GOTCHA: Physical Intrusion Detection with Active Acoustic Sensing using a Smart Speaker

Hoang Quoc Viet Pham[1][0009−0004−3760−2184], Hoang D. Nguyen[1][0000−0003−2541−3269], and Utz Roedig[1][0000−0002−4020−0889]

[1]University College Cork, Cork T12 K8AF, Ireland
{v.pham,hn,u.roedig}@cs.ucc.ie

Abstract. Voice Assistants (VAs) in the form of smart speakers such as Amazon Alexa, Google Assistant or Apple Siri are now commonplace. Due to their ubiquitous presence they can be useful as home alarm systems. 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 sensing. GOTCHA can execute on simple off-the-shelf smart speaker hardware. Periodic audible chirps are employed to gather data which 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 sensing solutions for physical intrusion detection.

Keywords: Physical Intrusion Detection · Active Acoustic Sensing · Anomaly Detection · Autoencoder · Machine Learning · Deep Learning

1 Introduction

Voice Assistants (VAs) such as Amazon Alexa [6], Google Assistant [14] or Apple Siri [7] are now commonplace. They are used to interact naturally, via voice, with digital infrastructure and services. For example, VAs are used to read out email, to provide weather forecasts or to interact with smart home devices. Often, VAs come in the form of standalone devices referred to as smart speakers. They are sold at affordable prices and they can be found in nearly every home.

As VAs are ubiquitous, users and VAs providers are looking for novel ways of using them to maximise their utility. One application domain considered is home security. Utilising available smart devices such as a smart speaker in a home as a security device can be a very cost-effective solution. The underlying idea here is to analyse anomalous sounds recorded by the smart speaker when a home is supposed to be unoccupied. 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).

Contrary to the aforementioned passive sensing approaches, it is promising to use active acoustic sensing for targetted monitoring, better noise handling and lower false alarms. Here an acoustic signal is sent periodically and the room response is analysed to determine the presence of an intruder. This approach has been explored less and existing work relies on dedicated hardware (e.g. ultrasound transducers, microphone arrays) and makes use of relatively simple methods such as statistic analysis and traditional machine learning.

In this work we detail the novel human presence detection system GOTCHA based on active acoustic sensing. GOTCHA can execute on simple off-the-shelf smart speaker hardware. It uses periodic audible chirps between 6kHz and 8kHZ. The core of GOTCHA is a deep autoencoder trained on the acoustic profile of the empty room. GOTCHA achieves a 99.2%, 99.0% and 91.4% of F1-score testing set with low false alarms rate. Thus, a practical alarm system can be constructed that can identify intruders very quick within miliseconds with minimal false alarms. While our work here focuses on active acoustic sensing it has to be noted that GOTCHA can be combined with existing classical passive acoustic sensing approaches. Passive sensing can execute in silent periods between GOTCHA chirps further improving system effectiveness.

The specific contributions of the paper are:

- *GOTCHA Design*: We provide an end-to-end design of the human presence detection system GOTCHA. It employs an autoencoder based anomaly detection using audio spectrograms from four microphones recording responses to acoustic chirps. The design enables implementation on simple commodity smart speaker hardware.
- *GOTCHA Evaluation*: We 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.
- *Public Available Dataset*: We make the datasets (chirp responses in an empty and occupied room) publicly available[1]. Thus, we enable other researchers to validate and reproduce our results and also enable further work such as bench-marking of alternate detection algorithms or improved training.

The remainder of this work is organised as follows. In Section 2 we describe related work on active acoustic sensing in monitoring applications and human presence detection studies. The GOTCHA design is presented in Section 3. In Section 4, we show an experimental evaluation of GOTCHA. In Section 5 we discuss limitations and future work. Section 6 concludes the paper.

2 Related Work

Human presence detection is a prominent research topic as it is necessary for a number of applications such as security surveillance, health care and traffic

[1] https://github.com/viet98lx/GOTCHA.git

management. A vast range of sensing modalities can be used for human presence detection including vision [2, 9, 10], Infrared (IR) [19, 20], Radio Frequency (RF) [12], and audio. In the case of RF, existing communication systems such as WiFi [18] 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 a 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 [8]. Acoustic sensing has been employed to observe humans. For example, Xie et al. [24] turn smart speakers into active sonars to determine exercise types using a method based on Doppler shift and short time energy. Xu et al. [25] utilized commercial off-the-shelf equipment to generate acoustic signals to distinguish individuals based on fine-grained gait information. In this work we aim to determine the presence of a person in a room instead of deriving information about a person's behaviour.

