MONASH University
Malaysia

Fish Bombing Detection System

TRC4001 Final Report

Name: BRENDON SOONG YUFENG

Student ID: 27953378

Supervisor: MR. KHOO BOON HOW

Abstract

Fish bombing is an illegal activity that is still practiced in Malaysia. The current initiatives either do not detect fish bombing activity in real-time, or, even if they can detect fish bombing activity in real-time, they are very expensive which make it difficult to deploy many units around the islands of Malaysia. A low-cost hydrophone is also designed to capture sound. This hydrophone was tested with an audio recorder to record sounds in a pool environment. A microcontroller-based system that uses a Raspberry Pi 4, is designed to use a deep learning network to recognize fish bombing sounds. The networks tested are LSTM and CNN networks. The LSTM network was built in the MATLAB environment whereas the CNN network was built in the TensorFlow environment. To test the Raspberry Pi network implementation, a USB microphone was used.

The results confirm that the hydrophone design can be used to record sound data. Also, a Raspberry Pi 4 is able to run a CNN network to detect fish bombing sound.

Acknowledgements

We would like to thank and acknowledge the Reef Check Malaysia team for providing us with the fish bombing audio recordings and information about fish bombing in Malaysia.

Declaration

This thesis contains no material which has been accepted for the award of any other degree or diploma at any university or equivalent institution and that, to the best of my knowledge and belief, this thesis contains no material previously published or written by another person, except where due reference is made in the text of the thesis.

Student signature: *Soong*

Student name: Brendon Soong Yu Feng

Student ID: 27953378

Date: 3/6/2021

Table of Contents

Table of Figures

Table of Tables

1.0 Introduction

1.1 Project Background

Coral reefs are vital for people and marine species around the world. They are home to more than 25% of all marine life and help to absorb wave energy which reduces damage caused by storms, hurricanes, cyclones, and tsunamis. Fish bombing (also known as blast fishing) is an illegal fishing method that threatens the health, biodiversity, and aesthetic value of coral reefs [1]. It is one of the major causes of reef degradation in South-east Asia [2]. This destructive fishing technique uses underwater explosions that stun and kill fish where the fish are harvested more easily as they float to the surface of the water. These bombs vary from hand grenades, dynamite and home-made bottle bombs made from ammonium nitrate and kerosene. Besides the loss of biodiversity, fish bombing negatively impacts the economic value of reefs for the tourism industry and for non-destructive fisheries. Studies have shown that reefs that have been prone to frequent fish bombings have been unable to recover naturally and it is estimated that the recovery for reefs around the world to range from 40 to several hundred years. The impact and risk to coral reefs are so severe that the United Nations have set a goal to eliminate destructive fishing practices as a framework for global sustainable development from 2015 to 2030.

Although illegal, fish bombing is still rampant today due to its large catch of fish with minimal effort and cost. Efforts to stop fish bombing have proven to be extremely difficult due to several reasons. Firstly, patrolling authorities are unable to cover the entire reef and open water areas where fish bombers usually operate due to the large area as well as logistical reasons such as manpower and boats. Furthermore, bomb fishermen usually operate on shallow reefs where security patrol boats are unable to follow and these fishermen usually fish during specific times of the day where no one is around. Once the bomb has been detonated, the explosion can be heard for many kilometres and its shockwave devastates the surrounding reef to a pile of rubble and destroys its inhabitants.

Currently, there are a few solutions implemented to minimize fish bombings. One strategy is where the local authorities work together with village communities to identify and apprehend bomb fishermen. Engaging the local community by conducting awareness activities and educating them is another way to tackle this issue. Also, a fish bomb detection system that alerts local authorities in real-time is being implemented. This technology is based on a system called ShotSpotter that is used to detect gunshots in US cities and determines their locations [1]. Finally, a passive acoustic monitoring system has been developed to monitor coral reefs for fish bombing activities in Sabah [2].

Therefore, there exists a need for a simple real-time fish bombing detection system to help in mitigating the fish bombing occurrences.

1.2 Problem Statement

This project aims to tackle the disadvantages of the current solutions implemented to mitigate fish bombing activities. The fish bombing detection system that utilizes the ShotSpotter technology requires expensive and complex equipment as well as has limited communication range and battery life because it uses cellular communication. It only has a range of 9 kilometres and the system can only last for 4-5 hours. Besides the ShotSpotter system, there is no system developed that can alert the local authorities in real-time. The solutions that have been implemented can only passively monitor the area for fish bombing activities. They only record audio data for a day or two. A human operator needed retrieve the data from the hard drive and then process the data using a computer. This project seeks to address these issues by developing a simple, low-cost system that can detect fish bombing.

1.3 Objectives

The objective of this project is to develop microcontroller-based system that utilizes a hydrophone and a deep learning algorithm to detect fish bombing activity. This is achieved by adhering to the following objectives:

1. Develop low-cost hydrophone that works with an audio interface.
2. Analyse audio data given by Reef Check Malaysia.
3. Develop a deep learning network for fish bombing detection.
4. Develop a cost-effective microcontroller system with hardware and software to detect fish bombing sound.

2.0 Literature Review

2.1 Land Acoustic Monitoring Systems

This section discusses the current solutions used to monitor acoustic activity from wildlife or to detect distinct sounds from the environment. The hardware and software used by the different sources will be explored here. Firstly, [6] describe using Autonomous Recording Units (ARUs) for monitoring singing passerine birds. These ARUs consists of a USB Voice Recorder SK-001 with a AC1517D72772-C processor. This processor is integrated with a single-channel electret microphone. The recorder was powered by a 12V/1.8 mAh LiPo batteries. A digital timer was used to actuate the recorder at selected intervals. The recordings were saved to a microSD card and the entire system was housed in a portable and weatherproof plastic box. Similarly, Hill [7] and a team from the University of Southampton, have developed a low-cost, small-size and low energy acoustic detector. This device is open-source and can be programmed for various applications. It consists of a printed circuit board (PCB) with a low-power microcontroller and a microelectromechanical systems (MEMs) microphone that has on-board real-time acoustic analysis capabilities. The AudioMoth device is developed to overcome the higher costs and the inefficient power consumptions faced by other devices like the Raspberry Pi. The AudioMoth microcontroller is an ARM Cortex-M4F microcontroller. The M series was chosen due to it being the most energy efficient microcontroller cores currently available in the market. The AudioMoth can process data at sample rates up to 384kHz which means that it can process ultrasonic frequencies. It can also store recordings in the microSD card. One of the main strengths of the AudioMoth is that it is modular. It can accommodate extensions to the boards such as external sensors and wireless network units. Users of AudioMoth devices can customise and design their on-board software for any sound filtering or classification using the programming language, C. Their real-time classification algorithm is built using the Goertzel filter. A spectrogram along with the Goertzel response is used to classify different sounds. Some applications of this device are seen in detecting cicada sounds as well as gunshot detection. The design can be seen in Figure 6 below.

Figure 1: AudioMoth Device housed in a plastic bag

Next, a forest fragmentation experiment in Malaysian Borneo conducted by Sethi *et al* [8], employs a Raspberry Pi based autonomous acoustic monitoring devices to monitor illegal logging activities. This system is also open source and capable of uploading real-time audio to the database and website using cellular data. The audio data is then processed using an automated audio analysis algorithm on a virtual machine (VM) hosted on Amazon Web Services. The design can be seen in Figure 3 below.

Likewise, Roe *et al* [9] has designed a passive acoustic monitoring system with real time alerts to detect the calls of an unwanted species: cane toads. Their designs include an early warning system for detecting toad calls using big audio data and Raspberry pi hardware along with cloud computing (Dropbox). The hardware of the system includes a Raspberry Pi 3B with built in Wi-Fi, a custom power management expansion board, a real-time clock, together with 4 USB devices. The USB devices include a microphone, 2TerraByte (TB) hard drive, 4G network card and a USB connection to the power management board. This system is housed in an IP67 rated and UV proof case to ensure that the system is robust to the environments. The sensor is powered through a commodity solar power system comprising: 80W solar panel, 60Ahr AGM battery and a 10A 12v solar charge controller. The microphone is an omni-directional USB microphone which is modified to be water resistant. It records in mono and at 44.1kHz using 16-bit samples. Besides storing data in the hard

drive, the audio data is also uploaded to Dropbox which is a cloud file storage system. The system employs an edge computing-based architecture with solar powered sensors and cloud computing and analysis services. The architecture can be seen in Figure 2.

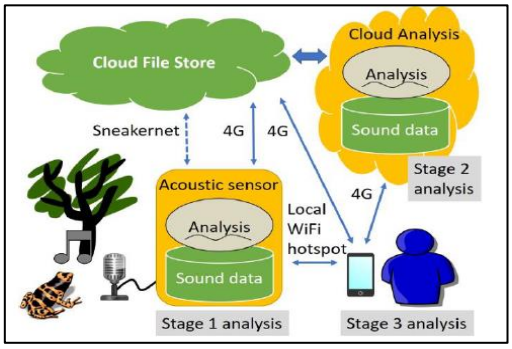


Figure 2: Architecture of Acoustic sensor system with Cloud storage and analysis.

Once a suspicious toad call is detected and analysed by the sensor, it is filtered by the cloud analysis and then reported to the user. The user then checks the data to verify that it is not a false positive. This architecture ensures 3 level checking where the sensor, cloud and user are able to check the validity of the data. The design can be seen in Figure 8 and 9 below. A table of acoustic monitoring tools can be found in Appendix A.

Figure 3: Microcontroller system (Raspberry Pi 3, Arduino, microphone, power management board and Hard disk).

2.2 Marine Acoustic Monitoring and Fish Bombing Detection Systems

This section discusses the current solutions and acoustic detection/monitoring technology in marine environments. Important studies for fish bombing detection are also discussed here. To start with, Wood and Ng [1] have collaborated with Sabah Parks to develop and install an underwater detection system that monitors fish bombing incidents and to provide real-time location data. They first constructed a hydrophone frame that is used to collect underwater acoustic data such as fish bombing sounds. The hydrophone frame is built such that the hydrophones are equal distance from each other and are 1m apart, at an angle of 90 degrees to the sea surface. The hydrophones are then connected to a Decimus unit via cable and they convert sounds into electrical signals where the Decimus unit transmits these signals to the base station laptop computer. The frame and Decimus unit can be seen in Figure 10 below.

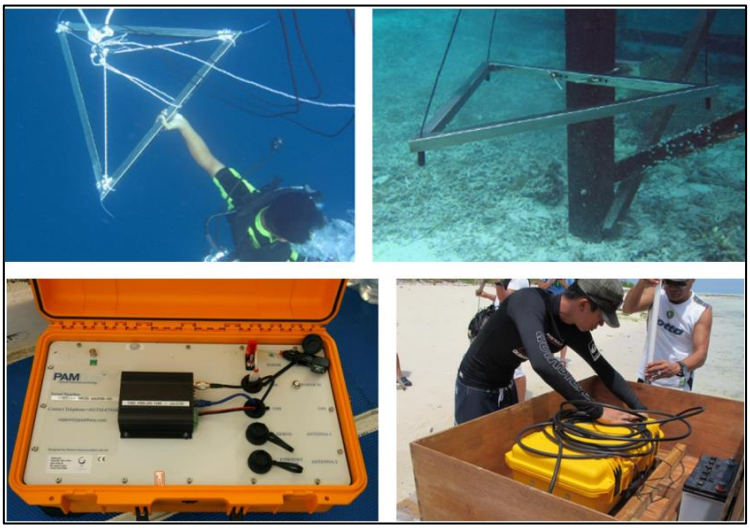


Figure 4: Hydrophone frame attached to cable on to boat. Hydrophone frame attached to the leg of a jetty.

While testing the frame underwater, the team found out that it was difficult to keep it firm at an angle of 90 degrees to the sea surface. Thus, the hydrophone was attached to the leg of a jetty. This can be seen in Figure 11. However,

they found that the fish bomb noise is blocked by the reefs nearer to the shore. 2 solutions were implemented to tackle this issue. One was to attach the frame to concrete blocks for the structure to be anchored to the ocean floor. Another method was to weld the hydrophone frame on to a long pole and hook the pole to the side of a boat. These solutions can be seen in Figure 12, 13 and 14.

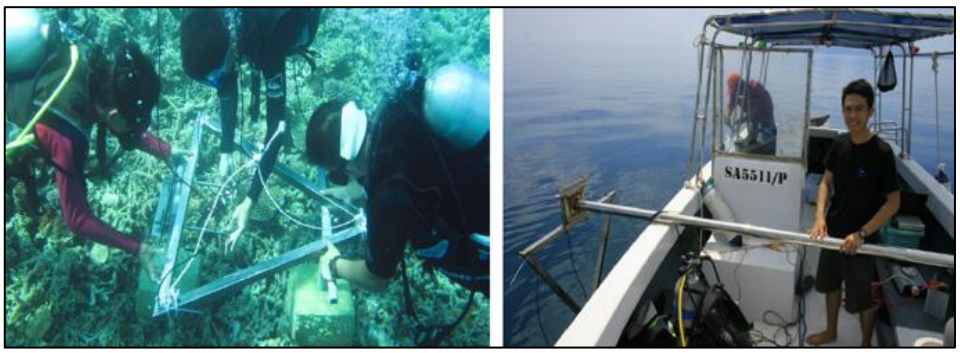


Figure 5: Hydrophone frame tied to concrete blocks to anchor it to the ocean floor. Hydrophone frame welded to metal pole.

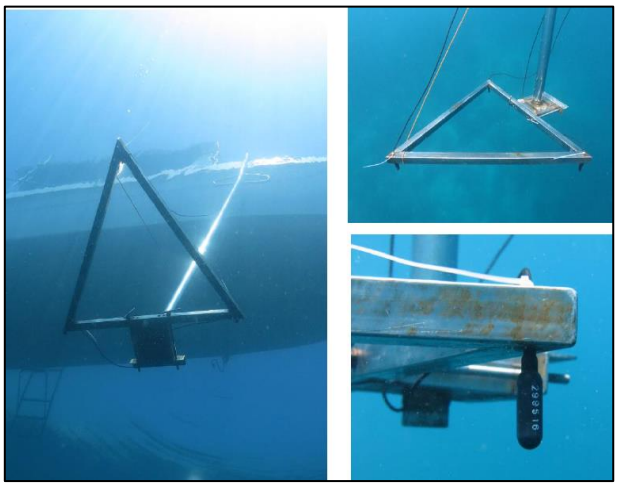


Figure 6: Hydrophone placement and frame placed parallel to water surface.

To record and collect sound data for sound classification and configuration of the PAMGuard software and Decimus hardware system, the hydrophone on the frame are connected via cable to an audio interface which is connected to a laptop. They recorded 102 hours of data and used that to calibrate the hardware and software. The Decimus system consists of the Decimus passive acoustic detection system with 2,4GHz communications option, a 2.4GHz collinear antenna, a portable 12V 100W solar battery charging system with

regulators, cables and battery connectors, hydrophone with 8m cables, a 2.4GHz wireless modem receiver and a laptop with PAMGuard software. The hydrophones used were Bruel and Kjaer Type 8103 hydrophones.

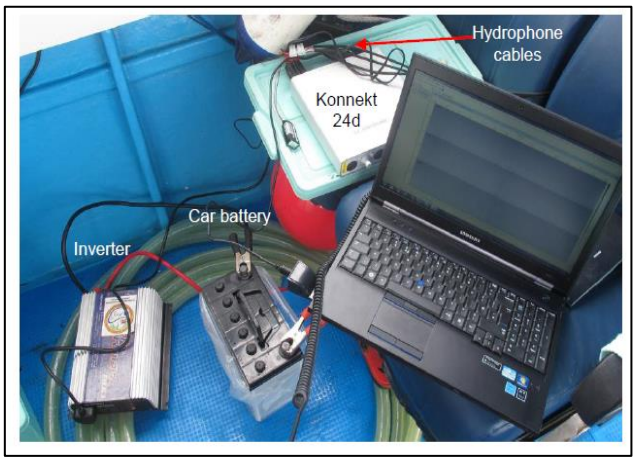


Figure 7: Hydrophones connected to audio interface (Konnekt 24d), inverter, car battery and to a laptop.

Next, the team had to determine suitable locations to establish the base station and the locations for the detection systems (hydrophones and Decimus units). The base stations had to have reliable 24-hour electricity supply, permanent staff and had to be clear of any obstructions to ensure that the signals from the Decimus unit did not interfere with surrounding reefs or islands. The detection system had to be placed in locations where there was no island blocking it from the base station. An example of ideal hydrophone locations can be seen in Fig, 16. Locations of the hydrophones are tracked using GPS.

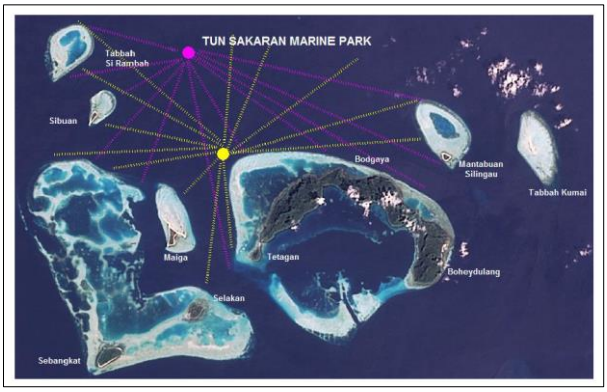


Figure 8: Potential placement of hydrophone and Decimus units.

Factors that affect the accuracy of the hydrophone data are diffraction of sound waves and depth of hydrophone placement. Sound waves can be diffracted around obstacles such as island and reefs and can affect detection accuracy. Also, hydrophones need to be placed at a depth of 4m to maintain a stable temperature and to avoid distortion of sound waves that can occur close to the water surface.

The PAMGuard software was used to monitor and analyse the fish bomb incidents over time. The signal can be displayed in waveform plots, Wigner plots, spectrogram, and a spectrum plot. The incidents can also be plotted on the map of the surrounding area.

Some limitations of this system to note of is that it is susceptible to theft from the fishermen who practice fish bombing. Also, the system communicates via cellular 3G data which limits the transmission range for communication between the Decimus unit and the home base. Besides this, the costs of the hydrophone and Decimus unit is very high for larger scale implementations. In addition, even if fully charged, the Decimus unit has a battery life of 405 hours.

Similarly, Oswald *et al* [10] utilizes PAMGuard software and its click detector module to monitor presence of killer whales in the north-eastern Pacific Ocean. PAMGuard also contains newly developed tools within the Real-time Odontocete Call Classification Algorithm module to analyse different parameters of the killer whale echolocation. To collect the data recordings, they used a 2-element hydrophone array that was towed 200-300m behind a research vessel. We can see that this system is an active system which requires user's effort compared to the previous Decimus system which is a passive monitoring system. These recordings were stored in a computer hard drive using 96kHz sample rate. To classify the distinct whale sounds, the team used a random forest classifiers model. The model takes a combination of information from echolocation clicks, whistles and pulsed calls to identify the groups of killer whales. The towed array of hydrophones and classification method in this study is a possible method in detecting fish bombing sounds.

