Ambulance alert

Áron Kuna, Hussein Al-Saidi

Supervisor: Sune Thomas Bernth Nielsen

May 2023

Roskilde University

Bachelor project

Abstract

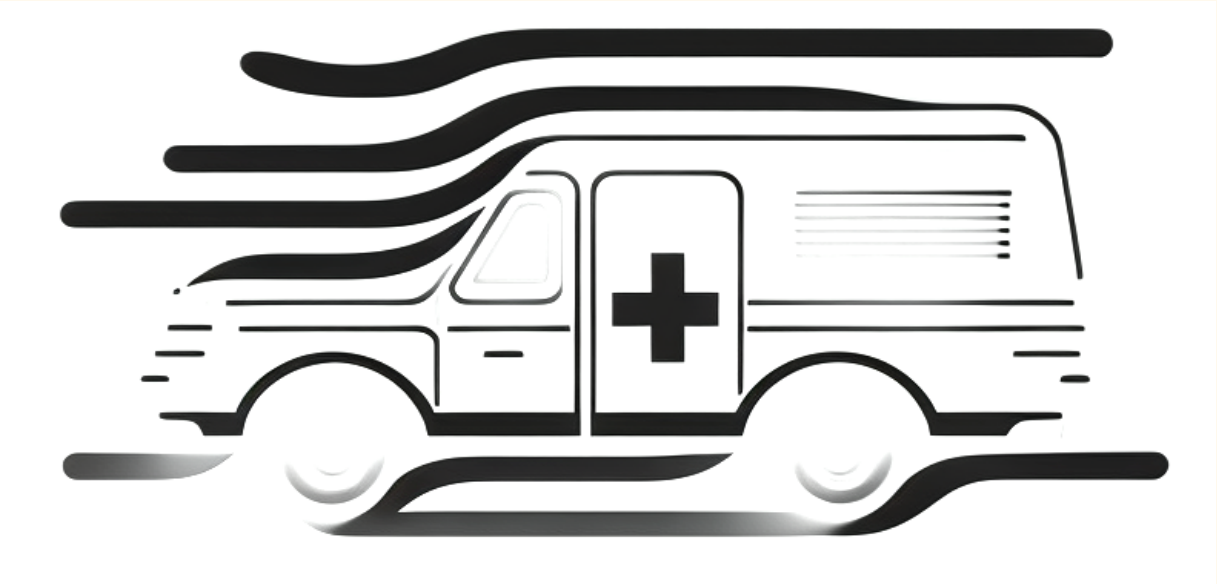
This study investigates the viability of using sound detection for road safety. Specifically looking into the possibilities of detecting emergency vehicle sirens. The advancements in soundproofing and noise canceling require new tools to improve road safety. A study made by Dr. Simon Moore in 2017, stated that drivers who listen to music louder than 96dB, will lose about 20% of their concentration. A driver in a soundproofed car only has a very short time to react to the ambulance. Our prototype to offer a solution has the capabilities to recognize ambulance sirens from low sound levels, in traffic situations as well. It connects to an Android App, sending alerts within 2 seconds after a siren sound becomes audible. The prototype provides a proof of concept, however, it amplifies the faults of the idea as well. Due to not being able to assess the full traffic situation, the prototype cannot decide if the user is in the ambulance's way or not. This can lead to several false alarms, limiting the real-life usability of the system. The research starts by looking at advancements in sound detection in the past decade. Several studies are reviewed about ambulance detection, which paths the way for the design choices. For the prototype the YAMNet pre-trained sound classifier neural network is implemented on a Raspberry Pi, communicating to an Android App. To determine the true viability of the prototype more excessive testing is required. However, this study elucidates a promising avenue for further exploration and development in sound-detecting driver assistance systems.

RUC Bachelor Project
Supervised by Sune Thomas Bernth Nielsen

Ambularm

A prototype to reach advancements in road safety

By Áron Kuna, Hussein Al-Saidi



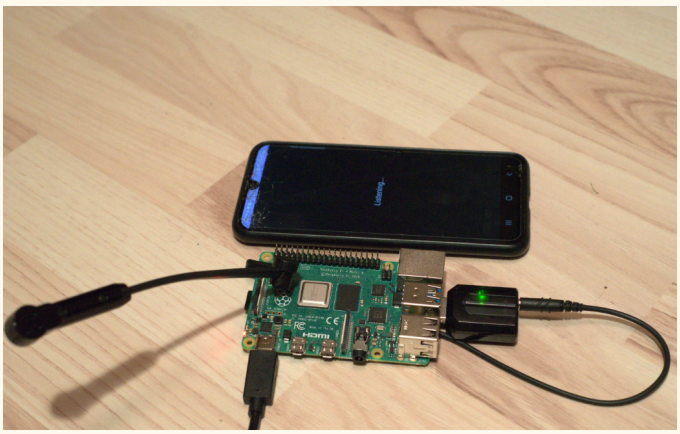
INTRODUCTION

NSC (National Safety Council) does research every year, to find statistics on the number of lives lost in different kinds of accidents. In 2010 did 131 Americans lose their lives in car crashes involving an EMV (emergency motor vehicle). 86 lives were lost to a collision with a police car, 31 with an ambulance, and 14 with a firetruck. These numbers only went up since then and are now up to approximately 200 lives per year. In the year 2021, 134 Americans lose their lives in a car crash with a police car. 39 died in a collision with an ambulance and 24 with a firetruck. Accidents could happen for numerous reasons. But in all the accidents, there is a discussion about concentration, alertness, reaction time, etc. both from the driver of the EMV (Emergency Motor Vehicle) and other drivers, pedestrians, and bystanders.

Professor Dr. Simon Moore did research back in 2017, that stated, that drivers lose approximately 20% of their focus by listening to louder music than 96dB while driving, which will end up in fatality. Dr Simon Moore also discovered that music with 100 BPM (beats per minute) or above, typically results in drivers driving faster, because the driver subconsciously matches the heartbeat to the beat of the music. Another group of Australian scientists did a test in 2011 that concluded that the sound of an emergency siren, loses 6dB per doubling distance, so the emergency vehicles don't need to be far away, for the driver not to hear them approach, essentially when the music is louder than 96dB.

The prototype

There are multiple solutions to solve that problem, but we want to focus on alerting the surroundings of the emergency vehicle. Our solution is to install a device that will detect the sound of the emergency vehicle siren and give an alert. The device will be mounted on the car, with the microphone on the outside, so the siren can be detected. The microphone must be placed strategically, so it won't pick up too much background noise and engine noise. When the microphone detects the siren, the device will send an alert to the connected phone, having the potential to automatically turn the volume down, or off altogether.



As seen above, the device next to the smartphone, the device is not connected to the phone by wire, therefore can be separated.

Findings

A lot has improved over the last few years, the technology is speeding up alongside the future. But it's not the future yet, there are still some complications with detecting sounds. The

detection of the sound itself is possible, the microphone and the software can do the task, as presented with our prototype. The challenge is that it will get complicated if we would go deeper into the problem. e.g. if the driver is sitting in the vehicle, and the device hears the emergency siren. But the driver neither sees nor hears the siren, since the ambulance is cruising by on another street. This “false positive” effect is unavoidable with the current implementation and can cause frustration to users. A similar problem is that the device does not recognize where the sound is approaching. Imagine the driver is stuck in traffic, the device goes off, all the drivers make way for the emergency vehicle, and an ambulance drives by on the other side of the street, or worse, the ambulance is driving on a parallel street. The device can’t hear the speed of the approaching emergency vehicle either, which could be a problem as well, some drivers might think that they still have a bit of time, whether it is to reach their turn or get to their destination, and that could result in a collision or even worse. There are methods to make the device more precise, for example by mounting multiple microphones on the device, in that case, the device would know if the sound is coming from the front of the car, back, or one of the sides. However, using purely sound detection, knowing if the car is in the way of the ambulance or not seems unreachable.

Future improvements

This device can be upscaled so it would be available for bicycles, pedestrians (with headsets), and people on other types of transportation vehicles. These upgrades or rather upscales, might have some complications as well. For this project to be upscaled, some of the technology needs to be upgraded as well, both software and hardware. The whole device needs to be more advanced, for it to be suited for more than cars. For the device to be mounted on bikes, motorcycles, or even be a technology on smartphones so pedestrians could benefit from it, it will need a lot of improvements. The device will need to be more precise, meaning no false positives, a more advanced approach to the problem, and easier assistance for the user. For a device like this to be a success, it needs to be flawless. For the drivers to choose to use this device, they need to be ensured that the device is helpful and that it will save lives. One of the most important reasons is that the user (driver, motorcyclist, bicycle, or pedestrian) is that the device won’t take over and be in control for no reason. The device would only take over and turn off the music, when it’s a matter of life and death, when the emergency vehicle is in a hurry, but cars are blocking the streets. Therefore, the device needs to be 100% flawless. The bare minimum is that the microphone listens to everything, but only takes over when there is an emergency vehicle approaching, not driving by, on the other side of the street, or even on a different street. In the future of upscaling/upgrading the device, GPS could be added to the

software. In that case, the route of the emergency could be online on a map, and the drivers in traffic could make way, even though the emergency vehicle isn't visible. In that way, all that is needed is that the device is connected to the satellite, and if the vehicle is on the same route or crosses paths with the emergency vehicle, the driver will receive an alert.

In future upgrades, ANC can be included in this technology. ANC is the technology we know as Active noise cancellation. That technology is getting more and more advanced. Big companies are starting to create headsets with advanced noise canceling, even cars have noise canceling. So, ANC could be a critical factor for the safety of drivers, pedestrians, motorcyclists, and cyclists. As mentioned earlier, a combination of, 100 BPM and sound higher than 96dB, now with noise cancellation, can be crucial, that's a matter of life and death.

The device could have a feature to not only turn off the music but also deactivate the noise cancellation. For all kinds of users. It could be used by cars, as a device mounted on, motorcycles, in the helmet, and pedestrians, built in the headset. By the time all these features are implemented on the device, and the device is publicly used, there will be a lot of difference. The response time for emergency vehicles will be shorter, and as a result, emergency responders will arrive faster at their destination. People will be more aware of the approaching vehicles, and make way, which results in fewer collisions.