2.1 Human Presence Detection - Passive Sensing

Acoustic passive sensing is a method that uses acoustic sensors to hear and analyze ambient sounds. 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. [3] 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. [21] proposed an end-to-end system for intrusion detection by sensing the sound of footsteps amongst background noise. The algorithm uses 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. [1], [15], [5]). 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. [13], [17], [23]) 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.

Research	Emitting sound	Omni-directional speaker	No of channels	Approach	Label needed in training	Room size
[4]	Tone (1kHz)	✓	8	Random Forest	✓	Medium
[11]	Chirp (12.5–15kHz)	✓	1	Statistical Method	✗	Medium
[22]	Chirp (20-21kHz)	✓	1	DBSCAN + Regression	✓	Large
[26]	Pulse (6000-7750Hz)	✓	8	Statistical Method	✗	Medium
GOTCHA	Chirp (6–8kHz)	✗	4	Autoencoders	✗	Medium

Table 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 - 4(m)x3(m)).

2.2 Human Presence Detection - Active Sensing

For acoustic active sensing a sound signal is emitted and reflections from the surroundings are analysed to characterize the environment. Table 1 provides a summary of the key characteristics of existing works which we discuss in the next paragraphs. The main difference of all existing works to our proposed method is that simple statistical methods are used while we employ an deep autoencoder.

Shih and Rowe [22] 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.

Alanwar et al. [4] detect 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 from our work; there is an acoustic interaction before a pulse is sent.

Cheong et al. [11] employ 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 uses 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) away from the speaker. However, in their work, their system prototype requires an omnidirectional microphone and 4 speakers facing 4 cardinal directions.

Yoneoka et al. [26] analyse signal power correlation between a pulse and received sound to detect a human. They show that there is a different pattern in power attenuation of reflected sound with and without human presence. Their work focuses on detection speed instead of detection accuracy.

3 GOTCHA System Design

The aim is to detect an intruder entering a room using an acoustic signal emitted and analysed by a standard smart speaker device placed in the room.

3.1 Assumptions

We assume standard sized rooms. We do not consider large rooms (halls, open spaces). We assume a quiet environment, no noisy machinery or significant noise from the outside is present. We assume the smart speaker is placed on a table or shelf at a height of around 1m. We consider a smart speaker that is not obstructed by other items directly in front of it.

3.2 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 re-profiling 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.

3.3 Sensing Signal

A sonar system such as GOTCHA should ideally use ultrasonic sound as commonly used in sensing application [16]. However, low-cost microphones on the market often support only a maximum sampling rate of 16kHz and the highest

frequency that can be captured is therefore 8kHz. 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. [11]. Moreover, using a chirp in active acoustic sensing demonstrated good performance in previous work (see [11, 22, 26]). We did not use the entire frequency range as the higher frequencies provide more information. We decided to use a chirp signal generated by Audacity[2] starting from 6kHz and gradually rising up to 8kHz at the end of the 10ms period.

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.

