CORA Cork Open Research Archive
Cartlann Taighde Oscailte Chorcaí

Title	Acoustic sensing for voice assistants
Authors	Pham, Hoang Quoc Viet
Publication date	2025-05-31
Original Citation	Pham, H. Q. V. 2025. Acoustic sensing for voice assistants. MRes Thesis, University College Cork.
Type of publication	Masters thesis (Research)
Rights	© 2025, Hoang Quoc Viet Pham. - https://creativecommons.org/licenses/by-sa/4.0/
Download date	2026-11-05 17:55:58
Item downloaded from	https://hdl.handle.net/10468/18798

UCC
University College Cork, Ireland
Coláiste na hOllscoile Corcaigh

Acoustic Sensing for Voice Assistants

Hoang Quoc Viet Pham
123121186

Thesis submitted for the degree of
Master of Science



NATIONAL UNIVERSITY OF IRELAND, CORK

School of Computer Science and Information Technology

May 2025

Head of School: Professor Dirk Pesch

Supervisor: Professor Utz Roedig
Dr. Harry Nguyen

Contents

List of Figures . . . iii
List of Tables . . . v
List of Acronyms . . . vi
Acknowledgements . . . viii
Abstract . . . ix

1 Introduction **1**
1.1 Motivation and Research Questions . . . 2
1.2 Problem Statement . . . 3
1.3 Research Contribution . . . 4
1.4 Publications . . . 5
1.5 Outline . . . 5

2 Background and Related Work **6**
2.1 Physical Security Intruder Detection . . . 6
2.2 Smart Speaker and Acoustic Sensing . . . 7
2.2.1 Smart Speaker . . . 7
2.2.2 Acoustic Sensing Mechanism . . . 8
2.3 Human Presence Detection - Passive Sensing . . . 9
2.4 Human Presence Detection - Active Sensing . . . 10
2.5 Anomaly Detection . . . 11
2.5.1 Autoencoder . . . 12
2.6 Environment Change Adaptation . . . 14
2.6.1 Transfer Learning . . . 14
2.7 Summary of Literature Review . . . 15

3 Human Presence Detection by AAS **16**
3.1 Threat Model . . . 16
3.2 GOTCHA System . . . 17
3.2.1 Assumptions . . . 17
3.2.2 Application Scenario . . . 18
3.2.3 Sensing Signal . . . 18
3.2.4 GOTCHA Autoencoder Processing Chain . . . 19
3.2.5 GOTCHA Prototype . . . 22
3.3 GOTCHA Evaluation . . . 23
3.3.1 Experiment Setup . . . 25
3.3.2 Experiments and Dataset . . . 26
3.3.3 Comparison and Metrics . . . 27
3.3.4 Hyper Parameters Tuning . . . 28
3.3.5 Data Analysis . . . 29
3.3.6 Detection Performance . . . 30
3.3.7 Discussion . . . 34
3.4 Summary . . . 35

4 GOTCHA System Adaptation **36**

4.1 The Need for Adaptation: Motivating Robust Performance in GOTCHA 36
4.1.1 Intra-room Environmental Adaptation 37
4.1.2 Cross-room Environmental Adaptation 38
4.2 Model Adaptation for GOTCHA 40
4.3 Environmental Adaptation Evaluation 42
4.3.1 Experiment Design 42
4.3.2 Experiment Results 44
4.3.3 Discussion 55
4.4 Summary 55

5 Conclusion and Future Work **56**
5.1 Conclusion 56
5.2 Future Work 58

List of Figures

2.1 The Autoencoder network architecture: Encoder part, which maps the input x into a latent representation z, and Decoder parts, which reconstructs z back into the original input space. 13

3.1 GOTCHA processing chain. (1) Audio signal (2) Recording on N microphones (3) Signal transformation into Spectogramms (4) Combining inputs into a single input matrix (5) Autoencoder processing (6) Reconstructed output matrix (7) Reconstruction error ϵ_x computation (8) z_score computation (9) Threshold τ comparison (10) Decision output (11) Windowing (12) Final decision output. 19
3.2 An example of input signal and reconstructed spectrogram of 4 microphone channels for an example (Output of the processing chain Step (6) as shown in Fig 3.1). The reduced noise in the reconstructed output is clearly visible. 22
3.3 GOTCHA prototype design and microphone array structure. . . 23
3.4 The environment of 3 rooms used in the experiments: room-1 and room-2 are typical office rooms furnished with desks, chairs, and cabinets, while room-3 is a standard bedroom equipped with a bed, curtains, and more objects. 24
3.4 Room layout and regions with possible human presence. In white regions, a human subject can be located; grey cells represent obstacles (bed, cabinet, table, etc.). 25
3.5 Hyper-parameter tuning for the selection of μ, σ and τ in step (8) and step (9) in the signal processing chain in room-1 speaker position-1 as depicted in Fig. 3.1. 28
3.6 Effects of windowing method on 6 cases (3 rooms with 2 speaker positions of each room. r1, r2 and r3 stand for room-1, room-2 and room-3 while p1 and p2 stand for speaker position-1 and 2, respectively). 31
3.7 Accuracy of detecting a human at different positions in the room after windowing. The black figures show accuracy for speaker at position-1 and the red figures show accuracy for speaker at position-2. 32
3.8 F1-score on testing set of 3 rooms at speaker position-1 when using different sizes of training set. 34

4.1 The process of adaptation with environmental changes happening within a room. It is required data recollecting and model retraining for every single new position. 38
4.2 The process of adaptation with environmental changes across rooms. It is required data recollecting and model retraining for every single new room. 39

4.3 The chart compares the F1-score with confidence intervals across different training set sizes. The blue line represents the method of retraining from scratch, the orange line corresponds to the method of fine-tuning the best model of initial room on new data, and the green line indicates our proposed approach — fine-tuning only the decoder part of the best model of initial room on new data. (Part 1). 46
4.3 The chart compares the F1-score with confidence intervals across different training set sizes. The blue line represents the method of retraining from scratch, the orange line corresponds to the method of fine-tuning the best model of initial room on new data, and the green line indicates our proposed approach — fine-tuning only the decoder part of the best model of initial room on new data. (Part 2). 47
4.4 The chart compares the False positive rate with confidence intervals across different training set sizes. The blue line represents the method of retraining from scratch, the orange line corresponds to the method of fine-tuning the best model of initial room on new data, and the green line indicates our proposed approach — fine-tuning only the decoder part of the best model of initial room on new data. (Part 1). 50
4.4 The chart compares the false positive rate with confidence intervals across different training set sizes. The blue line represents the method of retraining from scratch, the orange line corresponds to the method of fine-tuning the best model of initial room on new data, and the green line indicates our proposed approach — fine-tuning only the decoder part of the best model of initial room on new data. (Part 2). 51
4.5 The chart compares the execution time of retraining process with confidence intervals across different training set sizes. The blue line represents the method of retraining from scratch, the orange line corresponds to the method of fine-tuning the best model of initial room on new data, and the green line indicates our proposed approach — fine-tuning only the decoder part of the best model of initial room on new data. (Part 1). 53
4.5 The chart compares the execution time of retraining process with confidence intervals across different training set sizes. The blue line represents the method of retraining from scratch, the orange line corresponds to the method of fine-tuning the best model of initial room on new data, and the green line indicates our proposed approach — fine-tuning only the decoder part of the best model of initial room on new data. (Part 2). 54

List of Tables

2.1 Summary of related works in acoustic active sensing for human presence detection task. Room size is categorized as large (lecture hall - 9(m)x6(m)), medium (common living room - 5(m)x5(m)) . 10

3.1 Datasets summarization (r1, r2 and r3 stand for room-1, room-2 and room-3 while p1 and p2 stand for speaker position-1 and 2, respectively) 27

3.2 F1-score and False Positive Rate (FPR) on testing set of 6 experiments (r1, r2 and r3 stand for room-1, room-2 and room-3 while p1 and p2 stand for speaker position-1 and 2, respectively) . . . 30

List of Acronyms

DNN Deep Neural Network
FPR False Positive Rate
TPR True Positive Rate
IoT Internet of Things
LSTM Long Short-Term Memory
STFT Short-Time Fourier Transform
TOA Time-of-Arrival
IDS Intrusion Detection System
EER Equal Error Rate
VA Voice Assistant
DOA Direction of Arrival
AE Autoencoder
RF Radio Frequency
IR Infrared
ML Machine Learning
MSE Mean Square Error

This is to certify that the work I, Hoang Quoc Viet Pham, am submitting is my own and has not been submitted for another degree, either at University College Cork or elsewhere. All external references and sources are clearly acknowledged and identified within the contents. I have read and understood the regulations of University College Cork concerning plagiarism and intellectual property.

Hoang Quoc Viet Pham

Acknowledgements

To complete this work, I would like to express my sincere gratitude to my supervisors, Prof. Utz Roedig and Dr. Harry Nguyen, who have given me the opportunity to undertake this research, and have provided invaluable guidance and support throughout the development of this thesis. Their insights and encouragement have not only shaped my academic thinking but also deeply influenced the way I approach problems in life.

I would also like to thank my parents, who have always cared for and supported me from afar in my homeland, Vietnam. In addition, I want to express my deep appreciation to my friends back home in Vietnam, whose encouragement and kind words have kept me going forward.

My heartfelt thanks go to my friends and colleagues in lab 2.18 at WGB, as well as members of the ReliableAI group, for their companionship and support during the entire journey of completing this thesis in Ireland.

Finally, I gratefully acknowledge the financial support from the Science Foundation Ireland (SFI). This work has been carried out with the support of SFI under Grant numbers 19/FFP/6775 and 13/RC/2077_P2.

Abstract

Recent technological breakthroughs, such as AI assistants, smart devices, and IoT, are significantly enhancing the quality of human life. Over time, the commercialization of smart devices at affordable prices has made these applications more accessible, allowing users to integrate them into their daily lives more widely. Voice Assistants (VAs) in the form of smart speakers such as Amazon Alexa, Google Assistant, or Apple Siri are now commonplace. With their increasing presence in households, smart devices are becoming more versatile. Beyond serving as virtual assistants, these devices can be leveraged to create various useful applications. Transforming a smart speaker into a home alarm system allows end users to utilize their existing hardware as a valuable intrusion detection system without incurring additional costs beyond the smart speaker itself.

In this work, we present the development and evaluation of a novel physical intrusion detection system GOTCHA based on human presence detection with active acoustic detection. GOTCHA can execute on simple off-the-shelf smart speaker hardware. Periodic audible chirps are employed to gather data that are then processed by a deep autoencoder trained on the acoustic profile of the empty room. As our evaluation shows, GOTCHA achieves promising results of up to 99. 2% for the F1 score. Our experiments show that a person can be detected without fail while the system barely generates false alarms. GOTCHA is a viable alternative to passive detection solutions for the detection of physical intrusion.

Moreover, ensuring that GOTCHA can quickly adapt to changes in its surrounding environment is a critical challenge. We conducted experiments by retraining GOTCHA when transitioning from one background environment to another and found that with just 100 samples in the training set, the system was able to achieve an F1-score above 90% while maintaining an acceptable false alarm rate. Additionally, by applying transfer learning, the retraining time for GOTCHA was significantly reduced compared to fully retraining the entire model. This demonstrates that GOTCHA is entirely feasible for real-world deployment.

Chapter 1

Introduction

With the development of novel technology such as Internet of things and machine learning, smart home applications are becoming increasingly popular and having more impact to daily life. Integrating sensors and smart devices in the house enhances users experience and help users automatically handle daily task such as turning on/off light, open/close the doors, etc. As a result, people can improve their life quality and better optimize their time. In line with that trend, VA such as Amazon Alexa "Amazon" [n.d], Google Assistant "Google" [n.d] or Apple Siri "Apple" [n.d] are now commonplace. People can naturally interact with VA via voice and VA also can interact with other digital infrastructure and services. For example, VA are used to read out email, to provide weather forecasts or to interact with smart home devices. VA often come in the form of services that are integrated in devices such as phone, laptop, ... but the most common form is standalone devices referred to as smart speakers. As smart speakers are sold at very competitive prices, they can now be found in nearly every home. According to the Irish times Times [2023], almost half of adults in Ireland own a smart speaker. Meanwhile, smart speaker market size is forecasted over USD 13 billion by 2024 according to a new research report by Global Market Insights Insights [2018]. These figures are showing that smart speakers are increasingly emerging and having big impact to daily life in digital era.

Due to the popularity of smart speakers, users and VA providers are looking for novel ways of using them to maximise their utility. Many applications have been developed to utilize smart speakers as virtual assistants; however, there are still numerous potential applications of smart speakers that have yet to be explored such as alarm devices, voice control devices, monitoring devices, etc. Among these application domains, one that has been recently emerging is home security

domain. Turning an available smart speaker into a home security device can be a potentially cost-effective solution. The use of smart speakers as a home alarm system for intruder detection is considered. The underlying idea here is to analyse sound recorded by the smart speaker when an intruder enters the house or the room. In a simple setup, any sound above a set threshold will raise the alarm, as an empty house should be quiet. More sophisticated setups analyse the sound and perform classification to only raise the alarm when sounds attributed to human presence are detected (e.g. closing a door, footsteps, speech). Moreover, to be more feasible in real-world scenarios, the home security devices transformed from a smart speaker must also be capable of adapting to environmental changes as they occur. Regardless of environmental changes, the home security device must be able to quickly adapt to maintain reliable operation and minimize the impact of surrounding variations. We propose a low-cost hardware-based solution to the physical intrusion detection problem using active acoustic sensing. However, the term physical intrusion detection broadly encompasses various types of intruders, including humans, animals, or robots. Given the scope of this thesis, we are unable to cover all such cases. We limit our focus to human intruders, which are the most common in real-world scenarios. Specifically, we consider the problem of human presence detection as a proxy to demonstrate the effectiveness of our proposed method in addressing the broader task of physical intrusion detection. Therefore, throughout this thesis, the terms physical intrusion detection and human presence detection will be used interchangeably.

1.1 Motivation and Research Questions

Currently, on the market, surveillance and intruder detection devices often come in the form of camera-based devices. With breakthroughs in computer vision technology, camera-based approaches have proven to be effective in intruder detection problem. However, camera-based approaches also raise concerns about privacy and they also face difficulties in blur-light conditions as well as obstacle appearance. Therefore, there are some other approaches proposed to address these issues such as RF-based or acoustic-based approaches. Among these methods, acoustic-based solutions are emerging as potential option with cost-effectiveness and ease of setup for end users. Acoustics-based approaches are often divided into acoustic passive sensing and acoustic active sensing. Acoustic passive sensing focus on hearing the sound and do sound classification to detect closing door sound, footsteps, human movement, etc. There are many types of sound that can occur,

so acoustic passive sensing often requires a large amount of data with annotation. Moreover, acoustic passive sensing also has difficulty dealing with gentle movements that only emit low level of loudness. Contrary to above approach, acoustic active sensing actively emits sound and analyse the reverberation to detect abnormal state. However, this approach has been explored less and existing works often use dedicated hardware (e.g. ultra-sound transducers, microphone arrays) to analyse sound signal by statistic analysis and traditional machine learning. These approaches requires some background knowledge to effectively tune the model when the environment changes. Generally, this thesis will answer 2 main research questions proposed as follows:

- Q1: What approach can be used to design a human presence detection algorithm leveraging active acoustic sensing that is suitable for implementation on affordable hardware for alarm systems ?
- Q2: How efficiently can a human presence detection system adapt to environmental changes with minimal retraining effort for human presence detection ?

To answer the two above mentioned research questions, this thesis has two main objectives. The first objective is to build an end-to-end system that utilizes low-cost device to construct a smart speaker prototype to detect human presence in the room. This aims to develop an approach to reduce the cost while maintaining high efficiency. The second objective is to propose a novel way to make the human presence detection system quickly adapted when the environment change without requiring much effort in retraining.

1.2 Problem Statement

Physical intruder detection is the task that raises an alarm when an object intrudes into an area or space where it is not permitted. Detecting intruder plays an important role in security systems. With the recent development of smart devices and IoT technology, intruder detection by smart devices such as camera-based, RF-based systems are increasingly popular. However, above-mentioned methods have some drawback about privacy or require costly devices to set up. Using acoustic signal is becoming potential approach to deal with those problem of camera-based and RF-based existing approaches. Especially, active acoustic sensing is an approach that can detect intruder quickly with easy set up for end users. A system of active acoustic sensing for physical intruder detection can be

described as follow: a sound signal will be continuously emitted and reverberation will be collected by microphones, after that sound signal will be analysed to detect anomalous points which is considered as potential signal of an intruder enter the area. Let $x \in \mathbb{R}^{n \times L}$ be the input acoustic signal of n channels with the length L, f is the anomaly detection model and ϵ is a threshold to identify anomalous samples. The problem can be formulated as follow:

$$anomaly_score = f(x) \tag{1.1}$$

$$y = \begin{cases} 1, & \text{if } anomaly_score \geq \epsilon \\ 0, & \text{otherwise} \end{cases} \tag{1.2}$$

The anomaly detection model transforms the input x into an anomaly score, which is then compared with a threshold to determine whether the sample is an anomalous point or not. In this context, $y = 1$ indicates that the sample is anomalous, corresponding to a detected intruder. Conversely, $y = 0$ indicates that the sample is normal, representing the room in a normal state with no intruder detected.

1.3 Research Contribution

In this thesis, there are four main contributions:

- A novel end-to-end design to detect physical human presence is proposed. It employs an autoencoder based anomaly detection using audio spectrograms from recording acoustic chirps. It also enables implementation on simple commodity smart speaker hardware.
- We conduct experiments to evaluate GOTCHA in three environments and show that 99.2%, 99.0% and 91.4% F1-score on the testing set can be achieved respectively with low false alarms rate making GOTCHA a practical system.
- An audio dataset including recordings of chirp responses in two cases (empty and occupied rooms) is publicly released. This dataset enables other researchers to validate and reproduce our results and also enable further work such as bench-marking of alternate detection algorithms or improved training for human presence detection by acoustic signals.

- We also conducted experiments to evaluate the use of transfer learning as a means to enable GOTCHA to rapidly adapt to new environments while maintaining high detection accuracy (above 90%) and a low false alarm rate. The results demonstrate GOTCHA's capability to effectively adapt to environmental changes, thereby validating its practical feasibility for real-world deployment.

