selvakumar Manickam, and Nur Firdous. Impact of dos/ddos attacks in iot environment: A study. page 020020, 01 2023.

[RS07] Lawrence R. Rabiner and Ronald W. Schafer. Introduction to digital speech processing. *Foundations and Trends® in Signal Processing*, 1(1–2):1–194, 2007.

[RSAP21] Ribil Mary Roy, Blessy Sabu, Nisha A, and Aneesh R P. Voice controlled home automation system. In *2021 Seventh International conference on Bio Signals, Images, and Instrumentation (ICBSII)*, pages 1–6, 2021.

[Sch20] Julian Schwieren. Conversing with automated cars : Exploring voice assistants for trustful user interactions with fully automated vehicles, June 2020.

[SCR+20] 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. USENIX Association, August 2020.

[SCZ20] Ke Sun, Chen Chen, and Xinyu 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, 2020*, pages 298–311, New York,United States, 2020. ACM.

[SDJ09] Ervin Sejdić, Igor Djurović, and Jin Jiang. Time–frequency feature representation using energy concentration: An overview of recent advances. *Digital Signal Processing*, 19(1):153–183, 2009.

[SDSV20] Kalyanam Supriya, Anemu Divya, Gedala Ram Sai, and Balaga Vinodkumar. Trigger word recognition using lstm. 09, 2020.

[Sec24] Secured Data Recovery. Listening in: Privacy concerns of voice assistants, 2024.

[SGS+15] Gaurav Somani, Manoj Gaur, Dheeraj Sanghi, Mauro Conti, and Rajkumar Buyya. Ddos attacks in cloud computing: Issues, taxonomy, and future directions. *Computer Communications*, 107, 12 2015.

[SHK+14] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958, 2014.

[SHRL20] Emre Sezgin, Yungui Huang, Ujjwal Ramtekkar, and Simon Lin. Readiness for voice assistants to support healthcare delivery during a health crisis and pandemic. *NPJ Digital Medicine*, 3(1):122, 2020.

[Sir17] Team Siri. Hey siri: An on-device dnn-powered voice trigger for apple's personal assistant. *Apple Machine Learning Journal*, 1(6), 2017.

[SJT+21] Filipo Sharevski, Peter Jachim, Paige Treebridge, Audrey Li, Adam Babin, and Christopher Adadevoh. Meet malexa, alexa's malicious twin: Malware-induced misperception through intelligent voice assistants. *International Journal of Human-Computer Studies*, 149:102604, 01 2021.

[SKMR13] Tara N. Sainath, Brian Kingsbury, Abdel-rahman Mohamed, and Bhuvana Ramabhadran. Learning filter banks within a deep neural network framework. In *IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, pages 297–302, Olomouc, Czech Republic, 2013. IEEE.

[SKZ+18] Lea Schönherr, Katharina Kohls, Steffen Zeiler, Thorsten Holz, and Dorothea Kolossa. Adversarial attacks against automatic speech recognition systems via psychoacoustic hiding. *arXiv preprint arXiv:1808.05665*, 2018.

[Smi] Dale Smith. 'alexa' isn't the only thing your amazon echo can listen for. here's what else. https://www.cnet.com/home/smart-home/alexa-isnt-the-only-thing-your-amazon-echo-can-listen-for-heres-what-else/.

[Soc] IEEE Signal Processing Society. Signal processing 101. `https://signalprocessingsociety.org/our-story/signal-processing-101` . [Accessed 24-09-2024].

[Sol24] UL Solutions. Unprotected firmware puts iot devices at greater risk. `https://www.ul.com/news/unprotected-firmware-puts-iot-devices-greater-risk`, 2024. [Accessed 04-12-2025].

[S.p] Eidosmedia S.p.A. The growing power of voice search - | Eidosmedia — eidosmedia.com. `https://www.eidosmedia.com/updater/technology/The-Emerging-Power-of-Voice-Assistants-and-Voice-Search#:~:text=Oberlo%20reports%2050%25%20of%20the,assistants%20in%20use%20by%202024.` [Accessed 10-06-2024].

[SSR24] Prathyusha Sagi, Arun Sankar, and Utz Roedig. Personal voice assistant wake word jamming. In *2024 IEEE International Conference on Pervasive Computing and Communications Workshops and other Affiliated Events (PerCom Workshops)*, pages 19–24, 2024.

[ST20] Khairunisa Sharif and Bastian Tenbergen. Smart home voice assistants: A literature survey of user privacy and security vulner-

abilities. *Complex Systems Informatics and Modeling Quarterly*, 0:15–30, 10 2020.

[Sta25] Amazon Staff. 50 things to try with alexa+, 2025.

[STE22] STEPHAN. The Babel Noise Generator, 2022.

[STIM18] Shibani Santurkar, Dimitris Tsipras, Andrew Ilyas, and Aleksander Madry. How does batch normalization help optimization? *Advances in neural information processing systems*, 31, 2018.

[Sup24] NetSim Support. Modelling ddos attack in iot network in netsim. `https://support.tetcos.com/support/solutions/articles/14000141596`, 2024. Demonstrates 25-50% throughput degradation with 1-3 attackers at 20 packets/sec.

[SWW15] Ahmad-Reza Sadeghi, Christian Wachsmann, and Michael Waidner. Security and privacy challenges in industrial internet of things. In *Proceedings of the 52nd Annual Design Automation Conference*, DAC '15, New York, NY, USA, 2015. Association for Computing Machinery.