3.4 GOTCHA Autoencoder (AE) Processing Chain

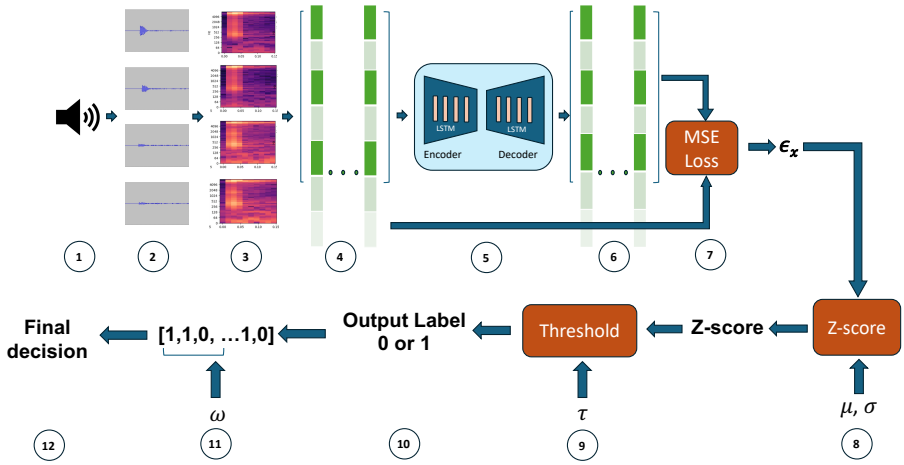


Fig. 1. GOTCHA processing chain. (1) Audio signal (2) Recording on N microphones (3) Signal transformation into Spectogramms (4) Combining inputs into a single input matrix (5) Autoencoder processing (6) Reconstructed output matrix (7) Reconstruction error ϵ_x computation (8) z_score computation (9)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

The flow of GOTCHA is described in Fig. 1. GOTCHA periodically sends an acoustic signal (Step 1) and then records the response on N microphones (Step 2). Fig. 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

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

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:

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

$$X = [X_1, X_2, X_3, ..., X_T] \tag{2}$$

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:

$$z = f_e(X) \tag{3}$$

$$\hat{X} = f_d(z) \tag{4}$$

$$\{\theta^*{}_{f_e}, \theta^*{}_{f_d}\} = \underset{\theta_{f_e}, \theta_{f_d}}{\operatorname{argmin}} \sum_{X \in \chi} \|X - \hat{X}\|^2 \tag{5}$$

$$\epsilon_X = \|X - \hat{X}\|^2 \tag{6}$$

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 $\hat{X}$.

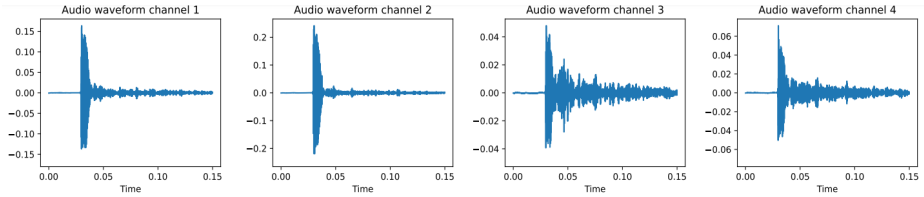
Because training samples only include normal signals, any anomaly sample fed into our AE will result in a large reconstruction error. Thus, a distribution of reconstruction error from the training set is established to identify anomaly samples. In the inference phase, we feed new data into our trained 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. 7 is a classical tool to detect outliers from a distribution (Step 8):

$$z\_score = (\epsilon_x - \mu)/\sigma \tag{7}$$

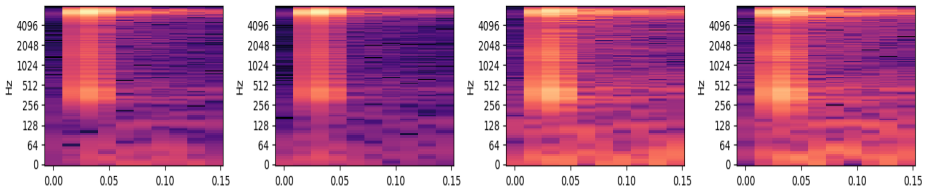
where μ and σ are the mean and standard deviation derived from the AE output on the training set which only includes normal samples. An example input

and the reconstructed signal is shown in Fig. 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).

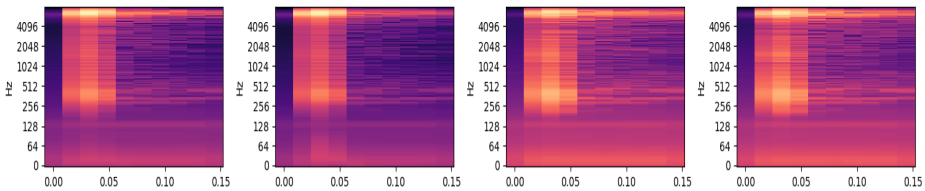
To eliminate false alarms, windowing is used. A window W is used to evaluate the last w decisions of the AE (Step 11). The majority label of all decisions within the window W are used as label for the entire window (Step 12). Thus, short-term noise does not lead to potential false alarms.