1.4 Publications

The following publications cover parts of the work presented in this thesis:

- Pham, H.Q.V., Nguyen, H.D., Roedig, U. (2025). *GOTCHA: Physical Intrusion Detection with Active Acoustic Sensing Using a Smart Speaker*. In: Horn Iwaya, L., Kamm, L., Martucci, L., Pulls, T. (eds) Secure IT Systems. NordSec 2024. Lecture Notes in Computer Science, vol 15396. Springer, Cham. https://doi.org/10.1007/978-3-031-79007-2_18

1.5 Outline

The remaining parts of this thesis includes 4 chapters from chapter 2 to chapter 5 as follows: Chapter 2 provides an overview of the background, including key concepts and operational mechanisms related to physical security intruder detection and smart speakers. It also covers foundational knowledge and related research on the use of acoustic sensing for human presence detection, anomaly detection, and transfer learning. Chapter 3 presents a detailed description of GOTCHA, including its overall design and the operational mechanisms of its components. In addition, this chapter outlines the experiments conducted to evaluate the performance of GOTCHA in real-world environments. Chapter 4 discusses the adaptability of GOTCHA to environmental changes. It evaluates and identifies the most suitable adaptation strategy for GOTCHA, and presents experimental results demonstrating the system's ability to maintain performance when deployed in varying environmental conditions. Finally, Chapter 5 presents the concluding remarks and outlines potential directions for future development and improvement of GOTCHA.

Chapter 2

Background and Related Work

The detection of physical intruders is a critical aspect of modern security systems, yet it presents significant challenges due to environmental variability and the need for reliable, cost-effective solutions. This chapter provides an overview of existing approaches to tackle this problem, focusing on human presence detection using acoustic sensing. First, we briefly introduce the mechanism of smart speakers and how acoustic sensing—both passive and active—can be leveraged for intrusion detection. We then explore contemporary methods for human presence detection, highlighting the advantages and limitations of passive and active sensing techniques. Additionally, we provide background on anomaly detection using Autoencoder (AE), offering insights into the role of deep learning in this domain. Finally, we discuss environmental adaptation techniques, particularly transfer learning, as a foundation for our proposed method to enhance the robustness of GOTCHA against environmental changes.

2.1 Physical Security Intruder Detection

Human presence detection is a prominent research topic as it is necessary for a number of applications such as security surveillance, healthcare, and traffic management. A vast range of sensing modalities can be used for human presence detection including vision Ahire et al. [2023]; Bharadwaj K H et al. [2018]; Chen et al. [2016], Infrared (IR) Netinant et al. [2022]; Sahoo and Pati [2017], Radio Frequency (RF) Denis et al. [2019], and audio. In the case of RF, existing communication systems such as WiFi Liu et al. [2020] might be used for human presence detection. Vision-based approaches might not work under low illumination conditions and also raise privacy concerns. RF-based detection based on

existing communication equipment is often complex to deploy and calibrate. IR and RF based approaches require dedicated hardware and effort to deploy. In this work we consider human presence detection using acoustic signals. We consider an already deployed smart speaker to perform this task.

Speakers and microphones are now ubiquitous, integrated into mobiles and many Internet of Things (IoT) devices, facilitating acoustic sensing applications Bai et al. [2020]. Consequently, acoustic sensing is attracting attention from the research community. Specifically, acoustic sensing has been employed to observe humans. For example, Xie et al. [2021] turn smart speakers into active sonars to determine exercise types using a method based on Doppler shift and short time energy. Xu et al. [2019] utilized commercial off-the-shelf equipment to generate acoustic signals to analyze how much each part of the human body responds to acoustic signals while walking. Their work tries to distinguish individuals based on fine-grained gait information. In this work we aim to use acoustics to determine the presence of a person in a room instead of deriving specific information about a person's behaviour.

2.2 Smart Speaker and Acoustic Sensing

A smart speaker is an advanced device equipped with voice recognition and virtual assistant technology, enabling users to interact hands-free through voice commands. When a user speaks, the sound waves travel through the environment, interacting with surfaces like walls, furniture, and other obstacles, which causes reflection, absorption, and scattering of the sound. These interactions affect the audio signal that reaches the speaker's microphones. Smart speakers use sophisticated algorithms to filter out background noise, isolate the user's voice, and accurately interpret commands despite environmental challenges.

2.2.1 Smart Speaker

VAs often come in the form of stand alone devices referred to as smart speakers. Smart speakers are becoming increasingly popular in households. According to "Statista" [2024], 64 percent of Americans owned an Amazon Echo in 2023, and, in that same year, a survey showed that half of surveyed middle-aged British answered that they owned a smart speaker at home. A smart speaker usually consists of a speaker and a microphone array to listen to the sound from any direction. The microphone array is in listening mode all the time so that it never

misses any command from a user. A smart speaker needs to be activated using a wake word before a command can be issued. For example, "Alexa", "OK! Google" or "Siri" are used. Once the smart speaker is activated, the microphone array uses beamforming to increase audio sensitivity in the direction of the speaker Alanwar et al. [2017]. The speakers are usually used to provide audio feedback to user commands. However, speakers and microphones may be used to implement additional applications. Besides ordinary applications such as controlling a smart home or playing music, a smart speaker may be turned into a security device. We consider here the use of active acoustic sensing for this purpose.

2.2.2 Acoustic Sensing Mechanism

Acoustic-based sensing and its applications have a long history. The basic concept of active acoustic sensing is to send an acoustic signal and thereafter analyse the received response. Systems using active acoustic sensing differ significantly in terms of *transmission*, *signal*, *reception* and *processing*. These design dimensions of an active sensing system cannot be treated fully independent. In this work, the design space is significantly constrained by the requirement to perform active sensing using an off-the-shelf smart speaker.

Transmission:

The capability of a system is constrained by the sensing signal transmission. A system may use a single speaker or multiple speakers in different locations to emit a measurement signal. A signal may be transmitted omnidirectionally or focused in a specific direction using a movable speaker or beamforming using a speaker array. The design and quality of the speaker(s) influence the usable frequency range, frequency response, sensitivity, noise and more. In this work, we consider a single low quality speaker which may not have a perfect omnidirectional transmission pattern and is limited to a low frequency range optimised for human audible sound.

Signal:

Using short sound pulses with high frequencies are usually better for sensing than long signals with lower frequencies. In this work, we consider relatively long pulses with a duration of several milliseconds in the audible frequency range as these are supported by smart speaker hardware.

Reception:

Similar to the speakers, a system may use a single microphone, multiple microphones (also as an array) or multiple distributed microphones. Transmission and reception may also occur at different locations. Usually, sensing capabilities improve if transmission and reception are separated and multiple microphones improve sensing. In this work, we consider transmission and reception to be colocated. We consider multiple microphones but not the ability for beam steering. While smart speakers have this ability it is usually not accessible to an application programmer on the device.

Processing:

Received signals can be analysed using different techniques. A variety of techniques can be used to analyse signals regarding strength variation, phase change, Doppler shift, Time-of-Arrival (TOA), Direction of Arrival (DOA), etc. Simple statistical methods may be used or sophisticated deep neural networks may be employed. In this work, we assume that even complex analysis methods can be used as the processing of a smart speaker can be offloaded to a powerful back end system (as it is done in the case of transcriptions of speech to text).

2.3 Human Presence Detection - Passive Sensing

Acoustic passive sensing is a method that uses acoustic sensors to hear and analyze ambient sounds. The recent development of this technology bring many potential application, especially in human presence detection. Human presence is either detected by registering noise above a threshold within a specific time window or in more advanced systems by identifying sound linked to human presence (e.g. footsteps).

Al-Khalli et al. [2023] built a home security system that can detect burglars by analyzing acoustic signals. An algorithm named *short-time-average through long-time-average* is used to detect noise above a threshold to detect human presence. The decision threshold is computed in an adaptive manner to maintain a constant false alarm rate.

Sharma et al. [2023] proposed an end-to-end system for intrusion detection by sensing the sound of footsteps amongst background noise. The algorithm uses

Table 2.1: Summary of related works in acoustic active sensing for human presence detection task. Room size is categorized as large (lecture hall - 9(m)x6(m)), medium (common living room - 5(m)x5(m))

Researchs	Environment	Emitting Sound	Omnidirectional speaker	No of channels	Approach	Label needed in training	Room size
Shih and Rowe [2015]	Indoor	Chirp (20-21kHz)	✓	1	DBSCAN + Regression Model	✓	Large
Yoneoka et al. [2019]	Indoor	Pulse (6000-7750Hz)	✓	8	Statistical Method	✗	Medium
Alanwar et al. [2017]	Indoor	Tone (1kHz)	✓	8	Random Forest Classifier	✓	Medium Medium
Cheong et al. [2022]	Indoor	Chirp (12.5–15kHz)	✓	1	Statistical Method	✗	Medium
GOTCHA	Indoor	Chirp (6–8kHz)	✗	4	Autoencoders	✗	Medium

energy information (amplitude) and spectral information (size of zero crossing intervals) to identify footsteps; an Machine Learning (ML) approach is not used.

A number of security systems based on ML sound classification have been proposed (e.g. Agarwal et al. [2021], Kim et al. [2020], Alsina-Pagès et al. [2017]). Often they aim to identify extreme acoustic events such as a glass break, a gunshot or a fire alarm. Previous studies have used ML approaches in the context of human behaviour classification (e.g. Do et al. [2021], Li et al. [2019], Sim et al. [2015]) but not in the context of human presence detection although they could be used for this purpose.

Our work differs from these existing solutions as we propose an active acoustic sensing approach instead of using passive acoustic sensing.

2.4 Human Presence Detection - Active Sensing

For acoustic active sensing a sound signal is emitted and reflections from the sourroundings are analysed to characterize the environment. Existing works have used active sensing for human presence detection. Table 2.1 provides a summary of the key characteristics of existing works which we discuss in the next subsubsections. The main difference of all existing works to our proposed method is that simple statistical methods are used while we employ an autoencoder, a type of artificial neural network.

Shih and Rowe [2015] use active ultrasonic sensing to count people in a room. DBSCAN combined with a regression model is applied. Good performance is achieved with an average error of 5% relative to the maximum room occupancy. However, the system requires high-end audio equipment to handle the used high-frequecy signal. The system also can distinguish an empty room from an occupied

room. However, different to our approach, labelled training data is required for human presence cases. Our approach employs an unsupervised training method.

Yoneoka et al. [2019] analyses signal power correlation between a pulse and received sound to detect a human. They showed that there is a different pattern in power attenuation of reflection sound in the case with and without human presence. Their work focuses on detection speed instead of detection accuracy. Hence, the application scenario is slightly different and a direct comparison with our work is not meaningful.

Alanwar et al. [2017] detected human presence after a critical command is issued to a VA to verify that a user is present. Machine learning models SVM, RF, MLP with sound signal characteristics such as mean, standard deviation, skewness, kurtosis and MFCC as input are used. They can achieve 93% in accuracy but their training process required labels for both empty room and human presence cases. The application scenario is also slightly different; there was an acoustic interaction with the device before a pulse is sent.

Cheong et al. [2022] employs a two-dimensional spectro-temporal filtering mechanism on a Fourier spectrogram to measure the total energy of the reverberations of the chirp signal to detect the change in the acoustic scene. Their work used reverberation energy difference between two consecutive chirps to detect a change in the background scene. Their work can achieve 100% accuracy without any false alarms in the case that the human position is 2-5(m) far away from the speaker. However, in their work, their system prototype required an omnidirectional microphone and 4 speakers facing 4 cardinal directions, which is not easy to build with commodity devices on the market.

2.5 Anomaly Detection

Anomaly detection has been a significant challenge studied for centuries. It involves identifying patterns in data that deviate from expected behavior Nassif et al. [2021]. Various unique approaches have been developed and applied across different domains to identify anomalies. Not falling outside that trend, many applications of anomaly detection have been applied in the security domain and now anomaly detection is a core element for a number of security applications. Any Intrusion Detection System (IDS), either used to detect intrusion within a computer system or as in our case a physical intrusion, uses at its core an anomaly detection algorithm. Thus, a number of anomaly detection methods

have been developed over the last decades to propose good solutions for intrusion detection problem. These approaches can be classified as two main categories: traditionally statistical methods and ML-based methods. Statistical anomaly detection techniques are among the earliest algorithms developed for identifying anomalies. These methods construct a statistical model representing the normal behavior of the given data. To identify anomalous points in dataset, a statistical inference test can be performed. Various approaches are employed in statistical anomaly detection, including proximity-based, parametric, non-parametric, and semi-parametric methods Bhuyan et al. [2013]. Apart from statistical approaches, ML-based techniques are becoming more prevalent as a method for anomaly detection. These techniques aims to "automate the process of acquiring knowledge from examples" Bose and Mahapatra [2001]. This approach involves creating a model that differentiates between normal and anomalous instances. As a result, ML-based approaches can be categorized into three main types based on the type of training data used to develop the model. Those are supervised learning, semi-supervised learning and unsupervised learning. Supervised anomaly detection involves training a model using labeled normal and anomalous data. The approach builds predictive models for both classes and compares them. However, challenges include the imbalance between normal and anomalous instances and the difficulty of obtaining precise and representative anomaly labels. Semi-supervised anomaly detection trains only on normal class data, marking anything deviating from it as anomalous. These methods assume labeled instances are available only for the normal class and are more commonly used than supervised methods since they do not require anomaly labels. Unsupervised anomaly detection does not require training datasets and assumes that normal instances are more frequent than anomalies in test data. In the context of human intrusion detection, semi-supervised and unsupervised methods are more widely used than supervised approaches due to the rarity of abnormal cases. Using datasets with only labeled normal instances or entirely unlabeled data significantly reduces the cost of data collection. Consequently, many deep learning-based techniques have been developed to handle such datasets, with Autoencoder being one of the most effective networks.

2.5.1 Autoencoder

An AE network is one relatively recently developed technique for this approach. An AE network contains two parts: encoder and decoder 2.1. The encoder is trained to compress an input signal of normal data (data that represents the

normal state of operations without an intrusion) into a low-dimensional representation space. The decoder is trained to recover the original signal from the compressed representation. A loss function is chosen to measure how closely the recovered output resembles the original input. The encoder and decoder are trained to minimize the overall reconstruction error (the difference between the input and output signal of the AE). The retained information after encoding is required to be as much relevant as possible to the normal data Pang et al. [2021]. Thus, the reconstruction error can be used as an indicator to identify anomalous data input. If an unusual data input is provided to the autoencoder it will not be able to reconstruct the output as well as in cases where the input is from the normal data pool used to train the autoencoder.

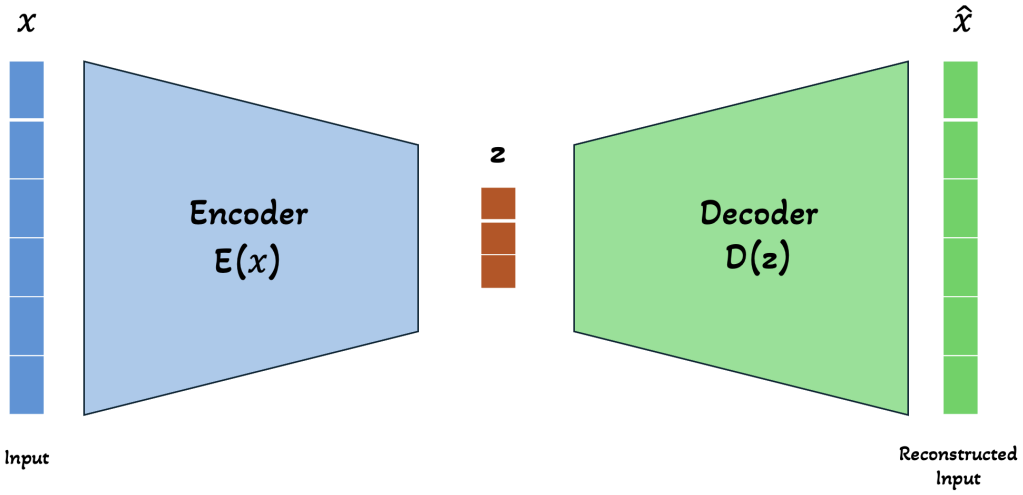


Figure 2.1: The Autoencoder network architecture: Encoder part, which maps the input x into a latent representation z, and Decoder parts, which reconstructs z back into the original input space.

In an intrusion detection context an autoencoder is particularly useful as for training only data from the normal operation state of the system is required. As it is usually difficult to obtain sufficient data examples representing an intrusion or it is not possible to enumerate all possible intrusions this is a clear benefit to other methods, specifically classifiers that need to be trained using data representing all different classes of intrusion.

The AE network can be mathematically described as follows:

$$z = f_e(x) \tag{2.1}$$

$$\hat{x} = f_d(z) \tag{2.2}$$

$$\{\theta^*{}_{f_e}, \theta^*{}_{f_d}\} = \underset{\theta_{f_e}, \theta_{f_d}}{\operatorname{argmin}} \sum_{x \in \chi} \|x - \hat{x}\|^2 \tag{2.3}$$

$$\epsilon_x = \|x - \hat{x}\|^2 \tag{2.4}$$

where f_e and f_d are the encoder and the decoder of the AE with the parameters θ_{f_e} and θ_{f_d}, respectively. ϵ_x is the reconstruction error for input x and recovered output $\hat{x}$.

2.6 Environment Change Adaptation

Traditional ML/Deep Neural Network (DNN) training methods can achieve quite impressive performance on task of physical intrusion detection. However, these training methods assumes that data in training phase and inference phase come from the same distributions. However, this assumption often does not hold in real-world scenarios. When the probability distributions of training and testing data differ, model performance tends to decline due to domain distribution gaps. Collecting data from all possible domains for training is costly and, in many cases, infeasible. As a result, improving the generalization capability of ML models remains a crucial challenge in both industry and academia Wang et al. [2021]. Recently, many researches has been arising to solve domain shift challenge. Notably, an approach that are gaining significant attention is transfer learning.

2.6.1 Transfer Learning

Transfer learning is a powerful machine learning approach designed to address the challenge of domain differences by transferring knowledge across domains. For example, a person who has learned to play the piano may pick up the violin more quickly than someone with no prior musical experience. Inspired by this human ability to apply knowledge from one area to another, transfer learning utilizes information from a related domain (known as the source domain) to enhance learning efficiency or reduce the need for labeled data in the target domain Zhuang et al. [2019]. An approach has been proposed by Danti and Innocenti [2023] that utilizes a fine-tuning process to update the autoencoder model on a weekly basis in order to capture the system's dynamics during the different seasons, effectively reducing false alarms. In domain of anomalous sound detection, Han

et al. [2024] utilized transfer learning by using large scale pre-trained models to detect anomalous machine sound and Zheng et al. [2024] employed Low-Rank adaptation to tune pre-trained audio models for anomalous sound detection task.

2.7 Summary of Literature Review

The application of ML in physical intrusion detection has been gaining significant attention. However, existing methods primarily rely on either traditional statistical analysis models or conventional ML models with manual feature extraction, both of which have certain limitations that need to be addressed.

Given its proven effectiveness in anomaly detection across various domains, autoencoder presents a strong potential for application in physical intrusion detection. Additionally, deploying these models efficiently on low-cost hardware remains an important consideration. Moreover, real-world variations in acoustic environments pose a challenge in terms of minimizing the cost of data collection and model retraining.

In this thesis, we propose an autoencoder-based approach for human intrusion detection and implement it on a prototype simulating a low-cost smart speaker to tackle the aforementioned challenges. Furthermore, we conduct experiments and propose strategies to reduce the cost of data collection and model retraining in response to environmental changes.