Read More about the accidents

https://injuryfacts.nsc.org/motor-vehicle/road-users/emergency-vehicles/

Table of Contents

1 Introduction

1.1 Background and motivation

Emergency vehicles sometimes have difficulty reaching their destination on time, especially during rush hour. The driver of any vehicle must make sure to drive fast but safely. The sirens must be on so everyone can hear its approach, but sometimes the driver of a private vehicle can turn the music up enough not to hear the sirens, or for some other reason not to see the light of the emergency vehicle. The problem also concerns bikers, especially in populated areas. More and more people are now using noise-cancelling headphones, some of which are capable of cancelling up to 70% of all noises. Combining that with a loud song or a phone call makes it almost impossible for a cyclist to perceive their surroundings. Lack of awareness or alertness can be crucial, and especially in traffic, it can be catastrophic. Drivers of private vehicles that tend to not pay attention can harm themselves and others. In 2021, 198 lives were lost in crashes with an EMV (Emergency Motor Vehicle) [24]. We were motivated to find a solution to this problem. This solution manifested in a device that will spread awareness and alert drivers in case of emergencies. The device detects the emergency vehicle when it is approaching, and lets the driver know by turning down the music and sending a notification.

1.2 Objectives and goals of the project

Research Question:

How to alert car drivers for an ambulance siren, to avoid delaying emergency vehicles?

The goal of the project is to research and get an overview of up-to-date sound detection methods. Then propose and make a prototype for a device that can recognize the sound of an ambulance in an urban environment. The sound detection has to be quick and able to work in low signal levels. The device is to be placed outside the passenger area to avoid loud music and ANC (Active Noise Cancelling) and will be connected to the user's smartphone. In the case of siren detection, the device sends signals to an app on the user's phone, having the potential to disrupt loud music listening.

The project looks into the possibilities of alerting drivers to emergency vehicle sirens. Looking at the state-of-the-art sound recognition, the viability of the concept, and research and products made in this field. The goal is to create a proof of concept and discuss the viability of the results.

2 Theory

2.1 Real-life situations

2.1.1 Response time of emergency vehicles

Response time is different from one country to another. Below, is shown a list of 8 countries. This list shows the response time for emergency vehicles, as well as the year, country, city, HDI (human development index), and life expectancy. We are not going to take the HDI as a factor since we only focus on the response time. As can be seen, the bigger the city is, the longer the response time gets. These numbers are average and could differ from responding on empty roads and throughout the rush hour.

Year of publication	Countries	City. Region and State	Response Time (Minutes)	HDI	Life expectancy (Years)
2014	Taiwan	Taoyuan[15]	5	0.738	76.0
2016	United States	Salt Lake City[16]	5	0.920	79.2
2017	Republic of Korea	Seoul[17]	7	0.901	82.1
2015	United Kingdom	Wiltshire,Glouceste-rshire and Avon in Southwest England[18]	6	0.909	80.8
2016	United States	Seattle[19]	6.1	0.920	79.2
2015	Singapore	Singapore[20]	7.25	0.925	83.2
2015	Sweden	Stockholm[21]	7.8	0.913	82.3
2012	Australia	Melbourne[22]	8	0.939	82.5

Figure 1: A list of response times for different countries

In Denmark, the response time for ambulances is set to be 5 minutes, although studies show that only 25% of the ambulances reach their destination within the time limit. For the Fire Department as well as the police, they need to be at their destinations as soon as possible. [27]

2.1.2 Effects of loud music and noise canceling

Dr. Simon Moore is a psychologist and a professor who, in 2017 conducted a study on drivers listening to loud music, with a fast beat. The study showed that if the driver is listening to music that had a BPM of more than 100, the driver tends to drive faster. BPM is beats per minute, and a higher BPM makes the driver drive faster subconsciously so that the heart will beat at the same speed as the music. Professor Moore has also concluded that the fast beat, combined with a sound level of about 95dB, would decrease reaction time by 20% [24].

2.2 Literature Review

Multiple research has been conducted on how to recognize the sound of an ambulance. The common goal is to be able to alert drivers in soundproof environments to the sirens and thus, speed up the way of the ambulance. This section starts with giving an insight into the advancements in sound recognition technology in the past decade, then takes a look into research conducted specifically on siren sound detection.

2.2.1 Sound recognition

The two main branches of sound recognition are speech and non-speech recognition. The non-speech recognition tasks are commonly known as automatic sound recognition (ASR). Other names for it are sound event recognition (SER) and, in some cases, acoustic event detection. [33]. An ASR system is made for automatic sound recognition by processing a sound signal and using machine learning techniques. Essentially, it is similar to a speech recognition system, with the primary difference being

that it processes non-verbal audio input. There is a wide range of usage of ASR systems, including music genre classification [37], musical instrument sound classification [38], audio surveillance [32], sound event recognition [13], and environmental sound recognition [10]. Audio surveillance and sound event recognition are used for room and public transport monitoring, guarding wildlife areas, and in healthcare, for monitoring elderly people.

While ASR is used in many different areas, the principle of the approaches is very similar, inspired by speech recognition systems [33]. There are three key steps for each ASR system. It starts with signal processing, then feature extraction, and finally classification. The goal of signal processing is to prepare the input data for feature extraction. This includes dividing the data into smaller frames, typically into 10-30 ms. Feature extraction commonly includes transforming the time-domain signal to the frequency domain (see more in chapter 2.5.1) or time-frequency domain (see more in chapter 2.5.2). Finally, during the classification, the unknown audio events are assigned into classes with some confidence. The classes are defined by the training data where a big amount of audio events are classified.

The two main classifiers used in the last decade are Support Vector Machines (SVM) and Deep Neural Networks (DNN) [33]. An SVM finds a hyperplane that maximizes the distance between two given classes. In other words, uses statistical learning to find what separates the features of distinct classes. SVMs generally give better predictions than other traditional classifier methods like kNN (k-nearest neighbors), and NC (nearest center) [33].

In recent years deep learning algorithms gained popularity in most pattern recognition fields. Similarly in sound recognition, deep neural networks are researched by many groups including Microsoft Research, Google, IBM Research, etc [33]. They also found that DNNs perform better than most other classification methods [19]. DNNs also generally outperform SMVs [33].

2.2.2 Siren sounds

The ambulance siren has a distinctly recognizable sound for the human ear, which suggests that the automatic recognition of it should not be too hard. However, this is not always the case. (The struggles with sound detection are discussed further in section 2.6).

The research paper "Identification of Ambulance Siren sound and Analysis of the signal using statistical method" [34] uses a simple approach to identify the ambulance sound. It has a great advantage in that it does not include the use of any computationally heavy machine learning algorithm, avoiding the need for a huge training dataset. It uses Python to process the sound input, and then properties like Mean and Standard deviations are compared to siren sound properties. It has a great success identifying siren sounds. An additional advantage is the light processing power needed and a lack of need for complex tasks. This sounds ideal, however, the paper states that "This algorithm can be used when there is less noise in the surroundings." which makes it ineffective in a general city environment with traffic noise.

Similarly, the research paper *Recognition of the Ambulance Siren Sound in Taiwan by the Longest Common Subsequence* [26] avoids using neural networks as well to speed up the recognition process and avoid complex calculations. The researchers are looking for the Longest Common Subsequence of high and low-frequency sounds that match the pattern of an ambulance siren. They use several setups to check the efficiency. The result is 85% for The true Positive Rate (ambulance sound is there and detected) when there is noise in the car. However, it is 77.1% when there is noise outside the car. Besides, there is a 10% False Positive Rate.

The following two research has a similar goal to our report and they are important sources of inspiration.

The research *Detection of an ambulance and a fire truck siren sounds using neural networks* [36] points out the dangers of soundproof vehicles as well as the possibilities in siren detection for road safety. The researchers test out the capabilities of two different neural networks (MLP and LSTM-RNN) in classifying ambient sound into three classes: ambulance, fire truck, and road noise. The

identification of fire trucks is not as strong, however, the accuracy of ambulance sound recognition is over 97% for both neural networks.

Researchers of *Detection of Ambulance Siren in Traffic* [31] are using a smartphone's microphone to identify ambulance sirens. The goal is to alert the drivers to the ambulance in soundproof vehicles so the ambulance can move through the traffic effectively. In the paper, the authors train a Bayesian regularized artificial neural network (BRANN) and observe that in identifying ambulance sirens, it reaches an "accuracy of greater than 99 percent in simulated conditions using sound data from prerecorded audio".

State of the art algorithms

The paper *CNN architectures for large-scale audio classification* [18] compares the performance of the VGG, AlexNet, ResNet-50, and Inception V3 neural networks in audio classification, finding ResNet-50 as the best-performing model. The research paper *Audio Interval Retrieval using Convolutional Neural Networks* [25] compares neural networks in classifying audion events. It compares YAMNet, AlexNet, and ResNet-50 pre-trained models and finds that YAMNet slightly outperforms the other two models, however, it is important to mention that their test dataset is the same which was used for the training of the YAMNet model.

2.3 Machine Learning

2.3.1 Machine Learning

ML, also known as Machine Learning, is a kind of artificial intelligence (AI), that focuses on using algorithms and data to imitate the human way of thinking. This technology has been advancing over the last couple of decades, the advances are within storing and processing data, for instance, Netflix recommending the users specific movies, and Spotify recommending specific songs. ML algorithms are in general used to make predictions and/or classifications. The algorithm will use the input data to produce a pattern [22].

2.3.2 Neural Networks

Neural networks, also known as artificial neural networks (ANN), are a subfield of ML. The name and structure of ANN are inspired by the human brain. Artificial neural networks (ANNs) are made of node layers. Each node connects to another and has a linked weight and threshold. If the output of any node is more than the specified threshold value, that node will be activated, and start sending data to the next layer of the network. Neural Networks train on data sets to improve their accuracy, so over time, and a lot of training, the neural networks become a powerful tool within computer science and AI, making it easier to classify and work with data fast [23].

2.3.3 Deep Learning

Deep learning is a subfield of neural networks, which is the subfield of ML. Deep learning is almost the same as neural networks, the difference is just the number of layers. Neural networks generally have one input layer, a hidden layer, and one output layer, but on the other side, Deep learning has one input layer, multiple hidden layers, and one output layer. So, with additional hidden layers, deep learning can optimize and refine for accuracy [21].

2.4 YamNet

YAMNet [17] is a sound classifier deep neural network made by Google. It is an open source and freely available. YAMNet uses a mel spectrogram to extract features of the sound. It is trained on the AudioSet-YouTube corpus dataset [16] which includes 2,084,320 human-labeled 10-second sound clips drawn from YouTube videos. The trained machine-learning model takes an audio waveform as an input and then makes independent predictions for 521 audio event classes. It is available as a TensorFlow as well as a TensorFlow light model. TensorFlow light is developed for computers with

low power, this means machine learning models can be used on a system-on-chip like an Arduino as well.