(a) Waves form of chirp signal



(b) Spectrograms of chirp signal



(c) Reconstructed spectrograms of chirp signal

Fig. 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 1). The reduced noise in the reconstructed output is clearly visible.

3.5 GOTCHA Prototype

We implemented a prototype of GOTCHA to carry out the experimental evaluation described in the next section.

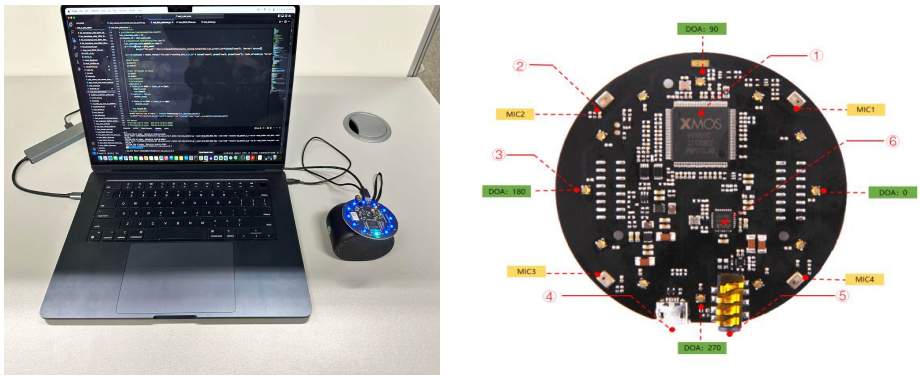
We use an HP S01 speaker as the transmitter of the chirp sensing signal. The speaker is a low-end device. As microphones we use the ReSpeaker Mic Array

v2.0 from Seed Studio (see Fig. 3(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. 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(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.



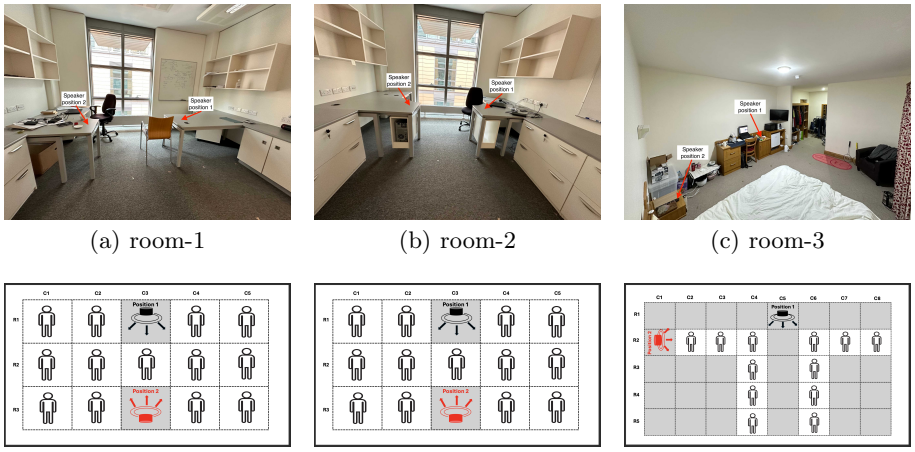
(a) GOTCHA prototype (b) 4-microphones array

Fig. 3. GOTCHA prototype design and microphone array structure.

4 GOTCHA Evaluation

4.1 Experiment Setup

We use the prototype described in Section 3.5 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. 4).



(a) room-1 (b) room-2 (c) room-3

(d) Human positions layout in room-1 (e) Human positions layout in room-2 (f) Human positions layout in room-3

Fig. 4. Room layout and regions with possible human presence. In white regions a human subject can be located, grey cells are obstacles (bed, cabinet, table, ...).

The GOTCHA prototype is placed on a table located in one corner of the room. The layout is shown in Fig. 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).

4.2 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 2. 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 dataset and code of GOTCHA are available via our Github repository[3].