Chapter 3

Human Presense Detection by Active Acoustic Sensing

The application of active acoustic sensing for physical intrusion detection remains relatively novel. By analyzing the differences between reflected signals and the emitted sound under varying room conditions, it becomes feasible to distinguish between human presence and absence. This approach enables a privacy-preserving solution to the human presence detection problem, offering a significant advantage over camera-based or passive acoustic sensing methods. Moreover, the proposed algorithm can be efficiently implemented on low-cost smart speakers, unlocking new application opportunities for such devices and reducing deployment costs for end users in smart home systems.

3.1 Threat Model

We consider a scenario where an adversary seeks to stealthily enter and occupy an enclosed indoor environment, such as a residential room or a private office, monitored by our active acoustic sensing system. The primary objective of the adversary is to perform an unauthorized intrusion while evading detection. We assume a non-expert adversary who has no technical understanding of the underlying sensing mechanism, the acoustic frequencies employed, or the machine learning model's architecture. Consequently, the adversary does not attempt to electronically jam the system or craft adversarial acoustic signals. Instead, the adversary relies on natural, intuitive methods of evasion, such as moving at an extremely slow pace, crawling, or tip-toeing to minimize physical noise and sudden movements.

Our system is designed to operate on commodity, affordable hardware readily available to end users, such as standard smart speakers or IoT devices. Our model operates under the assumption that the smart speaker's hardware and its standard operating environment remain uncompromised, forming a Trusted Computing Base (TCB) that ensures the reliable acquisition of acoustic data. This allows us to focus on the physical security of the space rather than cybersecurity vulnerabilities. Therefore, attacks involving the physical destruction of the sensing device, remote software exploits to disable the microphone, or the use of specialized electronic jamming equipment are considered out of scope for this work.

3.2 GOTCHA System

The goal of developing GOTCHA is to detect the presence of a person in a room using active acoustic sensing. A predefined sound signal is periodically emitted, and its corresponding reflections are captured. The system analyzes and compares the emitted and reflected signals to identify differences between the room's acoustic states when it is occupied versus unoccupied. The underlying algorithm is implemented on low-cost hardware to ensure practical applicability and affordability for real-world use.

3.2.1 Assumptions

In real-world settings, numerous factors and scenarios can affect the performance of physical intrusion detection systems. To keep the scope of this thesis manageable, we simplify the problem by constraining several conditions. Firstly, we assume that GOTCHA operates in rooms of standard size—neither too large nor too small—comparable to typical bedrooms, living rooms in households, or meeting rooms in office environments. Secondly, the surrounding environment is assumed to be relatively quiet. Specifically, we conduct experiments in residential rooms during working hours, aligning with the context of intrusion detection needs when homeowners are away. Thirdly, the interior layout of the room includes basic furniture such as tables, chairs, bookshelves, and cabinets. The rooms considered are not equipped with any specialized soundproofing systems or sound-absorbing layer on the walls. The GOTCHA device is positioned on a table surface, free from obstruction by nearby objects within a one-meter radius. This setup reflects common user behavior in real-world scenarios, where smart speakers are typically placed in open areas within a room to facilitate effective

interaction between the end user and the device. Ensuring an unobstructed placement is crucial to maintaining the quality of active acoustic sensing signals and preserving the reliability of the system's detection performance. While these constraints simplify the problem space, they still reflect practical conditions under which GOTCHA would be expected to operate.

3.2.2 Application Scenario

The user activates GOTCHA, an intrusion detection application on the smart speaker and then leaves the room. After a short time GOTCHA becomes active to detect intruders, similar to a standard burglar alarm. A sound pulse is emitted periodically and microphones of the smart speaker record responses to analyse the acoustic state of the room. Immediately after activation of GOTCHA it is known that the room is empty and the first responses collected by the device can serve as training data to have a clear understanding of the acoustic properties of the empty room. Training of the GOTCHA system can be performed every time the system is activated to accommodate changes in the room layout. This might be for example necessary if furniture has been moved. However, in most cases reprofiling of the empty room may not be necessary. After the system is initialised, GOTCHA sends periodic pulses and uses the recorded responses at multiple microphones to determine if the room state differs significantly from the expected empty room state. If that is the case GOTCHA records a detection but does not raise an alarm yet. Instead, windowing is applied and only if multiple detections are observed within the window an intrusion alarm is raised. Windowing avoids false alarms which is an important design consideration. Windowing increases detection delay, however, a delay of a few seconds is usually not a problem in the context of detecting a human intruder.

3.2.3 Sensing Signal

A sonar system such as GOTCHA should ideally use ultrasonic sound as commonly used in sensing application Lee et al. [2020]. However, ultrasonic systems typically require high-precision hardware manufacturing and are often associated with high costs. Apart from ultrasonic signals, there are various kinds of sounds that can be used, such as a tone, a chirp, etc. A chirp signal can be robust against background noise as reported by Cheong et al. [2022]. Moreover, using a chirp in active acoustic sensing demonstrated good performance in previous work (see Cheong et al. [2022]; Yoneoka et al. [2019]; Shih and Rowe [2015]). We did not

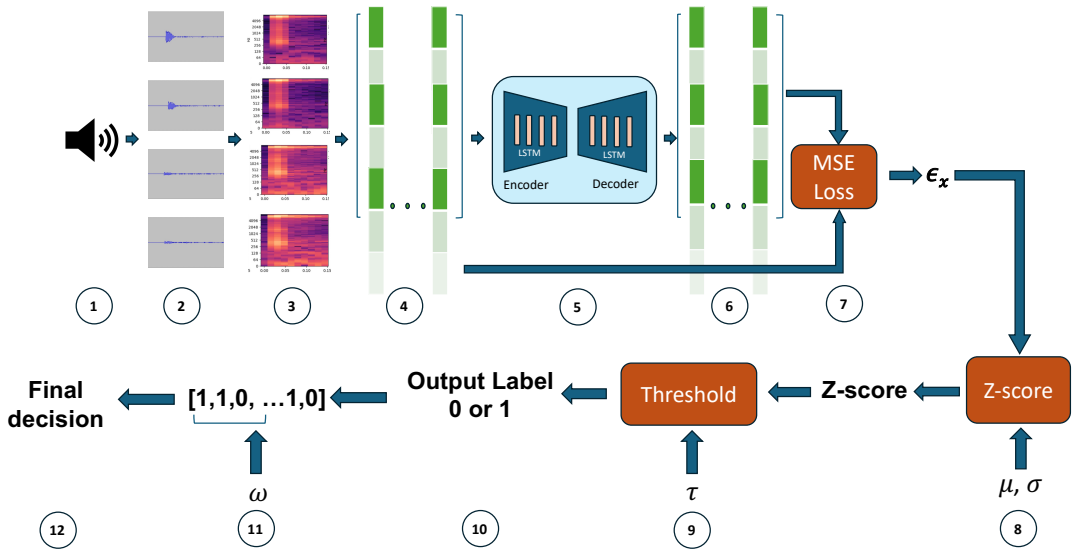


Figure 3.1: GOTCHA processing chain. (1) Audio signal (2) Recording on N microphones (3) Signal transformation into Spectogramms (4) Combining inputs into a single input matrix (5) Autoencoder processing (6) Reconstructed output matrix (7) Reconstruction error ϵ_x computation (8) z_score computation (9) Threshold τ comparison (10) Decision output (11) Windowing (12) Final decision output.

use the entire frequency range as the higher frequencies provide more information. We decided to use a chirp signal generated by Audacity[1] starting from 6kHz and gradually rising up to 8kHz at the end of the 10ms period. According to the Nyquist theorem, the sampling rate must be at least twice the bandwidth of the signal to avoid aliasing. Therefore, we select a frequency band from 6kHz to 8kHz to ensure that the emitted sound signals are compatible with most affordable microphones in the market, which can accurately capture signals sampled at 16kHz.

The sensing signal is audible (but not terribly annoying) and a person entering the space will notice it. That might be useful for the owner as a reminder when returning to disable the system. An intruder may seek out the device and disable it but this would also reveal their presence. This also serves as a motivation for us to develop an efficient approach capable of detecting intruders within a sufficiently short time frame, before they can physically approach the device.

3.2.4 GOTCHA Autoencoder Processing Chain

The flow of GOTCHA is described in Fig. 3.1. GOTCHA periodically sends an acoustic signal (Step 1) and then records the response on N microphones

[1]https://www.audacityteam.org/

(Step 2). Fig. 3.1 depicts GOTCHA for $N = 4$. We apply Short-Time Fourier Transforms (STFTs) to the acoustic signal received on each microphone and transform the input into 2D spectrograms (Step 3). The 2D representations are then fed to the AE to identify whether the current state is anomalous or not. To generate the input for the AE we stack the spectrograms from all N microphones (Step 4). For each time frame t of the spectrogram, we concatenate the frequency values:

$$X_t = [a_{1,t} \oplus a_{2,t} \oplus a_{3,t} \oplus ... \oplus a_{N,t}] \tag{3.1}$$

$$X = [X_1, X_2, X_3, ..., X_T] \tag{3.2}$$

where T is the number of time frames in the spectrogram, while N is the number of channels of microphones. Vector $a_{n,t}$ represents t^{th} column of spectrogram from channel n. Now, the raw audio signals from all channels are transformed into temporal representations. The encoder and decoder in our AE are two Long Short-Term Memory (LSTM) networks that are suitable to handle sequential data.

After receiving input, AE in GOTCHA aims to reconstruct the input signal as accurately as possible (Step 5 and Step 6). The encoder is trained to compress an input signal of normal data (data that represents the normal state of operations without an intrusion) into a low-dimensional representation space. The decoder is trained to recover the original signal from the compressed representation. Mean Square Error (MSE) loss is employed to measure differences between input representation and reconstructed output (Step 7). By minimising the MSE loss, our AE can learn the way to reconstruct the profile of an empty room as normal signals accurately. The result of MSE loss also known as the reconstruction error reflects how well our AE did. The AE network can be mathematically described as follows:

$$z = f_e(X) \tag{3.3}$$

$$\hat{X} = f_d(z) \tag{3.4}$$

$$\{\theta^*{}_{f_e}, \theta^*{}_{f_d}\} = \underset{\theta_{f_e}, \theta_{f_d}}{\operatorname{argmin}} \sum_{X \in \chi} \|X - \hat{X}\|^2 \tag{3.5}$$

$$\epsilon_X = \|X - \hat{X}\|^2 \tag{3.6}$$

where f_e and f_d are the encoder and the decoder of the AE with the parameters θ_{f_e} and θ_{f_d}, respectively. ϵ_X is the reconstruction error for input X and recovered

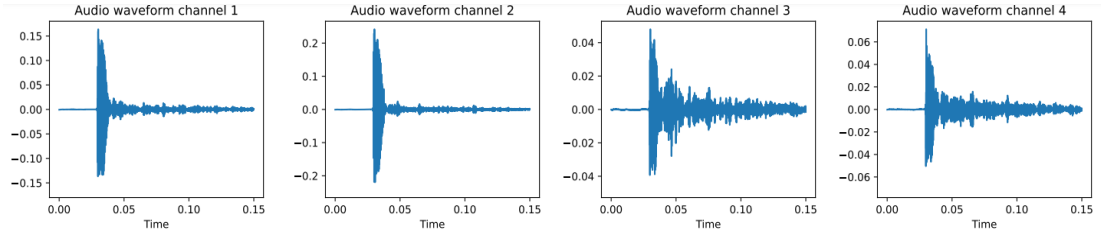
output $\hat{X}$.

Because training samples only include normal signals, any anomaly sample fed into our AE will result in a large reconstruction error. Thus, a distribution of reconstruction error from the training set is established to identify anomaly samples. In the inference phase, we feed new data into our trained AE and compare the newly computed reconstruction error to the distribution of training reconstruction errors to identify if the sample is normal or abnormal. Z-score as shown in Eq. 3.7 is a classical tool to detect outliers from a distribution (Step 8):

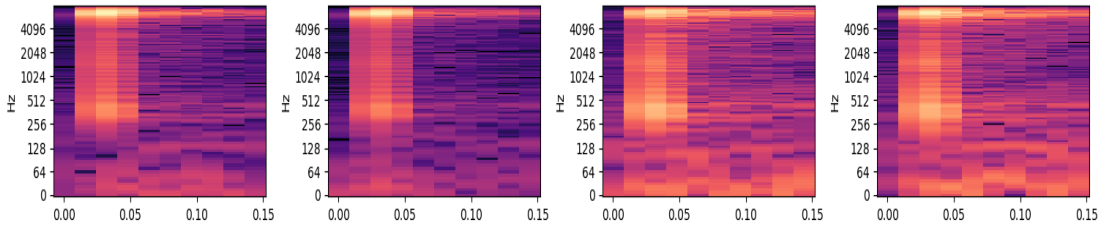
$$z_score = (\epsilon_x - \mu)/\sigma \tag{3.7}$$

where μ and σ are the mean and standard deviation derived from the AE output on the training set which only includes normal samples. An example input and the reconstructed signal is shown in Fig. 3.2. An input is considered as an anomaly if the z-score exceeds a pre-defined threshold τ_z (Step 9). Threshold τ is a hyperparameter that will be chosen during the hyperparameter tuning process. We label normal cases as 0(s) and anomalous cases as 1(s) (Step 10).

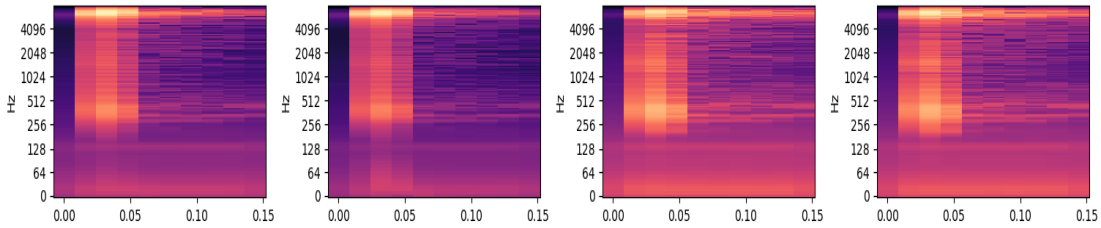
To eliminate false alarms, windowing is used. A window W is used to evaluate the last w decisions of the AE (Step 11). The majority label of all decisions within the window W are used as label for the entire window (Step 12). Thus, short-term noise does not lead to potential false alarms.



(a) Waves form of chirp signal



(b) Spectrograms of chirp signal



(c) Reconstructed spectrograms of chirp signal

Figure 3.2: An example of input signal and reconstructed spectrogram of 4 microphone channels for an example (Output of the processing chain Step (6) as shown in Fig 3.1). The reduced noise in the reconstructed output is clearly visible.

3.2.5 GOTCHA Prototype

We implemented a prototype that mimics a smart speaker using low-cost components, including a speaker and a microphone array. Both devices are connected to and controlled by a program running on a laptop. This setup is used to conduct the experiments for our human presence detection system.

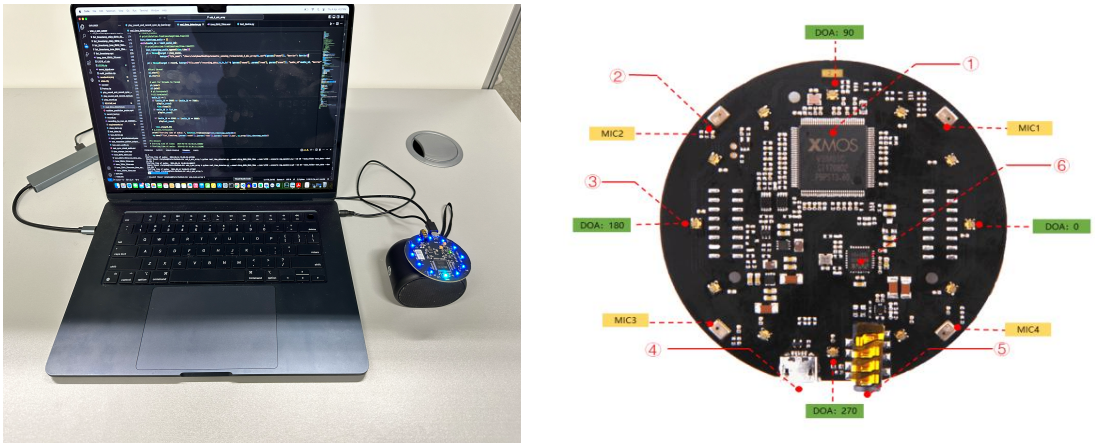
We use an HP S01 speaker as the transmitter of the chirp sensing signal. The speaker is a low-end device. As microphones we use the ReSpeaker Mic Array v2.0 from Seed Studio (see Fig. 3.3(b)). The ReSpeaker provides four microphones and the ability to record sound on all microphones in a synchronised fashion (i.e. recording starting points are synchronised). The platform is designed as a component to prototype smart speakers. The speaker and microphones are connected to a laptop that executes the GOTCHA algorithms.

Each sensing cycle has a length of 150ms. At the start of each period, a chirp

ranging from 6kHz to 8kHz is emitted by the speaker and the microphones are triggered to start recording. The chirp signal duration is 10ms and the process of emitting sound and recording is synchronised. In our prototype, recordings of cycles are stored for later analysis. However, in a practical system the analysis by the processing chain as shown in Fig. 3.1 would be executed after each cycle.

GOTCHA is implemented in Python using the API provided by ReSpeaker Mic v2 and to control the speaker via laptop. AE in GOTCHA was trained by Pytorch which is a well known framework for deep learning model and it also supports intergration into embedded system.

We place in the experiments the ReSpeaker microphones on top of the speaker (see Fig. 3.3(a)). This is not ideal as sound from the speaker is directly transferred to the microphones. However, we chose this configuration as it represents the closest real smart speaker implementation where both components are included in one casing.



(a) GOTCHA prototype (b) 4-microphones array

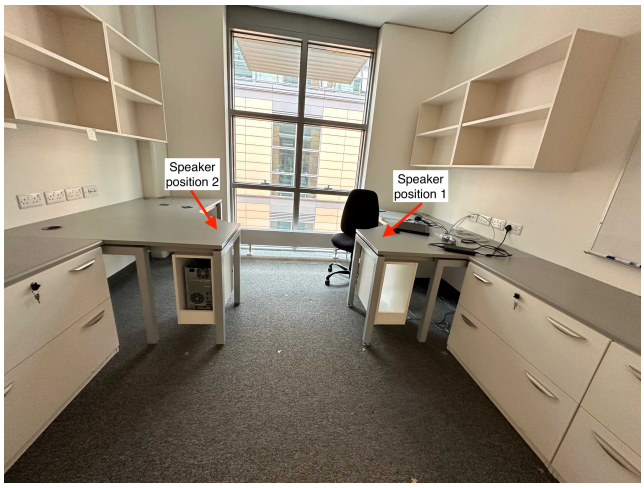
Figure 3.3: GOTCHA prototype design and microphone array structure.