2.5 Sound signal

The sound propagates through the air in the form of longitudinal waves. These waves arrive at the receiver which can be a human ear or a microphone. In the case of a microphone, the waves in the air make the diaphragm vibrate. The vibrations are turned into electrical signals with a transducer. The electrical signal is analog at this point. To turn the signal into digital, an ADC (Analog to digital converter) [8] measures the amplitude of the signal at discrete time intervals. This creates a series of values representing the sound signal in a digital form. The process of measuring the analog signal's amplitude at given time intervals is called sampling. The frequency of taking a sample is called the sampling rate. It is measured in Hertz (Hz) which tells how many samples were taken in one second. The accuracy of the measurement depends on the bit rate. The bit rate tells the number of bits used to represent each sample. For example, when the bit rate is 16 (CD quality) [28] it means 16 bits were used so $2^16 = 65,536$ different values possible for each amplitude measurement. The values are often represented as integers from -32,768 to 32,767. The higher the bitrate the higher the quality.

The sound arriving at the microphone is made of sound waves with different frequencies. For a given sound the frequency of the wave determines the pitch and the amplitude sets the volume. The building frequencies add up to make an accumulated waveform that is recorded. The digital signal is a series of snapshots of the original analog signal at discrete time intervals. To be able to retrieve the underlying waves in the signal the snapshots have to be taken frequently enough so the building waves show up. The Nyquist-Shannon theorem [30] states to accurately capture a signal the sampling rate has to be at least twice the highest frequency present in the signal. This is called the Nyquist rate. The human hearing range is from 20Hz to 20000Hz so the Nyquist rate for recording sounds humans can hear is 40000Hz. In practice, the sampling rate is usually set to 44.1 kHz.

Finding the building frequencies in a signal can give a lot more information about the sounds recorded than just seeing the waveform Fig2. To retrieve the underlying frequencies the well-known method is using a Fourier transform.

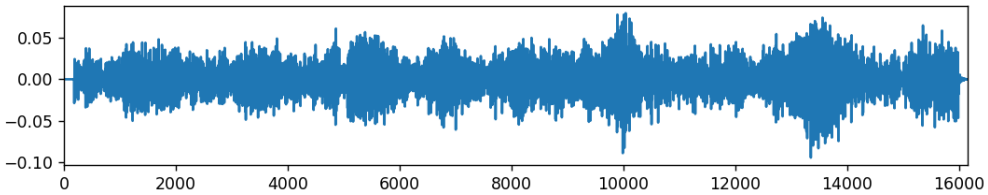


Figure 2: Recorded waveform of an ambulance siren. The x-axis is the sample number and the y-axis is the amplitude of the signal. *Created with Python*

2.5.1 Fourier transform

The Fourier transform is based on the principle that any periodic signal can be described as a sum of sine and cosine waves with different amplitudes and frequencies. After a Fourier transform, a signal in the time domain results in a signal in the frequency domain [8]. This means after a Fourier transform the change in the signal over time cannot be observed. Performing a Fourier transform on a sound signal gives the frequencies that are present in the sound. By essentially breaking down a signal to its "building frequencies" computers (and humans as well) can get more information about the sound signal. This opens the possibilities for algorithms that look for a specific frequency. Fig 3 shows an example of how the frequencies are usually shown after a Fourier transform on audio data.

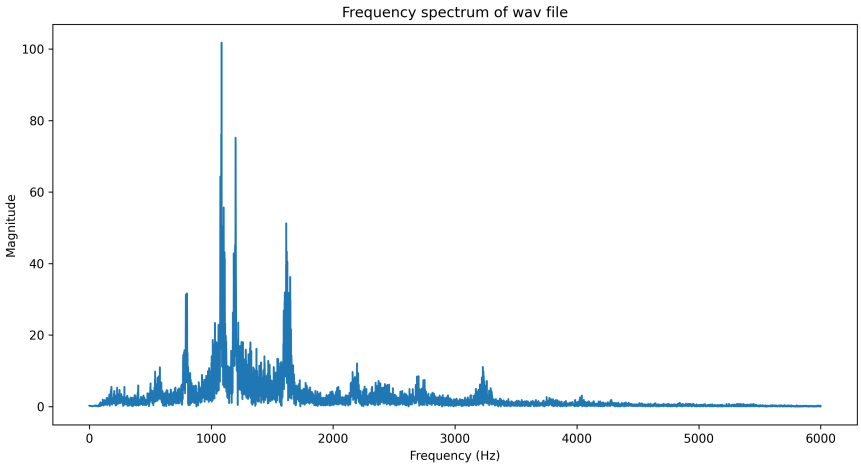


Figure 3: The FFT of the ambulance sound shown in Fig 2. The x-axis is the frequencies and the y-axis is the amplitude of the frequencies.
Created with Python

2.5.2 Spectrograms

Spectrograms are useful to get information on the frequency change over time. To create a spectrogram a Short Time Fourier Transform (STFT) is used which means performing a Fourier transform for shorter time slices. This way the frequency changes can be observed over time by constructing a spectrogram. In a spectogram the X-axis is the time and the Y-axis is the frequencies. The amplitude of the frequencies is shown as colors. Fig 4 shows the spectrogram of 1 second of traffic noise with ambulance sound.

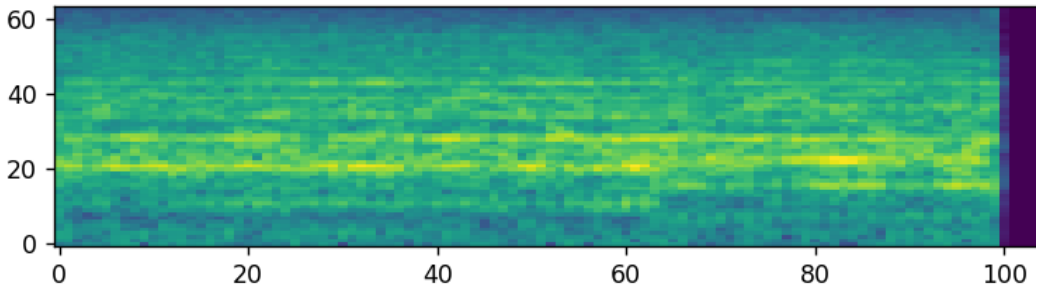


Figure 4: The spectrogram of the ambulance sound shown in Fig 2. The x-axis shows the time in centiseconds and the y-axis shows the frequencies. The amplitude of the different frequencies are represented by colors, the lighter the color the higher the frequency.
Created with Python

A special case of spectrograms is a mel spectrogram. Humans can distinguish between lower frequencies more effectively than between high-pitched sounds. For example, the change appears to be more significant from 400Hz to 500Hz than the change from 10000Hz to 10100Hz even though the frequency change is the same (100Hz). To account for this, a Mel spectrogram instead of using a regular scale on the y-axis (frequency-axis) uses a Mel scale [11], which is a logarithmic scale. This way the representation of sounds on the Mel Spectrogram is closer to how we perceive sound. This is important in the case of machine learning, models trained using Mel Spectrograms can often perform better in tasks that mimic human hearing. Fig 5 shows an example of a Mel Spectrogram compared

to a Spectrogram.

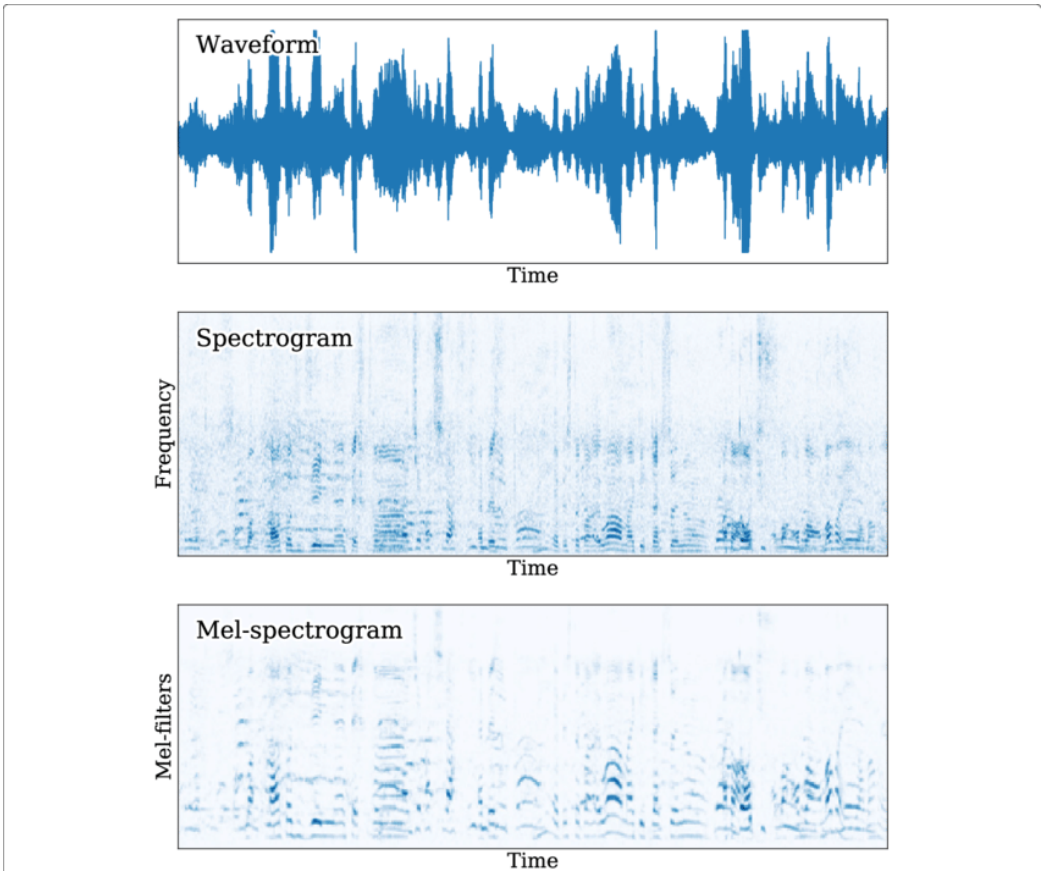


Figure 5: Showing the difference between Spectrogram and Mel Spectrogram. The Mel Spectrogram uses a logarithmic scale for the y-axis (frequencies) giving a representation which aligns more with how we (humans) perceive sound. *from EURASIP Journal on Audio Speech and Music Processing*[11]

2.6 Obstacles with sound recognition

There are several obstacles to sound detection. There is always sound pollution in the surrounding areas. Sound pollution is all the background noise we daily listen to, without hearing. Sounds like planes, machines, car exhausts, etc. These sounds are always around us, but the brain chooses not to acknowledge them, but sensors will. The sound sensor will detect all sounds within the range. For the sound to be used after being detected, it needs to be saved, the problem is that it needs memory storage. 1 hour of a 24-bit BWAY audio will require about 1GB of storage [39] [12].