Moving on, Showen *et al* [5] discusses the use of the ShotSpotter system, which is a gunshot detection system used in America. They took this system and implemented it to detect fish bombing sounds. The system hardware was modified from the original ShotSpotter system. Mainly, the microphones were removed from the circuit board and replaced with Aquarian model H2a hydrophones. Figure 9 shows the system used. Each ShotSpotter processor board comes with GPS receivers to provide the timing and location details for the triangulation. The software used is the ShotSpotter proprietary software. This system transmits data similarly to the system in [1][11] where cellular data is used. Once the system was constructed, it was mounted on to piers and tested using make-shift fish bombs that were detonated at a distance where the reefs would not be harmed. The method to triangulate the fish bombing will be discussed in another section.

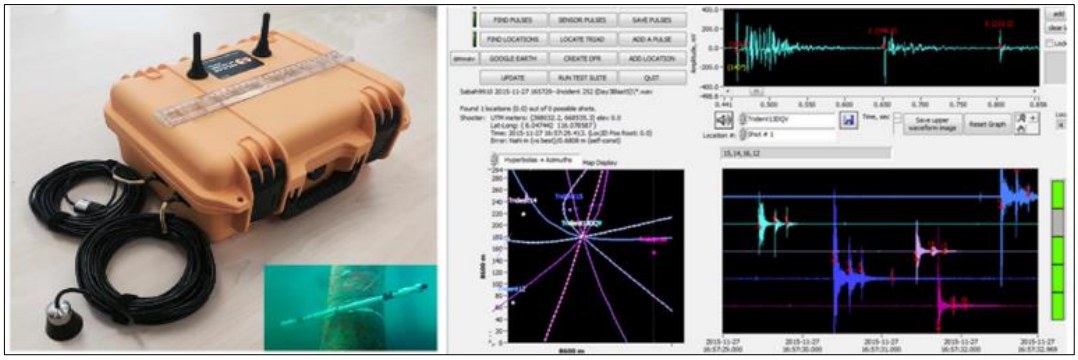


Figure 9: (Left) ShotSpotter system housed in a waterproof plastic box with 2 hydrophones are tied on to a pole and attached to the leg of a pier. (Right) ShotSpotter proprietary software.

Another study done on acoustic monitoring of blast fishing was conducted by Cagua *et al* [11] from the Worldwide Fund for Nature (WWF). The deployed Digital Spectrogram Long-Term Acoustic Recorders (DSG) from Loggerhead instruments for a span of 2 days and another for a span of 2 months. These recorders consume low power and are designed to sample sound up to 200kHz. They recorded 45 underwater explosions on to a 128 GB SDHC card by using a sample sound of 80kHz. Using these recordings, they developed a neural network that semi-automatically identifies and characterizes explosion-like recordings. Results show that the range of the DSGs are around 7-15km. Using a sample sound of 80kHz resulted in a lot of information. The team then

divided the recordings into several thousand files, each of size 10MB and containing about 1 minute of sound. The instruments were set up to record sound in 5-minute intervals from 0600 hours to 1900 hours daily. To find the explosions within the recordings, the root mean square (RMS) value of the signal was calculated in a 20ms window and signals that surpassed the 128dB threshold were identified as explosions. The bimodal distribution of the sound was identified and used to determine the blasts and other high-intensity sounds. The acoustic characteristic of the blasts is discussed in another section.

For the neural-network classification used was a 2-layer feed-forward neural network with 30 hidden and 1 output sigmoid neuron. The inputs to the network are the peak-to-peak maximum pressure, the equivalent energy, the pulse length, and the RMS value of the signal in the time between the equivalent energy reaches 5 and 95% of its total value. The equivalent energy is a measure of the amount of acoustic energy contained in a signal. It is obtained from integrating the square of the sound pressure level (in uPa) over time and is expressed in decibels (dB rel. uPa^2s). The neural network was trained with 70% of the samples; 15% was used for testing and the other 15% for validation. Scores from 0 to 1 were assigned to reflect either a low probability of a blast or a high probability respectively.

Lastly, Braulik *et al* [4] describes a survey conducted to monitor fish bombing activity in Tanzania. This system consists of a Vanishing Point stereo towed hydrophone array that was deployed on 100m of cable from the rear port-side of a boat. The hydrophone elements were connected to a HPO2 preamplifier with 28dB gain and a low-pass filter. These were also connected to a TASCAM DR-680 recorder with 2 continuous channels, 192 kHz and 24-bit recordings. These audio files were saved in .wav format into hard drives at the end of the day. For the analysis of acoustic data, the team used PAMGuard software as well. The entire dataset was manually examined by the researchers. Based on the acoustic characteristics of fish bombing sounds, the researchers developed an automatic detector for blasts using a simple energy band

detector available within PAMGuard. The team also developed a custom MATLAB script to extract time delays from detected bomb blast.

2.3 Fish Bombing Signal Characteristics

Fish bombing acoustic signals have distinct characteristics. From [3], the results from the tests done show the peak pressures recorded at 390m from the blasts. This is shown in Figure 10. At 2 km from the blast, a well-defined shock wave is observed. The signal shows a vertical leading edge followed by falling peaks. As the distance from the blast increases, the signal signal's leading-edge decays rapidly. Also, as the distance increases, the time delay between the leading edge and the reflected signal decreases as seen in Figure 11. At distances greater than 3km, the maximum pressure peak starts to occur in the main body of the signal as seen in Figure 10 and 11. It can also be seen in Figure 11 of an inverted signal after the leading edge. This is caused by signal reflections from surfaces. This is important to note of as future tests may be affected by this. Figure 12 shows a blast signal with time delay that was recorded by a few hydrophones. Each hydrophone recorded similar blast signals with sharp peaks.

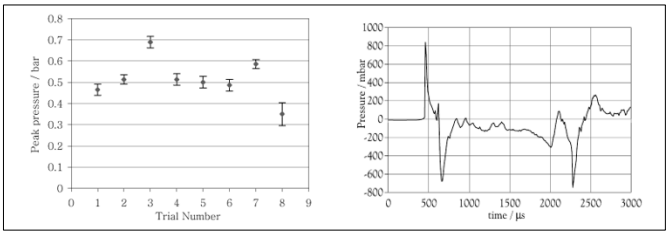


Figure 10: Peak Pressure of repeated blasts at 390m, 250 distance

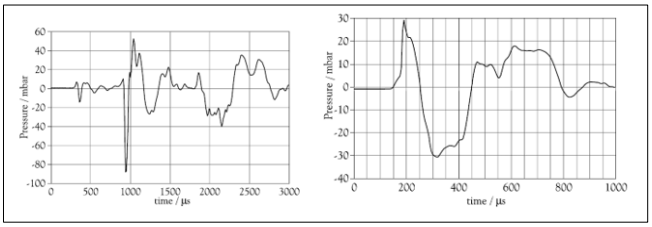


Figure 11: Blast signal at 2600m with 80us time delay, 3000m

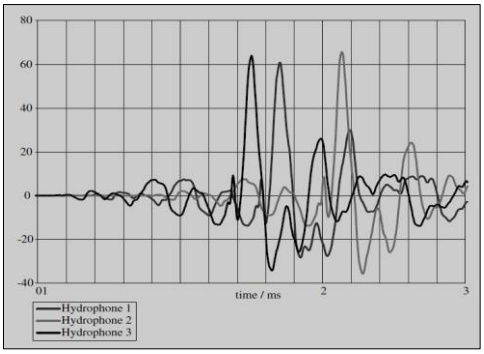


Figure 12: A blast signal recorded by 3 hydrophones

Similar results can be seen in [4], where they noted that the blasts had an intense arrival followed by a tail several seconds long. It can also be seen that most of the energy is contained within the first 0.2 seconds. Blasts frequencies range from 5kHz to 10kHz depending on the distance from the hydrophone. The peak pressure of the blasts approximates to 161 dB. Next, according to [5], they mentioned that an underwater blast produces rapid oscillations of gas bubbles. This oscillation is a key feature for identifying the blasts. They state that the period of the bubbles is predicted by the Rayleigh-Willis formula $T = \frac{(0.0452Q^{\frac{1}{3}})}{(\frac{D}{0.3048}+33)^{\frac{5}{6}}}$ where T is the time period of the oscillation in seconds, Q is the source in energy in Joules, and D is the depth of the bubble centre in meters.

2.4 Multi-Lateration/Source Triangulation

There are numerous ways to determine the location of a fish bombing blast. Directional microphones may be used to capture acoustic information from certain orientations. However, omnidirectional microphones can sample sound with similar efficiency in all directions. [acoustic terrestrial] describes performing cross-correlations among the waveforms from all microphones to extract their relative time delays from which the source location can be estimated in multiple ways [12]. This method is suited for single-sound sources. Another method proposed by [12] is the correlation sum approach which is to compute cross-correlations between waveforms from each channel

and determining the source location with the highest probability by using the Hilbert amplitude envelope of the cross-correlation functions.

[13] discusses the use of the Time Difference on Arrival (TDOA) method for determining the direction of a stationary source. 2 microphones pick up sounds emitted by the source and sounds are sampled by a soundcard and is processed. The processing function consists of a method to extract the timing information from the signals and this information is subtracted from each other resulting in a difference in time. This difference can describe a hyperbola which goes through the position of the source. The asymptote of the hyperbola will provide a direction. The illustration of this method can be seen in Figure 13 and 14.

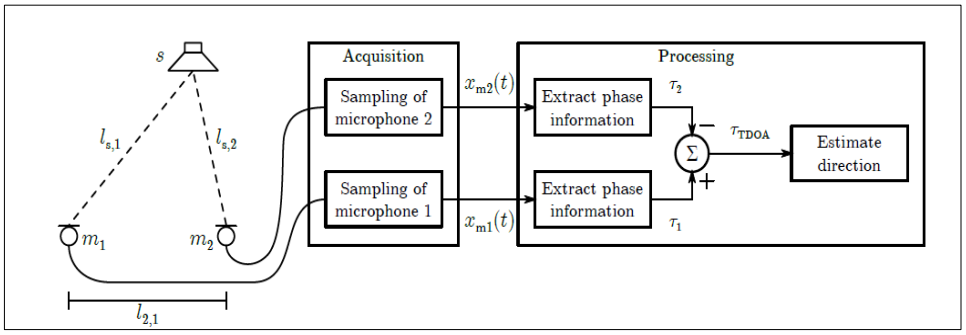


Figure 13: Overview of method

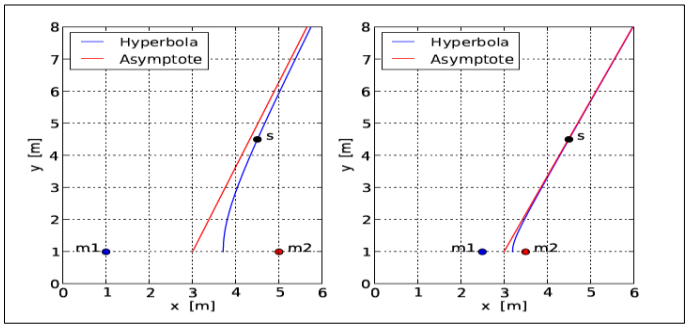


Figure 14: Asymptote in two cases with equal parameters except for the microphone separation, which in the left-hand side is 4 m and in the righthand side 1 m.

To calculate the TDOA using the general cross-correlation (GCC) method, the 2 signals are cross-correlated and the argmax of the correlation will be the offset in time. However, this method requires a high sampling rate and a large microphone separation to get a good angular resolution for angles close to the axis of the microphones. Another method apart from the GCC method is to calculate the complex Fourier transform of the signal and extract the phase information. Then, the phase has to be converted to a time value which will have a better resolution largely independent of the sampling rate. While calculating the TDOA, an analysis of the wave propagation from the source is required. The derivation of the entire formula to calculate TDOA and Source Direction can be found on page 9-12 of [13].

[3] presents 2 approaches to determine blast direction. In Figure 15, the first approach is to record the time of arrival of a blast signal at 3 widely spaced sensor units. The origin of the signal can be determined using coordinate geometry. However, this approach requires synchronization between the timing circuits at different sensor units. This is because electronic clocks running on different power sources are susceptible to drift with respect to each other. Even an error of 1s is enough to produce an error of 1.5km from the predicted location. Of the blast event. Approach 2 is to use an array of detectors, each spaced 1m apart, at each detection station. Since the detectors will measure different timings, it is possible to determine the direction of travel of the blast wave. This method requires a minimum of 2 detection stations for triangulation and is represented in Figure 15 (right).

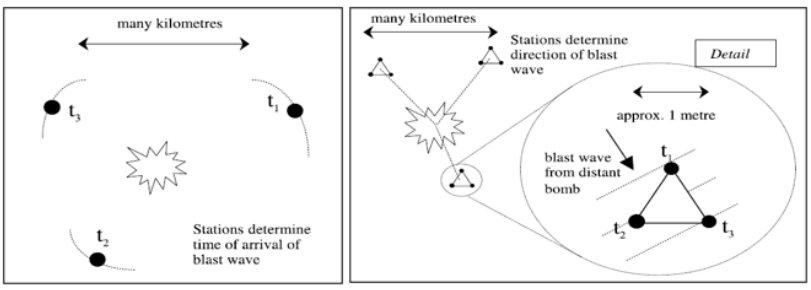


Figure 15: Method using single widely spaced detectors for a detection station. Method using 3 evenly spaced detectors for a detection station

2.5 Internet of Things (IoT) in ocean environments, Smart Ocean

This section explores the literature done in the area of Smart Ocean or Internet Underwater of Things where IoT technology is implemented in marine environments. The challenges, limitations, methods, and designs for integrating sensors and microcontrollers in underwater/water surface environments are discussed here. When researching about the different communication technologies available for underwater applications, [14] states that the propagation of radio waves underwater is not good due to large propagation delays of acoustic waves and high bit error rates of the underwater acoustic channel that disrupt communication. Not only this, but the speed of sound also varies with depth, temperature and salinity of the water. Acoustic propagation is especially affected in shallow water (water with dept<100m) because of temperature gradients, surface ambient noise and multipath propagation due to reflection and refraction. Besides this, underwater devices are prone to many causes of failure like fouling, corrosion and rust caused by sea water. Another issue concerning underwater devices is their limited battery capacities. Some of the promising energy harvesting technologies include solar energy, ocean thermal energy and piezoelectric energy harvesting. Piezoelectric energy harvesting involves using long strips of piezoelectric polymers which undulate in water flow and converts mechanical energy into electrical energy.

Next, Figure 16 showcase an overview of underwater device systems design (left) and a schematic depicting the functions and positioning of electronic equipment when deployed underwater.

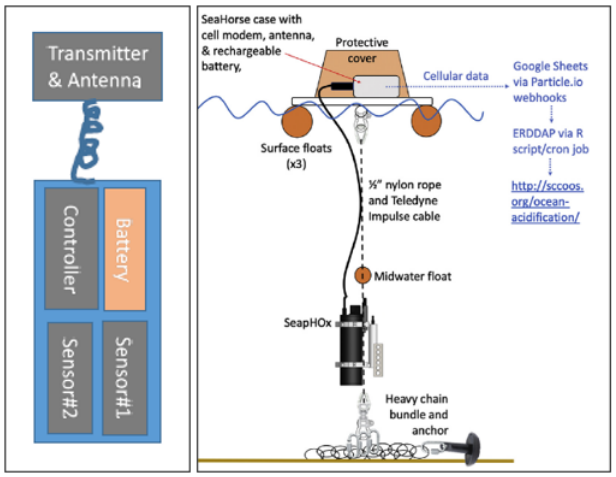


Figure 16: (Left) General diagram of an underwater device system; (Right) Schematic of electronic components anchored to the sea floor

2.6 Extra features and hardware to consider

Some of the hardware used in literature are those that can be bought of the shelf. These include an Arduino with the LoRa and GPS module that can be used for the detection and communication between nodes and gateways. An AudioMoth board can be used to record the audio data. A Raspberry Pi can also be used however they usually consume more energy compared to the Arduino or AudioMoth devices. The LoRa module chosen is the Cytron LoRa shield/RN2903 LoRaWAN module which operates at 919MHz to 023MHz which is regulated in Malaysia. For the GPS module, VK2828U7G5LF GPS Module is recommended due to its cost effectiveness and its off-the-shelf availability. To power these components, a LiPo battery can be used. To further sustain the battery life, the LiPo battery can be connected to a Solar LiPo charger with 5V 2A solar panels.

Once the timestamp and GPS location has been received by the gateway and uploaded to the cloud server via WiFi/Cellular, the data will be extracted by a web app that constantly waits for an entry to the cloud server. The web app, written in the JavaScript language, can extract the data from the cloud server and display the information on its interface. The app can also make use of APIs like Google Maps to show the rough location of the explosion from the given

GPS coordinates. Some of the cloud servers that can be used include Microsoft Azure and Google Cloud. The Things Network database can be used with the gateway.

As for wireless communication between nodes and servers. Initially, LoRa was thought to be the suitable communication protocol for this application. However, it is found that LoRa gateways are expensive and not easily accessible here in Malaysia as seen in Figure 17. The LoRa range between gateways and nodes is not very large as well. Thus, it is decided to use Cellular or NB-IOT as the mode of communication because the sensor will most likely be deployed n 5-10km from the shore. This means that the node will still be able to receive cellular connection.

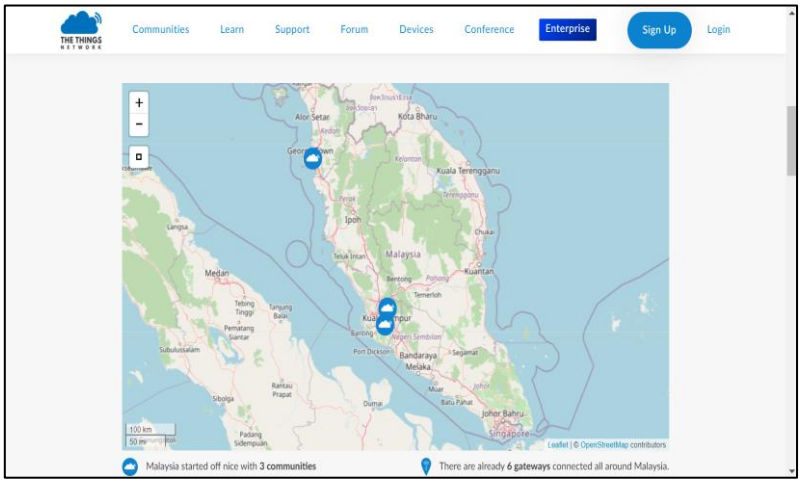


Figure 17: Current available Lora Providers in Malaysia

2.7 Hydrophone Design

Hydrophone Method 1

This method is the simplest and only involves using a 6.3mm guitar cable, contact microphone, 6.3 to 3.5mm adapter, electrical tape and Plasti Dip spray. First, connect the guitar cable and contact microphone together using the adapter. Then, wrap the connections with electrical tape and then spray layers of Plasti Dip to rubber coat the cables to make it waterproof. This rubber

coating fills any gaps or leaks that may not be visible to the human eye. Link for the video:

https://www.youtube.com/watch?v=E8UUa7wKAuA&t=12s&ab_channel=RebaDarling

Hydrophone Method 2

This method [15] and [16] uses 2 acrylic sheets and placing an O-ring in between and the piezo ceramic disk in between. The acrylic sheets are 2-3mm thick. Holes are drilled around the acrylic for the screws. These screws can be made of Nylon or Brass as they are resistant to corrosion from seawater. The piezo ceramic element must be smaller than the O-ring. It is glued to the surface of one of the acrylic sheets. The coaxial cable is fed through a hole in the top sheet or through the O-ring, and is soldered with the piezoelectric connections. Then, the hole is is sealed with epoxy. The coaxial cable can be connected to a 6.3mm jack or a 'coaxial to USB' adapter may be required.