3.3 GOTCHA Evaluation

To evaluate the capability of GOTCHA, we place our prototype in standard rooms (e.g., bedroom, office room, ...) as previously described, and run GOTCHA in both scenarios: with and without human presence. The collected data will be used to train and evaluate GOTCHA's ability to detect human presence. Details of the experiments and evaluation methods will be presented in the following.



(a) room-1



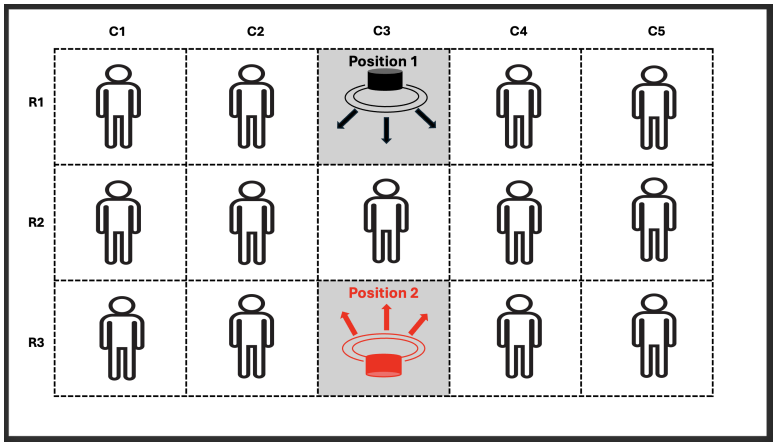
(b) room-2



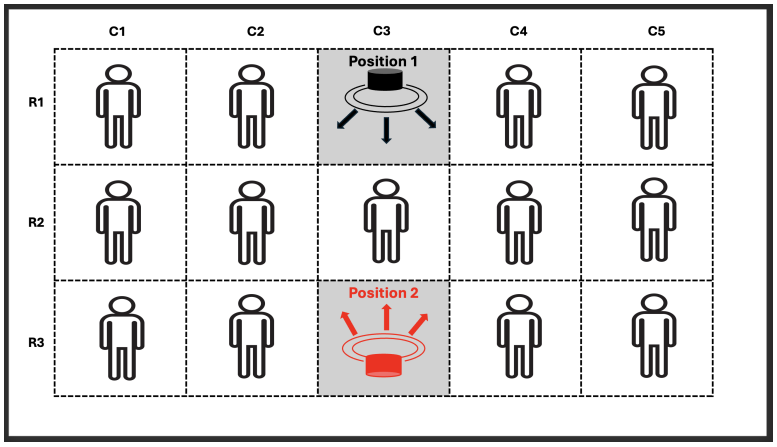
(c) room-3

Figure 3.4: The environment of 3 rooms used in the experiments: room-1 and room-2 are typical office rooms furnished with desks, chairs, and cabinets, while room-3 is a standard bedroom equipped with a bed, curtains, and more objects.

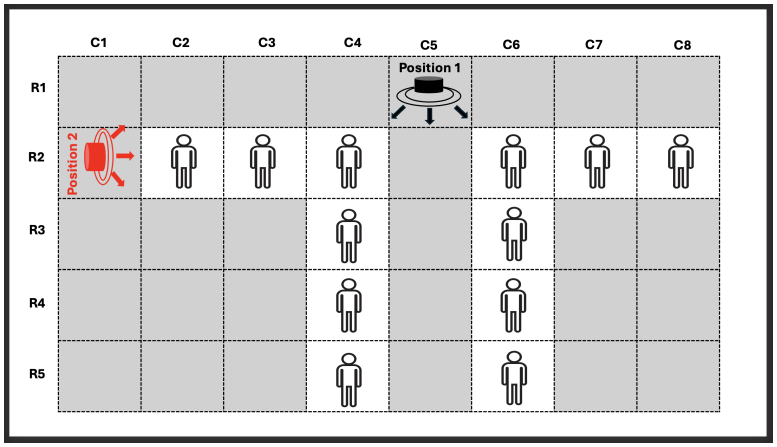
3.3.1 Experiment Setup



(a) Human positions layout in room-1



(b) Human positions layout in room-2



(c) Human positions layout in room-3

Figure 3.4: Room layout and regions with possible human presence. In white regions, a human subject can be located; grey cells represent obstacles (bed, cabinet, table, etc.).

We use the prototype described in Section 3.2.5 for evaluation. The experiments are conducted in a total of 3 rooms, 2 rooms of size 3.5m x 3.3m containing a table and cabinets and 1 bed room of size 6m x 4m (see Fig. 3.4).

The GOTCHA prototype is placed on a table located in one corner of the room. The layout is shown in Fig. 3.4. In each room two experiment runs are conducted with the speaker placed in different locations (shown as black and red speaker symbol). We divide the floor space into multiple regions, each region is a rectangle of size 80cm x 65cm. A region where a person can be are shown in white, grey regions are occupied by furniture and a person cannot be present. We place markers on the floor in the centre of each white square which we use as marker for a test subject to stand. Room-1 and room-2 are divided into 13 regions, while room-3 has 12 regions. Thus, we are able to collect data in a repeatable fashion. We conduct our experiments at a random time of the day; In room-1 and 2 we collected data in the morning while for room-3 data was collected in the afternoon. Some noise from the outside environment was present during experiments but not at significant level (usual background noise).

3.3.2 Experiments and Dataset

The GOTCHA prototype is started and executes a sensing cycle every 150ms. A chirp ranging from 6kHz to 8kHz over 10ms is emitted and the room response is recorded. The furniture is constant throughout the experiment so that it can be ensured that all changes recorded are due to a human that might be present. Initially, we execute a number of sensing cycles in the empty room for around 50 minutes to collect training data for the AE. Thereafter we collect validation and testing data. First, sensing cycles in the empty room are collected. Then, a person enters the room for approximately 20 minutes to record sensing cycles with human presence. The person is moving during this time from one of the regions marked on the floor to the next. We record the positions together with the sensing data such that a ground truth is available. This process is repeated two times, one to collect the validation set and one to gather the testing set.

We collected 9600 recording samples for each speaker position in room-1 and room-2, as well as 9400 for each position in room-3. A summary description of the data is shown in Table 3.1. For all 3 rooms, the first 5000 samples are fed to the AE in the training phase. For the validation and testing phase, each of human position corresponds to 100 samples and the rest represent the empty room. The

Table 3.1: Datasets summarization (r1, r2 and r3 stand for room-1, room-2 and room-3 while p1 and p2 stand for speaker position-1 and 2, respectively)

Data	Label	Number of samples					
		r1_p1	r1_p2	r2_p1	r2_p2	r3_p1	r3_p2
Training	Empty	5000	5000	5000	5000	5000	5000
Validation	Empty	1000	1000	1000	1000	1000	1000
	Human presence	1300	1300	1300	1300	1200	1200
Testing	Empty	1000	1000	1000	1000	1000	1000
	Human presence	1300	1300	1300	1300	1200	1200

dataset and code of GOTCHA are available via our Github repository[2].

3.3.3 Comparison and Metrics

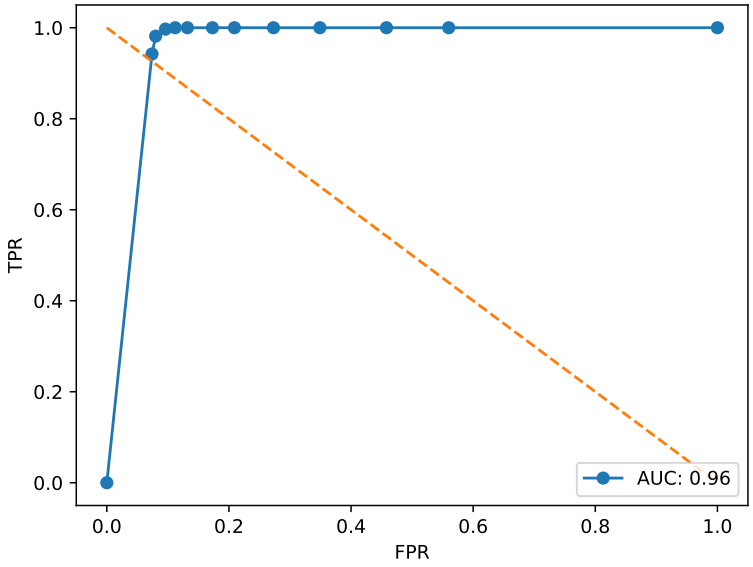
Comparison

Among existing works mentioned in Chapter 2 - Section 2.4, we found that work by Yoneoka et al. [2019] is closest to our work. We reimplemented their method and applied it to our dataset. However, as they showed in their paper, their method was designed to record continuous sound and uses a full 30 second recording for analysis. That is different from our recording process because we only require a chirp signal recording in discrete periods. The result of the reimplemented method by Yoneoka et al. [2019] on our dataset is poor (almost a random prediction with an AUC at 50%). GOTCHA improves significantly on existing work as it can work with short pulses and short recordings using off-the-shelf commodity hardware. A comparison against existing work under these conditions is not meaningful and therefore a comparison with other solutions is not provided.

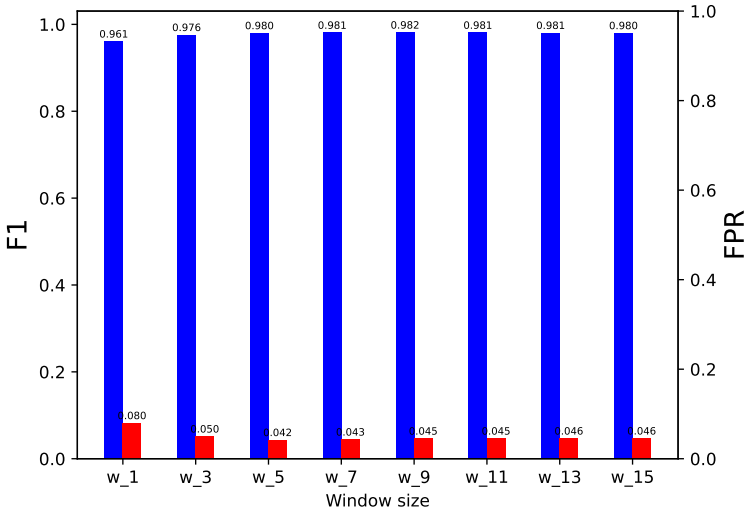
Metrics

We consider an anomaly as label 1 and the empty room as label 0. We use F1-score and FPR as main metrics to evaluate our system, we also use accuracy to show the result of GOTCHA when detecting humans at different positions. It has to be noted that the metrics can be applied to the classification result with or without the last step of applying windowing.

[2]https://github.com/viet98lx/GOTCHA



(d) ROC curve for selection of threshold τ by tuning on validation set of room-1 speaker position-1



(e) F1 (blue) and FPR (red) for different window sizes w on validation set of room-1 speaker position-1

Figure 3.5: Hyper-parameter tuning for the selection of μ, σ and τ in step (8) and step (9) in the signal processing chain in room-1 speaker position-1 as depicted in Fig. 3.1.

3.3.4 Hyper Parameters Tuning

All hyper parameters tuning process are conducted on validation set. GOTCHA requires parameters μ, σ and τ in step (8) and step (9) in the signal processing chain as depicted in Fig. 3.1. Furthermore, a window size w should be selected

in step (11), used to reduce FPR for the final decision in step (12). Fig. 3.5(d) and Fig. 3.5(d) describe the process of tuning parameters τ and w for the dataset of speaker at position-1 in room-1. We applied the same approach for the other datasets tuning parameters for these settings.

Threshold τ and z-score parameters μ, σ:

After training of the AE with the training set we obtain the distribution of the reconstruction error ϵ_x. μ and σ are the mean and standard deviation of this distribution. We use the z-score in Equation 3.7 to identify an anomalous sample. For every sample in the validation and testing set, we calculate the reconstruction error and then the z_score. We plot the ROC curve for all possible thresholds τ as shown in Fig. 3.5(d) for the speaker at position 1 in room-1. W. We chose τ by using the Equal Error Rate (EER) point such that the highest True Positive Rate (TPR) can be achieved while still keeping FPR at a low rate.

Window size w:

An alarm system benefits from preventing false alarms even if the detection accuracy is reduced. An operator will ignore alarms after a while if too many false alarms are issued and the system becomes ineffective. We tested window sizes of odd values 1, 3, 5, ... to 15 to choose the best value. In Fig. 3.5(e) we can see that for a window size of $w = 5$ the best F1-score of 98.0% on the validation set is achieved while the FPR is kept at 4.2% for speaker position-1 in room-1.

3.3.5 Data Analysis

Fig. 3.2(a) shows the signal waveform and spectrograms of 4 channels for one sensing sample in dataset of speaker at position-1 in room-1. It can be seen that the chirp starts at around 30ms and the signal amplitude attenuates after 40ms. In the spectrogram, the signal is as expected strongest at frequencies ranging from 6kHz to 8kHz. However, we also see that there are signal artifacts at lower frequencies. The samples in the other datasets have similar visualisation.

We also notice that the signal amplitude of channels 3 and 4 is often smaller than signal amplitude of channel 1 and 2. This is due to the fact that the speaker does not have an omnidirectional characteristic and sound is projected forward and channel 3 and channel 4 receive a stronger signal (see Fig. 3.3(a) and Fig. 3.3(b)).

After training, the AE provides the reconstructed spectrogram as output. The

Table 3.2: F1-score and FPR on testing set of 6 experiments (r1, r2 and r3 stand for room-1, room-2 and room-3 while p1 and p2 stand for speaker position-1 and 2, respectively)

Metrics	r1_p1	r1_p2	r2_p1	r2_p2	r3_p1	r3_p2
F1-score	99.2%	98.4%	99.0%	98.8%	91.4%	90.2%
FPR	1.9%	3.5%	2.3%	2.8%	9.4%	5.7%

comparison between input spectrograms and reconstructed output spectrograms are shown in Fig. 3.2. It can be seen that reconstructed spectrograms are smoother than the original input and regions at the low frequencies range become darker while the region at high frequencies remains the same. It indicates that the AE can learn to preserve the main signal of our chirp (6kHz-8kHz) while noise at lower frequencies is suppressed.

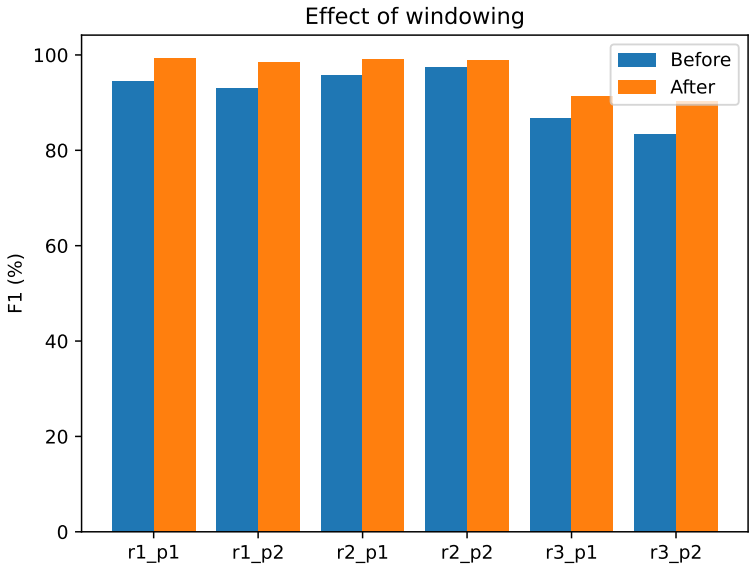
3.3.6 Detection Performance

Overall Performance

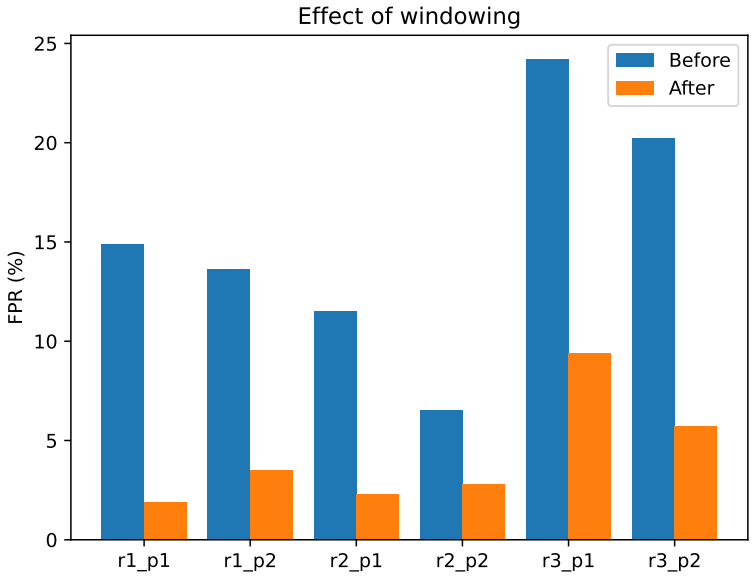
Our system can perform very well on the task of human presence detection. The results on testing set for 6 experiments are shown in Tab. 3.2. All test cases achieved F1-score above 90%, while the FPR remained below 10%. The best result was obtained in the case of room-1 speaker position-1, with a highest F1-score of 99.2% and a lowest FPR of 1.9%. Fig. 3.5(e) shows an example for room-1 speaker position-1 that window size 5 has the best F1 score while keeping FPR at a low rate. The window size of $w = 5$ means that 5 sensing cycles are required before a decision can be made. Our cycle is $150ms$ long and this means a decision can be made after $750ms$ once a person appears in the room. Given the speed with which humans normally move through a room a detection frame of under one second is sufficient. A person may not be detected within one window analysed; however as active sensing is continuous an intruder will eventually be noticed.

Effect of Windowing Method

Fig. 3.6 shows that after applying windowing method, for all cases, the F1-score can be improved while the FPR is significantly reduced. By using majority voting as the representative prediction for a window, GOTCHA is able to correct a significant number of mispredictions. As shown in Fig. 3.6(a), the F1-score can increase by 2%–5% after applying the windowing method. Meanwhile, in



(a) F1 score on testing set of 6 cases before and after windowing



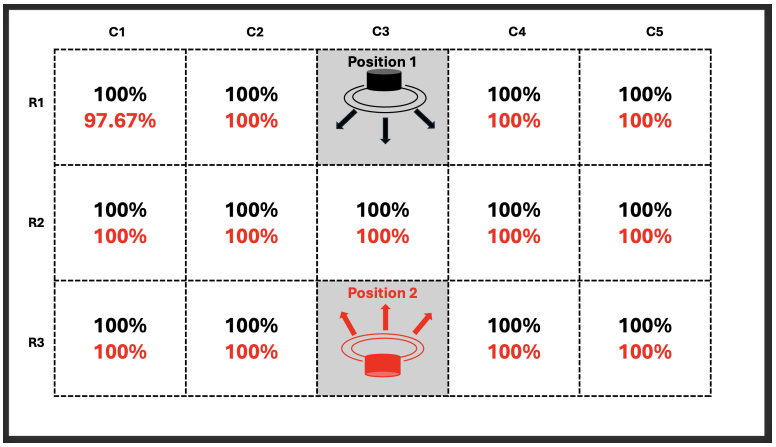
(b) FPR on testing set of 6 cases before and after windowing

Figure 3.6: Effects of windowing method on 6 cases (3 rooms with 2 speaker positions of each room. r1, r2 and r3 stand for room-1, room-2 and room-3 while p1 and p2 stand for speaker position-1 and 2, respectively).