2.6.1 The Doppler effect

The Doppler effect also known as the Doppler shift is the name for the change of frequency for a wave in relation, to an observer. An example of the Doppler effect is the change of sound heard when a car is pressing the horn while approaching. The frequency will be higher when the car is approaching, and lower when the car is driving away. The idea behind the Doppler effect is when the source of the waves is approaching, each wave will be sent from a location closer to the observer, than the previous wave. So, each wave received will be a bit faster, until the vehicle has passed the observer. On the other hand, if the vehicle is driving way, the waves will each be slower and further

away than the other [29]. The change in pitch makes it hard to listen for a specific known sound given by moving objects.

2.6.2 Attenuation

Attenuation is the loss or weakening of signal strength. The closer someone is to the source, the better signal received. Attenuation can happen due to many reasons. One factor that can weaken the signal, is the physical surroundings, Buildings, cars, people, and other signal interference [40].

2.6.3 Variability in sound sources

Sounds don't travel at the same speed, because sound is a vibration of kinetic energy passed from molecule to molecule. The closer the molecules are to each other the faster sound can travel. Sound waves travel easily through solid matter, and the bonds between the molecules are stronger than for instance liquid and gasses [40]. Sound also reflects from physical objects, then the reflected waves interfere with the original signal, potentially changing it.

2.6.4 Acoustic properties of ambulance sirens

In an emergency response in America, there are 3 different types of sirens. The driver of each vehicle can choose which siren to use. Depending on the situation the driver can choose a lower-frequency siren. In situations where the emergency vehicle driver needs to go through traffic, the sirens can be changed to a higher and faster frequency. The first type of siren is the "wail". This siren has a continuous rising and falling sound, that goes between 500 and 1800Hz, this siren is a slow siren, that goes 11 cycles/minute. The second type is the "Yelp", that sound has a continuous warbling sound, also between 500 and 1800Hz, but this one is faster, it has 55 cycles/minute. The third type is called Hi-Lo, and the name comes from the sound because it has a two-tone sound. This type is as fast as "Yelp" but with a lower frequency of 670-1100Hz. [9]

3 The prototype

In this section we propose a concept to implement an ambulance alert for car drivers. The goal is to see the viability of an ambulance alert using modern available tools. The section first starts off with the requirements for the prototype and design ideas. Then the created prototype is presented and the technical implementations are explained. The next step is the testing. Although the scope of the project did not allow for rigorous testing, a testing procedure is proposed which could be used in future work. The section ends with an analysis on the capabilities of the prototype.

3.1 Design

The goal of the prototype is a proof of concept to realize the capabilities of modern available tools. The aim is a device that can recognize the sound of emergency vehicles (in the project specifically focusing on ambulance sound) and give near real-time alerts to an application. Our ambition with the prototype is to give insight into the limits of the concept and enable future work in the field.

The major design decision for the "ambulance alert" is, where does the recording takes place. The paper *Detection of Ambulance Siren in Traffic* [31] proposes an idea to use the phone's microphone. However the goal is to avoid the sound-proofed area of the car. To do that, the design idea is to use a system-on-chip with microphones connected outside of the passenger area, which communicates to the driver's phone.

There are several reasons for this design choice. Firstly, this makes a modular system that can be used in any car, the system-on-chip can be placed under the hood, while the connected microphones can be placed further away to avoid engine noise. Secondly, most of the drivers use Bluetooth connection in the car to connect to the speaker with their phone. Communicating to the phone enables notification to the driver. Additionally, the connected app could lower the volume of the music, and maybe turn off active noise canceling, if in use, however, these features are not required for the proof of concept and are out of the scope of this project. Finally, by making the sound detection run on an external device, it does not drain the power of the phone, the external device can be connected to the car's electricity.

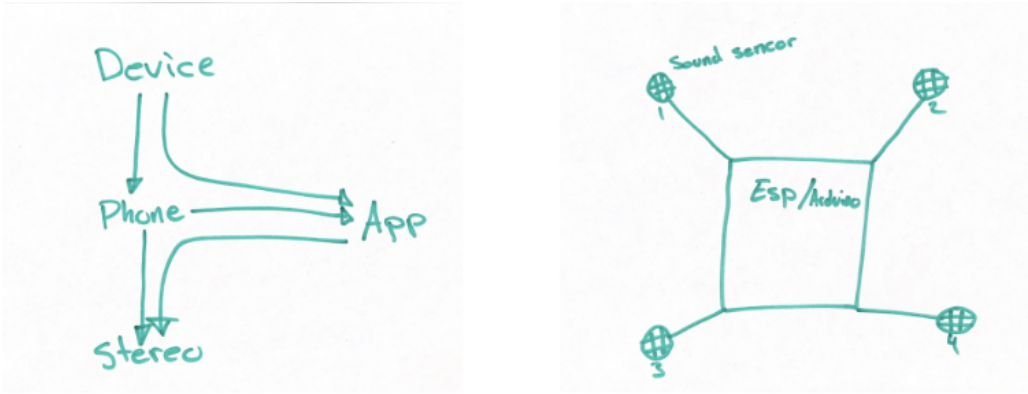


Figure 6: The design sketch of the product. The Device has microphones placed outside of the passenger area in a car and it connects to the driver's phone. By communicating through the app it can lower the music volume.

3.2 Overview of the prototype

The created prototype is a multiple-platform system, where an environment sound classifier neural network is implemented on a Raspberry Pi 4 which communicates through Bluetooth with an Android app. The Raspberry Pi and the phone is paired with Bluetooth. The app is open for connections, the Raspberry Pi initiates a connection, using the RFCOMM protocol. When the connection is established, the app stops looking for connections and instead starts to listen for incoming

messages The app is a simple application with a text message in the middle, displaying its status to the user. Using threading the app simultaneously listens for incoming messages and updates the text message. When the Raspberry Pi detects an emergency vehicle siren, it sends a message to the app, with the confidence of the detection, which is then displayed in the app. Figure 7 shows the final setup running on the Raspberry Pi 4 and the Android phone.

The program code for the prototype is available on GitHub. See Appendix A.

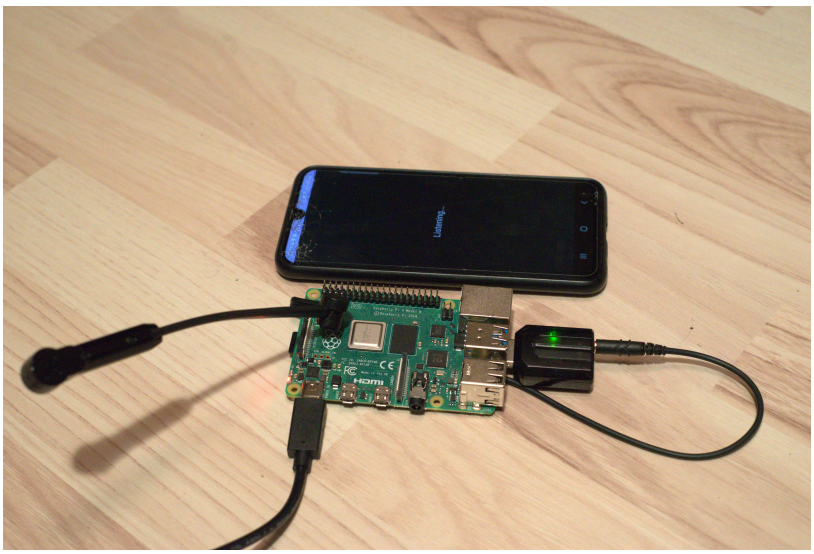


Figure 7: Image of the final setup. The Raspberry Pi 4 is actively listening for siren sounds and the app is waiting for messages.

3.3 Technical implementation

3.3.1 Hardware setup

The requirements for the microchip are the following:

- Strong enough processing power to be able to run the YAMNet ML model to classify the sound in the environment. Additionally, it has to run fast enough to give near real-time predictions.
- Needs to have a big enough memory to handle about two seconds of sound recording, store the ML model and the code to process the recording, and give predictions.
- Available Bluetooth connection to communicate with the driver's phone to send alerts when an emergency vehicle is detected.
- Finally, it has to be able to handle inputs from microphones.

Original choice

The initial choice for the microchip was an ESP32 D1 mini. An ESP32 is a low-cost microcontroller with low power consumption [2]. It has an integrated Wi-Fi and Bluetooth module. It is an ideal choice for prototyping as it is easily programmable through the Arduino IDE (Arduino Integrated Development Environment) [1]. Additionally, it has a built-in ADC (analog-to-digital converter (see section 2.5) which makes it easy to pair it up with a low-cost microphone and record digital audio samples. The ESP32 worked properly to record sounds in digital format and to transfer them to a laptop through Bluetooth. At this stage, the audio data was analyzed on the laptop using

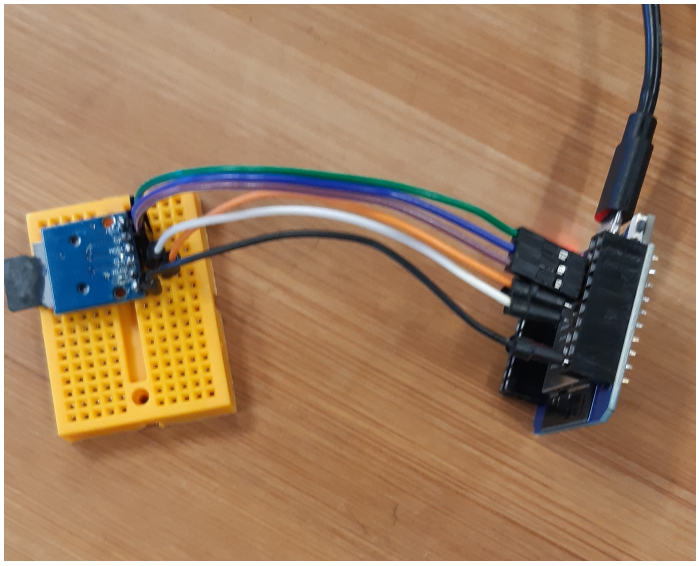


Figure 8: ESP32 runs a program to record audio data and transfer it to a laptop with Wi-Fi. At this stage of the development, the data was analyzed on the laptop. Implementing the data processing and analyzing process on the ESP32 faced storage limitations.

Python. However, when trying to implement the analyzing process on the ESP32, limitations were faced.

Encountered limitations

The limits of the ESP32 were reached with the available RAM on the chip. The available space did not occur to be enough to make a 1-second audio recording and store it on the ESP32. Besides the recording, the chip should have enough space to store the program code and the trained neural network. The first solution was to connect an SD card with a module, however, this complicated the development significantly. Figure 8 shows the setup for the ESP32. The communication with the SD card adds an additional layer to the program, complicating significantly the use of the pre-trained neural network. Additionally, it requires extra care to ensure that the program's storage demands at any point never exceed the ESP32's available RAM.