[SYO+18] Sining Sun, Ching-Feng Yeh, Mari Ostendorf, Mei-Yuh Hwang, and Lei Xie. Training augmentation with adversarial examples for robust speech recognition. *arXiv preprint arXiv:1806.02782*, 2018.

[SZWX18] Changhao Shan, Junbo Zhang, Yujun Wang, and Lei Xie. Attention-based end-to-end models for small-footprint keyword spotting, 2018.

[TGCEKK20] Ozan Topal, Selen Gecgel Cetin, Ender Eksioglu, and Gunes Karabulut Kurt. Identification of smart jammers: Learning-based approaches using wavelet preprocessing. *Physical Communication*, 39:101029, 04 2020.

[TJ18] Li Tan and Jean Jiang. *Digital signal processing: fundamentals and applications*. Academic press, 2018.

[TK09] Sergios Theodoridis and Konstantinos Koutroumbas. *Pattern Recognition*. Academic Press, Burlington, MA, 4th edition, 2009.

[Tor18] Don Torrieri. *Principles of Spread-Spectrum Communication Systems*. Springer, 4th edition, 2018.

[TS20] George TERZOPOULOS and Maya SATRATZEMI. Voice assistants and smart speakers in everyday life and in education. *Informatics in Education*, 19(3):473–490, 2020.

[USL20] Sahar Ujan, Mohammad Same, and Rene Landry. A robust jamming signal classification and detection approach based on multi-layer perceptron neural network. 03 2020.

[W+21] Yiming Wang et al. *Wake word detection and its applications*. PhD thesis, Johns Hopkins University, 2021.

[War18a] P. Warden. Speech Commands: A Dataset for Limited-Vocabulary Speech Recognition. *ArXiv e-prints*, April 2018.

[War18b] Pete Warden. Speech commands: A dataset for limited-vocabulary speech recognition. *CoRR*, abs/1804.03209, 2018.

[WEK+15] Zhizheng Wu, Nicholas Evans, Tomi Kinnunen, Junichi Yamagishi, Federico Alegre, and Haizhou Li. Spoofing and countermeasures for speaker verification: A survey. *Speech Communication*, 66:130–153, 2015.

[WPS+18] Minhua Wu, Sankaran Panchapagesan, Ming Sun, Jiacheng Gu, Ryan Thomas, Shiv Naga Prasad Vitaladevuni, Björn Hoffmeister, and Arindam Mandal. Monophone-based background modeling for two-stage on-device wake word detection. In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2018*, pages 5494–5498, Calgary,Canada, 2018. IEEE.

[XTZW05] Wenyuan Xu, Wade Trappe, Yanyong Zhang, and Timothy Wood. The feasibility of launching and detecting jamming attacks in wireless networks. In *Proceedings of the 6th ACM International Symposium on Mobile Ad Hoc Networking and Computing*, pages 46–57, 2005.

[YCZ+18] Xuejing Yuan, Yuxuan Chen, Yue Zhao, Yunhui Long, Xiaokang Liu, Kai Chen, Shengzhi Zhang, Heqing Huang, XiaoFeng Wang, and Carl A. Gunter. CommanderSong: A systematic approach for practical adversarial voice recognition. In *27th USENIX Secu-*

rity Symposium (USENIX Security 18), pages 49–64, Baltimore, MD, August 2018. USENIX Association.

[YJW+22] Chen Yan, Xiaoyu Ji, Kai Wang, Qinhong Jiang, Zizhi Jin, and Wenyuan Xu. A survey on voice assistant security: Attacks and countermeasures. *ACM Comput. Surv.*, 55(4), nov 2022.

[YLZ+20] Qiben Yan, Kehai Liu, Qin Zhou, Hanqing Guo, and Ning Zhang. Surfingattack: Interactive hidden attack on voice assistants using ultrasonic guided waves. 01 2020.

[YZ22] Qiuchen Yu and Ruohua Zhou. Wake word detection based on res2net. *arXiv preprint arXiv:2209.15296*, 2022.

[ZCL+18] Rongjunchen Zhang, Xiao Chen, Jianchao Lu, Sheng Wen, Surya Nepal, and Yang Xiang. Using ai to hack ia: A new stealthy spyware against voice assistance functions in smart phones, 2018.

[ZLC+19] Xiao Zhang, Jiqiang Liu, Si Chen, Yongjun Kong, and Kui Ren. Priwhisper+: An enhanced acoustic short-range communication system for smartphones. *IEEE Internet of Things Journal*, 6(1):614–627, 2019.

[ZMF+19] Nan Zhang, Xianghang Mi, Xuan Feng, XiaoFeng Wang, Yuan Tian, and Feng Qian. Dangerous skills: Understanding and mitigating security risks of voice-controlled third-party functions on virtual personal assistant systems. In *2019 IEEE Symposium on Security and Privacy (SP)*, pages 1381–1396, 2019.

[ZSLC17] Yundong Zhang, Naveen Suda, Liangzhen Lai, and Vikas Chandra. Hello edge: Keyword spotting on microcontrollers. *arXiv preprint arXiv:1711.07128*, 2017.

[ZXZ+15] Shengkui Zhao, Xiong Xiao, Zhaofeng Zhang, Thi Ngoc Tho Nguyen, Xionghu Zhong, Bo Ren, Longbiao Wang, Douglas L. Jones, Eng Siong Chng, and Haizhou Li. Robust speech recognition using beamforming with adaptive microphone gains and multichannel noise reduction. In *2015 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, pages 460–467, 2015.

[ZYJ+17] 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, CCS 2017*, pages 103–117, Dallas, USA, 2017. ACM.