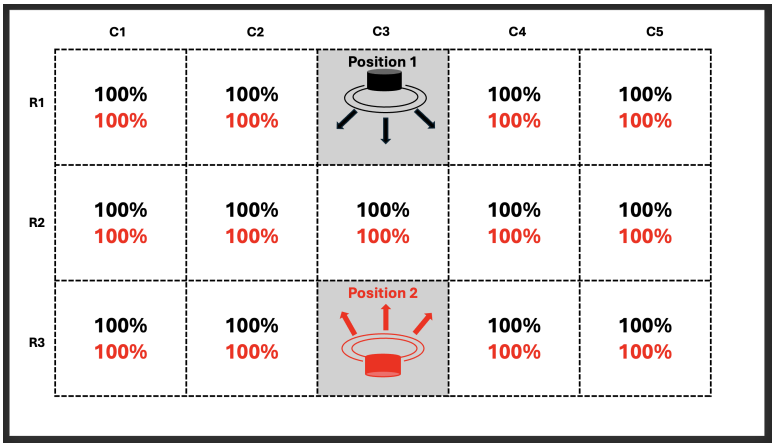
Fig.3.6(b), the FPR drops sharply by around 10%–15%, bringing the FPR of all experimental rooms down to below 10%.

Accuracy on Positions

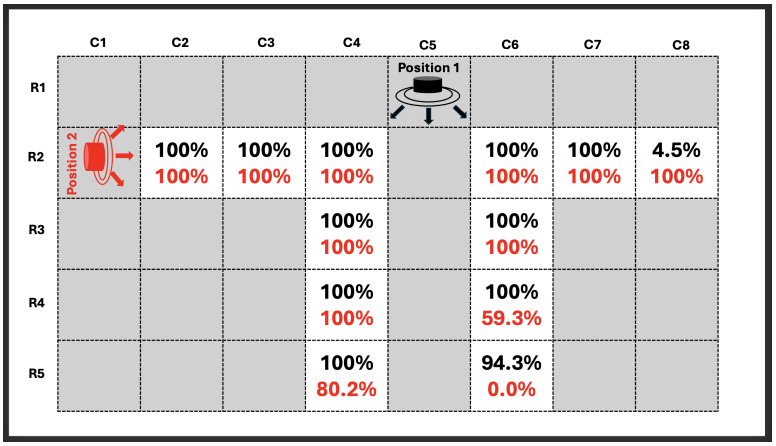
An important question is if GOTCHA is able to detect an intruder equally well at all positions within a room. More importantly, are there any blind spots in which



(a) Accuracy on testing set of room-1 after windowing



(b) Accuracy on testing set of room-2 after windowing



(c) Accuracy on testing set of room-3 after windowing

Figure 3.7: Accuracy of detecting a human at different positions in the room after windowing. The black figures show accuracy for speaker at position-1 and the red figures show accuracy for speaker at position-2.

the system cannot detect an intruder ? To analyse these aspects we designed our experiment such that data is collected with a person present at different marked

locations in the room and we conducted two sessions for two different position of speaker as shown in Fig. 3.4.

Fig. 3.7 shows the achieved accuracy at each position on testing set after applying windowing method for two different positions of speaker. Accuracy is defined as the percentage of correct classification after a sensing cycle; i.e. an 80% accuracy means that 80 out of the 100 cycles were accurately classed as a human is present. The results are shown in Fig. 3.7(a), Fig. 3.7(b) and Fig. 3.7(c) for room-1, room-2 and room-3, respectively.

In room-1 and room-2, we can see that almost all areas achieve 100% of accuracy for both speaker positions while there is no area in the room where the accuracy reaches zero. This means that within the time frame of sample collection at each point at least one alarm was raised; at any position an intruder is detected at least once within the 750ms window. It has to be noted that this result is achieved while keeping very low FPR.

In room-3, there are more obstacles compared to the other two rooms, it showes that there are a few blind-spots that make GOTCHA perform worse than in open areas. For speaker position-1, the position (R2/C8) is partly blocked by a wall so the accuracy drops significantly while at the position (R5/C8), the accuracy drops a bit even though it is in the direct line of sight to the speaker. For speaker position-2, in areas (R5/C4), (R4/C6) and (R5/C6), the accuracy drops significantly, especially at (R5/C6). It can be explained by the fact that these positions are far away and not in direct line of sight (the speaker is not perfectly omnidirectional), as well as that they are close to the wall with a curtain and especially, (R5/C6) is overlapping with a couch and curtain (see Fig. 3.4(c)).

In summary, GOTCHA is able to well detect an intruder in reasonable range while the system is not raising many false alarms.

Training Set Size

In previous sections, we used in our evaluation the entire collected dataset of 5000 sensing cycles within an empty room to train the AE. In a practical setting the system will be activated and the user will leave the room. Thereafter active sensing cycles will be executed in the empty room, the AE will be trained and then intrusion detection is activated. Thus, it may be necessary to shorten the time for the collection of empty room data in order to switch to the detection state early. Thus, we investigated how the reduction of training data impacts on

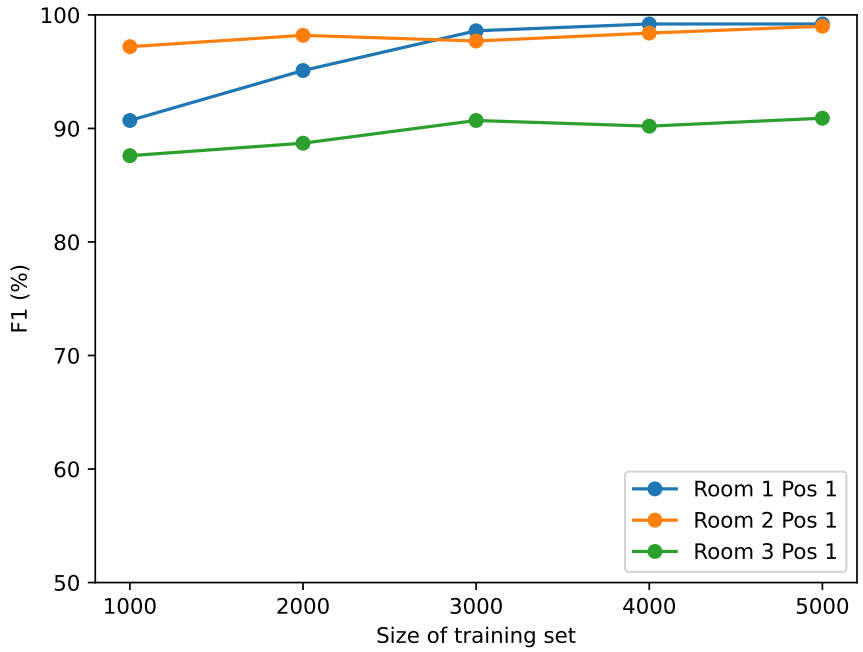


Figure 3.8: F1-score on testing set of 3 rooms at speaker position-1 when using different sizes of training set.

the resulting accuracy. We can see in Fig. 3.8 that all of 3 rooms share a similar trend: the F1 score of testing set tends to increase when the size of training set increases. However, the performance is not significantly improved after the size of training set is over 3000 samples. Clearly, it is possible to reduce the amount of necessary training data significantly.

3.3.7 Discussion

The results from the above experiments demonstrate that GOTCHA is capable of making highly accurate predictions. In all tested scenarios, GOTCHA achieved F1-scores above 90% and maintained a FPR below 10%, with the best results reaching 99.2% for F1-score and 1.9% for FPR. The application of the windowing method, which uses majority prediction within a time window, significantly improved performance—evidenced by increases in F1-score across all cases and a substantial reduction in false alarms, bringing FPR below 10% in every case. While GOTCHA performs well in most room positions, some blind spots still exist where the system fails to detect human presence. This limitation needs to be addressed in future work. Additionally, experiments on training set size indicate that approximately 3000 samples are sufficient for GOTCHA to achieve reliable intrusion detection performance.

3.4 Summary

This chapter introduced GOTCHA, a human presence detection system (i.e. physical intrusion detection system) using active sensing with low-cost devices. To the best of our knowledge our work is the first to employ an AE to detect human presence by active acoustic sensing. This is particularly significant as the AE can be trained with only samples from an empty room; data examples representing an intrusion are not required which are usually hard to obtain. We demonstrated that our system is highly practical with good results in our experimental evaluation. Our system achieved up to 99.2%, 99.0% and 91.4% F1-score on corresponding testing sets of 3 different rooms with low rate of false decisions. In almost all evaluated situations an intruder is always detected except the case that intruder is blocked from the path of signal. Our system can be easily deployed using low-cost devices and can be integrated in ubiquitous deployed smart speakers. We also collected a dataset for human presence detection by an active acoustic signal that can be useful for future research in the field of active acoustic sensing. We noted that it is impossible to compare approaches across the literature as different hardware and application scenarios are used. We hope that our dataset can be used to establish a benchmark for comparison.

Chapter 4

GOTCHA System Adaptation with Environmental Changes

For a number of reasons it is desirable to adapt the GOTCHA model. It would be useful to ship a GOTCHA device with a pre-trained model which is subsequently adapted to the room in which it is deployed. The device may also be relocated from one room to another which requires model adaptation. Finally, the deployment environment is also not static and may be subject to change. For GOTCHA to function effectively, model adaptation is a required feature. This chapter outlines our approach to ensuring that GOTCHA can remain operational despite variations in it surroundings. Moreover, we also provide an experimental evaluation to demonstrate the effectiveness of GOTCHA in changing environments.

4.1 The Need for Adaptation: Motivating Robust Performance in GOTCHA

GOTCHA has demonstrated the ability in detecting human presence in enclosed environments. As we have shown it is possible to place a device in a room and to collect data for training. Once sufficient data has been collected a model can be built for the room and presence detection is possible with high accuracy (See Section 3.3). The constructed model is static and designed for a specific room.

However, in real-world conditions, there are numerous factors that can lead to changes in the surrounding environment. For instance, the acoustic environment may vary throughout the day — mornings may feature more ambient sounds

such as human conversations or animal noises, whereas nighttime tends to be quieter. Environmental changes can also stem from spatial rearrangements, such as modifications in furniture layout or the addition/removal of objects within a room. All of these factors can influence the performance and reliability of GOTCHA.

The proposed system GOTCHA employs an active sensing method that primarily relies on pre-defined emitted sounds. As demonstrated by the experimental results in Chapter 3 - Section 3.3, GOTCHA performs robustly across different time periods. Therefore, this chapter does not focus on environmental changes by background acoustic variations over time. Instead, we focus on environmental changes caused by structural variations in spatial layout, which directly affect the propagation path of the acoustic signals emitted from the speaker. Based on these structural variations in spatial layout, two main scenarios can be considered. First, a room may not remain invariant and can undergo changes over time. Adaptation in this case is necessary to maintain reliable of intruder detection model. Second, it is not always desirable to train a model in situ, as this requires time and computational resources. A more practical approach is to deploy a pre-trained model and adapt it quickly to a specific room using only a limited number of samples. This situation typically occurs at initial deployment or when the device is relocated from one room to another. Next, we discuss these two adaptation scenarios in more detail.

4.1.1 Intra-room Environmental Adaptation

In practice, spatial structure changes within a room can occur frequently. Objects may be added or relocated to different positions at any time, for example a table or a chair is moved from a specific position to another one in the room, altering the room's acoustic landscape. Additionally, users might reposition the smart speaker within the same room based on personal preferences or usage needs. These daily changes in turn affects the propagation of acoustic signals. As a result, the signal pathways shift, potentially impacting the accuracy of detection.

Typically, to minimize deployment cost, users expect a single device to function reliably even under varying environmental conditions. This expectation raises the challenge of enabling a single GOTCHA to adapt effectively to structural changes in a single room environment.

Without adaptation, GOTCHA will classify samples in the new environment as anomalies since it has never encountered them in the training set. Therefore,

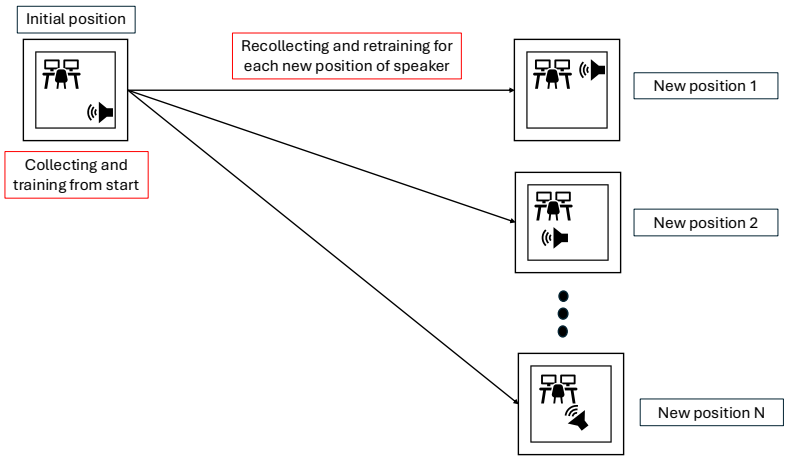


Figure 4.1: The process of adaptation with environmental changes happening within a room. It is required data recollecting and model retraining for every single new position.

recollecting data and retraining the model is necessary to maintain accuracy and reduce false alarms. The process of recollecting and retraining model are shown in Figure 4.1. However, the cost of recollecting data and retraining the model from scratch is non-trivial. As shown in Chapter 3-Section 3.3, achieving comparable performance requires over one hour of data collection and training on the new dataset. Therefore, full retraining is not considered the most practical option in real-world scenarios.

A GOTCHA device should be shipped with a pre-trained model. As we have shown in Section 3.3 it takes a significant effort and time to prepare GOTCHA for use in a specific location. It is therefore desirable to reduce the time until GOTCHA is ready for use and to reduce computational effort to prepare the system.

To achieve a more efficient and practical solution, leveraging a pre-trained model and fine-tuning it on the new dataset helps minimize retraining time and makes GOTCHA's adaptation more practical for real-world deployment.

4.1.2 Cross-room Environmental Adaptation

In addition to spatial changes occurring within a single room, there are also cases where end users may wish to move the smart speaker to a different room depending on their needs. This scenario can be considered a more generalized version of the within-room structural changes discussed earlier. When GOTCHA is relocated from its original room to a completely different one, the surrounding

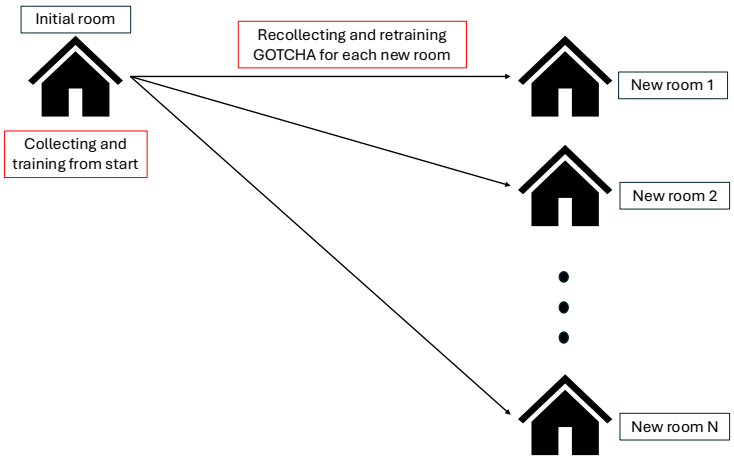


Figure 4.2: The process of adaptation with environmental changes across rooms. It is required data recollecting and model retraining for every single new room.

environment changes entirely. As a result, the intruder detection model trained on data from the original room may no longer produce accurate predictions in the new environment. This is due to the shift in spatial characteristics, which in turn affects the propagation patterns of acoustic signals.

Similar to the previously mentioned scenario,GOTCHA will misclassify the new samples in new environment, so it is required to recollect new dataset and retrain on new dataset as shown in Fig 4.2. And to reduce cost, end users are likely to prefer leveraging a single device that can operate across multiple rooms, rather than deploying a new device for each additional room. However, in this case, the change occurs on a much larger scale—the room's structure is almost entirely different, rather than being altered by the relocating of a few objects as in the single-room adaptation scenario. Moreover, background noise characteristics may also vary significantly when moving from one room to another. Nevertheless, as mentioned earlier, this thesis focuses on addressing environmental adaptation challenges primarily caused by spatial structural changes.

To address the challenge of adapting to a new environment, similar to the intra-room adaptation scenario, collecting new data and retraining the model is necessary. However, as previously discussed, such an approach requires considerable effort in terms of data recollection and model retraining. Therefore, leveraging a pre-trained model—previously trained in another environment—and fine-tuning it with only a small amount of new data offers a practical solution for achieving quick adaptation. Moreover, since this cross-room scenario is a more generalized case compared to intra-room adaptation, a successful adaptation approach in this

scenario can also be applied to solve the intra-room adaptation problem.

4.2 Model Adaptation for GOTCHA

In general, the ultimate goal of adaptation is to enable GOTCHA to quickly adapt to environmental changes, thereby offering a cost-effective hardware solution that remains functional across diverse environmental conditions. This adaptability helps reduce deployment costs for end users while maintaining system reliability. Due to the influence of environmental structural changes on the propagation path of sound, GOTCHA faces challenges in making accurate decisions when encountering previously unseen examples. This necessitates data recollection and model retraining in the new environment. However, collecting new data and retraining the model from scratch is time-consuming, making GOTCHA less practical for real-world deployment.

To address challenges raised by intra-room environmental changes or cross-room environmental changes that mentioned above, there are a lot of potential approaches that can be considered. Basically, an intruder detection model can be retrained from scratch or fine-tuned on small number of samples in new dataset to accommodate those environmental changes. The former method requires a lot of efforts on retraining and collecting new dataset while the latter approach can benefit from transferred knowledge acquired by pre-trained model, allowing adaptation through fine-tuning on a limited number of new samples. Thus, many existing work focus on utilizing pre-trained model on big dataset or retraining model with less data sample as possible. As mentioned in Chapter 2-Section 2.6, transfer learning is one of emerging method to solve the problem of domain shift or domain adaptation. Transfer learning can utilize pre-trained model on large dataset to transfer knowledge from related domain to solve the task on new domain. The concept of transfer learning provides a promising framework to address the above issues posed by environmental changes, as it enables the reuse of previously learned knowledge in new environment but related settings. Regardless of the type of environmental changes, data collected in the new environment can be regarded as samples from a new domain that is related to the original one, since they belong to the same kind of data and are served for the same task. From this perspective, GOTCHA adaptation naturally aligns with the principles of domain adaptation in transfer learning.