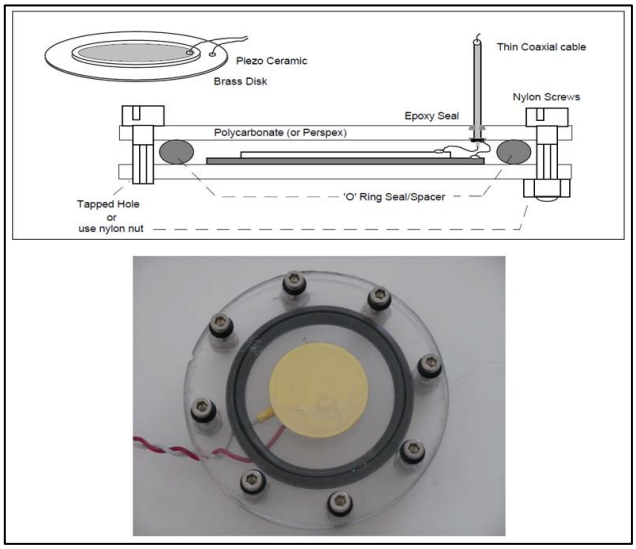


Figure 18: Simple hydrophone schematic for Method 2

Hydrophone Method 3

This method [17] is more complex compared to the previous methods. This guide has instructions for building the pre-amplifier circuit, attaching the circuit to cables and the hydrophone element, building a mould cast using resin and finally enclosing the circuit in the mould.

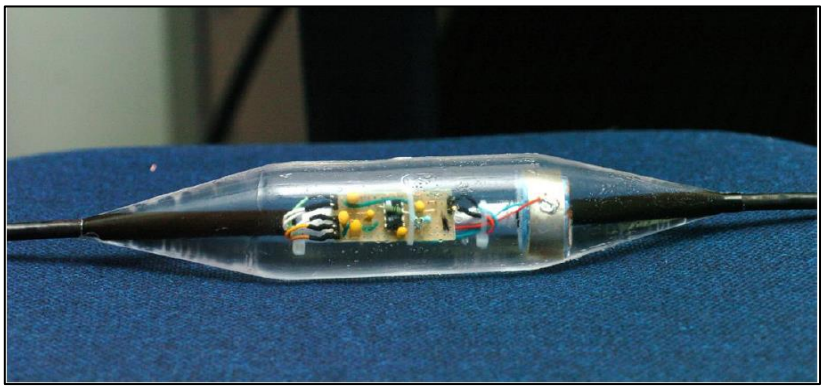


Figure 19: Assembled hydrophone for Method 3

2.8 Fish Bombing Activity Simulation

A few methods were tested to see if the recreation of fish bombing sound in a controlled environment is possible. Safety of the researcher while conducting the test is of high priority. The first method is to fill a balloon with air and submerging it underwater near the hydrophone before popping it with a needle. Next, a firecracker ignited and thrown into a body of water near the hydrophone can also be tested as well. Thirdly, a plastic bag bomb made from mixing vinegar and baking soda is a promising method to test. Lastly, a plastic bottle bomb made from either mixing vinegar with baking soda or mixing toilet cleaner solution (HCL) with aluminium foil can be used to recreate an underwater explosion sound. However, this method may be dangerous because of the exploded plastic bits that may hurt anyone standing within the explosion radius.

2.9 Detection, Multi-Lateration Algorithm

As mentioned in the literature review, there are many methods for detecting and locating a blast event. Using a frequency analysis algorithm can be used

to detect the bomb sound. Another way is once the signal crosses a certain threshold, it means that a blast it detected. One way of improving on this is to make sure that the surrounding nodes also detect that sound to be classified as an explosion. This is to minimize the error and false positives that can be caused by animals hitting the hydrophone or the sound of boats passing by. The Time Difference on Arrival (TDoA) and Angle of arrival method is one option. Another option is to use the GPS module on each detection unit. Once an explosion is detected, the nearby detection units will send their timestamps and location to the gateway via LoRa. The earliest timestamp is chosen and its corresponding detection node is chosen as the bomb's location. This method is favoured as the open ocean does not have many obstacles in its path so although the exact location of the blast event is not given, the authorities should still be able to spot the criminals.

2.10 Experimental Design/Approach

Firstly, the 2 methods of hydrophone from the literature mentioned were built. Figure 20 involves a prebuilt contact microphone attached to a 3.5mm audio cable. Figure 21 is to build the hydrophone from scratch using a piezoelectric element and soldering it with a coaxial cable.

METHOD 1

Figure 20: Method 1 Hydrophone

METHOD 2 (using hot glue gun adhesive for the piezo element)

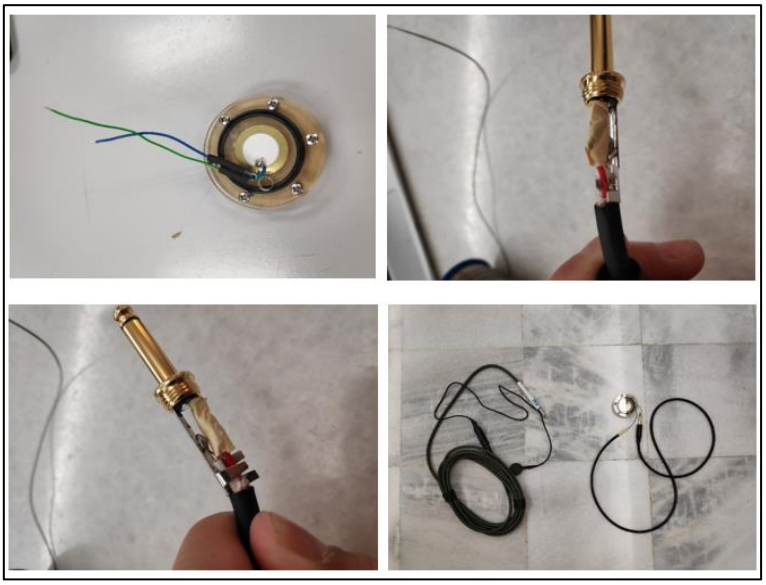


Figure 21: Method 2 Hydrophone

Once the prototypes were built, they were tested in various conditions:

a. Tapping on hydrophone in air
b. Hydrophone immersed in bucket filled with water. Tapping on the side of the bucket
c. Tapping on hydrophone in water
d. Metal clanging in water while hydrophone is submerged in bucket of water
e. Balloon explosion in bucket of water (hydrophone in water)
f. Firecracker in bucket of water (hydrophone in water)
g. Balloon explosion in an inflatable pool at different distances

The inflatable pool experiment guide can be seen below:

1. Inflate POOL with ELECTRIC AIR PUMP/BICYCLE PUMP.

2. Connect HYDROPHONE to AUDIO INTERFACE and LAPTOP. Run AUDACITY software. Fill POOL with WATER in the meantime.

3. Attach BALLOON to hose and use CLAMP/PEG to hold the end in place.

4. Use MEASURING TAPE, place HYDROPHONE at one end of the POOL. Use tape and measure 5, 10, 15, 20, 25, 30, 40, 60, 80, 100cm from the edge of HYDROPHONE to edge of HOSE connected to

BALLOON. Use BRICKS to secure HOSE and HYDROPHONE in place.

5. Press RECORD on AUDACITY. Blow to inflate balloon.
6. Once BALLOON explodes, turn OFF recording on AUDACITY.
7. Replace BALLOON and repeat steps 4-6 for different distances.

3.0 Deep Learning Network

The link for the repository for all codes and datasets can be found in

Appendix E.

3.1 Datasets used for training network

Various datasets were used for the training, validation, and testing of the deep learning network. For each network training trials, different combinations of datasets were used. The datasets used are:

Gunshot dataset

This gunshot dataset was taken from 3 online databases [19][20][21]. A total of 18469 audio files were pre-processed (padding or truncating) and used as the training dataset. This dataset contains audio of varying length and from various types of guns recorded by different types of phones.

Miscellaneous Dataset

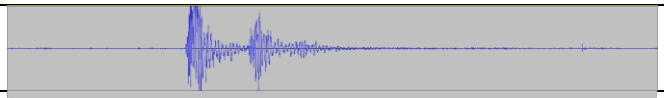
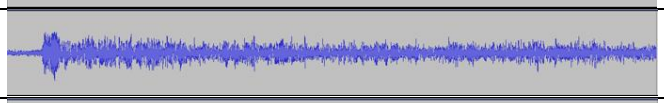
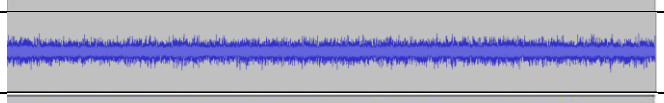
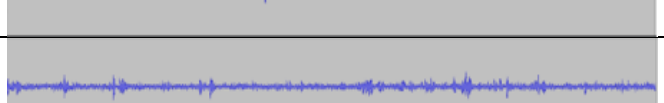
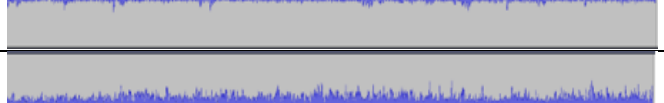
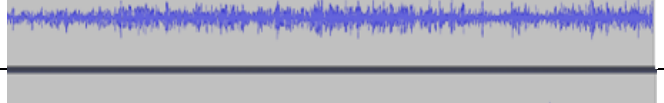
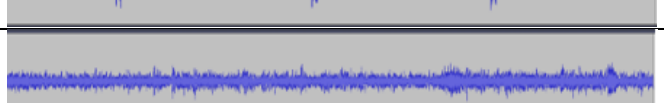
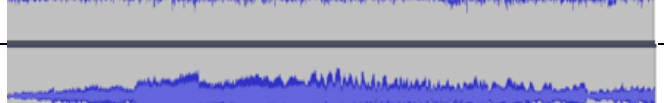
The miscellaneous dataset is comprised of randomly generated white, pink, brown noise and underwater noises. Underwater noises are noise that are commonly heard in the ocean environment. These noises include boat engines, ocean storms, dolphins, bubble curtain used by oil rigs, dredging, humpback whales, jet ski, sonar, lightning, rainfall, seismic air gun, underwater volcanic eruption and earthquakes, waves, wind turbine and snapping shrimp. These datasets were also pre-processed and augmented. A total of 1600 augmented underwater noises and 300 white, pink, and brown noises were used in training the network. A comparison of the various noises against the actual fish bombing audio can be seen in the table 1 below.

Reef Check Audio Clips Dataset

2 audio clips of actual fish bombing noise were given by the Reef Check Malaysia team. One audio clip was captured using a hydrophone and the other was captured using a GoPro video where the audio was extracted. The hydrophone recorded audio is around 1.7s whereas the GoPro video is around 1 minute. To increase the number of datasets for better network training, the hydrophone audio clip was augmented and used for network training.

The comparisons of some of the sample sounds can be seen in table 1.

Table 1: Comparison of sound signals

Hydrophone Fish Bomb	
LFA Sonar	
Lightning at sea	
Outboard Motor	
Rainfall at sea	
Seismic air gun	
Wind turbine	
Ship rotor noise	
Snapping shrimp	
Underwater volcanic eruption	
Underwater earthquake	
Sea waves	
Bottlenose dolphin	
Air bubble at oil rigs	
Dredging	
Humpback whale	
Jet ski.	

3.2 Pre-processing the datasets

The hydrophone fish bombing audio was chosen as the reference audio clip length. It is a 1.727 second Mono audio clip and has a sample rate of 24000 Hz. The other datasets are first converted to mono. Then, they are read into arrays and compared against the hydrophone fish bomb audio to see if their lengths are the same. They are either padded with zeros equally front and back if they are shorter than 1.727s or truncated if they are longer than 1.727s. If the length of the audio is the same as the reference, nothing is done. These arrays are scanned in a for loop to remove NaN values that will cause network training error. If NaN values are present, they are replaced with '0'. The processed clips are then saved as a new audio file. Each audio clips used must be labelled as Explosion or Noise because the network uses these labels to classify the audio signals. The flowchart for the pre-processing code can be seen below:

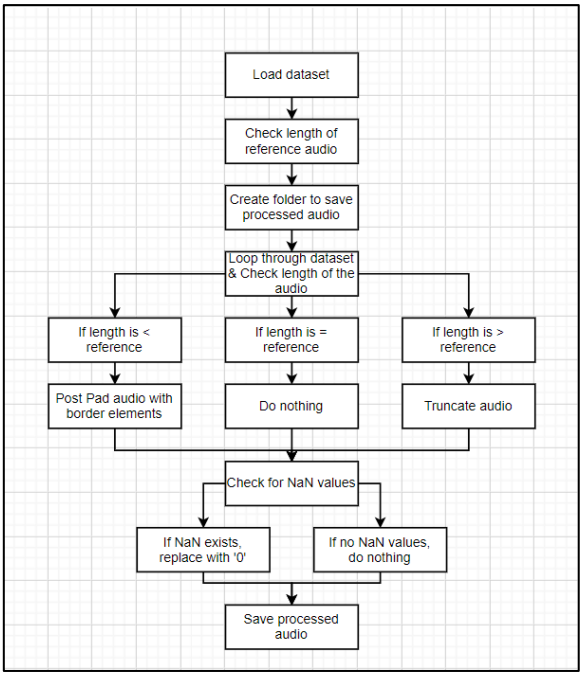


Figure 22: Flowchart of Pre-Processing Code

3.3 Data Augmentation of audio datasets

Data Augmentation is used to increase the number of audio files so that the deep learning network can be trained more accurately. The augmentation methods used are Random Independent Augmentation (RIA) and Random Sequential Augmentation (RSA). 'sequential' is an augmentation algorithm that is applied sequentially (in series) whereas 'independent' is applied independently (in parallel). 'random' is applied probabilistically using a probability parameter and a range parameter. 'specify' is applied using a logical parameter and a specified parameter value. The RIA augmented to produce 10,000 audio files while the RSA augmented 5000. The parameters that are augmented for RIA and RSA are:

- Augmentation Mode
- Augmentation Parameter Source
- Number of Augmentations
- Speedup Factor
- Time stretch
- Pitch shift
- Semitone shift
- SNR
- Volume control
- Apply time shift

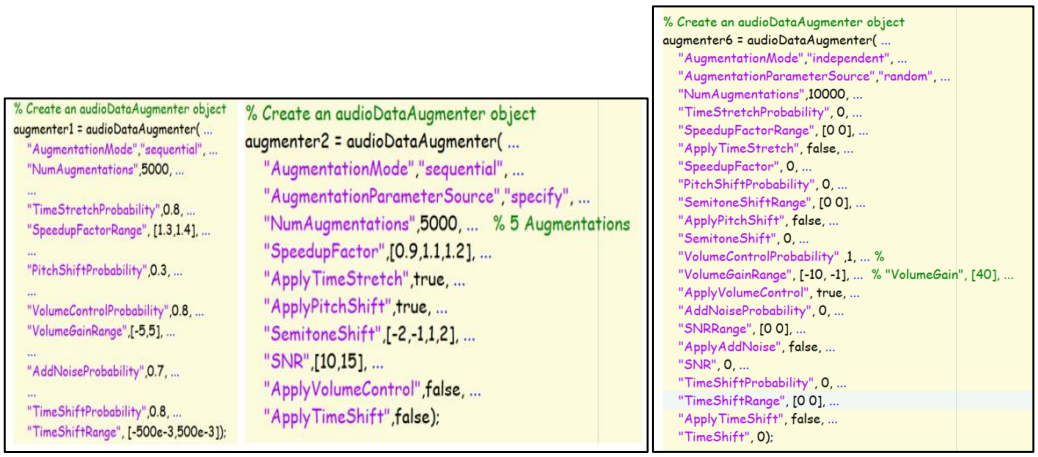


Figure 23: Parameters for Data Augmentation for (Sequential & Specify, Left) and (Independent & Random, Right)

The values chosen are arbitrary. However, for the sequential and specify augmenter object, the speedup factor and pitch parameters are not varied heavily. For the independent and random augmenter object, the volume gain was adjusted to produce lower volumes to simulate the bomb explosion far from the hydrophone. This is to preserve the unique characteristic of the fish bombing signal. An example of data augmentations can be seen below. Data Augmentation is applied to all training and validation datasets mentioned above. The flowchart for the Data Augmentation code can be seen below:

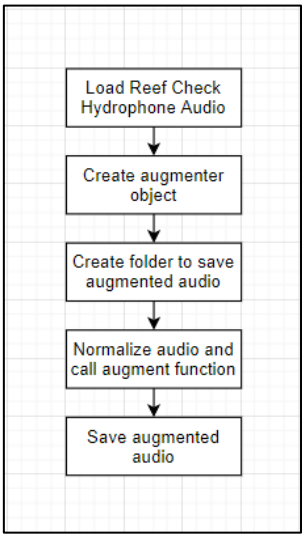


Figure 24: Flowchart for Data Augmentation code

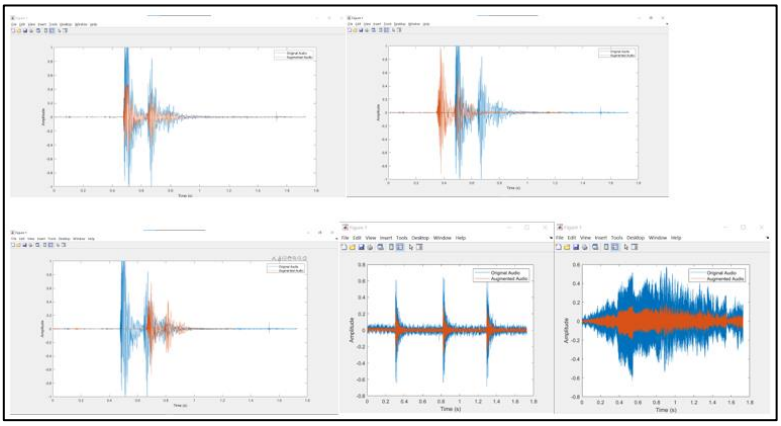


Figure 25: Augmented Hydrophone Fish bomb (Amplitude, time shifts) & Augmented Underwater Noise

3.4 Audio Feature Extraction

Audio features are extracted from an audio clip to be used as input to the deep learning network. The *audioFeatureExtractor* and *extract* in-built function in Matlab are used. First, the pre-processed training and validation audio clips are read into Matlab and stored into arrays. The training arrays are 41444 rows by X columns, where X depends on the number of training audio clips. There are 41444 rows because of the length of each pre-processed audio clip. From the image below, the aFE *audioFeatureExtractor* object is defined and is used with the *extract* function and training arrays. The extracted features are stored in the featuresTrain cell variable. The validation features are also extracted. In the featuresTrain or featuresValidation cells, each cell contains 79x30 double which represents the audio features of an audio file. This 79x30 is used as the input into the deep learning network.