The solution to surpass the limitations was to switch to a Raspberry Pi 4. This choice not only solves the issue but has several additional advantages for the project.

Switch to Raspberry Pi

Using a Raspberry Pi for the prototype gives several advantages over an ESP32 with the main drawback being the increased cost for prototyping. A Raspberry Pi is a credit-card-sized computer with available General-Purpose Input/Output pins to connect electronic components.

The first and foremost advantage to switch to a Raspberry Pi is the increased RAM and storage available. Besides, the Raspberry Pi is a more sophisticated device than the ESP32, it can manage the communication between the running program and the SD card automatically.

A second big advantage of switching to Raspberry Pi is that it has Python natively running on it. In early development, to test the initial ideas and to see what is possible we used the Python programming language. For the ESP32 this code should have been translated to C++ code. The Raspberry Pi can run code in Python, which made it possible to implement the already tested code with small changes to match the Raspberry environment. The final advantage the Raspberry Pi has is the easier use of Tensorflow Lite on it. TensorFlow Lite is a framework developed by Google to use machine learning models. It is lightweight, making it ideal to use ML models on devices with limited CPU power.

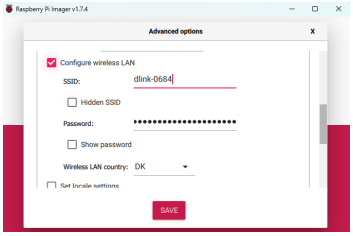
Even though switching to Raspberry Pi opens up easier development for the prototype, it required compromising as well. Switching meant that all the code programmed for the ESP32 are need to be rewritten for the new environment. However, development in Python is generally faster than in C++, therefore reusing the tested concepts in Python did not mean significant throwback.

In the final setup, an external microphone is connected to the Raspberry Pi using a USB sound card. This is automatically recognized by the Raspberry OS. It enables the use of the "sounddevice" Python module for recordings. The final configuration is showed on Figure 7

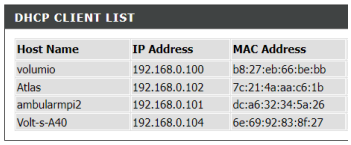
3.3.2 Software setup

Raspberry Pi

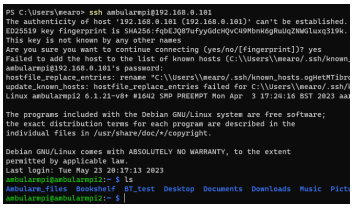
The Raspberry runs a Raspberry Pi OS which is a Linux-based operating system. The interface of the small computer can be reached in several ways. In our case the fastest development was possible, communicating to the Raspberry using ssh through the terminal. This involves connecting the laptop to the same Wi-Fi network as the Raspberry, then using the Raspberry's IP address and password to connect through the terminal. The IP address was found using the router's connection list. To make the Raspberry connect to the Wi-Fi, credentials can be set during the installation of the operating system.



(a) OS installation



(b) Finding the IP address



(c) accessing the Raspberry terminal

Figure 9: Setting up the Raspberry Pi to connect to the mobile hotspot Wi-Fi during installation of the Raspberry Pi OS (a). Then the IP address of the Raspberry shows up in the router's connection list (b), which can be used to connect through ssh in the laptop's terminal (c).

The Python environment of the Raspberry Pi is slightly different than what runs on a Windows laptop which means the same libraries cannot be used. Although most of the code can be tested on the laptop the development requires continuous checking if the code runs on the Raspberry Pi as well. The process of testing means changing the Python code to use the libraries in the Raspberry environment, then transferring the folder containing all the necessary parts for the code to run through SSH. Finally starting the program through the terminal.

The file structure for the Raspberry Pi program is important. The code uses the YamNet pre-trained machine-learning model, stored in a ".tflite" format, as well as a CSV file containing the name of the sound classes, the model can identify.

The required dependencies installed to run the program on the Raspberry Pi are stored in the "requirements.txt" file. (Available on the GitHub page, see Appendix A). However, the full list of installed libraries on the Raspberry Pi is available in the "requirements_full_list.txt" file, in order to make the system reproducible.

The Machine Learning model

The chosen method to recognize emergency vehicle sirens is using a machine learning model. The reason for this is the results of the research papers reviewed in section 2.2. Based on the review, the neural networks clearly outperform traditional algorithm methods. From the neural networks reviewed YAMNet is one of the best-performing models (see section 2.2.2), more information about the ML model YAMNet can be seen in 2.4. The pre-trained model is available through the TensorFlow framework [4], having the additional advantage that it is possible to further train the model. The

implementation of the neural network was done using the documentation of the model, available by Google[5]. To use YAMNet, the pre-trained model is downloaded from TensorFlow's website. The model is implemented with the tflite-runtime python module[3]. The newest version was found to be incompatible with the Raspberry Pi, therefore version 2.11.0, released on the 7th of December, 2022 is used in the project.

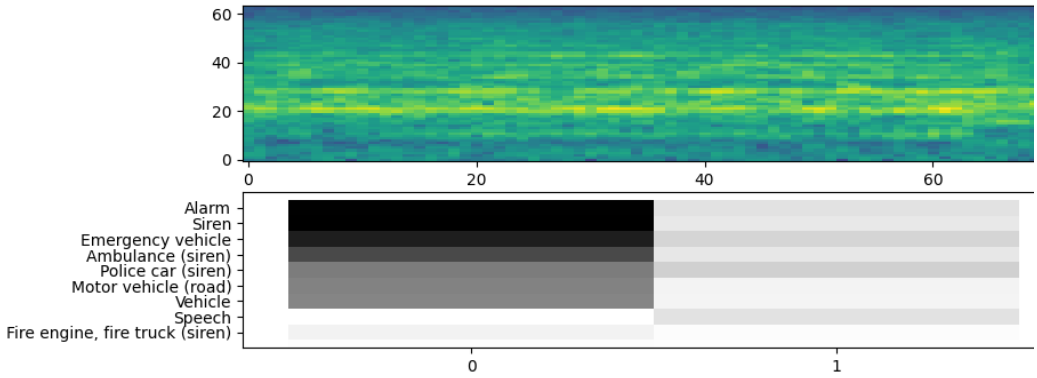


Figure 10: The YAMNet model gives predictions on a one-second audio clip. The darker color shows more confidence in the prediction. Out of 520, the 9 predictions with the highest confidence are shown. The Alarm class has the highest confidence score from them (black stripe).

3.4 The Raspberry Pi code

The program on the Raspberry Pi utilizes the YamNet pre-trained neural network, to classify the environment sounds. It listens for input using an external microphone, in one-second windows. It means it takes a recording every second and passes that recording to the neural network for classification. The neural network gives out predictions of what sounds can it hear. The program listens for the confidence level of identifying "emergency vehicle". It continuously checks the confidence level, and when a threshold is passed, it sends out a message through Bluetooth. Figure 11 shows a flowchart for the sound detection. Figure 12 shows the printouts in the Raspberry's terminal. The program prints out the confidence level for the emergency vehicle sound recognition and the predicted main environment sound as well. The prediction of the main sound shows that the program could be used for other purposes than siren detection as well.

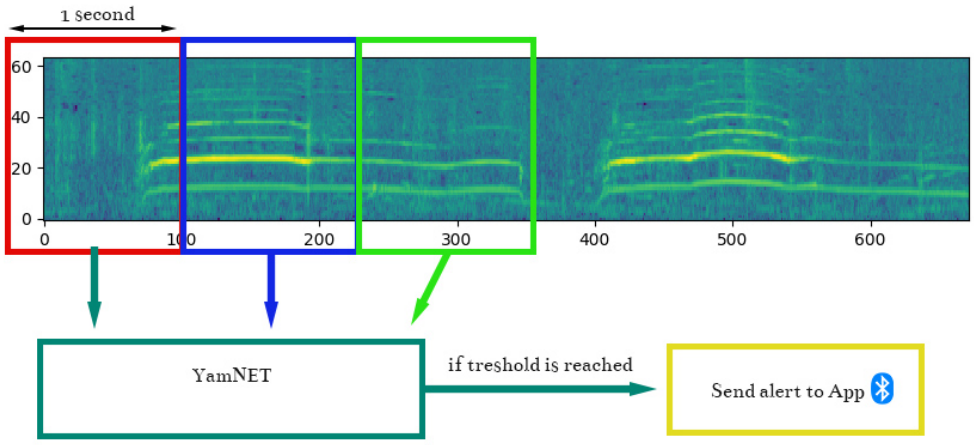


Figure 11: Flowchart for the program on the Raspberry Pi. The takes one second recordings of the environment sounds, which recordings are passed into the YamNet pre-trained neural network. When the confidence level of the "emergency vehicle" class reaches the threshold, an alert is sent out through Bluetooth.