By applying the concept of transfer learning to address the domain adaptation

problem, GOTCHA can leverage the knowledge embedded in the pre-trained model from previous environments, thus significantly reducing the effort required compared to training a new model from scratch. Initially, first approach that we considered is using a pre-trained model on large dataset as Audio Spectrogram Transformer Gong et al. [2021] for our downstream task. However, due to the big size of those models, even the fine-tuning process still takes longer than desired. Moreover, all the chirping sound (both normal and abnormal ones) often is included in a same class in big datasets for those pre-trained models. We also thought about employing LoRA Hu et al. [2021] to fine tune GOTCHA on new dataset. Although this method can work very well on fine-tuning Large Language Model, it still hurts performance of GOTCHA. Since the size of GOTCHA is not big compared to billion-parameters Large Language Models and rank of adaptation matrices is much smaller than main modules, applying LoRA Hu et al. [2021] to GOTCHA might lead to information loss.

To balance between information loss and the costs of retraining and collecting data from scratch, we propose an approach that fine-tunes GOTCHA decoder part for every single time it has to deal with new environment. AE network is particularly effective at learning compact representations of data in a lower-dimensional latent space. Leveraging this capability, GOTCHA maps acoustic signals into a compressed latent space using the encoder. The encoder, trained during the initial deployment of GOTCHA, is retained across environments. When adapting to a new environment, only the decoder part is fine-tuned, enabling efficient and practical adaptation without retraining the entire model from scratch. By utilizing the latent space representations learned during initial training, only the decoder component of GOTCHA is updated during each adaptation phase. As a result, only half of the model parameters need to be fine-tuned in the new environment. This significantly reduces the computational cost associated with retraining, making the adaptation process more efficient and practical. In real-world scenarios, it is challenging to accurately determine the exact moment when the environment surrounding GOTCHA changes. To address this issue, we propose two deployment settings for practical implementation. The first is a manual adaptation mode, which allows users to explicitly trigger the adaptation process whenever the surrounding environment changes. The second is an automatic adaptation mode, inspired by the approach in Danti and Innocenti [2023], where GOTCHA is periodically fine-tuned on newly collected data at predefined intervals (e.g., twice per day). This enables continuous adaptation to environmental changes without requiring user intervention.

4.3 Environmental Adaptation Evaluation

To evaluate the effectiveness of the adaptation process, we simulate a realistic scenario by relocating GOTCHA from one environment to another and monitoring its performance in each new setting. When GOTCHA is moved into new environment, it will have new start by collecting new samples and retrain on those new samples. This process will repeat for any new environment shifting. In this section, we conduct experiments to evaluate the adaptation capability of GOTCHA. We also aim to identify key factors that influence its performance, such as the optimal amount of data required in the new environment and the time needed to complete the adaptation process.

4.3.1 Experiment Design

On the first time GOTCHA is activated, it will be in collecting data mode. Following approximately one hour of data collection, the anomaly detection model undergoes its initial training phase. Subsequently, whenever GOTCHA is relocated to a different position or room, the adaptation mode will be activated. As previously described, the encoder of GOTCHA is retained from the initial training phase. During each activation of the adaptation process, only the decoder component is fine-tuned using newly collected data. Consequently, upon triggering the adaptation mode, a shorter data collection phase (compared to the initial training phase) is conducted, followed by the fine-tuning of the decoder to enable adaptation to the new environment.

In the experiment described in Chapter 3, we collected data and evaluated GOTCHA in three different rooms at various times, which can be considered as three distinct environments. Instead of repeating the entire experimental process, we simulate the environmental transition scenario by leveraging the previously collected data to evaluate the adaptation capability of GOTCHA across these three rooms. One of the rooms is selected as the initial environment, where the model is fully trained. This trained model is then fine-tuned on data from the other two rooms. It is worth noting that we also have data corresponding to intra-room changes, where GOTCHA was repositioned within the same room. However, as mentioned in this Chapter -Section 4.2, the cross-room environmental change scenario is a more general case of intra-room changes. Therefore, to avoid redundancy and reduce experimental workload, we focus our evaluation solely on the cross-room adaptation scenario. In this study, we primarily focus

on spatial differences that affect sound propagation rather than background noise. Consequently, reusing the previously collected data does not affect the objectivity or validity of the adaptation experiment. In addition, another point worth noting is the windowing method used in the experiment described in Chapter 3. In experiment in Chapter 3 - Section 3.3, we showed the effect of windowing methods on improving F1-score and reducing false alarms. However, windowing method use the most predicted label as label represent for a time window frame, so it might collapse the effect of environment adaptation approaches. Because when using majority as prediction for a window slide instead of single chirp prediction, the performance could be same although GOTCHA made more single false prediction in new environment than previous environment. Thus, in this chapter, we do not apply windowing method to final results to evaluate the performance of environment adaptation approaches.

To evaluate the effectiveness of our proposed approach, we establish two baseline methods for comparison and performance assessment. The first baseline involves fully retraining GOTCHA from scratch whenever the environment changes. This baseline is used to evaluate whether a model fine-tuned on a small subset of new data can achieve performance comparable to a model that is fully retrained from scratch. The second baseline reuses the pre-trained model from the initial environment, but fine-tunes the entire model (both encoder and decoder) on the new data. This serves to assess whether reducing the number of trainable parameters by freezing the encoder — as in our proposed approach — can still achieve comparable performance to full-model fine-tuning.

To ensure that performance comparison are statistically meaningful, we use three different random seeds for each approach when evaluating on a dataset. The final results are reported as the average over these three runs. This reduces the impact of randomness in model initialization and training, providing a more robust evaluation of each method's effectiveness. Furthermore, to address the question of the optimal quantity of new data necessary for successful adaptation, we perform experiments using varying training set sizes in the new environment. We gradually increase the number of training samples from 10 to 5000. The sizes follow an approximately exponential growth pattern, nearly doubling at each step: 10, 50, 100, 200, 500, 1000, 2000, and 5000.

4.3.2 Experiment Results

The experiments in this section are conducted to evaluate the effectiveness of our proposed approach for environmental adaptation. In addition, they aim to address two key research questions: how long it takes for GOTCHA to achieve acceptable performance in a new environment, and how much data is required for GOTCHA to perform effectively in the new setting. To evaluate and compare the performance of GOTCHA with the baselines, we use F1-score and False Positive Rate, consistent with the evaluation metrics employed in Chapter 3-Section 3.3. While the F1-score measures GOTCHA's ability in detecting anomalies (intruder), the false alarm rate evaluates its ability to minimize unnecessary alerts that may cause inconvenience to end users. We additionally measure and compare the retraining time on new data to highlight the computational efficiency of our proposed approach relative to the baseline methods.

F1-score Comparison

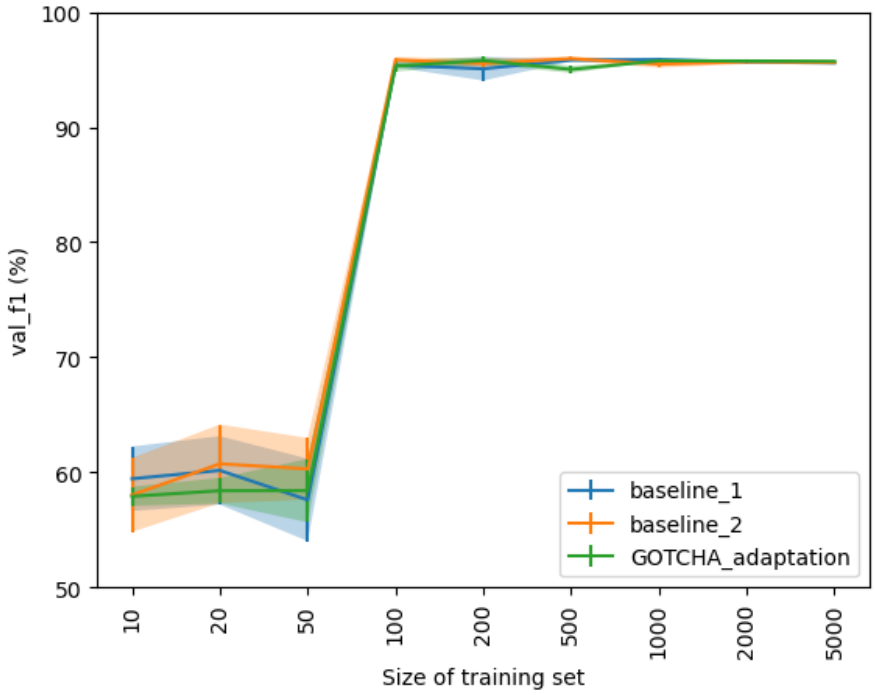
In term of F1-score, the higher F1-score indicate that a physical intrusion detection system is more effective at detecting intrusion threats. We present the F1-score results for experiments conducted in rooms R2 and R3 in Fig 4.3. The F1-scores are reported for both the validation and testing sets. Overall, the results across validation and testing sets show high consistency, following similar trends in both rooms. Two key phases can be observed: the first phase when the training set contains fewer than 100 samples, and the second phase when the training set contains 100 samples or more.

In the first phase, when the number of training samples is fewer than 100, the F1-scores on both the validation and testing sets for rooms R2 and R3 remain relatively low, all below 70%. Moreover, the variance across seeds is quite large, and there is no clear upward (or downward) trend in F1-score as the number of training samples gradually increases. This suggests that, with such a small amount of data, the GOTCHA anomaly detection model is not able to effectively learn the underlying patterns of the training data. As a result, many of its predictions appear to be random, yielding F1-scores only slightly better than chance. In fact, in room R3, the model trained from scratch even performs worse than random guessing, with an F1-score below 50%.

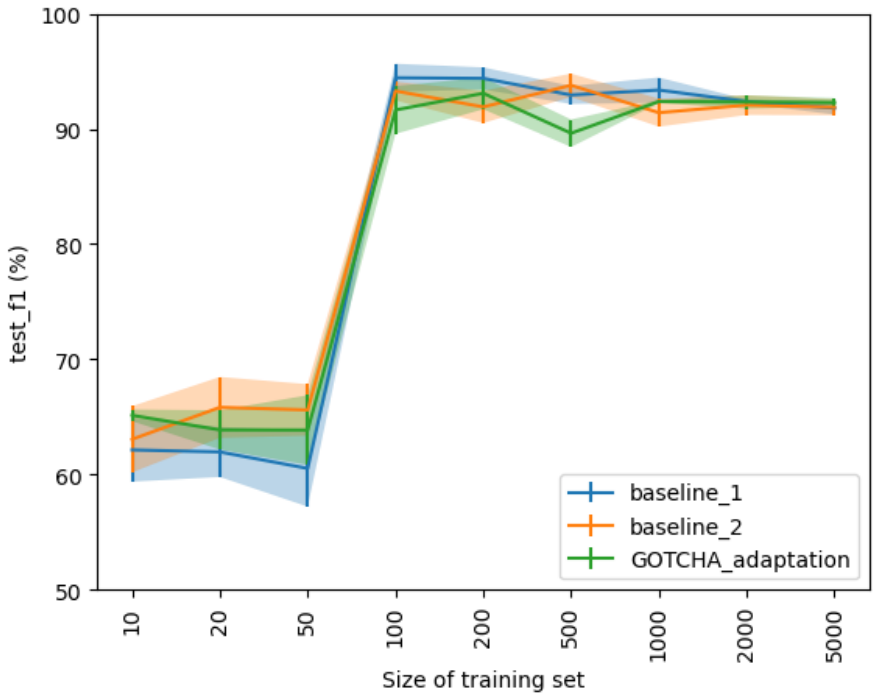
In the second phase, when the training set size reaches 100 samples and continues to grow, the performance of GOTCHA shows a clear improvement, and the variance across the three random seeds becomes significantly smaller compared

to the previous phase. In the data from room R2, the results on the validation set are notably stable, with very low variance across seeds, almost no noticeable performance gain as the training set size increases and no clear difference between three approaches. On the testing set, however, the variance is larger and no consistent trend is observed with increasing training size. Interestingly, the best testing results are achieved before the training set size reaches 1000. In the case of room R3, the results on both the validation and testing sets are more consistent compared to R2. While the performance of the two approaches — retraining the entire model from the best checkpoint of the initial room and fine-tuning only the decoder part — remains similar and stable (F1-score curves are almost flat) as the training size increases, the model retrained from scratch shows a performance drop at the training size of 200 samples. At this point, the F1-score falls below 80% and exhibits relatively high variance. This drop is likely caused by the randomness in sample selection for retraining, where the selected samples may include segments with more background noise or lower quality than usual. This phenomenon highlights an important issue for future work: how to ensure the quality of new data used for adaptation, as it directly impacts the model's ability to maintain consistent performance in new environments.

Overall, there is no significant difference in F1-score among the three approaches once the training set size reaches 100 samples or more.

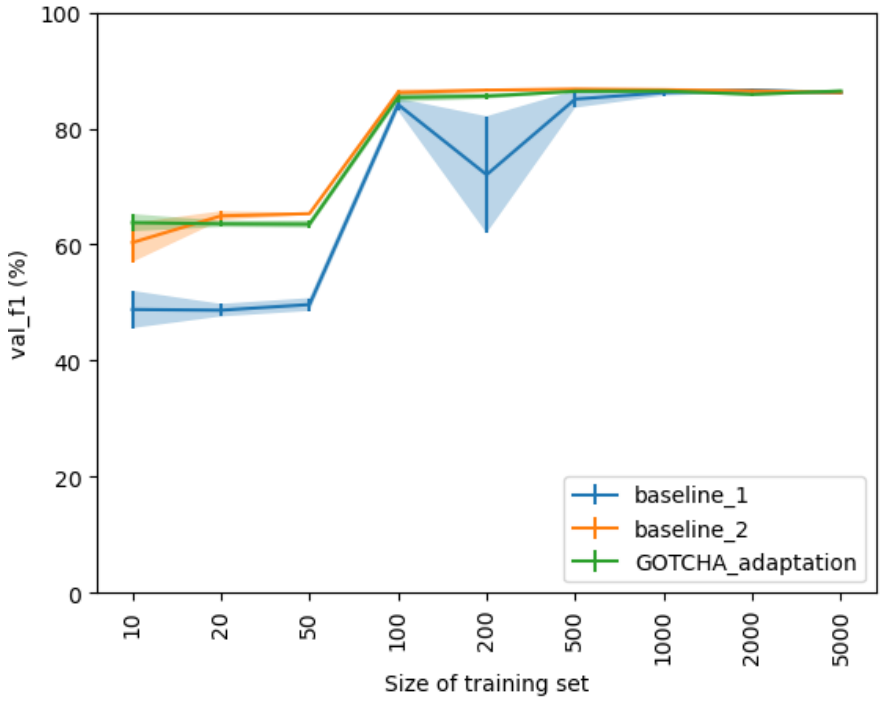


(a) F1 comparison between three approaches on validation set in room 2

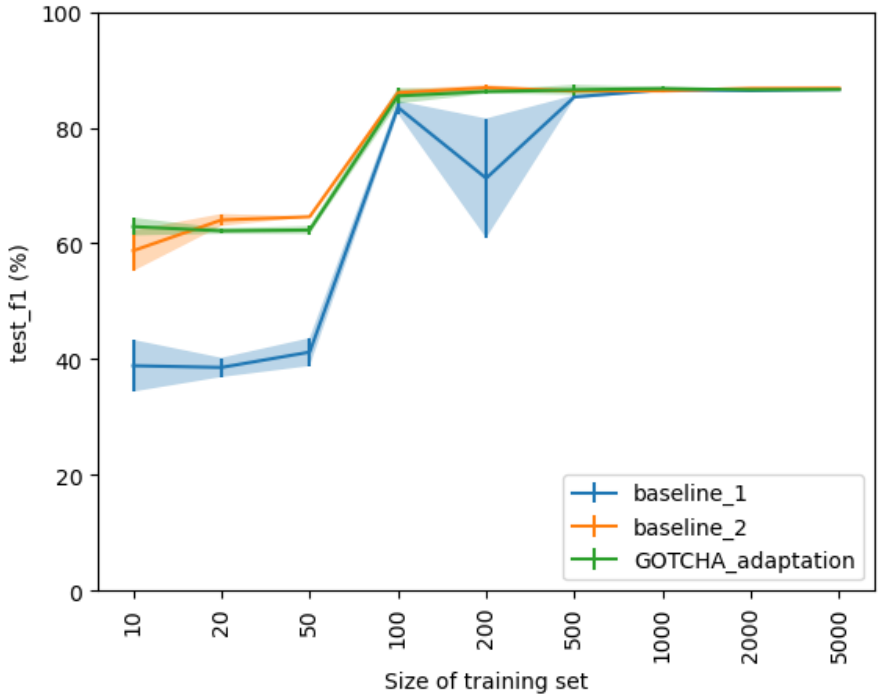


(b) F1 comparison between three approaches on test set in room 2

Figure 4.3: The chart compares the F1-score with confidence intervals across different training set sizes. The blue line represents the method of retraining from scratch, the orange line corresponds to the method of fine-tuning the best model of initial room on new data, and the green line indicates our proposed approach — fine-tuning only the decoder part of the best model of initial room on new data. (Part 1).



(a) F1 comparison between three approaches on validation set in room 3



(b) F1 comparison between three approaches on test set in room 3

Figure 4.3: The chart compares the F1-score with confidence intervals across different training set sizes. The blue line represents the method of retraining from scratch, the orange line corresponds to the method of fine-tuning the best model of initial room on new data, and the green line indicates our proposed approach — fine-tuning only the decoder part of the best model of initial room on new data. (Part 2).

False Positive Rate Comparison

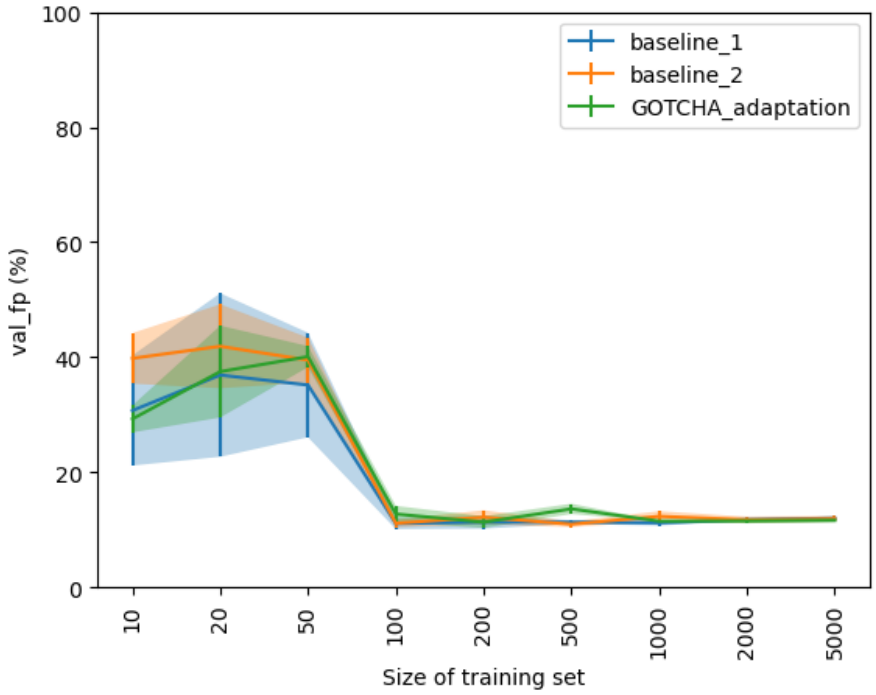
In addition to evaluating the prediction accuracy of GOTCHA, another crucial metric for any intruder detection system is the false positive rate. This metric quantifies how often the system generates false alerts; the lower the FPR, the more reliable the system is considered. The experimental results of FPR are presented in Fig 4.4. Similar to the trend observed with F1-score, we can identify two distinct phases separated by the point at which the training data size reaches 100 samples. Notably different behaviors in FPR are observed before and after this threshold.