The audio features to extract are: SampleRate, SpectralDescriptorInput: Mel Spectrum, mfcc, spectralCentroid, spectralEntropy, mfccDelta, pitch and harmonic ratio. As mentioned above, the sample rate of the hydrophone fish bomb audio from Reef Check Malaysia is sampled at 24000Hz thus the sample rate is set to this value. The SpectralDescriptorInput: Mel Spectrum and mfcc (Mel-Frequency cepstral coefficients) settings are used because they have been used as the dominant acoustic feature representations for audio analysis tasks such as speech recognition [22]. The spectralCentroid, spectralEntropy, mfccDelta, pitch and harmonic ratio settings are also commonly used audio features for speech recognition and are thus selected for this application. There are many other audio features that can be extracted but these few are selected to ensure that the when the network tests real data it does not have to extract too many features which can be time consuming in real-world application.

Figure 26: Audio Feature Extraction Settings

3.5 MATLAB LSTM Network

4 trials with varying network parameters and dataset were conducted. Each network has a 5-7 layers depending on the trial. The layers and network options used are based on speech recognition applications.

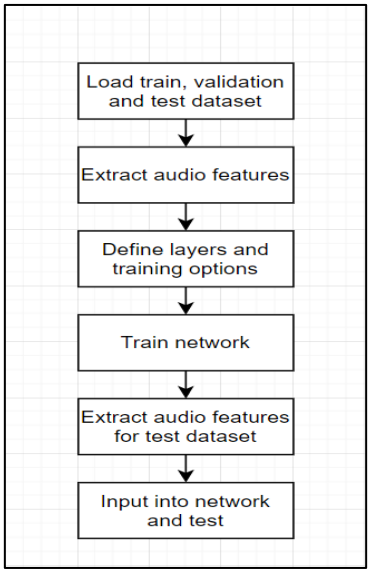


Figure 27: Flowchart for deep learning code

Layers tested:

- sequenceInputLayer
- lstmLayer
- bilstmLayer
- fullyConnectedLayer
- reluLayer
- softmaxLayer
- classificationLayer.

Options:

- miniBatchSize = 27,
- trainingOptions("adam"),
- 'ExecutionEnvironment','cpu',
- "Shuffle","every-epoch",
- 'MiniBatchSize',miniBatchSize,
- 'GradientThreshold',1,
- "ValidationData", {featuresValidation,LabelsVal},

- "Plots","training-progress",
- "Verbose",false;

For the options used, the Adam optimizer is used because it is computationally efficient, requires little memory requirement and many other benefits compared to other solvers like SGDM or RMSPROP [23]. A batch size of 27 is chosen because a range of 1-100 is recommended with 32 being a good default value [24]. Since mini-batch sizes are tuned to the computational architecture on which the implementation is being executed, which in this case is a Raspberry Pi, a value lower than 32 is chosen.

To train the network, the trainNetwork function was used. The output from this function is a 1x1 Series Network consisting of 5x1 Layers. The output from the neural network is a categorical array that contains either an “Explosion” or “Noise” for each audio clip tested. The figure below shows the layers of the network.

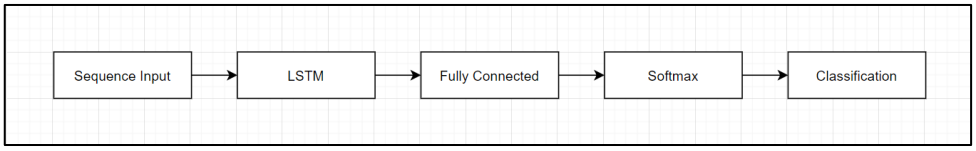


Figure 28: Flowchart for network layers

Table 2: Network layers summary

sequenceInput	biLSTM	FullyConnected	Softmax	ClassificationOutput
InputSize = 79	InputSize = 79	InputSize = 100	No. of Inputs = 1	No. of input = 1
		No. of Outputs = 1	No. of Outputs = 1	Class output: ‘Explosion’ and ‘Noise’
	No. of Hidden Units = 50 x 2 = 100			
	State Activation Function = tanh			
	Gate Activation Function = sigmoid			

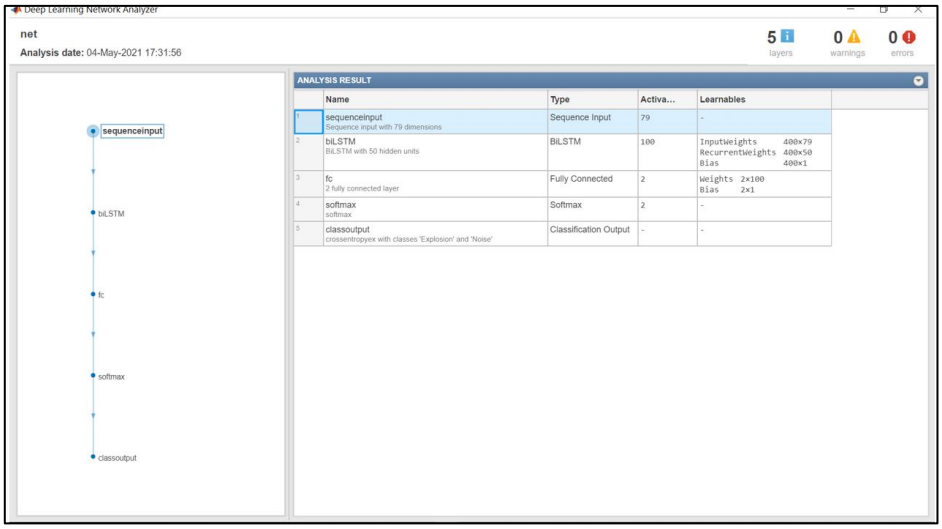


Figure 29: Deep Learning Network Analyser

From table 2, the input size of 79 refers to the length of the audio input required for the network. The number of hidden units corresponds to the amount of information remembered between time steps (the hidden state). The hidden state can contain information from all previous time steps, regardless of the sequence length. If the number of hidden units is too large, then the layer might overfit to the training data. This value can vary from a few dozen to a few thousand. The value of 50 chosen is referring to other literature examples of speech recognition networks that use this value as well. The StateActivationFunction is to update the cell and hidden state. A tanh function is used because with LSTMs, it distributes gradients very well. Hence, it prevents cell state vanishing or exploding which is a problem for RNNs when learning dependencies over long time windows. The GatewayActivationFunction is sigmoid because it constitutes smooth curves of a range 0-1 and the model remains differentiable. These options are what are commonly used in Speech Recognition Networks.

For the layers, the sequence input layer inputs sequence data to a network. The bidirectional LSTM (BiLSTM) layer learns bidirectional long-term dependencies between time steps of time series or sequence data. These dependencies can be useful when you want the network to learn from the

complete time series at each time step. BiLSTMs allow the network to have both backward and forward information about the sequence at every time step. It will run inputs in two ways, one from past to future and one from future to past. BiLSTM is chosen over a LSTM because studies have shown that BiLSTM can provide better predictions compared to LSTM models [25]. The fully connected layer multiplies the input by a weight matrix and then adds a bias vector. All neurons in a fully connected layer connect to all the neurons in the previous layer. This layer combines all of the features learned by the previous layers across the image to identify the larger patterns. The softmax layer applies a softmax function to the input. This layer and the classification layer must follow the final fully connected layer. It assigns decimal probabilities to each class in a multi-class problem. Those decimal probabilities must add up to 1.0. The classification layer then computes the cross-entropy loss for classification and weighted classification tasks with mutually exclusive classes. It outputs classes which are 'Explosion' and 'Noise' depending on the probability from the Softmax Layer.

A summary of the trials and its parameters can be seen in appendix D. The results from these trials are found in the Network Training and Testing Results of the Results & Discussions section below.

3.6 Running Window

After the successful training and testing of the neural network. A running window algorithm was developed to process audio clips longer than the training audio duration of 1.727s. To test the algorithm, the reef check GoPro audio was used. First, the audio is read and stored as an array. The length of the array is recorded and a for loop is used to iterate through the entire audio while extracting a window length of audio data features using the same audioFeatureExtractor function mentioned above. This window length is the same length as a Reef Check Hydrophone audio clip that was used to train the network. For each iteration, the window is incremented a value ¼ of the length. This is so that there is sufficient overlap between each window. This ensures

that an explosion will not be missed. Figure 30 visualizes how the windows overlap. The yellow region represents the overlapped portion. Various window lengths were tested to find the optimal window overlap increment. The running window algorithm was also tested with other noise datasets. The results are recorded in the results section.

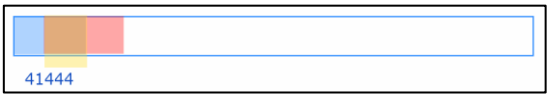


Figure 30: Visualization of Running Window Overlap

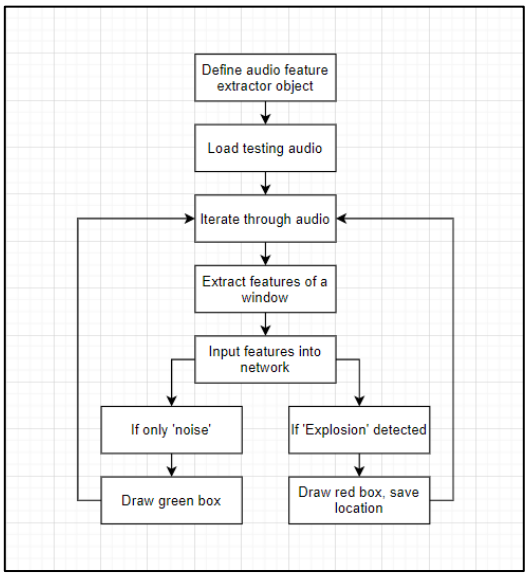


Figure 31: Flowchart for Running Window code

3.7 TensorFlow CNN Network

A simple CNN network architecture [26] built on the Jupyter notebook and Tensorflow framework was also tested. Unlike the previous MATLAB LSTM network, the CNN network is built on Python and its libraries. This network was trained using the same dataset as the LSTM network above. However, there are a few differences which are discussed below.

To pre-process the dataset, the noise and explosion datasets must be separated into 2 different folders where they are read and stored into an array. Some settings are then defined which is seen in Figure 33. The dataset is split into 60% train, 30% validation and 10% testing data. A sampling rate of 8kHz

is used compared to the original 24kHz so that the model can run faster on the Raspberry Pi. The datasets are also appended with ‘0’ or ‘1’. ‘0’ is for noise and ‘1’ for explosion. The datasets are then randomly shuffled. The number of mfcc is 16 but this is arbitrary and can be varied to increase the performance of the model. The length of mfcc is due to the size of each dataset and this shows 28 windows of time.

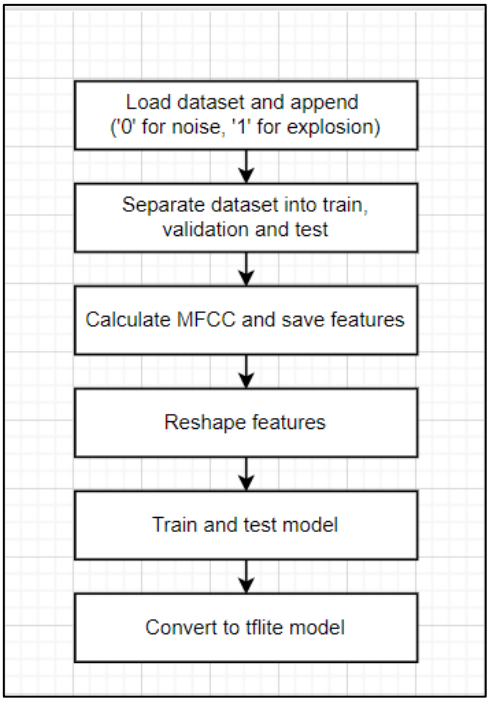


Figure 32: Flowchart for CNN code

```
# Settings
target_list = all_targets
feature_sets_file = 'all_targets_mfcc_sets.npz'
perc_keep_samples = 1.0 # 1.0 is keep all samples
val_ratio = 0.3 #0.1
test_ratio = 0.1 #0.1
sample_rate = 8000 # orignal =24kHz, use 8kHz for faster sampling on rpi
num_mfcc = 16
len_mfcc = 28 # (31 used) (most of my signals are 28)
```

Figure 33: Settings for network

Unlike LSTMs which take in sequence recognition, CNNs are mostly used for image recognition. The network takes in the spectrogram of a give sound data as input. The MFCCs are extracted for every set window of time as seen in Figure 34 of the red boxes from left to right. These mfccs are stored as an

array and make up one spectrogram. This is represented in Figure 34 which shows 16 rows of MFCC coefficients and 28 columns of window time. A comparison of spectrogram for noise and explosion datasets can be seen in Figure 35 The colours of the spectrogram give a relative representation of the value of each coefficient. The bottom row, or zeroth coefficient is dark because they are all large negative values compared to the rest.

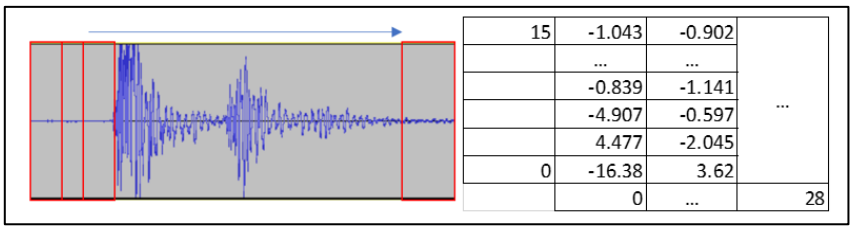


Figure 34: Extracting MFCC features

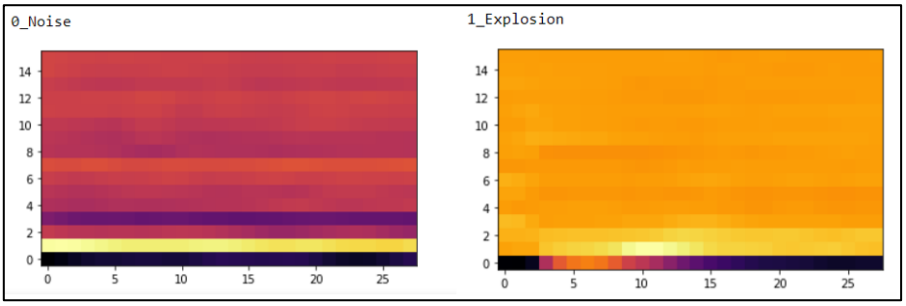


Figure 35: Comparison of Noise vs Explosion spectrogram

Tensorflow expects tensors in 4-D as input to Convolution in the form of Sample, Height, Width, Channel. However, the mfcc spectrogram has only one channel in the form of MFCC, time, samples. Thus, the train, validation and test datasets are reshaped to have an extra channel. In Figure 36 (left), it is the train dataset of 10620 files with 16x28 of the MFCC. Then, after reshaping, it has an extra ‘1’ channel. This is repeated for the 5310 validation and 1770 train datasets.

```
                        (10620, 16, 28, 1)
                        (5310, 16, 28, 1)
(10620, 16, 28)         (1770, 16, 28, 1)
```

Figure 36: Reshaping dataset to 4-D

The CNN network layers can be seen below in Figure 38 and 39. The spectrogram image is first passed into the convolutional layers, then classifier where it makes a prediction. The convolution layer consists of moving a window across the whole image as a filter to extract features. This sliding window is called a kernel and performs mathematical operations on 2x2 pixels. There are 32 nodes in the first convolutional layer. This results in 32 different filtered images containing fine details of the spectrogram image. The Relu activation simply sets all negative numbers in the feature maps to zero. This adds a layer of nonlinearity to the nodes for model training. The Maxpooling layer reduces the size of the feature maps by down sampling them. This reduces computation for the next layer. The second and third layer are repeated using the same steps. The final convolution layer has 64 nodes of 2-D feature maps. The classifier network expects a 1-D tensor or vector input so the flatten function is used to flatten the 2-D feature maps into one long string of numbers which is fed into the dense with 64 nodes. After passing it through another Relu activation, it is input to a dropout layer which reduces overfitting of the model. Then, it passes into the dense layer of 1 node passed through a sigmoid activation function which is commonly used in binary classification. The output is a prediction of whether the input MFCC image corresponds to noise or explosion. The model is compiled using 30 epochs and batch size =100.

```
history = model.fit(x_train,
                    y_train,
                    epochs=30, # can be changed to affect performance
                    batch_size=100, # " "
                    validation_data=(x_val, y_val))
```