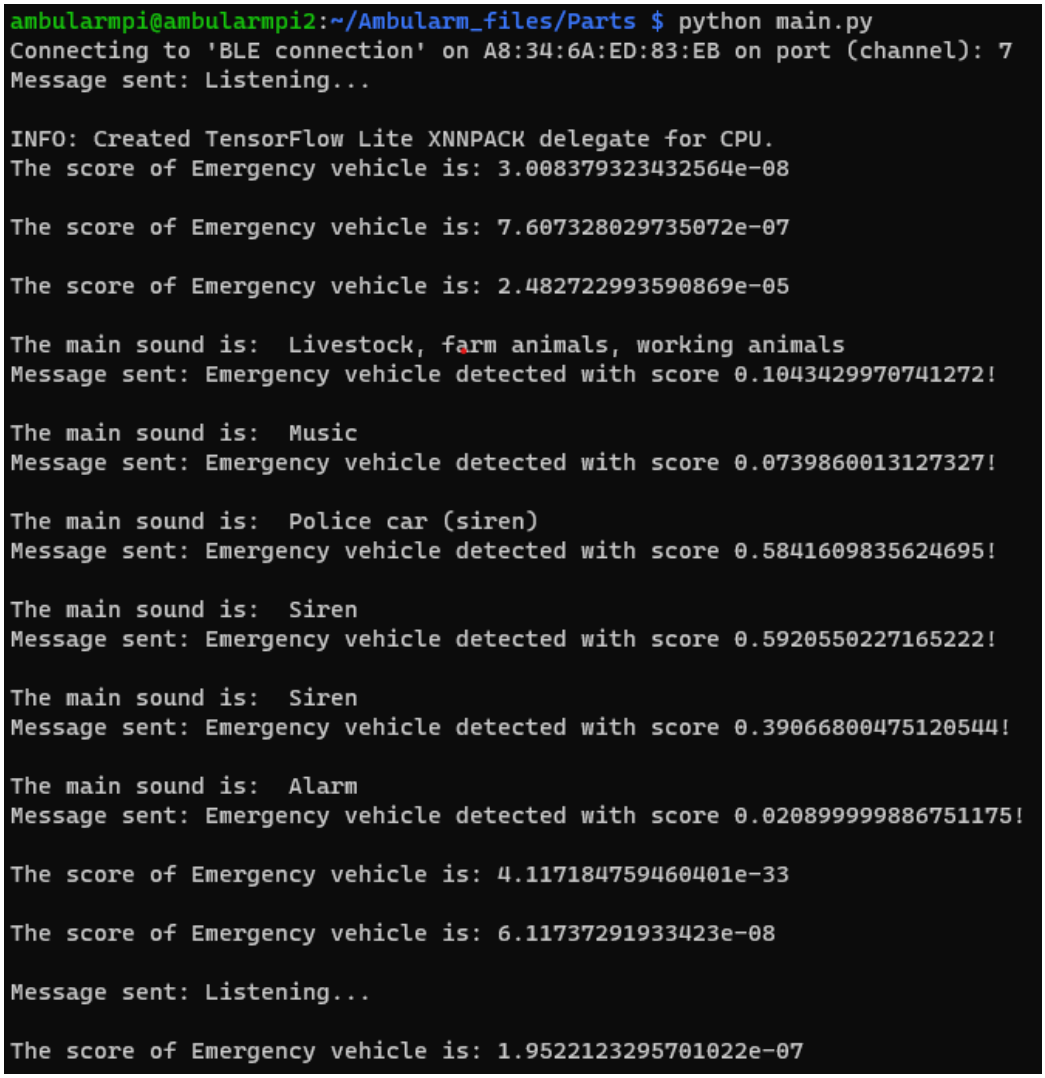


Figure 12: The Raspberry Pi terminal prints out the confidence level for the emergency vehicle sound recognition and the predicted main environment sound as well. The prediction of the main sound shows that the program could be used for other purposes than siren detection as well.

The implementation of the neural network and Bluetooth communication is done using Python classes. This design choice was made to make the program modular. This helps to make the code reusable in other projects and simplifies the debugging. The program runs from the "main.py" file, where the overall flow of the program is defined. For more details, look into the GitHub page (see Appendix A).

3.5 The Android App

The Android App is primarily used for presentation purposes. It helps to see the speed of the system, giving a view of implementing the concept can give fast enough alerts for the drivers. The app is created with Android Studio, using the Java language. It displays a text message to the user, giving information about the current state of the program. Additionally, it sets up a Bluetooth server and listens for connections. When a connection is received, it checks for new messages. When a message is received, it is displayed to the user.

It has two files, “MainActivity” and “Bluetooth”. Both files contain several classes. The design choice is to break down the program into two distinguishable parts. The MainActivity contains everything related to the user experience and manages the life cycle of the application. This includes the UI element and permission management. The Bluetooth file contains every class and method needed to establish a Bluetooth connection and receive messages. Figure 13 shows a UML diagram of the classes. The other important files are the “activity_main.xml” which contains the UI elements and the “AndroidManifest.xml” which lists the required permissions for the app to run.

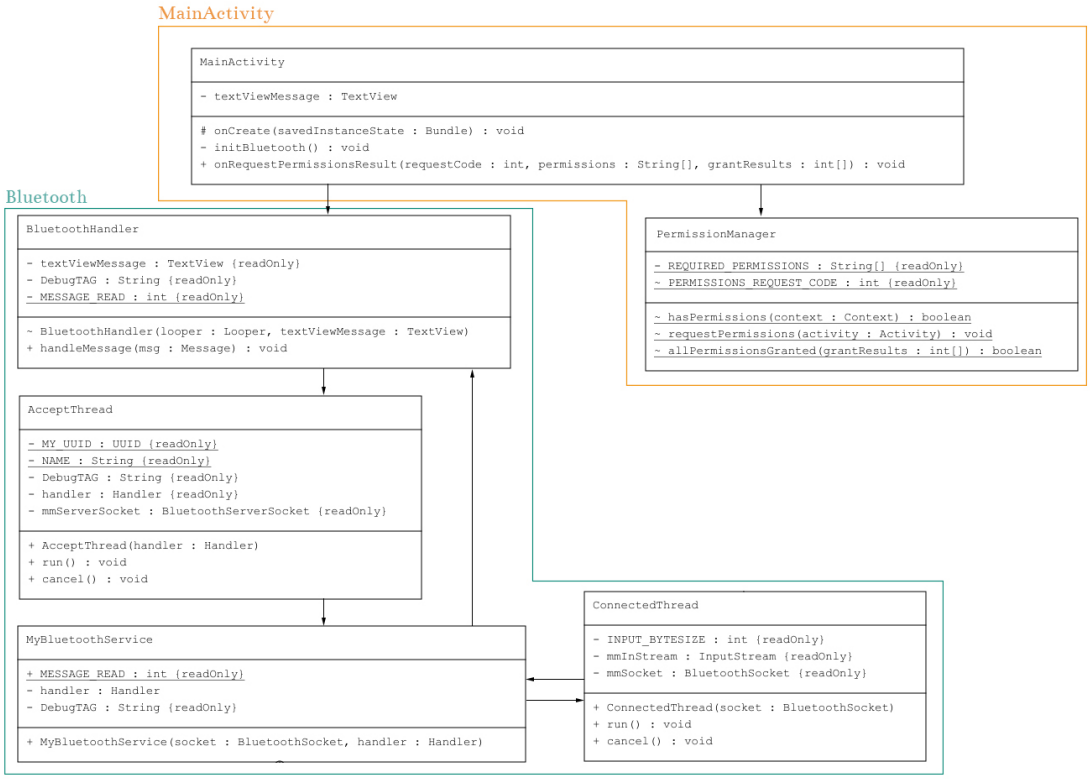
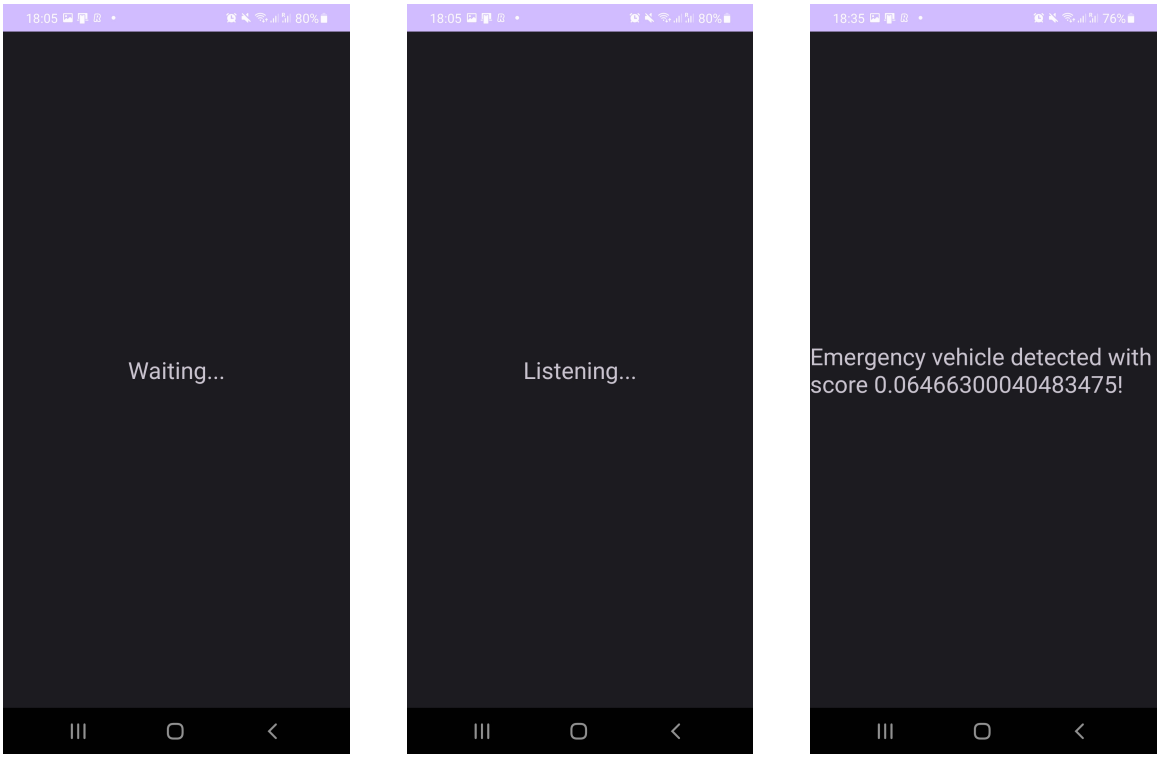


Figure 13: Class diagram of the Java classes made for the Android application. The arrows showing the flow of communication between classes.

The Android App was tested on a Samsung A40 smartphone, with Android version 11. Figure 14 shows the app running. The application was developed using Android’s “Build your first Android app” [7] tutorial. The main source for implementing Bluetooth is the Android Developers documentation on Bluetooth handling [6].



(a) Waiting for connection (b) Connection established (c) Emergency vehicle detected

Figure 14: Screenshots of the Android Application running. The application is run and tested on a Samsung A40 smartphone. It listens for Bluetooth messages from the Raspberry Pi.

3.6 Bluetooth connection

One of the trickiest parts of the program is the Bluetooth communication between the Raspberry device and the phone. The Android application has very specific requirements for permission management, set by Google. If the permissions are not managed properly, the app gives errors.

For the Bluetooth connection, the phone sets up a server to receive connections. The server has an identification, called the UUID. For the project, a UUID was generated using a random UUID generator. The Raspberry Pi and the phone is paired, then the Raspberry Pi looks for a connection, using the same UUID as specified in the phone application and the MAC address of the phone. It is important that the current implementation uses the MAC address of the phone, this means the program only works with the phone that has that MAC address. Since the MAC address is unique for every phone, this has to be changed in the Raspberry's program if the system is tested with another Android device. Using the UUID and the MAC address, the Raspberry finds the open channel on which the App is listening and initiates a connection. Once a connection is set, the Raspberry can continuously send messages to the phone. The prototype uses the RFCOMM connection protocol, which provides for binary data transport.