In the phase where the training set size is smaller than 100 samples, the FPR in Room R2 remains relatively high (ranging from 20% to 40%) on both the validation and testing sets, with large variance—similar to what was observed in the F1-score results. This can be explained by the same reasoning: the training data is too limited for the model to effectively learn the underlying patterns, resulting in largely random predictions. In contrast, Room R3 shows a lower and more stable FPR compared to Room R2. However, this is due to the model in Room R3 predominantly misclassifying abnormal samples as normal ones, which reduces the FPR but also significantly degrades the F1-score. This indicates that the model misses a large number of anomalies, making it unreliable for anomaly detection despite its seemingly low FPR.

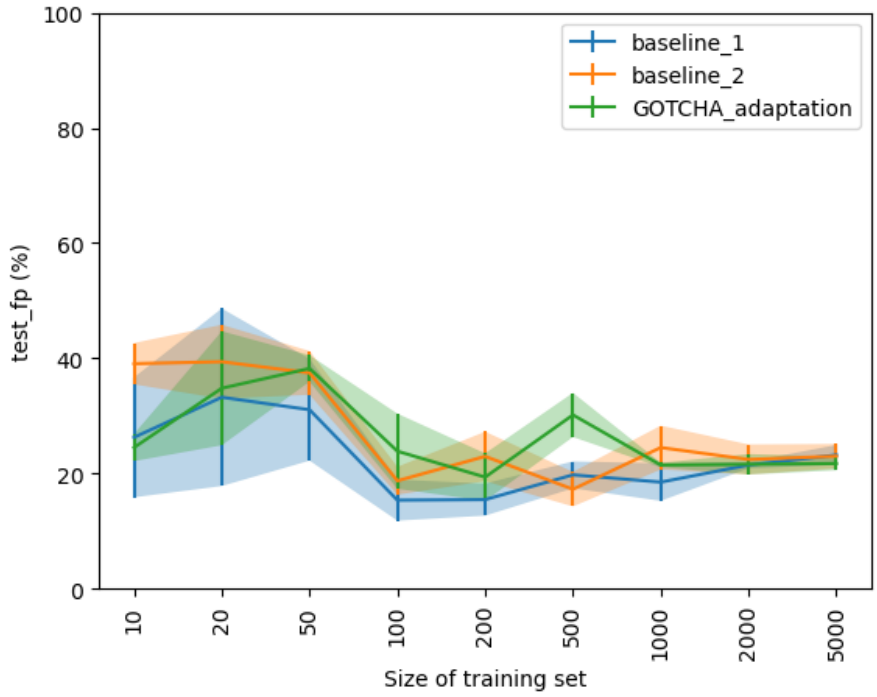
In the phase where the training set size exceeds 100 samples, the FPR in Room R2 shows noticeable improvement compared to the previous phase on both validation and testing sets. While the FPR on the validation set remains relatively stable at around 10%, the testing set exhibits a slight upward trend as the training set size increases. This indicates a trade-off introduced by using a larger training set—more data helps the model detect a greater number of anomalous points, but it also leads to an increase in false alarms. The lack of a clear trend and the relatively high variance in FPRs on the testing set of Room R2 can be attributed to the randomness in sampling training data across the three different seeds. Meanwhile, for Room R3, once the training set size reaches 100 samples or more, the FPR becomes relatively stable, staying below 20% on the validation set and around 20% on the testing set. Consistent with the F1-score results in Room R3, an outlier appears at the 200-sample mark, likely due to the effect of randomly selected samples for the training set. This again highlights the sensitivity of performance to the quality and representativeness of the adaptation data.

In general, there is no significant difference in false alarm rates among the three

approaches when the training set contains 100 samples or more. All approaches achieve their best false alarm rates at approximately 20%. However, it is important to note that in this experiment, we intentionally excluded the windowing method described in Chapter 3 in order to clearly observe the differences among the three adaptation strategies. As demonstrated in previous experiments, applying the windowing method can significantly enhance detection performance—GOTCHA is capable of achieving a false positive rate (FPR) below 10% when the method is employed.

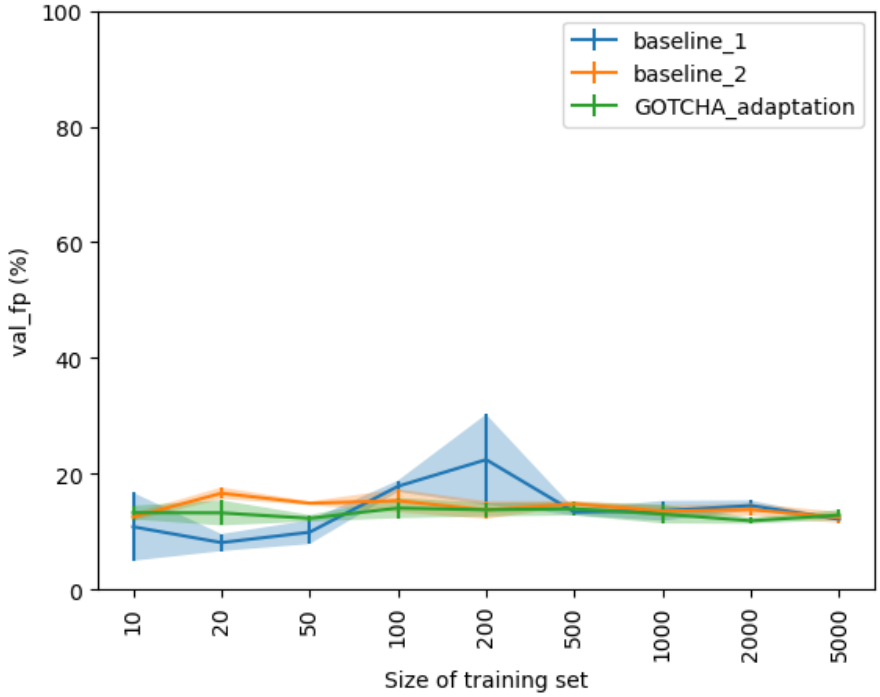


(c) False positive rate comparison between three approaches on validation set in room 2

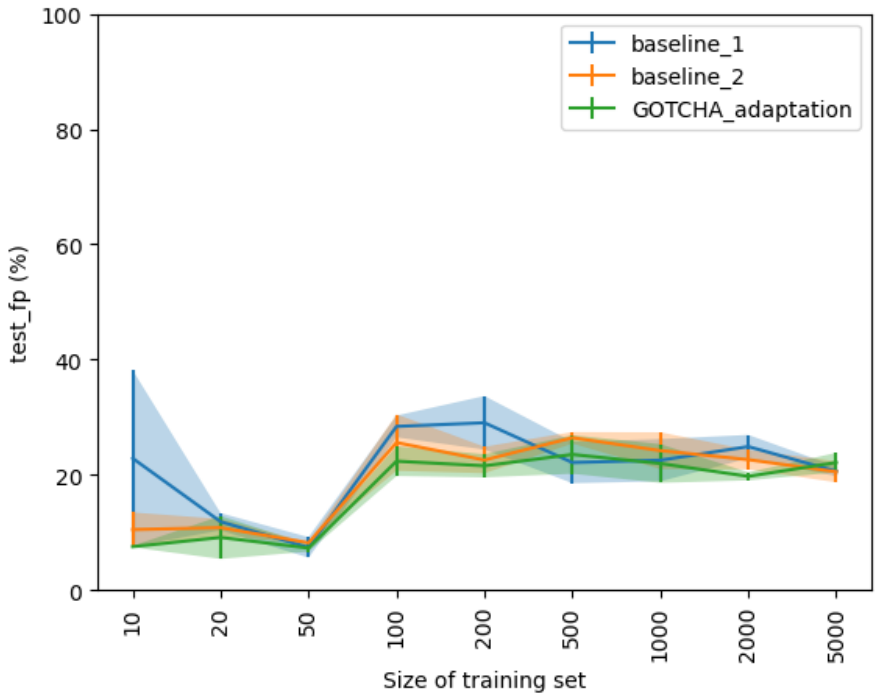


(d) False positive rate comparison between three approaches on test set in room 2

Figure 4.4: The chart compares the False positive rate with confidence intervals across different training set sizes. The blue line represents the method of retraining from scratch, the orange line corresponds to the method of fine-tuning the best model of initial room on new data, and the green line indicates our proposed approach — fine-tuning only the decoder part of the best model of initial room on new data. (Part 1).



(a) False positive rate comparison between three approaches on validation set in room 3

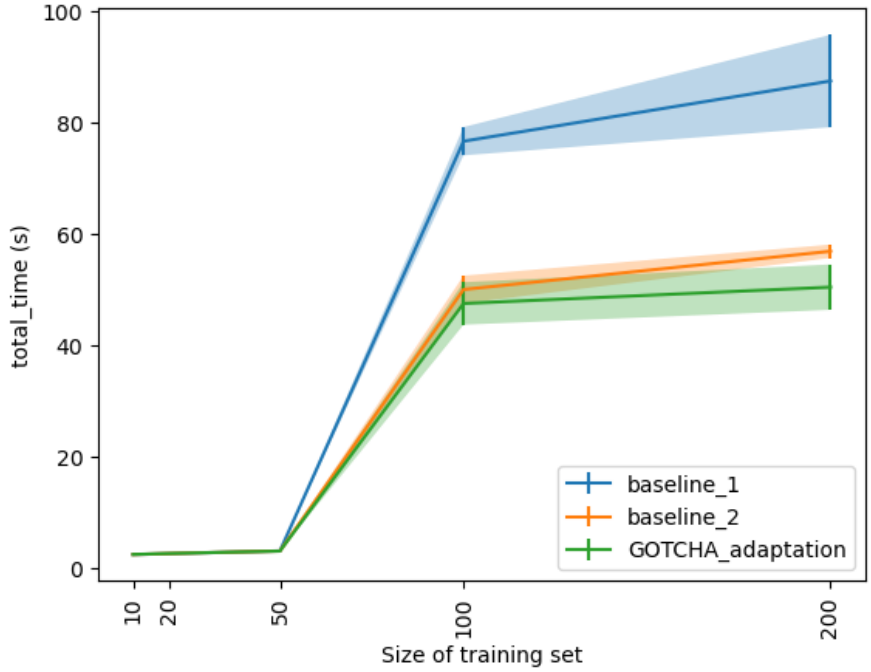


(b) False positive rate comparison between three approaches on test set in room 3

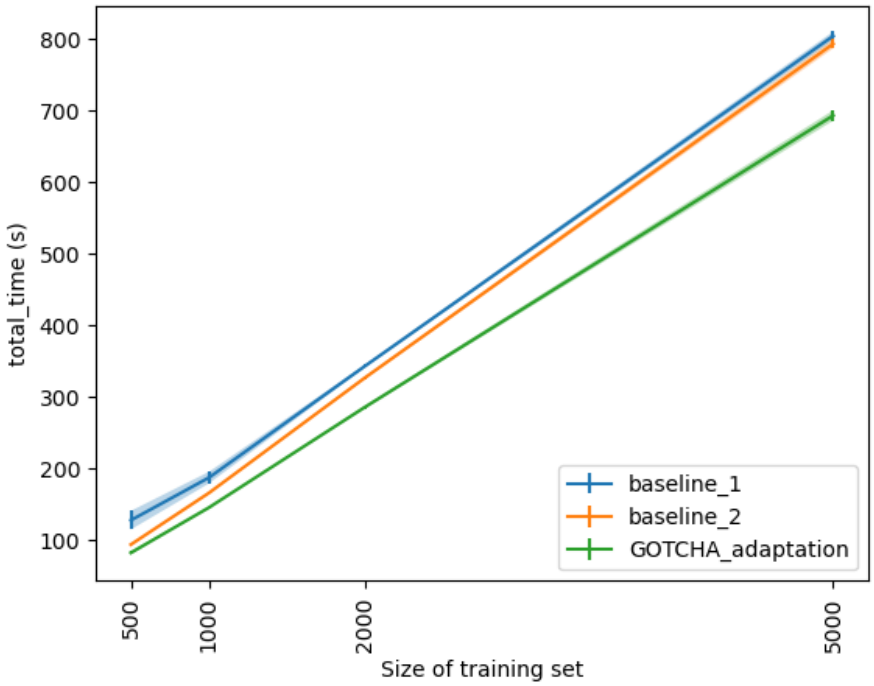
Figure 4.4: The chart compares the false positive rate with confidence intervals across different training set sizes. The blue line represents the method of retraining from scratch, the orange line corresponds to the method of fine-tuning the best model of initial room on new data, and the green line indicates our proposed approach — fine-tuning only the decoder part of the best model of initial room on new data. (Part 2).

Computational Analysis

In addition to accuracy and false alarm rate metrics, evaluating GOTCHA's adaptation also requires considering the system's computational complexity to estimate the cost of retraining process. We measure the training time from the start of the retraining process until the model stops improving on the validation set, at which point early stopping is triggered or retraining process reach a predetermined epochs. To determine the optimal number of training samples for the adaptation process, we conduct experiments with various training set sizes: 10, 20, 50, 100, 200, 500, 1000, 2000, and 5000. As previously mentioned, for each training set size, the reported results are averaged over three independent runs with different random seeds to ensure the statistical significance of observed differences. The results are showed in Fig 4.5. The observed results can be divided into two main phases based on the training set size: when the size is smaller than 500 samples and when it reaches 500 samples or more. During the adaptation experiments in Room 2, in the first phase, when the training set size is smaller than 50, the retraining time for all three approaches was very short (less than 5 seconds). However, as discussed in the F1-score and FPR results, models in this phase were almost unable to learn meaningful patterns. When the training set size increased to 100 and 200 samples, it was observed that retraining from scratch required significantly more time compared to the other two approaches: while retraining from scratch took approximately 80–90 seconds, the other two approaches needed less than 60 seconds. From 500 samples onwards, the retraining time for all three approaches exhibited a roughly linear increase with respect to the training set size. Furthermore, it can be noted that when the training set size becomes larger, the approach that freezes the encoder part consistently achieved about 10% faster retraining time compared to the other two methods. The experiments conducted in Room 3 yielded similar results in terms of retraining time and trends with respect to the training set size as observed in Room 2, demonstrating the consistency of GOTCHA's adaptation retraining time across different environments.

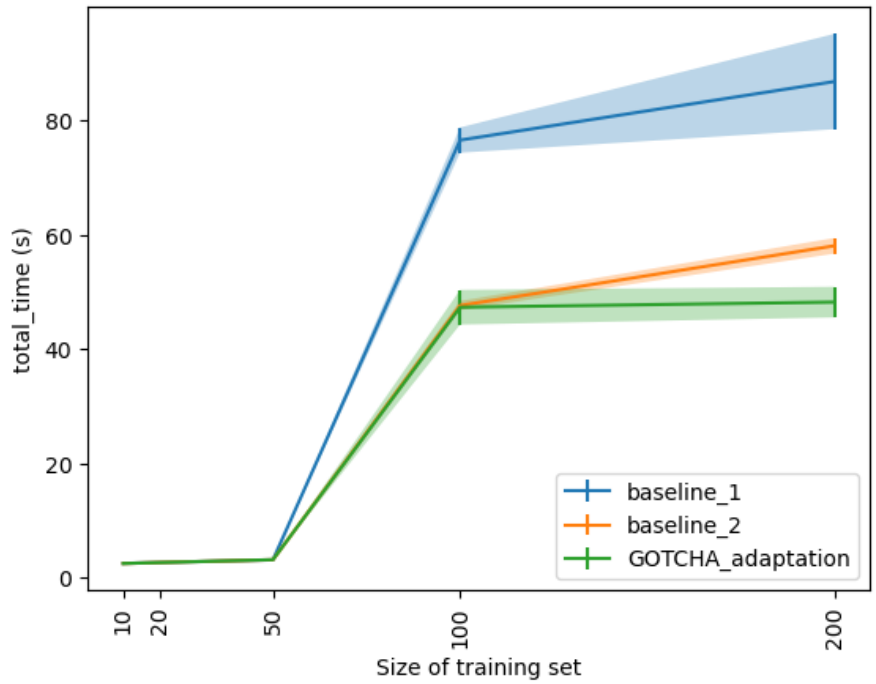


(c) Execution time comparison between three approaches in room 2

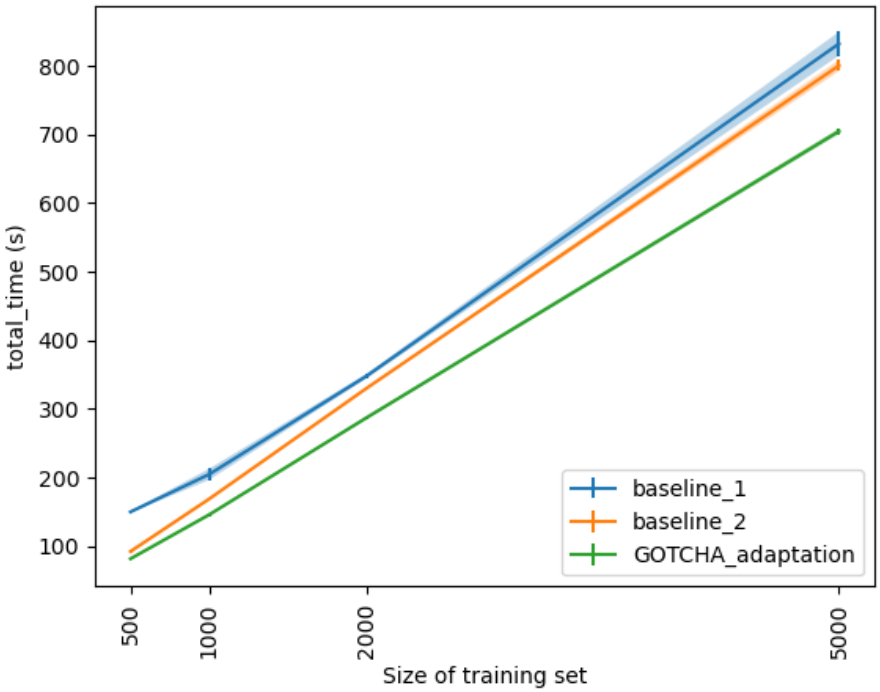


(d) Execution time comparison between three approaches in room 2

Figure 4.5: The chart compares the execution time of retraining process with confidence intervals across different training set sizes. The blue line represents the method of retraining from scratch, the orange line corresponds to the method of fine-tuning the best model of initial room on new data, and the green line indicates our proposed approach — fine-tuning only the decoder part of the best model of initial room on new data. (Part 1).