Figure 37: Model fit settings

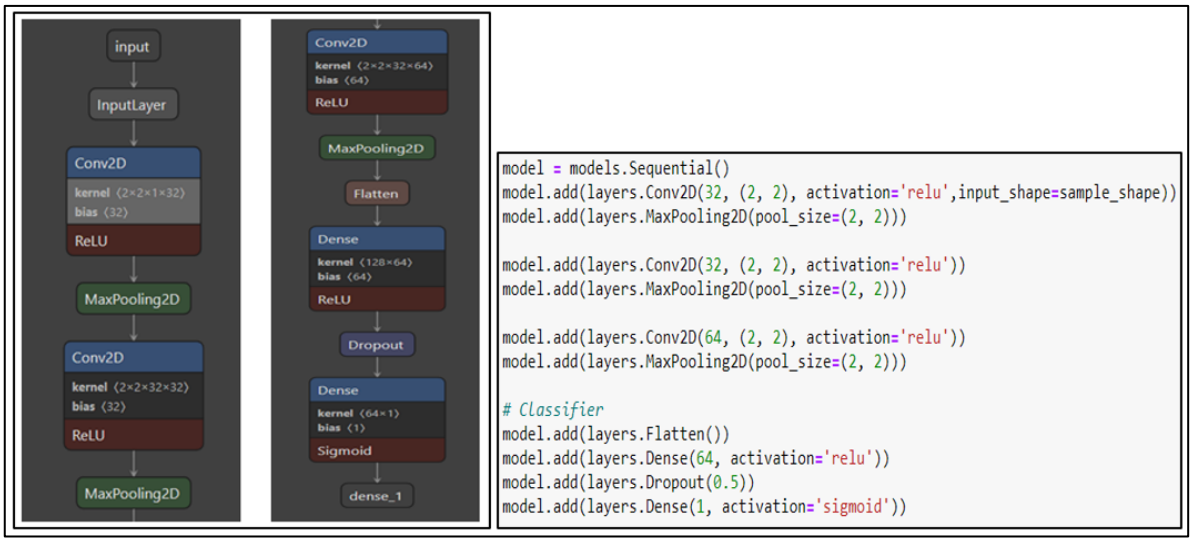


Figure 38: CNN layers

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape
=================================================================
conv2d (Conv2D)              (None, 15, 27, 32)
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 7, 13, 32)
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 6, 12, 32)
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 3, 6, 32)
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 2, 5, 64)
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 1, 2, 64)
_________________________________________________________________
flatten (Flatten)            (None, 128)
_________________________________________________________________
dense (Dense)                (None, 64)
_________________________________________________________________
dropout (Dropout)            (None, 64)
_________________________________________________________________
dense_1 (Dense)              (None, 1)
=================================================================
```

Figure 39: Summary of layers

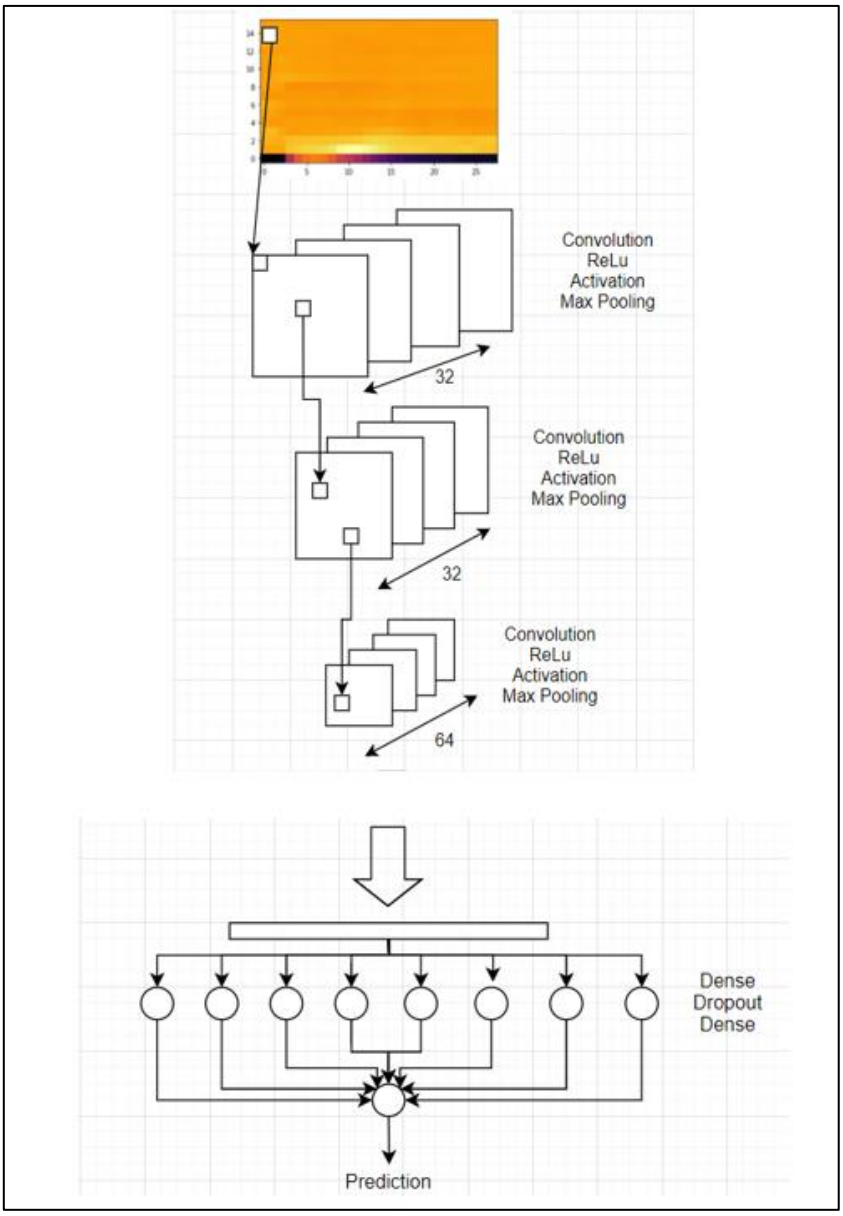


Figure 40: Flowchart of CNN network process

4.0 Raspberry Pi Implementation

4.1 Application of the Deep Learning Network on the Raspberry Pi 4

An experiment was conducted to test the networks on the Raspberry Pi. Figure 41 shows the experiment setup which consists of a Raspberry Pi 4, LED, breadboard, and USB microphone. The speaker used is a Logitech Z333 Speaker System with Subwoofer. The results are discussed in the results section below.

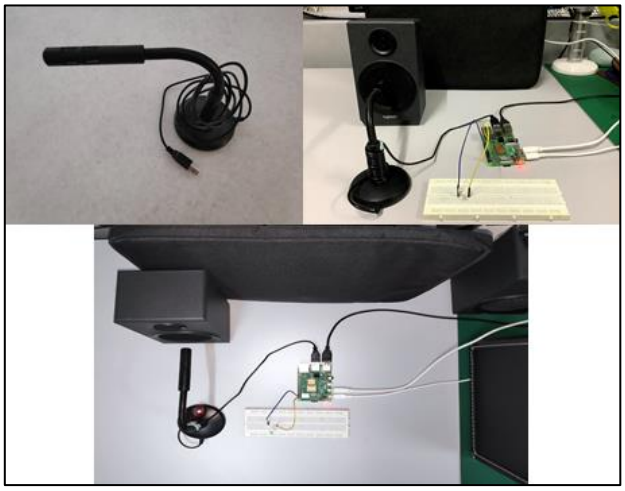


Figure 41: Raspberry Pi Implementation Experiment Setup

Initially, the trained Matlab LSTM network was converted to ONNX file format to be converted to tflite format before transferring to the Raspberry Pi. However, it was found that conversion from ONNX to tflite for LSTM networks is not currently supported as of now. Thus, the CNN network architecture discussed above was used to deploy on the Raspberry Pi. This CNN Tensorflow model is converted to a TensorFlow Lite model before it is uploaded to the board. The audio files that were used for testing are the same ones used to test the Matlab LSTM network. Figures 42 and 43 show the process flow of the system and a flowchart of the algorithm. The sound is picked up by the USB microphone where the MFCC features are extracted before inputting into the TFLite model for classification.

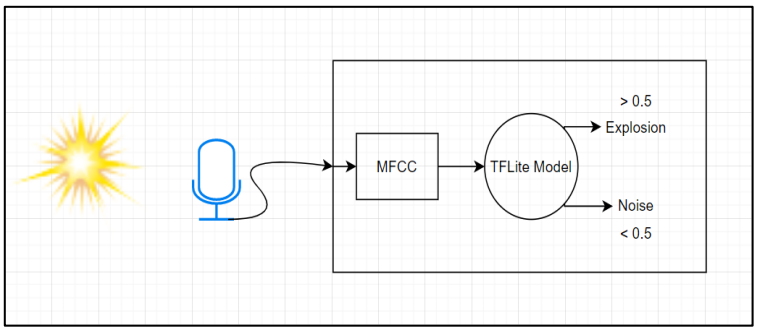


Figure 42: Overview of Raspberry Pi System

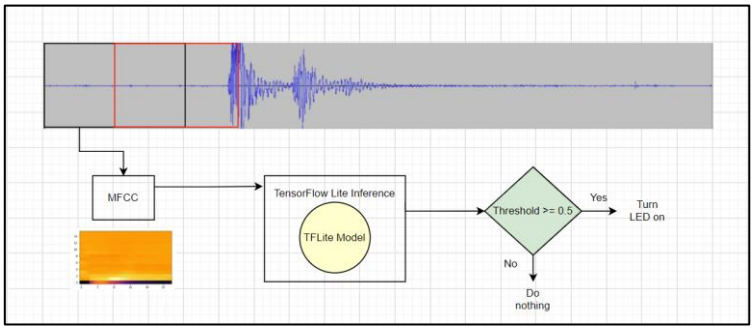


Figure 43: Extracting MFCCs

From Figure 43, the black box represents a buffer of 1.6s of audio data from the microphone and is first downsampled from 24kHz to 8kHz before its MFCCs are extracted. Then, the features are fed into the inference which contains the model. The process of down-sampling and filtering of the input audio is called decimation. The scipy library and scipy.signal.decimate function is used to decimate the audio. Decimation is required because the USB microphone cannot sample at 8kHz. The USB microphone was tested to obtain the ranges of frequency it can handle. From a sample python code found online, the USB microphone could handle 24000, 32000, 44100, 48000, 96000, 128000 Hz frequencies which is suitable for this application.

The python_speech_features library is used to extract the MFCC features. The output from the inference is a probability that buffer of audio contains the

explosion signal. Since the black section does not contain the explosion, it will have a low probability and will not meet the threshold. The window is then slide over to the red box by 0.8s and the process is repeated. To achieve this in real-time, the second half of the buffer is shifted to the first half and the newly captured 0.8 seconds audio is stored into the second half.

4.2 Proposed Solution Implementation on Mantanani Island

Figure 44: Location of current fish bomb detectors

Figure 45: Location of proposed fish bomb detectors

Figure 44 shows where the current Reef Check Malaysia passive fish bomb detectors are placed. These detectors are able to detect up to 10km. They are

placed 200-500m from the shore. Figure 45 shows the proposed locations of the deep learning fish bomb detectors. It is assumed that these fish bomb detectors can detect up to 5 km. They are placed 350m from the shore and are spaced 2 km apart. By using only 6 detectors, the entire island can be covered, and fish bombing locations can be determined more accurately compared to the current fish bomb detectors. Although spaced 2 km apart, research has shown that in standard atmospheric conditions, for an observer with eye level above sea level by 1.70 metres, the horizon is at a distance of 5 kilometres [27]. With the use of binoculars, it will be hard for the authorities to miss locating the fish bombing boats. Since this system is cheaper compared to the current detectors, it will be easier to scale up and maintain/replace the detectors.

Each microcontroller node will communicate with each other and a gateway using the LoRa Protocol. Real-time alerts containing the timestamp and location of a blast event will be sent to the phones of the authorities. Each node is equipped with a GPS module which will send the GPS coordinates of the node once a fish bomb is detected. These blast events along will be recorded on a server and displayed in real-time. The microcontroller will also be recording explosion audio on its SD card. The recorded occurrences of fish bombing activity can be used to monitor the surrounding environment for fish bombing activity and can also be used for environmental survey reports.

- Hardware Components:

  Hydrophone, Raspberry Pi microcontroller, LoRa, GPS module, Solar Panel, Power regulator circuit, Li-Po Battery, Gateway, USB microphone, LED, breadboard

-Process Diagram:

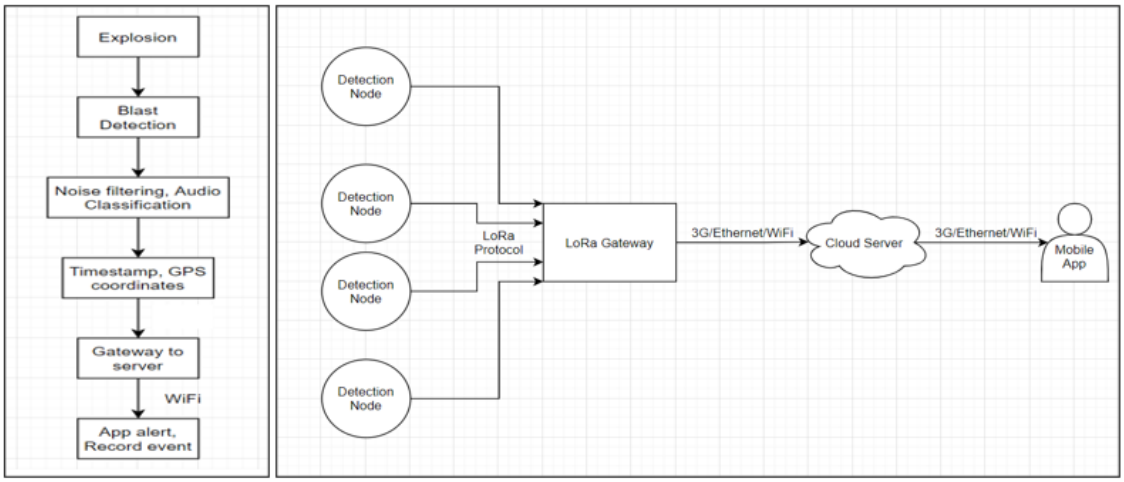


Figure 46: Flowchart of process from explosion to user notification. Internet of underwater things flowchart

5.0 Results and Discussion

5.1 Analysis of actual fish bombing audio

(AudioFishBomb.mp3)

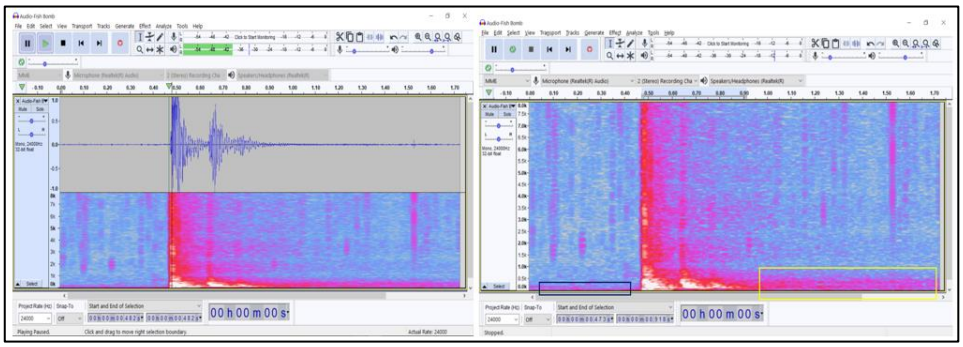


Figure 47: Signal in time domain and Spectrogram

This audio is taken by the researchers who used the ShotSpotter method. From the spectrogram, the characteristics of the fish bombing audio are it has a very sharp impulse followed by a smaller but noticeable impulse. The second impulse could be caused by the reflection of the sound wave against the surrounding reefs. Red colour represents higher amplitude and blue/grey colour represents silence. The black box represents background noise, and the yellow box is the noise after the explosion has taken place.