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

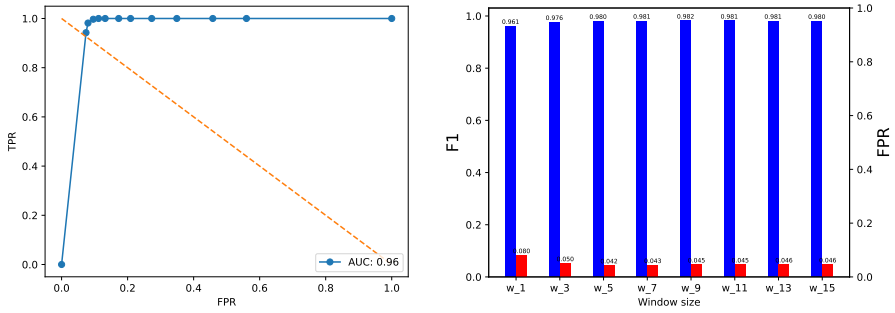
Table 2. Datasets summarization (r1, r2 and r3 stand for room-1, room-2 and room-3 while p1 and p2 stand for speaker position-1 and 2, respectively)

4.3 Comparison and Metrics

Comparison Among existing works mentioned in Section 2, we found that work by Yoneoka et al. [26] 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. [26] 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 comodity 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 False Positive Rate (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.

[3] https://github.com/viet98lx/GOTCHA.git



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

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

Fig. 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. 1.

4.4 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. 1. Furthermore, a window size w should be selected in step (11), used to reduce FPR for the final decision in step (12). Fig. 5(a) and Fig. 5(a) 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 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. 5(a) 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. 5(b) 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.

4.5 Data Observations

Fig. 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(a) and Fig. 3(b)).

After training, the AE provides the reconstructed spectrogram as output. The comparison between input spectrograms and reconstructed output spectrograms are shown in Fig. 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.

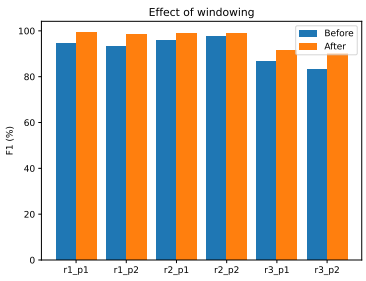
4.6 Detection Performance

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 . Fig. 5(b) 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.

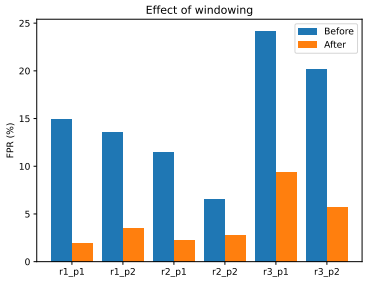
Metrics	r1_p1	r1_p2	r2_p1	r2_p2	r3_p1	r3_p2
F1-score	99.2%	98.4%	99.0%	98.8%	91.4%	90.2%
FPR	1.9%	3.5%	2.3%	2.8%	9.4%	5.7%

Table 3. 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)

Effect of windowing method Fig. 6 shows that after applying windowing method, for all cases, the F1-score can be improved while the FPR is significantly reduced.

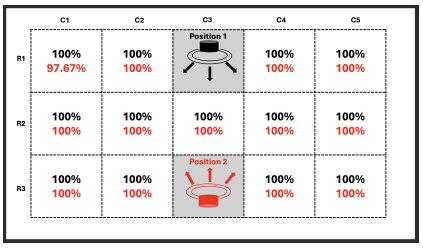


(a) F1 score on testing set of 6 cases before and after windowing

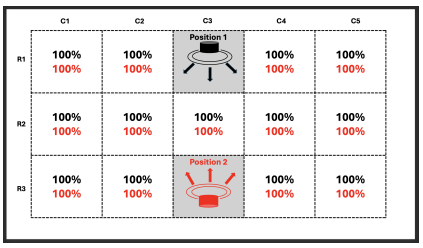


(b) FPR on testing set of 6 cases before and after windowing

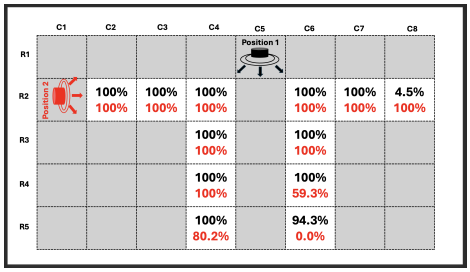
Fig. 6. Effects of windowing method on 6 cases (3 rooms with 2 speaker positions of each room. r1, r2 and r3 stand for room-1, room-2 and room-3 while p1 and p2 stand for speaker position-1 and 2, respectively).