(a) Execution time comparison between three approaches in room 3



(b) Execution time comparison between three approaches in room 3

Figure 4.5: The chart compares the execution time of retraining process with confidence intervals across different training set sizes. The blue line represents the method of retraining from scratch, the orange line corresponds to the method of fine-tuning the best model of initial room on new data, and the green line indicates our proposed approach — fine-tuning only the decoder part of the best model of initial room on new data. (Part 2).

4.3.3 Discussion

Based on the experimental results on F1-score, FPR, and retraining time for GOTCHA adaptation across two different environments, we summarize the following key observations. With a training set size of 100 samples, GOTCHA adaptation can start to operate effectively, achieving satisfactory performance in both F1-score and FPR. The adaptation approach that freezes the encoder part and fine-tunes only the decoder part proves effective, achieving comparable or even better results than fully fine-tuning the entire model or retraining from scratch, in terms of both F1-score and FPR. However, the random selection of samples for the training set also leads to some outlier results, raising concerns about the need to control the quality of data used for retraining, especially when only a small amount of data is available. In terms of retraining time, freezing the encoder and tuning only the decoder part significantly reduces retraining time compared to the other two approaches. Overall, the results demonstrate that GOTCHA can effectively handle environmental changes by freezing the encoder and fine-tuning the decoder in less than one minute while maintaining strong performance.

4.4 Summary

In this chapter, we presented the environmental adaptation strategy for GOTCHA. Given the practical requirements where surrounding environments continuously change, and the need to optimize costs for end users, it is crucial to develop an approach that allows GOTCHA to quickly adapt to new environments without significant setup efforts or extensive data collection. We surveyed and evaluated possible strategies and ultimately proposed an approach where the encoder part of the AE is frozen while only the decoder part is fine-tuned using new environmental data.

We conducted experiments and measured key metrics including F1-score, FPR, and retraining time for the GOTCHA adaptation process based on datasets collected from the experiments described in Chapter 3. The experimental results demonstrate that GOTCHA can achieve an F1-score exceeding 90% with only 100 new training samples, requiring less than one minute for retraining. These findings confirm the feasibility and effectiveness of our proposed method in balancing performance and deployment cost for practical applications of GOTCHA.

Chapter 5

Conclusion and Future Work

5.1 Conclusion

With the rapid advancement of technology in the IoT era, smart home security applications are becoming increasingly prevalent in modern life. Leveraging smart home devices to integrate additional security features alongside their primary functions allows users to maximize the value of their investments. Recent studies have explored the potential of utilising smart devices such as smart speakers, smartwatches, and smart cameras for home security applications, as discussed in Chapter 2. Integrating smart devices into home security presents several challenges, including accuracy, operational costs, and the complexity of setup and configuration.

In this thesis, we propose a method to employ smart speakers as intrusion detection devices using active acoustic sensing. We trained an autoencoder model and deployed it on a prototype simulating an affordable smart speaker for human detection. To evaluate its performance, we collected data and conducted experiments in three different rooms, including two office rooms and one bedroom. Experimental results demonstrate that our proposed model achieves up to 99.2%, 99.0% and 91.4% F1-score on corresponding testing sets of 3 different rooms while maintaining the false alarm rate below 10%. Our approach is deployed on low-cost hardware, making it accessible to a wide range of everyday users. However, it still has certain limitations in detecting intruders, such as susceptibility to environmental noise and blind spots within a room, which can reduce detection accuracy or lead to false alarms.

In practice, changes in the surrounding environment can significantly affect the

performance of intrusion detection systems based on acoustic sensing. To address this challenge and optimize cost-effectiveness for end users, we propose an efficient solution that enables GOTCHA to quickly adapt to new environments. Specifically, we investigate and evaluate a transfer learning approach that leverages the encoder part of the initially trained autoencoder (kept frozen) while fine-tuning only the decoder. By utilising previously collected data, we assess the adaptability of GOTCHA by designating one room as the initial environment and then applying fine-tuning and evaluation on data from two additional rooms. Experimental results show that GOTCHA achieves over 90% F1-score and under 20% false positive rate (FPR) on both target rooms using only 100 training samples (without applying windowing methods). Moreover, the adaptation time remains under two minutes, demonstrating that fine-tuning only the decoder not only accelerates the adaptation process but also yields comparable results to retraining the entire model or fine-tuning the full autoencoder. These findings confirm the practical feasibility of deploying GOTCHA in dynamically changing environments.

Based on the experimental results, we successfully addressed the two research questions posed in Chapter 1. Our proposed system, GOTCHA, has demonstrated its potential to function as a physical intrusion detection system by leveraging active acoustic sensing combined with an Autoencoder (AE) on low-cost hardware. Additionally, the proposed transfer learning strategy — fine-tuning only the decoder part of the AE — has proven to be an effective solution for enabling GOTCHA to adapt to environmental changes efficiently. These findings highlight the feasibility of integrating smart devices into home security systems and pave the way for applying machine learning on affordable smart devices, with the aim of reducing the cost of building smart homes for end users.

The results obtained in this thesis demonstrate that active acoustic sensing can be effectively employed for the task of human presence detection. Furthermore, we release a public dataset that can serve as a benchmark for future research in this domain. Subsequent algorithms can be evaluated on this dataset using our results as a baseline. In practice, the promising performance of our system on low-cost hardware also highlights the potential for integrating off-the-shelf smart devices, such as smart speakers, into alarm systems. This opens up new application directions for enhancing modern security systems, particularly in scenarios where conventional camera-based methods underperform, such as low-light environments or occluded areas.

5.2 Future Work

While GOTCHA has demonstrated its practical applicability with an F1-score above 90%, an acceptable FPR, and rapid adaptation to new environments with less than 2 minutes of retraining, we acknowledge that there remain several aspects that can be further improved in future work.

First, GOTCHA is not intended to replace passive acoustic sensing methods or other intrusion detection approaches based on camera or RF signals. Instead, it can be integrated with passive sensing in alternating cycles to enhance overall accuracy. Furthermore, combining acoustic-based and camera-based systems may help address blind spots inherent to each modality when used in isolation.

Second, the current use of chirp sounds may cause discomfort to users. Therefore, exploring alternative sound types that are more pleasant or deploying them in a way that minimizes user disturbance should be considered. We plan to investigate different audio signals and explore low-cost hardware alternatives that still support wide-frequency operations to enhance GOTCHA's usability and performance.

Finally, there are a few challenges arise during the adaptation process. These include ensuring the quality of the newly collected data when only a limited number of samples are available, and the need for automatic adjustment of hyperparameters such as anomaly detection thresholds and window sizes when encountering a new environment. We are currently exploring methods from the fields of domain shift and domain adaptation to further enhance GOTCHA's adaptability and robustness.

References

Shivam Agarwal, Kiran Khatter, and Devanjali Relan. Security threat sounds classification using neural network. In *2021 8th International Conference on Computing for Sustainable Global Development (INDIACom)*, pages 690–694, 2021.

Prashant Ahire, Meghana Lokhande, Rutuja Patil, Twinkle Shirsath, Sumedha Zaware, and Tejaswini Zalki. Intrusion detection using camera and alert management. In *2023 International Conference on Inventive Computation Technologies (ICICT)*, pages 1117–1121, 2023. doi: 10.1109/ICICT57646.2023.10134400.

Najeeb Faud Al-Khalli, Saud Alateeq, Mohammed Almansour, Yousef Alhassoun, Ahmed B. Ibrahim, and Saleh A. Alshebeili. Real-time detection of intruders using an acoustic sensor and internet-of-things computing. *Sensors (Basel, Switzerland)*, 23, 2023. URL `https://api.semanticscholar.org/CorpusID:259582958`.

Amr Alanwar, Bharathan Balaji, Yuan Tian, Shuo Yang, and Mani Srivastava. Echosafe: Sonar-based verifiable interaction with intelligent digital agents. In *Proceedings of the 1st ACM Workshop on the Internet of Safe Things*, SafeThings'17, page 38–43, New York, NY, USA, 2017. doi: 10.1145/3137003.3137014.

Rosa Maria Alsina-Pagès, Joan Navarro, Francesc Alías, and Marcos Hervás. homesound: Real-time audio event detection based on high performance computing for behaviour and surveillance remote monitoring. *Sensors (Basel, Switzerland)*, 17, 2017. URL `https://api.semanticscholar.org/CorpusID:23015307`.

"Amazon". "amazon alexa", n.d. URL `https://www.apple.com/siri.`

"Apple". "apple siri", n.d. URL `https://www.apple.com/siri.`

Yang Bai, Li Lu, Jerry Cheng, Jian Liu, Yingying Chen, and Jiadi Yu. Acoustic-based sensing and applications: A survey. *Comput. Networks*, 181:107447, 2020. doi: 10.1016/J.COMNET.2020.107447. URL https://doi.org/10.1016/j.comnet.2020.107447.

Akshay Bharadwaj K H, Deepak, V Ghanavanth, Harish Bharadwaj R, R Uma, and Gowranga Krishnamurthy. Smart cctv surveillance system for intrusion detection with live streaming. In *2018 3rd IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT)*, pages 1030–1035, 2018. doi: 10.1109/RTEICT42901.2018.9012234.

M. H. Bhuyan, D. K. Bhattacharyya, and J. K. Kalita. Network anomaly detection: Methods, systems and tools. *IEEE Communications Surveys & Tutorials*, 16(1):303–336, 2013. doi: 10.1109/SURV.2013.052213.00046. URL http://ieeexplore.ieee.org/document/6524462.

I. Bose and R. K. Mahapatra. Business data mining—a machine learning perspective. *Information & Management*, 39(3):211–225, 2001. doi: 10.1016/S0378-7206(01)00091-X.

Hang Chen, Dongfang Chen, and Xiaofeng Wang. Intrusion detection of specific area based on video. In *2016 9th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI)*, pages 23–29, 2016. doi: 10.1109/CISP-BMEI.2016.7852676.

Kah-Meng Cheong, Yih-Liang Shen, and Taishih Chi. Active acoustic scene monitoring through spectro-temporal modulation filtering for intruder detection. *The Journal of the Acoustical Society of America*, 151 4:2444, 2022. URL https://api.semanticscholar.org/CorpusID:248053139.

Piero Danti and Alessandro Innocenti. Anomaly detection in a micro-chp using convolutional autoencoders and fine tuning for domain adaptation. *2023 7th International Conference on System Reliability and Safety (ICSRS)*, pages 124–131, 2023. URL https://api.semanticscholar.org/CorpusID:266894015.

Stijn Denis, Rafael Berkvens, and Maarten Weyn. A survey on detection, tracking and identification in radio frequency-based device-free localization. *Sensors*, 19(23), 2019. ISSN 1424-8220. doi: 10.3390/s19235329. URL https://www.mdpi.com/1424-8220/19/23/5329.

Ha Manh Do, Karla Conn Welch, and Weihua Sheng. Soham: A sound-based human activity monitoring framework for home service robots. *IEEE Trans-*

actions on Automation Science and Engineering, 19:2369–2383, 2021. URL https://api.semanticscholar.org/CorpusID:236408129.

Yuan Gong, Yu-An Chung, and James Glass. AST: Audio Spectrogram Transformer. In *Proc. Interspeech 2021*, pages 571–575, 2021. doi: 10.21437/Interspeech.2021-698.

"Google". "google assistant", n.d. URL https://assistant.google.com.

Bing Han, Zhiqiang Lv, Anbai Jiang, Wen Huang, Zhengyang Chen, Yufeng Deng, Jiawei Ding, Cheng Lu, Wei-Qiang Zhang, Pingyi Fan, Jia Liu, and Yanmin Qian. Exploring large scale pre-trained models for robust machine anomalous sound detection. *ICASSP 2024 - 2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1326–1330, 2024. URL https://api.semanticscholar.org/CorpusID:268581877.

J. Edward Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *ArXiv*, abs/2106.09685, 2021. URL https://api.semanticscholar.org/CorpusID:235458009.

Global Market Insights. Smart speaker market size by intelligent virtual assistant (alexa, google assistant, siri, cortana, others), by application (personal, professional, commercial), industry analysis report, regional outlook, growth potential, competitive market share forecast, 2018 – 2024, 2018. URL https://www.gminsights.com/industry-analysis/smart-speaker-market.

Jinwoo Kim, Kyungjun Min, Minhyuk Jung, and Seokho Chi. Occupant behavior monitoring and emergency event detection in single-person households using deep learning-based sound recognition. *Building and Environment*, 181:107092, 2020. ISSN 0360-1323. doi: https://doi.org/10.1016/j.buildenv.2020.107092. URL https://www.sciencedirect.com/science/article/pii/S0360132320304686.

Yeonjoon Lee, Yue Zhao, Jiutian Zeng, Kwangwuk Lee, Nan Zhang, Faysal Hossain Shezan, Yuan Tian, Kai Chen, and XiaoFeng Wang. Using sonar for liveness detection to protect smart speakers against remote attackers. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.*, 4(1), mar 2020. doi: 10.1145/3380991. URL https://doi.org/10.1145/3380991.

Jiawei Li, Xiaoling Li, Bin Liu, Yang Liu, Weiwei Jia, and Jiarui Lai. Invisible-guardian: Using ensemble learning to classify sound events in healthcare

scenes. *2019 IEEE 4th Advanced Information Technology, Electronic and Automation Control Conference (IAEAC)*, 1:1277–1282, 2019. URL https://api.semanticscholar.org/CorpusID:211210752.

Jian Liu, Hongbo Liu, Yingying Chen, Yan Wang, and Chen Wang. Wireless sensing for human activity: A survey. *IEEE Communications Surveys & Tutorials*, 22(3):1629–1645, 2020. doi: 10.1109/COMST.2019.2934489.

Ali Bou Nassif, Manar Abu Talib, Qassim Nasir, and Fatima Mohamad Dakalbab. Machine learning for anomaly detection: A systematic review. *IEEE Access*, 9:78658–78700, 2021. doi: 10.1109/ACCESS.2021.3083060.

Paniti Netinant, Auttapon Amatyakul, and Meennapa Rukhiran. Alert intruder detection system using passive infrared motion detector based on internet of things. In *Proceedings of the 2022 5th International Conference on Software Engineering and Information Management*, ICSIM '22, page 171–175, New York, NY, USA, 2022. Association for Computing Machinery. ISBN 9781450395519. doi: 10.1145/3520084.3520112. URL https://doi.org/10.1145/3520084.3520112.

Guansong Pang, Chunhua Shen, Longbing Cao, and Anton Van Den Hengel. Deep learning for anomaly detection: A review. *ACM Comput. Surv.*, 54(2), mar 2021. ISSN 0360-0300. doi: 10.1145/3439950. URL https://doi.org/10.1145/3439950.

Khirod Chandra Sahoo and Umesh Chandra Pati. Iot based intrusion detection system using pir sensor. In *2017 2nd IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT)*, pages 1641–1645, 2017. doi: 10.1109/RTEICT.2017.8256877.

Suryansh Sharma, Prabhakar T. Venkata, Shalakha Singhal, Gogineni Gopi, Sunanth Kumar, and R. Venkatesha Prasad. B4w: A smart wireless intruder detection system. *ICC 2023 - IEEE International Conference on Communications*, pages 191–197, 2023. URL https://api.semanticscholar.org/CorpusID:264468882.

Oliver Shih and Anthony Rowe. Occupancy estimation using ultrasonic chirps. In *Proceedings of the ACM/IEEE Sixth International Conference on Cyber-Physical Systems*, ICCPS '15, page 149–158, New York, NY, USA, 2015. Association for Computing Machinery. ISBN 9781450334556. doi: 10.1145/2735960.2735969. URL https://doi.org/10.1145/2735960.2735969.

Jae Mun Sim, Yonnim Lee, and Ohbyung Kwon. Acoustic sensor based recognition of human activity in everyday life for smart home services. *International Journal of Distributed Sensor Networks*, 11, 2015. URL https://api.semanticscholar.org/CorpusID:39843971.

"Statista". "smart speakers - statistics & facts", 2024. URL https://www.statista.com/topics/4748/smart-speakers/#topicOverview.

The Irish Times. Almost half of adults own a smart speaker, finds radio industry report, 2023. URL https://www.irishtimes.com/business/2023/10/27/almost-half-of-adults-own-a-smart-speaker-finds-radio-industry-report/.

Jindong Wang, Cuiling Lan, Chang Liu, Yidong Ouyang, and Tao Qin. Generalizing to unseen domains: A survey on domain generalization. *IEEE Transactions on Knowledge and Data Engineering*, 35:8052–8072, 2021. URL https://api.semanticscholar.org/CorpusID:232110832.

Yadong Xie, Fan Li, Yue Wu, and Yu Wang. Hearfit: Fitness monitoring on smart speakers via active acoustic sensing. In *40th IEEE Conference on Computer Communications, INFOCOM 2021, Vancouver, BC, Canada, May 10-13, 2021*, pages 1–10. IEEE, 2021. doi: 10.1109/INFOCOM42981.2021.9488811. URL https://doi.org/10.1109/INFOCOM42981.2021.9488811.

Wei Xu, ZhiWen Yu, Zhu Wang, Bin Guo, and Qi Han. Acousticid: Gait-based human identification using acoustic signal. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.*, 3(3), sep 2019. doi: 10.1145/3351273. URL https://doi.org/10.1145/3351273.

Naoki Yoneoka, Yutaka Arakawa, and Keiichi Yasumoto. Detecting surrounding users by reverberation analysis with a smart speaker and microphone array. In *2019 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)*, pages 523–528, 2019. doi: 10.1109/PERCOMW.2019.8730674.

Xinhu Zheng, Anbai Jiang, Bing Han, Yanmin Qian, Pingyi Fan, Jia Liu, and Wei-Qiang Zhang. Improving anomalous sound detection via low-rank adaptation fine-tuning of pre-trained audio models. *2024 IEEE Spoken Language Technology Workshop (SLT)*, pages 969–974, 2024. URL https://api.semanticscholar.org/CorpusID:272593324.

Fuzhen Zhuang, Zhiyuan Qi, Keyu Duan, Dongbo Xi, Yongchun Zhu, Heng-

shu Zhu, Hui Xiong, and Qing He. A comprehensive survey on transfer learning. *Proceedings of the IEEE*, 109:43–76, 2019. URL `https://api.semanticscholar.org/CorpusID:207847753`.