3.7 Testing

We conducted a simple test to see the reaction of the device. Using a decibel meter on a phone. We tested when the confidence level rise significantly for the "emergency vehicle" detection by the YamNet neural network running on the Raspberry Pi. We played an ambulance sound from a speaker. The recording had other traffic noises in the background. We found that when the ambulance sound reached about 50dB, the confidence level of the detection raised significantly, showing about 0.005 confidence. It may seem very low, however, based on our experience this level

of confidence is rarely reached, it is a good indication that something similar to a siren can be heard. Figure 15 shows shows a screenshot of the UI of the sound meter application, used in the test. This test also showed the speed of the system. When the ambulance sound was played with a high enough volume, the sound classification went through the system with almost no delay. The program needs one second to record the sound, however, after that, the classification and sending the message to the phone happens seemingly instantaneously. The program is capable of alerting the ambulance sound after it's audible within 1.5 seconds.

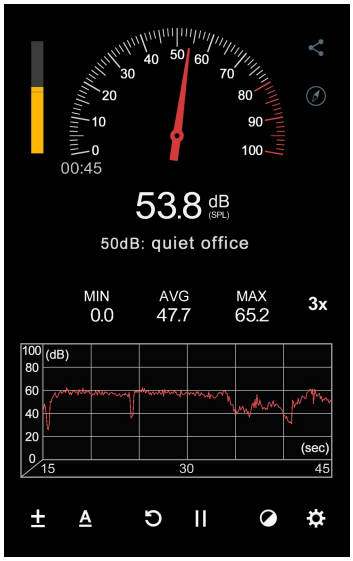


Figure 15: Conducting a simple test to see how the prototype reacts to ambulance sound. The confidence level of the neural network's prediction raised significantly around 50dB. The image shows a screenshot of the UI of the sound meter application, used in the test.

The time frame of the project did not allow for rigorous testing by the time the final prototype was created. Now we would like to propose a testing procedure that can be used in future work. The core of the siren detection is the threshold for the confidence level. When the confidence level of the "emergency vehicle" class reaches the threshold (given by the YamNet NN) it is considered a detection of an emergency vehicle. Therefore selecting the threshold is essential for the prototype. To do this we propose the following method: Placing the Raspberry Pi safely on a car, positioning the microphone preferably outside of the passenger area. Maybe fixing it to the roof. Log the predictions for the "emergency vehicle" category. Drive with no distractions (i.e. music etc.) and take notes of when the emergency vehicle siren is audible. Of course, considering safety first, maybe a second person could take notes, or using a logging application where it's possible to log with a press of a button. Analyzing the data can give a good idea about the threshold to set. Firstly looking at the maximum confidence reached by the program when there were no siren sounds. Then, compare it to what are the predictions when there were some alarm sounds. The threshold should be set somewhere between. This test can also give a good idea about the precision of the prototype.

3.8 Analysis

Using the YamNet model the prototype is capable to recognize other sounds than emergency vehicle sirens as well. It can recognize 520 different sounds, which could be used for further ideas and development. It can near real-time give predictions about the environment using purely sound input. The speed of the system is especially promising. The prototype shows that by using modern tools like a Raspberry Pi and freely available resources like the YamNet model, sound classification is possible within seconds. Based on preliminary testing it can be seen that the model reacts to low

levels of ambulance sounds. To learn more about its precision and required sound levels, further testing is necessary.

Analyzing the process to create the prototype, we can see some of the tools used are quite painstaking. Especially working with Bluetooth shows the weakness of the system. The Bluetooth connection cannot be fully trusted, which is an important factor for the potential implementation of the created concept.

Finally, an important factor is the exactness of the device. The device can recognize, that an ambulance is nearby but it cannot realize if the car which uses the device is in the way of the ambulance or not. This can lead to big amounts of False Positive alarms, even when the ambulance sound is recognized correctly. Table 1 shows a confusion matrix, of what False Positive means in this case.

	Detected	Not Detected
Car is in the ambulance's way	True Positive (TP)	False Negative (FN)
Car is not in the ambulance's way	False Positive (FP)	True Negative (TN)

Table 1: Confusion Matrix, explaining what False Positive means in the case of the project. Perfect detection of the emergency vehicle sound does not guarantee accuracy. A high number of False Positives are expected due to the car's position relative to the emergency vehicle's route.

4 Discussion

4.1 Discussing design choices

Why aren't today's smartphones being used to detect approaching emergency vehicles? Maybe in the future, but the technology is not there yet. There are some complications for why not. Firstly, the microphone on smartphones is not the most optimal option. The microphones on smartphones are designed to always detect the voice of the user, but that also means that the microphone also picks up a lot of unwanted background noise. The difference between a phone microphone and an external microphone is that the external microphone doesn't detect as much background noise [35]. If the device is the smartphone itself, the microphone is placed inside the passenger area, where the distractions (music, noise-canceling) are present. Using a separate device gives the opportunity to place the microphone in a less distracted location.

4.2 Real life usage

The created prototype provides valuable information about the viability of a similar product in real-life.

Based on the preliminary testing results, the device is capable of recognizing ambulance sounds from 50 decibels. In contrast, in a soundproofed car with music, the driver starts to hear the ambulance from 100dB [20]. This means implementing the device can alert drivers ahead of when they would realize the ambulance otherwise. That is especially true since the prototype shows, the sound is recognized and an alert has arrived to the phone within seconds. This can give valuable time for the drivers to react which can save valuable time for the ambulance to reach its destination. In the case of a juncture in a city, the driver cannot see the ambulance, it can only get information about its approach through sound. If the driver cannot hear the ambulance due to loud music, for example, it leaves a very short time for the driver to react to the ambulance. In such a situation the alerting device has the potential to save double life, by helping to avoid an accident. Saving the drivers and the patient as well.

This is increasingly true for bikers. Many people like to listen to music while biking. It is a vulnerable position in traffic, which makes it even more important for a biker to be aware of the environment. A similar device to the prototype can be made for bikers, using the electricity generated by a dynamo or from an electric bike's battery. Although it has to be considered, that such a device could make bikers with headphones braver, without offering significant safety.

The main beneficiary of a similar product would be the ambulance, and with that every person. However, the device focuses on a relatively rare traffic situation from the view of an average driver. It does not offer significant improvements for a regular road user making it unlikely to be able to sell in big amounts. Since the benefit is mainly for the common good (quick ambulance response) one possibility is to make such a system required by law built-in in new cars. It could be also spread through a campaign bringing attention to the importance of yielding to emergency vehicles.

Even if a siren is audible, it does not mean that the car is in the way of the emergency vehicle, which can lead to several false alerts. The device could be improved to lower the number of false alarms (see section 4.4, but it cannot be fully precise. The route of the ambulance is unknown and the device cannot know if the lane occupied by the car is in the way or not of the emergency vehicle. The false alarms can make the feeling for the driver that the device causes more trouble than good. It would make it a lot more appealing if such a device could recognize more traffic situations and help the driver react to them. Since the prototype can already recognize more sound than just an emergency vehicle, a similar device could be utilized for other usages as well.

Overall there are possibilities for real-life usage and there are scenarios when the device can be very useful, however, due to the low expected True Positive rate, the widespread applicability of the product is limited.

4.3 Limitations and challenges

The biggest challenge was that working with sound appeared to be significantly more complex than we imagined. Although sound signals are easy to understand for humans, it still appears to be a great challenge for computers. The classification of the signal reached good precision in recent years and the tools are publicly available, but identification of direction, speed, and angle from sound signals from traffic remains a challenging task.

The biggest limitation of the device is the incapability to truly assess the traffic situation. It cannot alert the driver of other important road situations either. This is an even more significant drawback in the case of bikers. A biker using such a device to make listening to music safer would expect defense from other dangerous situations as well. For example, a fast car approaches from behind. When a biker puts their trust in such a device, a faulty device, or imprecise results can put the biker's life in danger. Therefore the planned-out concept only seems to be a viable option for car drivers as an additional helping tool, mainly to help to give way for the ambulance.

A significant limitation was reached in the testing process. The testing of such a device would require plenty of hours driving in traffic while noting ambulance sounds. The final limitation is the time frame for the project. It enabled us to get to a rough prototype and ideas about limits and possibilities. However rigorous testing and development is not in the scope of the project.

4.4 Future work

The prototype and the development process gave us plenty of ideas of what further improvements are possible and what is necessary for a usable product. As discussed before in section 4.2, one of the important points is to lower the number of false positives. False positives can occur every time an ambulance approaches from the opposite direction as the car's direction. Or the ambulance is audible but does not even show up in the view, just passes by in a nearby street. To lower the number of false positives, a system can be implemented to check the direction and location of the ambulance. By implementing multiple microphones, then checking which microphone hears the ambulance first, the direction of the ambulance can be calculated. The change in the amplitude of the siren sound can give an indication that the ambulance is approaching or receding. If the amplitude is growing, it means the ambulance is approaching, the opposite means receding. However, this is further complicated due to the Doppler effect, since the frequency of the siren can change, making it harder to isolate. As an even more complex idea, the Doppler effect could be utilized to estimate the velocity of the ambulance compared to the car. A quickly approaching ambulance appears to have a higher-pitched siren than a slowly approaching one. A quickly receding ambulance appears to have a lower-pitched siren than a slowly receding one. Both of the mentioned methods require the isolation of the ambulance sound which is truly challenging, but it could be achieved by creating a complex ML model. It is especially challenging to reach great precision across different sounding sirens. The second, utilizing the Doppler effect part would additionally require knowing the exact pitch of the ambulance in a stationary situation, which is hardly possible for a system that works in several countries, old and new ambulances.

Although the project focused on ambulance sirens, the same principles can be applied for recognizing other emergency vehicles, and loud noise road events.

Based on the research, the need is there for a device that can alert the driver for the approaching ambulance. However, for widespread use, a sound recognizer does not look ideal due to the limited benefits for the car driver, and due to the high possible False Positives. The best approach is probably to combine it with a GPS-based approach. If the ambulance's route would be available online, when a siren is recognized, the app could check if the car's direction is crossed with the ambulance's way, and only alert the driver if there's some chance of risk. With such a system the False Positives can be brought to a minimum and if additional road situations are recognized as well, it can be a very useful tool for future drivers. Generally, it can be seen that there is plenty of potential in utilizing sound classifier programs and with that adding an important sensor to driver assistant systems. Therefore probably the highest usability and potential is achieved when combined

with other systems. That way higher precision can be reached for example for self-driving cars to assess their environment.

5 Conclusion

In conclusion, this project set out to examine the feasibility and potential utility of using sound detection systems to detect emergency vehicles. With the increasing amount of sound-proofed cars and the popularity of noise-canceling systems, the relevance and implications of this research become increasingly meaningful. The key findings from this study highlight that while there is considerable potential in such a system, there are notable limitations and challenges that have to be addressed for real-life implementation.

The overview of previous research and similar projects shows that there is a compelling need for emergency vehicle detection. While there is plenty of research, due to the complexity of the topic, the task remains challenging. The reviewed papers show that modern Deep Neural Networks outperform the precision and speed of classic algorithms. This opens up new possibilities.