(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

Fig. 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.

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 the system cannot detect an intruder, given certain

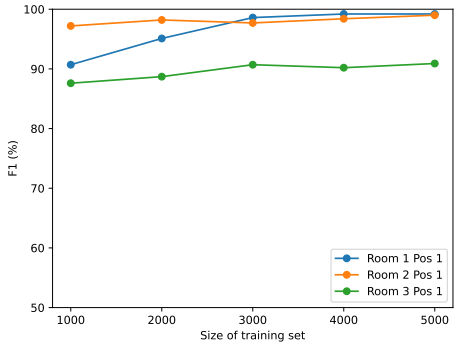


Fig. 8. F1-score on testing set of 3 rooms at speaker position-1 when using different sizes of training set.

hardware? 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. 4.

Fig. 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. 7(a), Fig. 7(b) and Fig. 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. 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 the resulting accuracy. We can see in Fig. 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.

5 Limitations and Future Work

As shown in our evaluation, GOTCHA is able to detect intruders within short time and barely make false alarms. However, there are a number of system aspects we aim to evaluate in future and improve upon.

We did not investigate the full design spectrum of the sensing chirp pulse we used. It might be possible to improve the system by using different pulses. Also, it could be investigated how to optimise the pulse for user acceptance (reducing nuisance, selecting a pleasant sound) while maintaining system performance.

GOTCHA currently only relies on active sensing. Although its performance is adequate for real-world adoption, it would be interesting to explore the combination of the active sensing approach with a passive sensing approach. In between sensing cycles (in our case of duration $150ms$) it would be feasible to switch to passive sensing and use a sound classification system to identify sound events such as footsteps, door opening/closing or light switching. Using two sensing approaches in combination might enable the system to be more robust.

6 Conclusions

In this paper we proposed 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.

Acknowledgements

This publication has emanated from research conducted with the financial support of Science Foundation Ireland under Grant number 19/FFP/6775 and 13/RC/2077_P2. For the purpose of Open Access, the author has applied a CC BY public copyright licence to any Author Accepted Manuscript version arising from this submission.

Bibliography

[1] Agarwal, S., Khatter, K., Relan, D.: Security threat sounds classification using neural network. In: 2021 8th International Conference on Computing for Sustainable Global Development (INDIACom), pp. 690–694 (2021)

[2] Ahire, P., Lokhande, M., Patil, R., Shirsath, T., Zaware, S., Zalki, T.: Intrusion detection using camera and alert management. In: 2023 International Conference on Inventive Computation Technologies (ICICT), pp. 1117–1121 (2023), https://doi.org/10.1109/ICICT57646.2023.10134400

[3] Al-Khalli, N.F., Alateeq, S., Almansour, M., Alhassoun, Y., Ibrahim, A.B., Alshebeili, S.A.: 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

[4] Alanwar, A., Balaji, B., Tian, Y., Yang, S., Srivastava, M.: Echosafe: Sonar-based verifiable interaction with intelligent digital agents. In: Proceedings of the 1st ACM Workshop on the Internet of Safe Things, p. 38–43, SafeThings'17, New York, NY, USA (2017), https://doi.org/10.1145/3137003.3137014

[5] Alsina-Pagès, R.M., Navarro, J., Alías, F., Hervás, M.: 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

[6] "Amazon": "amazon alexa" (nd), URL https://www.apple.com/siri.

[7] "Apple": "apple siri" (nd), URL https://www.apple.com/siri.