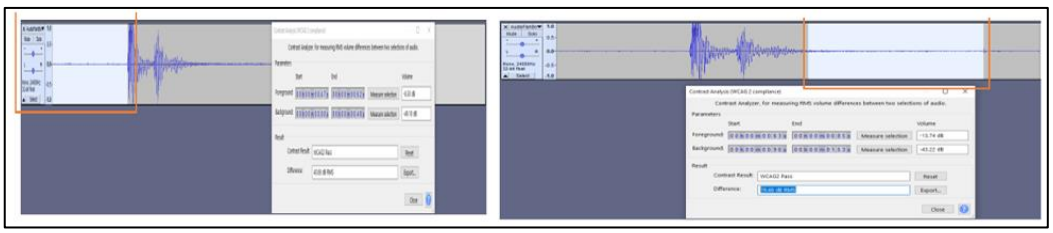


Figure 48: RMS values of the first explosion between background noise and explosion signal

RMS of first blast wave (average value of squared of amplitude of wave signal) (compared first wave with background) = **40.68 dB RMS**. The RMS of second blast wave (average value of squared of amplitude of wave signal) (compared second wave with background) = **29.48 dB** RMS. The fish bombing audio has a lot of low frequency and starts to steadily roll-off at -20dB. This is expected because the audio sounds like it has a lot of low frequencies and little high frequencies.

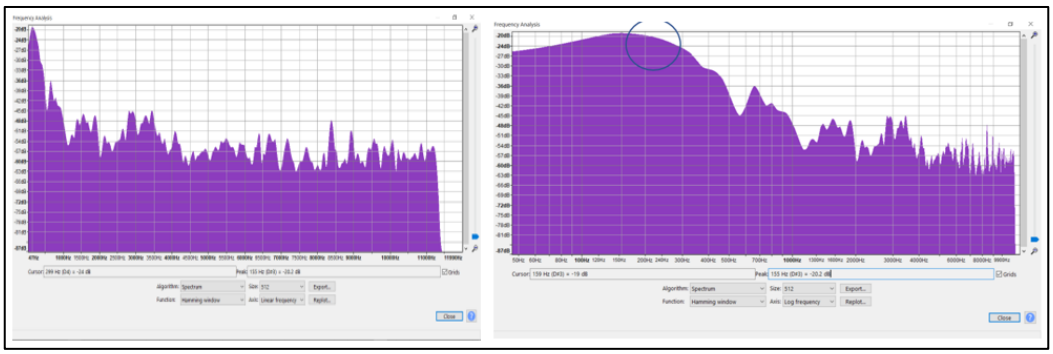


Figure 49: Linear & Log frequency representation of signal

(VID 20200825 WA0000)

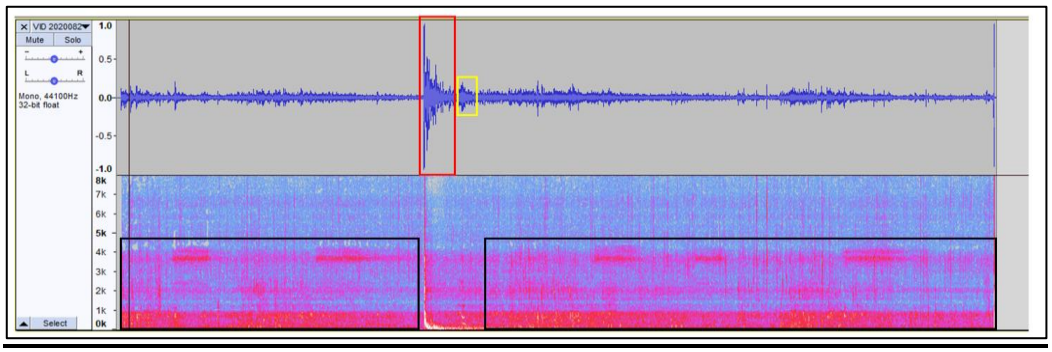


Figure 50: Audio from action camera

This audio is extracted from a video taken from a scuba diver using a GoPro action camera. It is similar to the previous audio where two impulses can be seen. The red box is the fish bombing explosion sound impulses whereas the yellow box is the voice of the diver reacting to the sound. From the spectrogram, it can be seen that the audio is very noisy compared to the spectrogram of the first audio file. The noise occurs at approximately 4kHz and is represented by the black boxes. RMS noise before explosion = **17.06 dB**

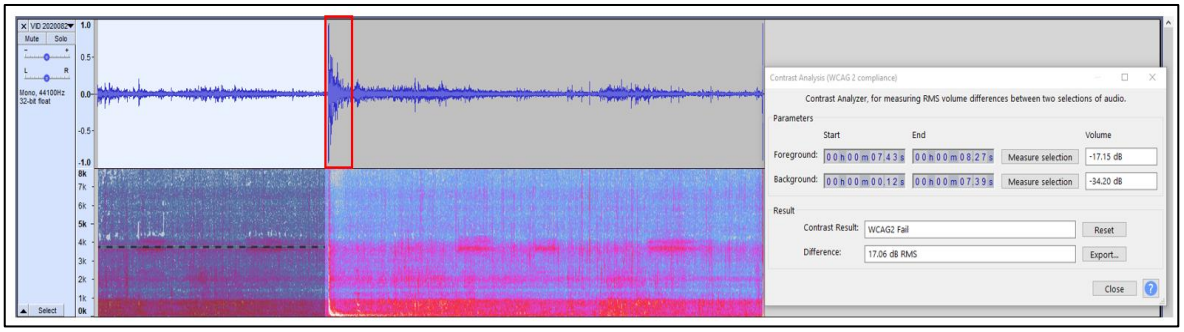


Figure 51: RMS value before explosion

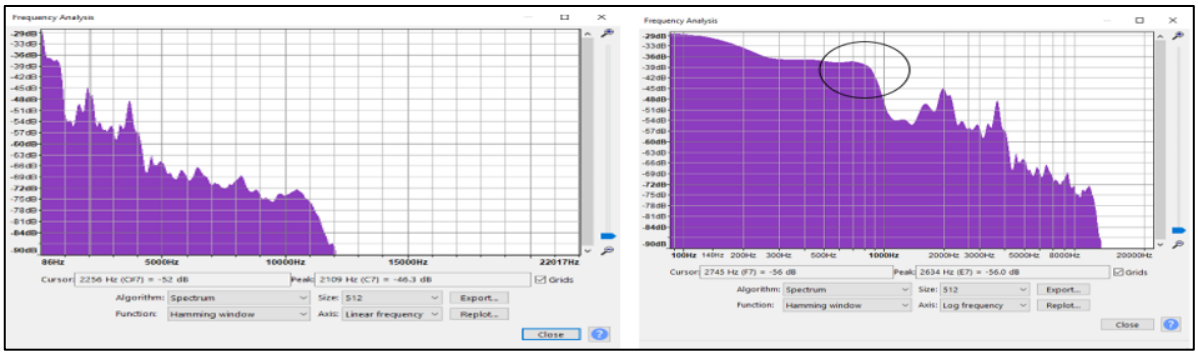
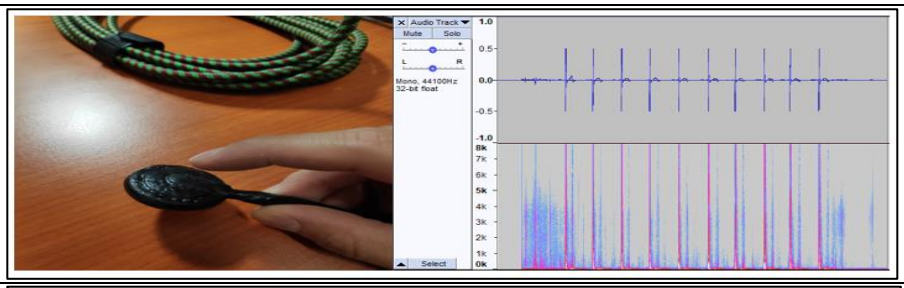
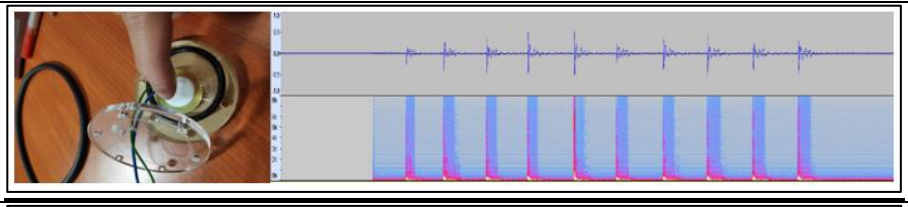
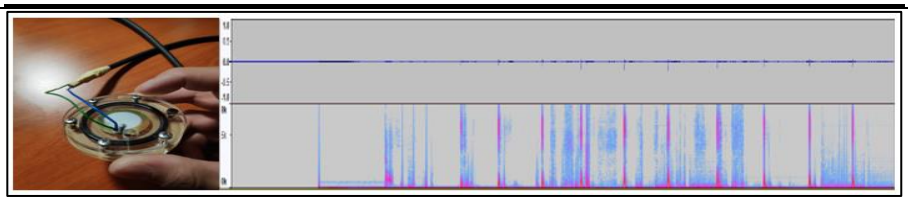
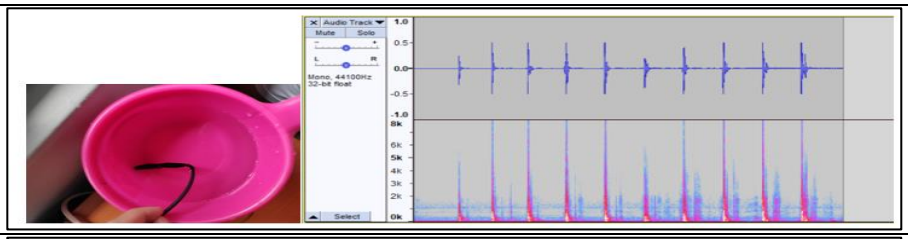
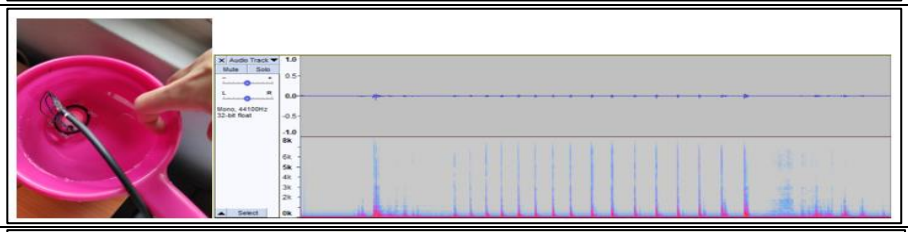
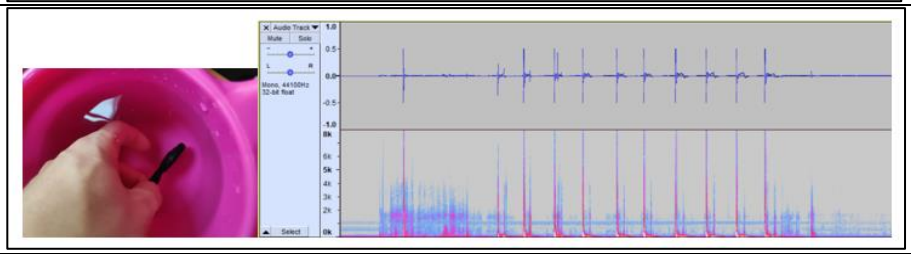
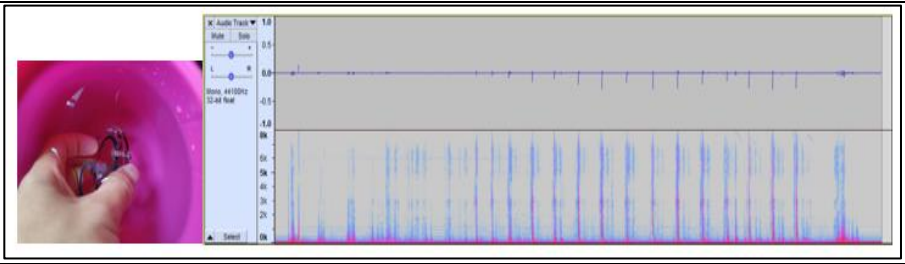
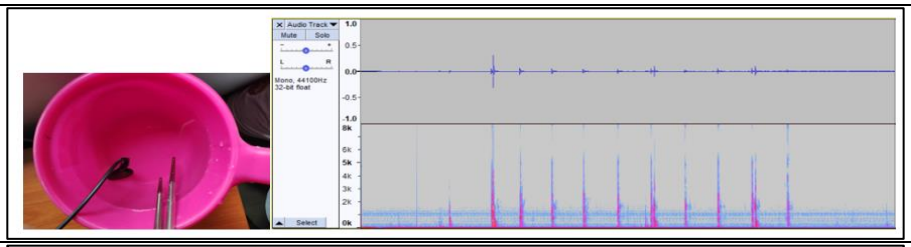
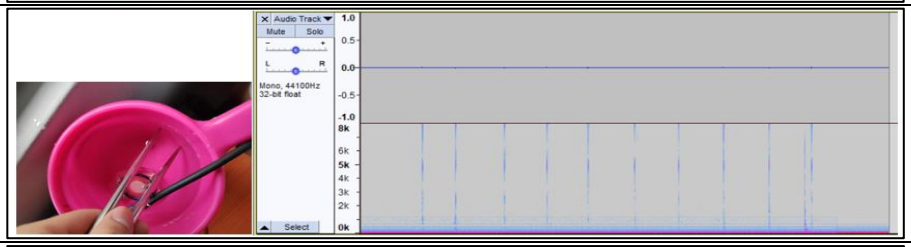
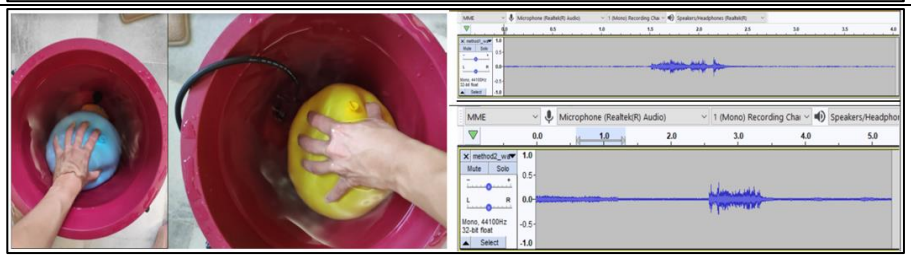
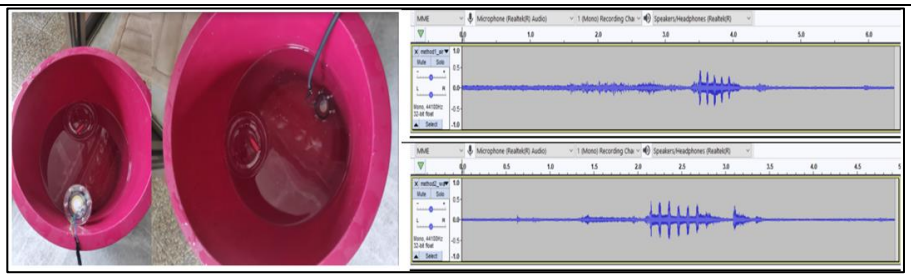


Figure 52: Linear & Log frequency representation

5.2 Testing of the hydrophone prototypes

Table 3: Comparing Method 1 and Method 2 hydrophones

Method 1 Tapping on hydrophone	
Method 2 (without top cover)	
Method 2 (with top cover)	
Method 1 Tapping side of bucket	
Method 2 Tapping side of bucket	
Method 1 Tapping on hydrophone in water	
Method 2 Tapping on hydrophone in water	

Method 1 Metal clank in water	
Method 2 Metal clank in water	
Method 1 (left, top) Method 2 (right, bottom) Popping balloon next to hydrophone	
Method 1 (left, top) Method 2 (right, bottom) Lighting a small firework on the surface of the water	

Method 1 is more responsive than the method 2, it could be because method 1 has a thinner case covering the piezoelectric element. However, Method 2 is more robust and suitable for ocean deployment due to it being made of coaxial cable and bolted acrylic sheets compared to the method 1 hydrophone which uses a music cable which is only water resistant and not submersible in water. Method 1 is less robust because of its Plasti Dip. The harsh ocean environments may corrode the rubber coating and destroy the electrical tape. Once the rubber coating and tape are gone, seawater will seep into the connection between the 6.3mm jack and the hydrophone. Moving forward, the method 2 hydrophone is more suitable for ocean application due to it being made of coaxial cable and bolted acrylic sheets which make it more robust

compared to the method 1 hydrophone which uses a music cable which is only water resistant. However, more testing is required with various acrylic sheet thickness to determine if there is a significant effect in the response of the hydrophone with varying thickness.

5.3 Inflatable pool testing

Equipment used: Electric air pump, hose, balloons, measuring tape, inflatable pool, weights, hydrophone, audio interface, laptop to run Audacity software, phone for video recording.

Figure 53: (From left to right) Electric air pump and hose, balloon clamped on to hose with paper clamp, inflated balloon using breath

Figure 54: (From left to right) Pool setup with weights to secure hydrophone and balloon, audio interface connected to laptop and hydrophone, weights securing measuring tape

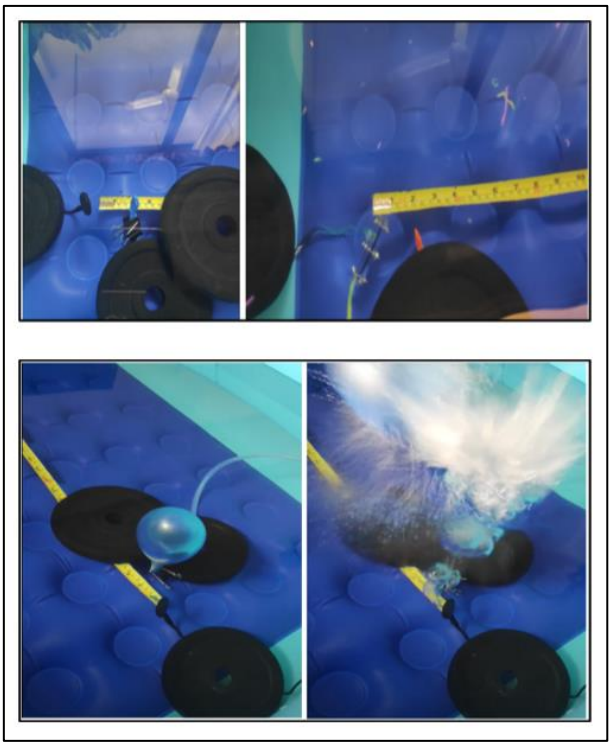
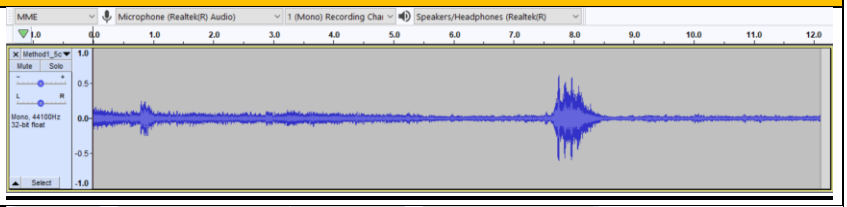
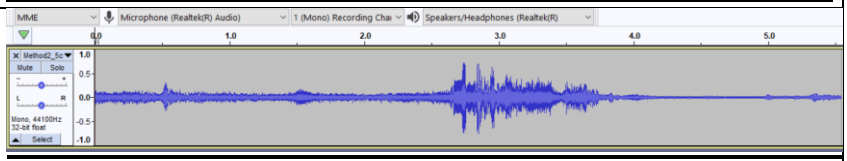
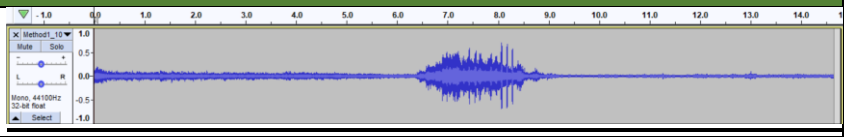
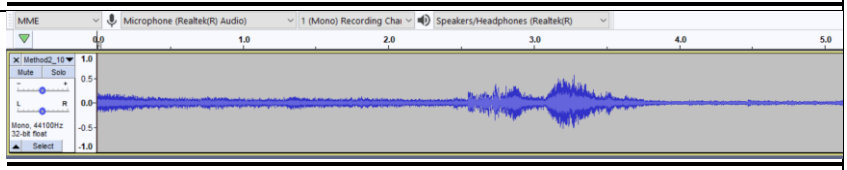
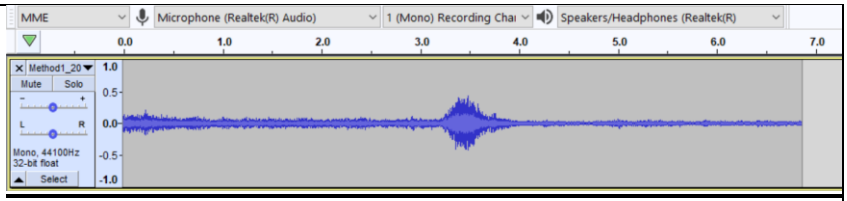
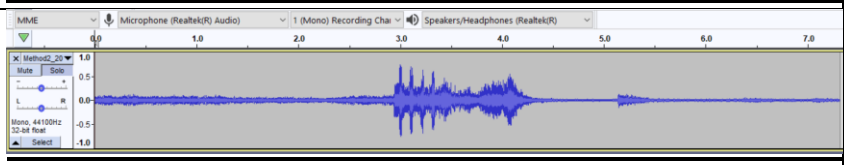
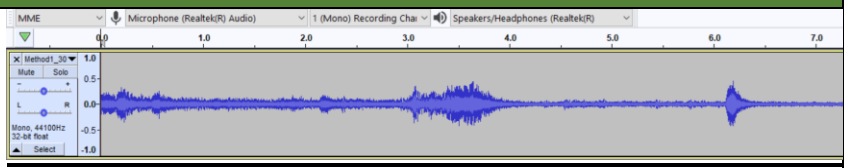
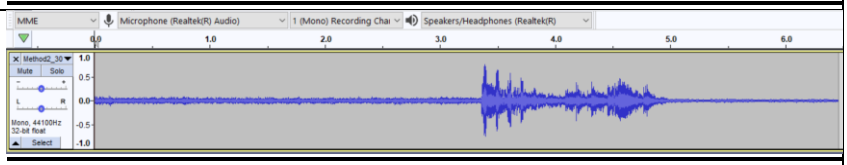
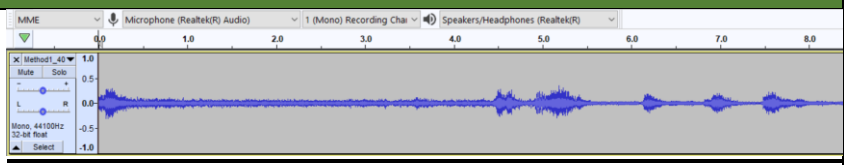
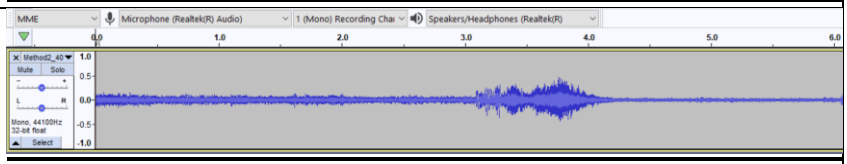
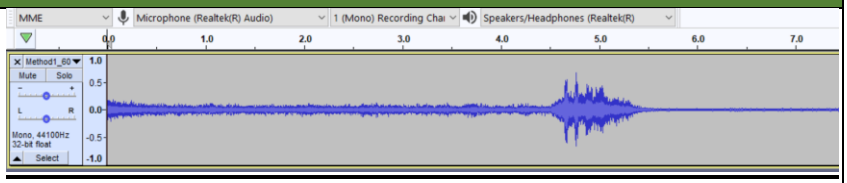
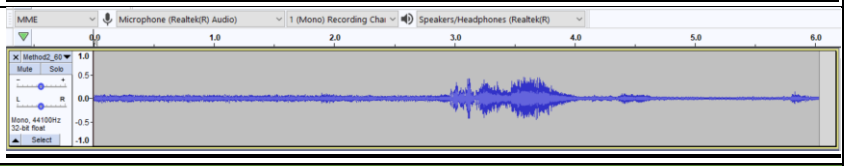
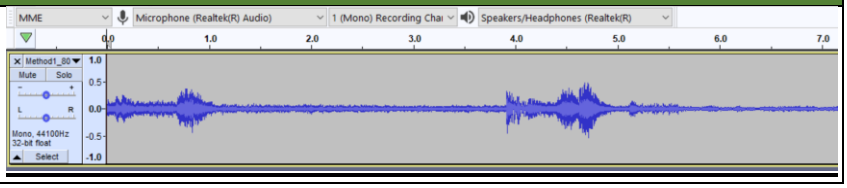
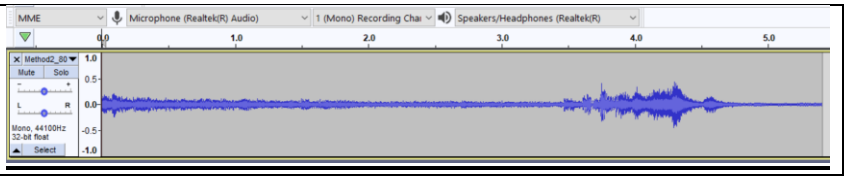
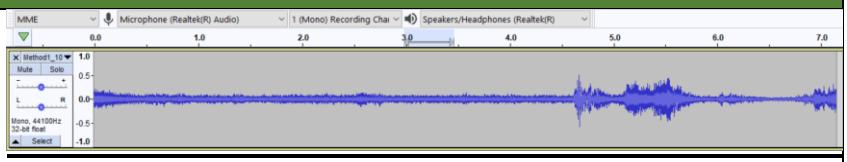
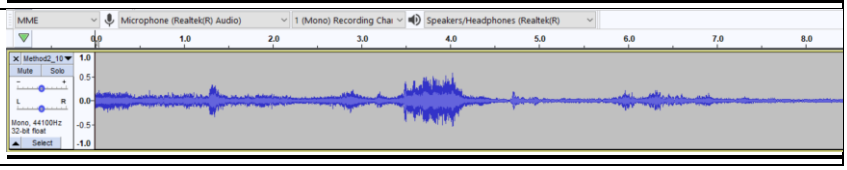


Figure 55: Experiment conducted at various distances for method 1 and 2 hydrophone

Table 4: Pool results for Method 1 & 2 hydrophones

Distance (cm)	Method	Results (Method 1, Method 2)
5	1	
	2	
10	1	
	2	

20	**1**	
	2	
30	**1**	
	2	
40	**1**	
	2	
60	**1**	
	2	
80	**1**	

	2	
100	**1**	
	2	

Comparison between Method 1 and 2 hydrophones

Based on visual inspection, the results between Method 1 and 2 are quite similar. It is noted that the duration of the measured signals from 5m to 100cm are all around 1 second except for the 100cm distance at slightly greater than 0.5 seconds. It is also seen that the signal amplitude decreases from ~0.75 to ~0.5 slowly as the distance is increased from 5cm to 100cm.

Comparison between Reef Check Hydrophone, GoPro recording sound and pool sound.

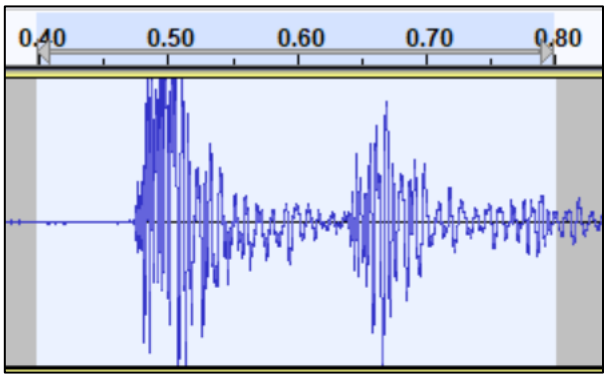


Figure 56: Reef Check Hydrophone Fish Bombing Sound

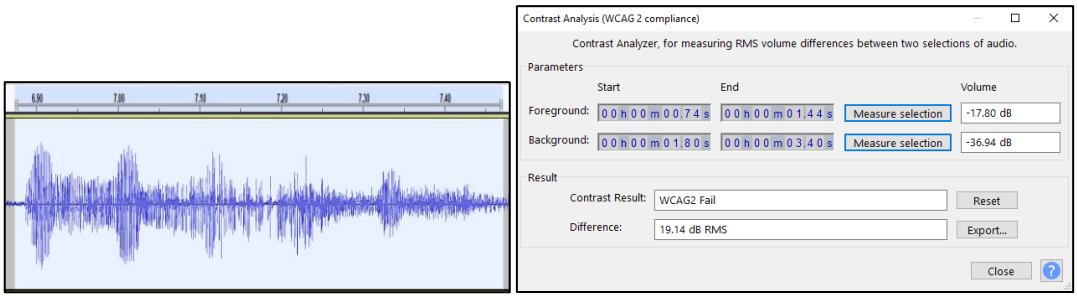


Figure 57: Pool recording for Method 1 @ 15cm, RMS @ 5cm

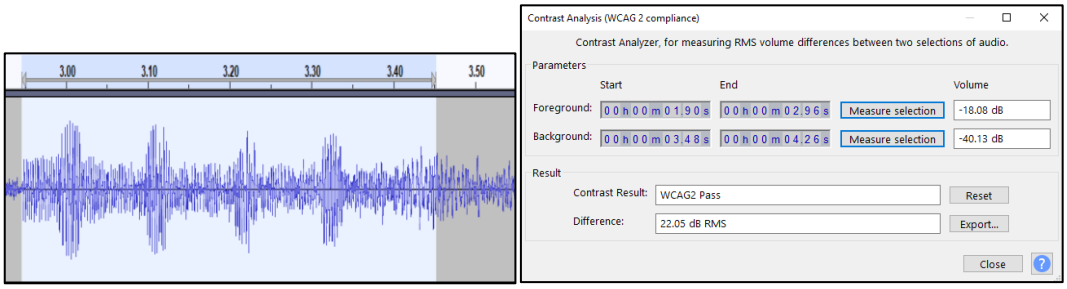


Figure 58: Pool recording for Method 2 @ 15cm, RMS @ 5cm

From Figure 56, the actual fish bombing audio signal has two impulse peaks. Compared to the results from method 1 and 2 in Figure 57 and 58, these results show multiple decreasing peaks. The multiple peaks could be caused by the reflection of the sound waves in the inflatable pool. As the actual fish bombing audio was captured in the ocean, the chance of sound reflection across the reefs would be less resulting in only 2 peaks. In the actual signal, the duration of the signal is around 0.325 seconds. This is similar to the recreated sound signals which also lasts around 0.3-0.4 seconds. From the actual fish bombing frequency response, the signal ranges from 50Hz to 8kHz whereas the simulated signal ranges from 100 to 16kKz. However, for all 3 signals, most of the sound frequency is from the 50 to 400Hz. The difference in the lower frequency response is probably due to the type of explosion used which is an actual fish bomb explosion versus a balloon explosion.