The prototype developed in this project gives a good indication of what is possible with modern tools. It showed great potential in speed and sensitivity. It offers valuable time for drivers to react to the ambulance which even has the potential to save lives. We got the device to send an alert to the smartphone, saying that the detected sound is from an ambulance, the alert comes with a certain confidence. That is by itself a proof of concept. The device can theoretically be used on a vehicle, to give awareness to the driver. The device's connection to the smartphone is wireless, so the device can be mounted on the outside of the vehicle, while the phone is inside the car.

However, the prototype pointed out the flaws of the concept as well. The device's potential benefits are tempered by its limited benefits to regular road users. Even if the emergency vehicle is detected with great precision, the limited capabilities in deciding if the user is in the way of the emergency vehicle or not can lead to a high number of false positive alarms.

The research and prototype made in the project pave the path for possibilities of future work. Implementing more advanced sound detection and pitch isolation can give tools to determine the direction and distance of approaching rescue vehicles. Furthermore, the biggest potential is combining sound detection with other sensors, for example with a GPS-based solution.

Overall there is substantial potential in utilizing sound classifier programs to enhance driver assistance systems. While the device currently struggles to accurately access the traffic situation, it holds promise as a tool to help give way for ambulances. Additionally, there are other ways where sound classification could be utilized since the device is capable of identifying numerous sound activities. To further validate and improve the prototype, more rigorous testing is necessary. Given the constraints of the current project, these are for future work. With the right amount of time, the device has the potential to become a powerful tool, used not only by drivers but bikers as well.

6 Bibliography

References

[1] Arduino ide. URL https://docs.arduino.cc/software/ide-v1/tutorials/arduino-ide-v1-basics. Accessed: 23.05.2023.

[2] Esp32 by espressif systems. URL https://www.espressif.com/en/products/socs/esp32. Accessed: 23.05.2023.

[3] Tensorflow lite module history. URL https://pypi.org/project/tflite-runtime/#history. Accessed: 23.05.2023.

[4] The tensorflow framework. URL https://www.tensorflow.org/about. Accessed: 23.05.2023.

[5] Sound classification with yamnet. URL https://www.tensorflow.org/hub/tutorials/yamnet. Accessed: 23.05.2023.

[6] Android developers bluetooth documentation. URL https://developer.android.com/guide/topics/connectivity/bluetooth. Accessed: 23.05.2023.

[7] Build your first android app. URL https://developer.android.com/training/basics/firstapp. Accessed: 23.05.2023.

[8] S. W. . Smith. *The scientist and engineer's guide to digital signal processing*. California Technical Publishing, 1999. ISBN 0-9660176-6-8.

[9] K. Catchpole and D. Mckeown. A framework for the design of ambulance sirens. 50(8):1287–1301, 2007. doi: 10.1080/00140130701318780. PMID: 17558670.

[10] S. Chu, S. Narayanan, and C. . J. Kuo. Environmental sound recognition with timefrequency audio features. *IEEE Transactions on Audio, Speech and Language Processing*, 17(6):1142–1158, 2009. doi: 10.1109/TASL.2009.2017438.

[11] D. de Benito, A. Lozano-Diez, D. Toledano, and J. Gonzalez-Rodriguez. Exploring convolutional, recurrent, and hybrid deep neural networks for speech and music detection in a large audio dataset. *EURASIP Journal on Audio, Speech, and Music Processing*, 2019, 06 2019. doi: 10.1186/s13636-019-0152-1.

[12] K. M. de Paiva Vianna, M. R. A. Cardoso, , and R. M. C. Rodrigues. Noise pollution and annoyance: An urban soundscapes study. 06 2015. URL https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4918656/.

[13] J. Dennis, H. D. Tran, and H. Li. Spectrogram image feature for sound event classification in mismatched conditions. *IEEE Signal Processing Letters*, 18(2):130–133, 2011. doi: 10.1109/LSP.2010.2100380.

[14] Exxa1. Ambularm github page. URL https://github.com/Exxa1/ambularm. Accessed: 23.05.2023.

[15] B. Fatimah, A. Preethi, V. Hrushikesh, A. Singh B., and H. R. Kotion. An automatic siren detection algorithm using fourier decomposition method and mfcc. In *2020 11th International Conference on Computing, Communication and Networking Technologies (ICCCNT)*, pages 1–6, 2020. doi: 10.1109/ICCCNT49239.2020.9225414.

[16] Google. Audioset, a large-scale dataset of manually annotated audio events, . URL https://research.google.com/audioset/. Accessed: 09.05.2023.

[17] Google. Yamnet audio event classifier, . URL https://tfhub.dev/google/yamnet/1. Accessed: 09.05.2023.

[18] S. Hershey, S. Chaudhuri, D. P. W. Ellis, J. F. Gemmeke, A. Jansen, R. C. Moore, M. Plakal, D. Platt, R. A. Saurous, B. Seybold, M. Slaney, R. J. Weiss, and K. Wilson. Cnn architectures for large-scale audio classification, 2017.

[19] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A.-r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath, and B. Kingsbury. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Processing Magazine*, 29(6):82–97, 2012. doi: 10.1109/MSP.2012.2205597.

[20] C. Q. Howard, A. J. Maddern, and E. P. Privopoulos. Acoustic characteristics for effective ambulance sirens. *Acoustics Australia*, 39(2), 2011.

[21] IBM. what is deep learning, . URL `https://www.ibm.com/topics/deep-learning`. Accessed: 24.05.2023.

[22] IBM. What is machine learning?, . URL `https://www.ibm.com/topics/machine-learning`. Accessed: 24.05.2023.

[23] IBM. what are neural networks?, . URL `https://www.ibm.com/topics/neural-networks`. Accessed: 24.05.2023.

[24] S. Januzzi and L. Wolfe. Can listening to loud music increase your chances of a car crash. URL `https://www.sholljanlaw.com/blog/2021/09/can-listening-to-loud-music-increase-your-chances-of-a-car-crash/l`. Accessed: 2.405.2023.

[25] I. Kuzminykh, D. Shevchuk, S. Shiaeles, and B. Ghita. Audio interval retrieval using convolutional neural networks. In *Internet of Things, Smart Spaces, and Next Generation Networks and Systems: 20th International Conference, NEW2AN 2020, and 13th Conference, RuSMART 2020, St. Petersburg, Russia, August 26–28, 2020, Proceedings, Part I*, page 229–240, Berlin, Heidelberg, 2020. Springer-Verlag. ISBN 978-3-030-65725-3. doi: 10.1007/978-3-030-65726-0_21. URL `https://doi.org/10.1007/978-3-030-65726-0_21`.

[26] J.-J. Liaw, W.-S. Wang, H.-C. Chu, M.-S. Huang, and C.-P. Lu. Recognition of the ambulance siren sound in taiwan by the longest common subsequence. In *2013 IEEE International Conference on Systems, Man, and Cybernetics*, pages 3825–3828, 2013. doi: 10.1109/SMC.2013.653.

[27] B. NETAVIS. Ambulancernes responstider i 4. kvartal 2022. URL `https://www.beredskabsinfo.dk/praehospital/ambulancernes-responstider-i-4-kvartal-2022/?fbclid=IwAR2W_lag8CwXpmQ_d6H9-lWMhmAl6XgfrADMCfQI4fjD7Gbc5ltYhVVJKlc`. Accessed: 09.05.2023.

[28] R. Parekh. *Fundamentals of IMAGE, AUDIO, and VIDEO PROCESSING Using MATLAB®*. CRC Press, First edition. — Boca Raton : CRC Press., Mar. 2021.

[29] M. Poessel. Waves, motion and frequency: the doppler effect. URL `https://web.archive.org/web/20170914003837/http:/www.einstein-online.info/spotlights/doppler`. Accessed: 20.05.2023.

[30] J. G. Proakis and D. K. Manolakis. *Digital Signal Processing (4th Edition)*. Prentice-Hall, Inc., USA, 2006. ISBN 0131873741.

[31] D. Rane, P. Shirodkar, T. Panigrahi, and S. Mini. Detection of ambulance siren in traffic. In *2019 International Conference on Wireless Communications Signal Processing and Networking (WiSPNET)*, pages 401–405, 2019. doi: 10.1109/WiSPNET45539.2019.9032797.

[32] R. V. Sharan and T. J. Moir. Noise robust audio surveillance using reduced spectrogram image feature and one-against-all svm. *Neurocomputing*, 158:90–99, 2015. ISSN 0925-2312. doi: https://doi.org/10.1016/j.neucom.2015.02.001. URL `https://www.sciencedirect.com/science/article/pii/S0925231215001113`.

[33] R. V. Sharan and T. J. Moir. An overview of applications and advancements in automatic sound recognition. *Neurocomputing*, 200:22–34, 2016. ISSN 0925-2312. doi: https://doi.org/10.1016/j.neucom.2016.03.020. URL https://www.sciencedirect.com/science/article/pii/S0925231216300406.

[34] H. V. Supreeth, S. Rao, K. S. Chethan, and U. Purushotham. Identification of ambulance siren sound and analysis of the signal using statistical method. In *2020 International Conference on Intelligent Engineering and Management (ICIEM)*, pages 198–202, 2020. doi: 10.1109/ICIEM48762.2020.9160070.

[35] P. Technology. Improve video recorded on your phone using a stabilizer and external microphone. URL https://mediacommons.psu.edu/2022/02/21/mic-stabilizer-for-phone/. Accessed: 23.05.2023.

[36] V.-T. Tran and W.-H. Tsai. Acoustic-based emergency vehicle detection using convolutional neural networks. *IEEE Access*, 8:75702–75713, 2020. doi: 10.1109/ACCESS.2020.2988986.

[37] G. Tzanetakis and P. Cook. Musical genre classification of audio signals. *IEEE Transactions on Speech and Audio Processing*, 10(5):293–302, 2002. doi: 10.1109/TSA.2002.800560.

[38] A. A. Wieczorkowska, Z. W. Ras, X. Zhang, and R. Lewis. Multi-way hierarchic classification of musical instrument sounds. In *2007 International Conference on Multimedia and Ubiquitous Engineering (MUE'07)*, pages 897–902, 2007. doi: 10.1109/MUE.2007.159.

[39] R. W. world. Advantage and disadvantages by detecting sounds. URL https://www.rfwireless-world.com/Terminology/Advantages-and-Disadvantages-of-Sound-Sensor.html. Accessed: 20.05.2023.

[40] R. W. worldCompTIA. What is attenuation in networking. URL https://www.comptia.org/content/guides/what-is-attenuation. Accessed: 18.05.2023.

A Appendix

To access the code created for the project, go to the following GitHub page:
https://github.com/Exxa1/ambularm