[8] Bai, Y., Lu, L., Cheng, J., Liu, J., Chen, Y., Yu, J.: Acoustic-based sensing and applications: A survey. Comput. Networks **181**, 107447 (2020), https://doi.org/10.1016/J.COMNET.2020.107447, URL https://doi.org/10.1016/j.comnet.2020.107447

[9] Bharadwaj K H, A., Deepak, Ghanavanth, V., Bharadwaj R, H., Uma, R., Krishnamurthy, G.: 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), pp. 1030–1035 (2018), https://doi.org/10.1109/RTEICT42901.2018.9012234

[10] Chen, H., Chen, D., Wang, X.: Intrusion detection of specific area based on video. In: 2016 9th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI), pp. 23–29 (2016), https://doi.org/10.1109/CISP-BMEI.2016.7852676

[11] Cheong, K.M., Shen, Y.L., Chi, T.: 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

[12] Denis, S., Berkvens, R., Weyn, M.: A survey on detection, tracking and identification in radio frequency-based device-free localization. Sensors

19(23) (2019), ISSN 1424-8220, https://doi.org/10.3390/s19235329, URL https://www.mdpi.com/1424-8220/19/23/5329

[13] Do, H.M., Welch, K.C., Sheng, W.: Soham: A sound-based human activity monitoring framework for home service robots. IEEE Transactions on Automation Science and Engineering **19**, 2369–2383 (2021), URL https://api.semanticscholar.org/CorpusID:236408129

[14] "Google": "google assistant" (nd), URL https://assistant.google.com.

[15] Kim, J., Min, K., Jung, M., Chi, S.: 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, https://doi.org/https://doi.org/10.1016/j.buildenv.2020.107092, URL https://www.sciencedirect.com/science/article/pii/S0360132320304686

[16] Lee, Y., Zhao, Y., Zeng, J., Lee, K., Zhang, N., Shezan, F.H., Tian, Y., Chen, K., Wang, X.: Using sonar for liveness detection to protect smart speakers against remote attackers. Proc. ACM Interact. Mob. Wearable Ubiquitous Technol. **4**(1) (mar 2020), https://doi.org/10.1145/3380991, URL https://doi.org/10.1145/3380991

[17] Li, J., Li, X., Liu, B., Liu, Y., Jia, W., Lai, J.: 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

[18] Liu, J., Liu, H., Chen, Y., Wang, Y., Wang, C.: Wireless sensing for human activity: A survey. IEEE Communications Surveys & Tutorials **22**(3), 1629–1645 (2020), https://doi.org/10.1109/COMST.2019.2934489

[19] Netinant, P., Amatyakul, A., Rukhiran, M.: 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, p. 171–175, ICSIM '22, Association for Computing Machinery, New York, NY, USA (2022), ISBN 9781450395519, https://doi.org/10.1145/3520084.3520112, URL https://doi.org/10.1145/3520084.3520112

[20] Sahoo, K.C., Pati, U.C.: Iot based intrusion detection system using pir sensor. In: 2017 2nd IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT), pp. 1641–1645 (2017), https://doi.org/10.1109/RTEICT.2017.8256877

[21] Sharma, S., Venkata, P.T., Singhal, S., Gopi, G., Kumar, S., Prasad, R.V.: B4w: A smart wireless intruder detection system. ICC 2023 - IEEE International Conference on Communications pp. 191–197 (2023), URL https://api.semanticscholar.org/CorpusID:264468882

[22] Shih, O., Rowe, A.: Occupancy estimation using ultrasonic chirps. In: Proceedings of the ACM/IEEE Sixth International Conference on Cyber-Physical Systems, p. 149–158, ICCPS '15, Association for Computing Machinery, New York, NY, USA (2015),

ISBN 9781450334556, https://doi.org/10.1145/2735960.2735969, URL https://doi.org/10.1145/2735960.2735969

[23] Sim, J.M., Lee, Y., Kwon, O.: 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

[24] Xie, Y., Li, F., Wu, Y., Wang, Y.: 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, pp. 1–10, IEEE (2021), https://doi.org/10.1109/INFOCOM42981.2021.9488811, URL https://doi.org/10.1109/INFOCOM42981.2021.9488811

[25] Xu, W., Yu, Z., Wang, Z., Guo, B., Han, Q.: Acousticid: Gait-based human identification using acoustic signal. Proc. ACM Interact. Mob. Wearable Ubiquitous Technol. **3**(3) (sep 2019), https://doi.org/10.1145/3351273, URL https://doi.org/10.1145/3351273

[26] Yoneoka, N., Arakawa, Y., Yasumoto, K.: 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), pp. 523–528 (2019), https://doi.org/10.1109/PERCOMW.2019.8730674