At a distance of 5cm, Method 1 had a RMS of 19 and Method 2 had 22.05 RMS. From Fig. 59, it can be seen that Method 1 has a maximum of 24dB compared to Method 2 which has 15dB.

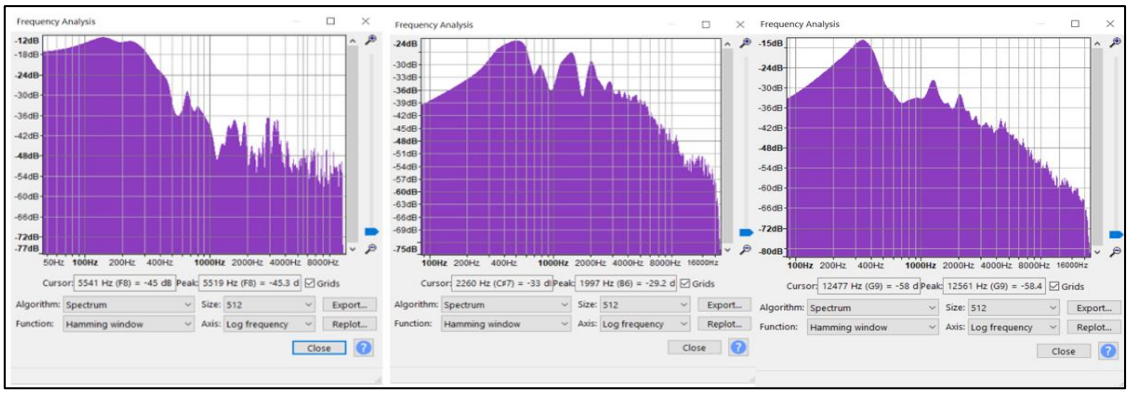


Figure 59: Frequency representation of the 3 signals above (Actual, Method1, Method2)

5.4 Network Training and Testing Results

Matlab LSTM & Running Window

From the trials, the results are summarized in table 5 below.

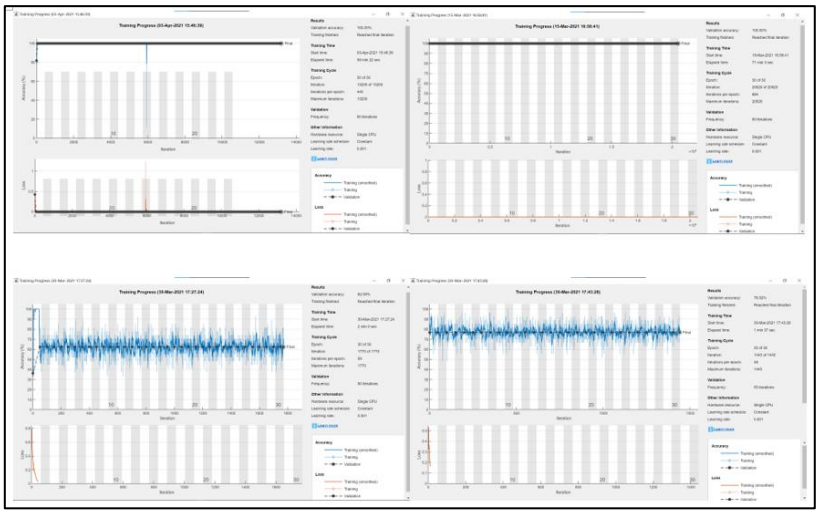


Figure 60: LSTM Network training

Table 5: LSTM Network testing results

Trial	Train Dataset	Validation Dataset	Neural Network Settings	Testing dataset	Results
1	18469 of gunshot and augmented Reef Check	5040 augmented Reef Check	Layers = sequenceInputLayer lstmLayer(50,"OutputMode","last") fullyConnectedLayer softmaxLayer classificationLayer	Reef Check Hydrophone Audio Clip	Some Noise categorized as explosions.
2	same	Same	- sequenceInputLayer -> featureInputLayer (error) - add bilstmlayer (error) - lstmlayer to bilstmlayer - add relulayer - mini batchsize =32 - pitch = true - harmonic ratio = true	Reef Check Hydrophone Audio Clip	Noise still classified as explosion
3	1300 Train with augmented reef check audio (small augmentation like reducing volume) and with white, pink, brown noise **Did not train using gunshot**	1300 of augmented reef check and white, pink, brown noise	Same as trial 1 settings.	1) Augmented reef check hydrophone 2) White, Pink, Brown Noise 3) Underwater Noise	1) 97% accuracy, 3/100 wrongly classified 2) 100% accuracy 3) 0% accuracy, all wrongly classified
4	10,000 augmented (small) Reef Check 1600 Aug UW Noise 300 Combined	5000 augmented (small) Reef Check 1100 of noise &	Same as trial 1 settings.	1) Augmented reef check hydrophone 2) White, Pink, Brown Noise	1) 100% accuracy, explosion detected 2) 100% accuracy

	white, pink. Brown noise **Did not train using gunshot**	Underwater noise		3) Underwater Noise 4) Gunshot 5) Pool generated balloon explosion	3) 100% accuracy, all noise 4) 100% accuracy, all noise 5) 100% accuracy, all noise

From the results, a few observations were made: From trials 1 and 2, training with the gunshot dataset is not suitable for this application as the datasets used were very noisy and had bad audio quality. Trials 3 and 4 were trained without the gunshot dataset and yielded better results. In trial 3, the network had an accuracy of 97% for detecting the explosions in the hydrophone reef check audio but it could not classify the underwater noise correctly. After training and validating with the underwater noise in trial 4, it could then classify the testing data 100% accurately and could even classify random testing data input to the network. Trial 4 was chosen as the final trained network that is implemented on the Raspberry Pi.

In addition, another observation was that the recorded pool explosions using the balloon was not a suitable audio to test for explosion as the network classified it as noise. This shows that this method of simulating a fish bomb explosion does not reproduce the same characteristics as a fish bomb. More research is needed to find a sustainable and safe way to recreate the fish bomb signal.

For the running window testing, the trial 4 network was able to distinguish explosions from noise. Table 6 summarizes the various window increment lengths tested. From this table, the optimal increment value is either 10000 or 12500 because the result is 1 explosion label which is expected from the tested audio clip. However, when the interval is too small (2000) or too big (20722),

the output labels are either 4 detected explosions (160:163) or all noise which means the running window missed the explosion.

The trained network is also called in the for loop where it takes this window of audio data as input into the network and classifies the window. The output is stored in the same type of categorical array that contains “Explosion” or “Noise” for each window tested. In Table 7, the running window can be seen processing various datasets. It manages to detect the explosion in the GoPro audio. It iterates through the entire audio clip and the window changes from green to red colour when it encounters an explosion.

Table 6: Running Window Increment Results

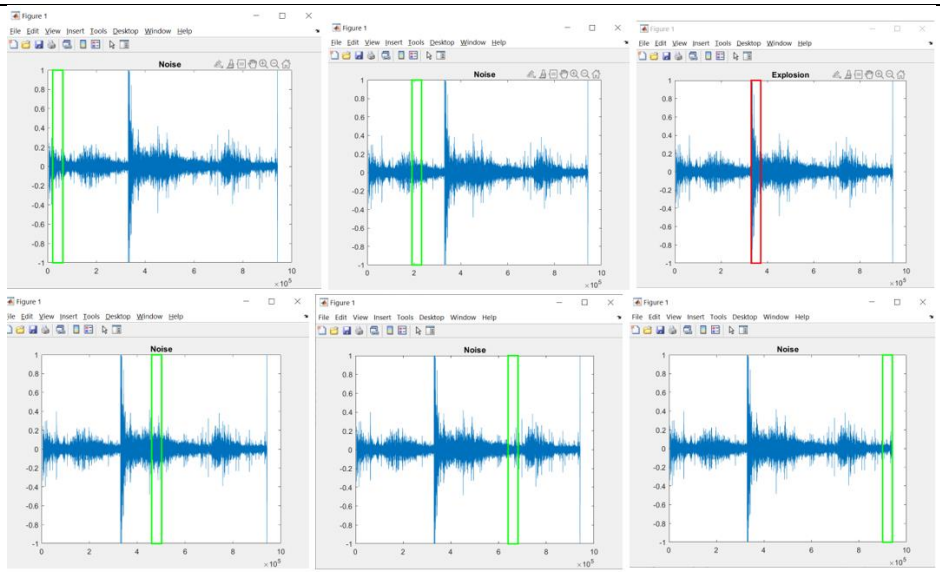
Running Window Increment	% Overlap (Running window-Increment)/(Running window)	Number of windows to process audio	Index of explosion label
2000	95	452	160:163
5000	88	452	65:66
10000	76	91	33
12500	69	91	27
15000	63	91	All noise
20722	50	91	All noise

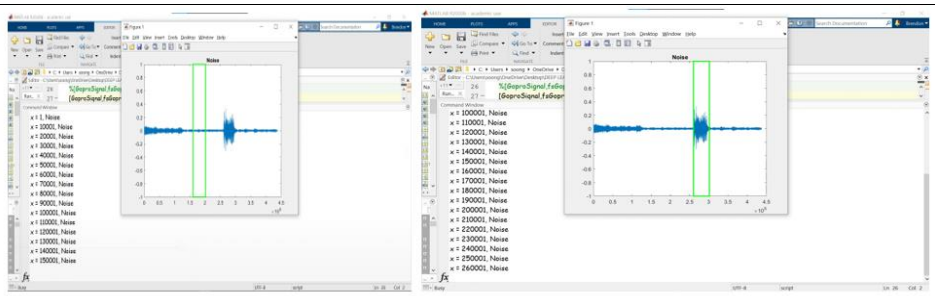
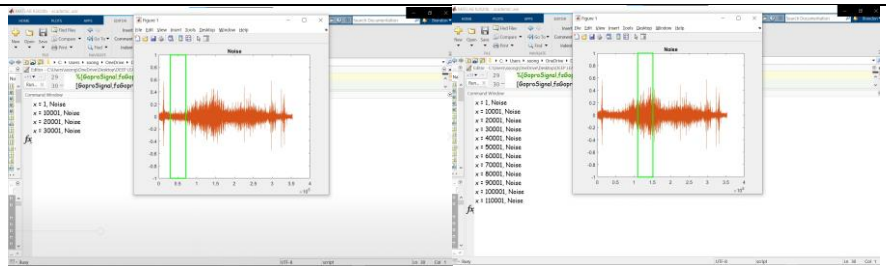
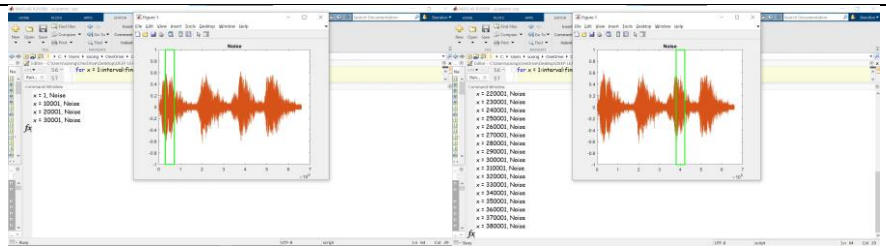
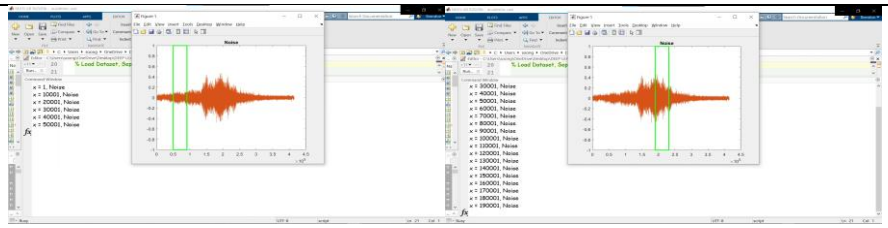
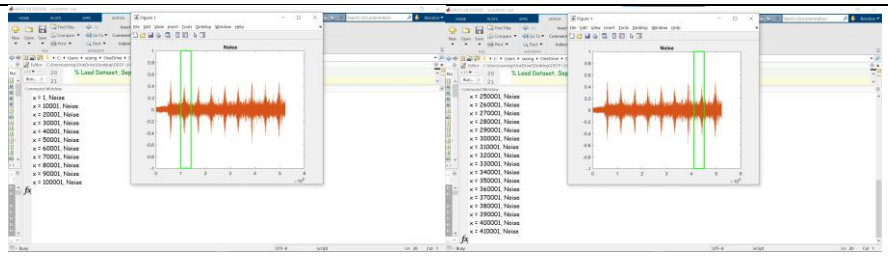
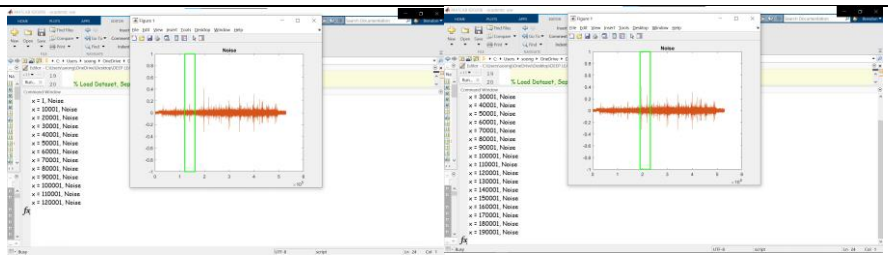
91x1 categorical

	1	2	3
26	Noise		
27	Noise		
28	Noise		
29	Noise		
30	Noise		
31	Noise		
32	Noise		
33	Explosion		
34	Noise		
35	Noise		
36	Noise		
37	Noise		
38	Noise		
39	Noise		
40	Noise		
41	Noise		

Figure 61: Running Window Explosion classification

Table 7: Running Window Processing Datasets

Dataset	Results
GoPro recording	

Balloon explosion pool recording	
Boat motor	
Humpback Whales	
Jet Ski	
Seismic Airgun	
Snapping Shrimp	

TensorFlow CNN

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 15, 27, 32)        160
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 7, 13, 32)         0
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 6, 12, 32)         4128
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 3, 6, 32)          0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 2, 5, 64)          8256
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 1, 2, 64)          0
_________________________________________________________________
flatten (Flatten)            (None, 128)               0
_________________________________________________________________
dense (Dense)                (None, 64)                8256
_________________________________________________________________
dropout (Dropout)            (None, 64)                0
_________________________________________________________________
dense_1 (Dense)              (None, 1)                 65
=================================================================
Total params: 20,865
Trainable params: 20,865
Non-trainable params: 0
```

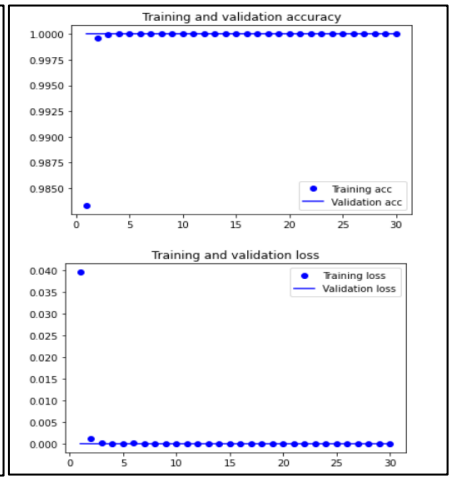


Figure 62: CNN Network training

```
Epoch 30/30
107/107 [==============================] - 2s 20ms/step - loss: 4.6071e-10 - acc: 1.0000 - val_loss: 6.0132e-20 - val_acc: 1.00
00
```

```
1 # Evaluate model with test set
2 model.evaluate(x=x_test, y=y_test)
```

```
56/56 [==============================] - 0s 3ms/step - loss: 1.4130e-19 - acc: 1.0000
```

Figure 63: CNN Network training result

From the figures above, the network was able to train successfully without much loss and had an accuracy of 100%. These results are similar to the LSTM network training where the loss is very small. When tested with the testing data, the CNN network managed to get a 100% accuracy. This high accuracy could be due to the very distinct sound of the fish bombing compared to the other noise datasets. It could also be because of the testing dataset was taken from original dataset which would mean that it can categorize data that it was trained from.

The model is then converted to the TensorFlow Lite version. A code snippet can be seen below.

```
from os import listdir
from os.path import isdir, join
import tensorflow
from tensorflow.keras import layers, models
import numpy as np
from tensorflow import lite
from tensorflow.keras import models
```

```
# Parameters
#keras_model_filename = 'C:\\Users\\soong\\fyp_explosion_CNN_model.h5'
keras_model_filename = 'fyp_explosion_CNN_model.h5'
tflite_filename = 'fyp_explosion_CNN_model.tflite'
```

```
# Convert model to TF Lite model
model = models.load_model(keras_model_filename)
converter = lite.TFLiteConverter.from_keras_model(model)
tflite_model = converter.convert()
open(tflite_filename, 'wb').write(tflite_model)
```

Figure 64: Code snippet of file conversion

5.5 Raspberry Pi Implementation

Figure 65: Explosion detected (LED Lights up)

```
Python 3.7.3 (/usr/bin/python3)
>>> %Run fyp_explosion_CNN.py
 [{'name': 'conv2d_input', 'index': 0, 'shape': array([ 1, 16, 28,  1]), 'shape_signature': array([-1, 16, 28,  1]), 'dtype': <class 'numpy.float32'>, 'quantization': (
0.0, 0), 'quantization_parameters': {'scales': array([], dtype=float32), 'zero_points': array([], dtype=int32), 'quantized_dimension': 0}, 'sparsity_parameters': {}}]
4.156035e-39
4.156035e-39
4.156035e-39
4.156035e-39
4.156035e-39
4.156035e-39
4.156035e-39
4.156035e-39
```

Figure 66: Threshold values during detection runtime

Noise	Threshold
Reef Check Explosion	>= 1
Boat	3.30E-27
Dredging	2.70E-26
Humpback Whales	3.30E-32
Jet Ski	1.24E-30
Firecracker	7.10E-21
Balloon	2.34E-24
Ocean storm	3.40E-22
Boat Motor	2.57E-37
Seismic airgun	7.00E-30
Ship Noise	4.15E-38
Snapping Shrimp	9.21E-29
GoPro	>0.5
Waves	8.90E-23
Silence	4.16E-39

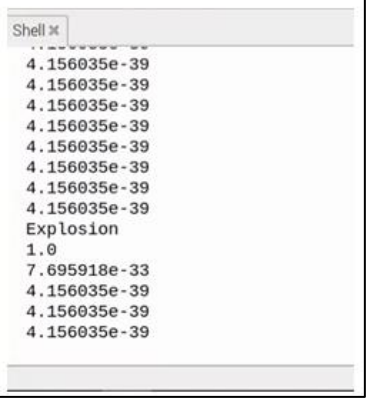


Figure 67: Threshold values for different sounds

It was observed that for the GoPro recording, the raspberry pi could not detect the explosions consistently and it had a very low average threshold value. This could be because of the microphone used to capture the signal or, that the CNN architecture is not suitable for this application. The CNN network has a lower performance compared to the Matlab LSTM network which had a higher threshold for the explosion in the GoPro recording. However, the CNN network could distinguish the explosion sounds that were used in its training as it had a high threshold value. It could also distinguish other noises accurately with a low threshold. It was also observed that running the CNN network is very computationally expensive as seen in Figure 67 where one of the CPU cores was 96% used.

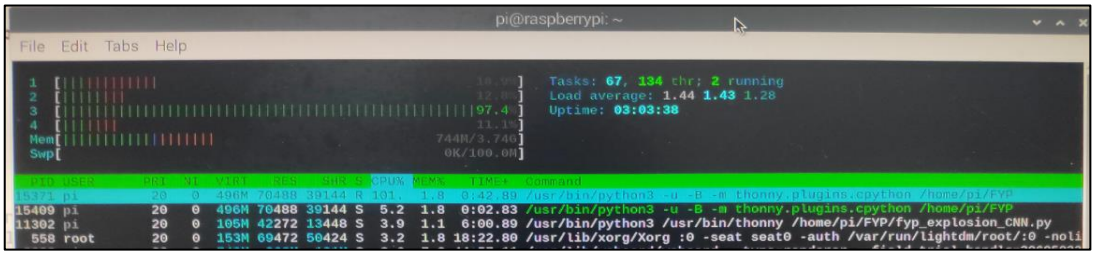


Figure 68: Raspberry Pi Resource Monitor during detection runtime

However, there were some challenges faced when running this network. When the Reef Check hydrophone audio was played, the network could not consistently detect it. If the audio was played back-to-back with very little interval between, the algorithm returned low threshold values and could not detect it. There were some cases where the detector registered medium threshold values but was not greater than the 0.5 minimum. The same goes for the GoPro recording, but this audio had a lower detection rate compared to the trained Reef Check hydrophone audio. There are a few reasons as to why the detection rate might not be consistently 100%. First, as mentioned above, the CNN network might not be suitable for this application as they are mostly used in image recognition. Computing and comparing the audio spectrogram might be due slow to process the sound signals in real time. Another reason could be the window overlap value. The window overlap value

6.0 Conclusion

In summary, a low-cost hydrophone has been designed. The results from testing the hydrophone show that it can capture audio signals. In addition, it is made from parts that can be found off the shelf. Besides this, the fish bombing audio files provided by Reef Check Malaysia were analysed. Unfortunately, replicating the sound was not successful as seen in the inflatable pool experiment.

Next, a LSTM and CNN deep learning network was shown to be able categorize fish bombing explosions or noise from the datasets input into it. The CNN network was successfully implemented on the Raspberry Pi 4 and can detect fish bombing sounds in real-time. The TensorFlow framework was shown to be suitable for Raspberry Pi implementation and it can be further explored for this application. This shows that a Raspberry Pi-based fish bombing detection system is a possible low-cost solution for ocean deployment. Re-iterating the objectives, this project is deemed successful as the scope of the project has been met.

1. Develop low-cost hydrophone that works with an audio interface.
2. Analyse audio data given by Reef Check Malaysia.
3. Develop a deep learning network for fish bombing detection.
4. Develop a cost-effective microcontroller system with hardware and software to detect fish bombing sound.

7.0 Future Work

For future work, the Method 2 hydrophone discussed earlier can also be redesigned to have a USB output instead of a music jack. This is so that it can be easily connected to the Raspberry Pi. Also, more testing is required to see if the response of this hydrophone method is similar to that of a professional hydrophone.

For the deep learning networks, the Matlab LSTM network can be reimplemented in the TensorFlow environment and uploaded to the Raspberry Pi for testing. In addition, to maximize CPU utilization, multi-threading can be used to separate different processes such as audio buffering, audio feature extraction and audio classification to be done by different cores.

Once a suitable network with high accuracy of detection has been chosen, an ocean buoy consisting of other hardware such as LoRa module, GPS module, solar panels, LiPo battery, waterproof housing and anchor can be integrated with the Raspberry Pi. The waterproof housing must be at least IP67 rated for it to be deployed in an ocean environment [28]. A good example can be seen in Figure 69 by Drifting Buoy [29]. This design can be considered for this application. Figure 70 also shows another type of buoy design.

Once an explosion is detected, the GPS module will transmit the GPS coordinates of the buoy via LoRa to a ‘surface station’ computer. An algorithm needs to be constructed to handle the GPS coordinates and timestamp to then alert authorities either using SMS or through an app.

IP65	Protected from total dust ingress.	Protected from low pressure water jets from any direction.
IP66	Protected from total dust ingress.	Protected from high pressure water jets from any direction.
IP67	Protected from total dust ingress.	Protected from immersion between 15 centimeters and 1 meter in depth.
IP68	Protected from total dust ingress.	Protected from long term immersion up to a specified pressure.
IP69K	Protected from total dust ingress.	Protected from steam-jet cleaning.

Figure 69: Waterproof ratings

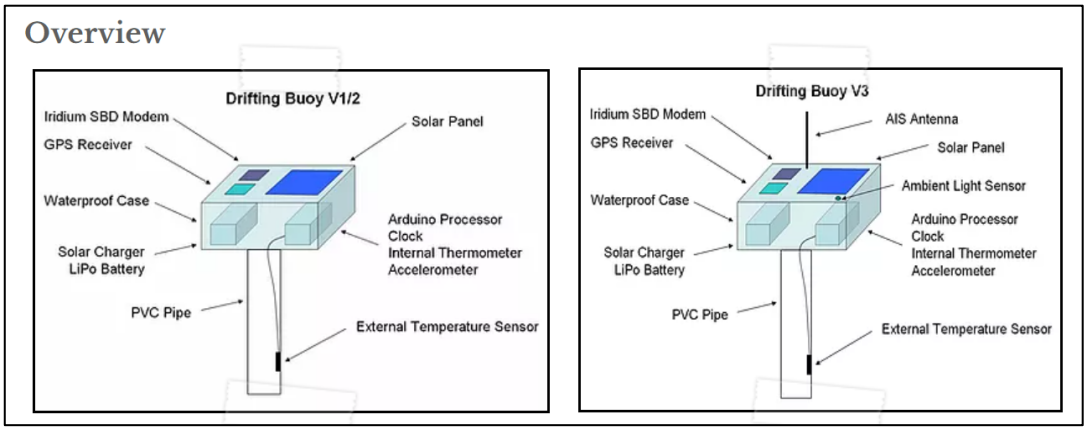


Figure 70: Floating buoy design

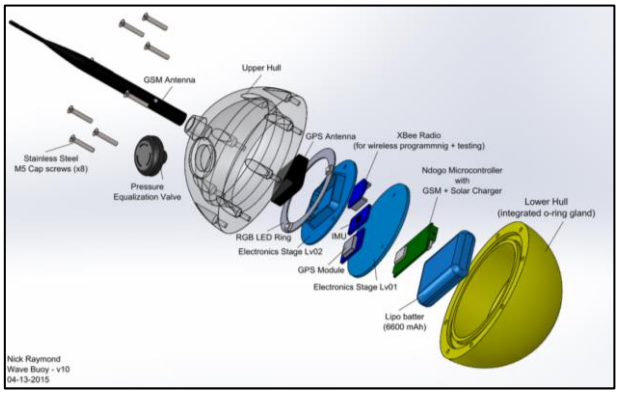


Figure 71: Other buoy designs

References

[1] E. Wood and J. Ng, "Acoustic detection of fish bombing: Final Report January 2016. Semporna Islands Project/Marine Conservation Society. 29 pages," Acoustic detection of fish bombing in Sabah, vol. 3, p. 3, 2016.

[2] G. H. Woodman, S. C. Wilson, V. Y. F. Li, and R. Renneberg, "A direction-sensitive underwater blast detector and its application for managing blast fishing," Marine Pollution Bulletin, vol. 49, no. 11, pp. 964-973, 2004/12/01/ 2004, doi: https://doi.org/10.1016/j.marpolbul.2004.06.022.

[3] G. H. Woodman, S. C. Wilson, V. Y. F. Li, and R. Renneberg, "Acoustic characteristics of fish bombing: potential to develop an automated blast detector," Marine Pollution Bulletin, vol. 46, no. 1, pp. 99-106, 2003/01/01/ 2003, doi: https://doi.org/10.1016/S0025-326X(02)00322-3.

[4] G. Braulik et al., "Acoustic monitoring to document the spatial distribution and hotspots of blast fishing in Tanzania," Marine Pollution Bulletin, vol. 125, no. 1, pp. 360-366, 2017/12/15/ 2017, doi: https://doi.org/10.1016/j.marpolbul.2017.09.036.

[5] R. Showen, C. Dunson, G. H. Woodman, S. Christopher, T. Lim, and S. C. Wilson, "Locating fish bomb blasts in real-time using a networked acoustic system," Marine Pollution Bulletin, vol. 128, pp. 496-507, 2018/03/01/ 2018, doi: https://doi.org/10.1016/j.marpolbul.2018.01.029.

[6] C. Pérez-Granados, G. Bota, D. Giralt, and J. Traba, "A cost-effective protocol for monitoring birds using autonomous recording units: a case study with a night-time singing passerine," Bird Study, vol. 65, no. 3, pp. 338-345, 2018/07/03 2018, doi: 10.1080/00063657.2018.1511682.

[7] E. Piña-Covarrubias, A. P. Hill, P. Prince, J. L. Snaddon, A. Rogers, and C. P. Doncaster, "Optimization of sensor deployment for acoustic detection and localization in terrestrial environments," Remote Sensing in Ecology and Conservation, vol. 5, no. 2, pp. 180-192, 2019/06/01 2019, doi: 10.1002/rse2.97.

[8] S. S. Sethi, R. M. Ewers, N. S. Jones, A. Signorelli, L. Picinali, and C. D. L. Orme, "SAFE Acoustics: An open-source, real-time eco-acoustic monitoring network in the tropical rainforests of Borneo," Methods in Ecology and Evolution, vol. n/a, no. n/a, 2020/06/24 2020, doi: 10.1111/2041-210X.13438.

[9] P. Roe, M. Ferroudj, M. Towsey, and L. Schwarzkopf, "Catching Toad Calls in the Cloud: Commodity Edge Computing for Flexible Analysis of Big Sound Data," in 2018 IEEE 14th International Conference on e-Science (e-Science), 29 Oct.-1 Nov. 2018 2018, pp. 67-74, doi: 10.1109/eScience.2018.00022.

[10] J. N. Oswald, Yack, T.M., Dunleavy, K.D., and Zoidis, A.M., "Development of a tool for acoustic identification of killer whale communities in the Pacific Ocean," 2015.

[11] C. Edgar fernando, B. Christopher, K. Robert, and R. Jason, Acoustic Monitoring of Blast Fishing: Pilot Study - Dar es Salaam. 2014.

[12] D. T. Blumstein et al., "Acoustic monitoring in terrestrial environments using microphone arrays: applications, technological considerations and prospectus," Journal of Applied Ecology, vol. 48, no. 3, pp. 758-767, 2011/06/01 2011, doi: 10.1111/j.1365-2664.2011.01993.x.

[13] Dalskov, "Locating Acoustic Sources with Multilateration - Applied to Stationary and Moving Sources," 2014.

[14] M. C. Domingo, "An overview of the internet of underwater things," J. Netw. Comput. Appl., vol. 35, no. 6, pp. 1879–1890, 2012, doi: 10.1016/j.jnca.2012.07.012.

[15] G. A. D. Lepper.P, "A simple hydrophone monitor for cetacean acoustics. In: P.J.Evans (ed) European Research on Cetaceans," European Cetacean Society, Cambridge, no. 9, 1995.

[16] E. Vivas and B. Leon-Lopez, "Construction, calibration, and field test of a home-made low-cost hydrophone system for cetacean acoustic research," The Journal of the Acoustical Society of America, vol. 128, p. 2438, 10/01 2010, doi: 10.1121/1.3508712.

[17] J. Barlow, S. Rankin, and S. Dawson, "A Guide to Constructing Hydrophones and Hydrophone Arrays for Monitoring Marine Mammal Vocalizations," 2013.

[18] E. Browning, R. Gibb, P. Glover-Kapfer, and K. Jones, Passive acoustic monitoring in ecology and conservation. 2017.

[19] http://cadreforensics.com/audio/“Gunshot Audio Analysis,” Gunshot Audio Forensics Dataset. [Online]. Available: http://cadreforensics.com/audio/. [Accessed: 29-Apr-2021].

[20] T. Spadini, “Sound Events for Surveillance Applications,” https://zenodo.org/record/3519845#.YIokLJAzbY8, 20-Oct-2019. [Online]. Available: https://zenodo.org/record/3519845#.YIokLJAzbY8. [Accessed: 29-Apr-2021].

[21] Raponi, Simone & Ali, Isra & Oligeri, Gabriele. (2020). Sound of Guns: Digital Forensics of Gun Audio Samples meets Artificial Intelligence.

[22] C. K. On, P. M. Pandiyan, S. Yaacob and A. Saudi, "Mel-frequency cepstral coefficient analysis in speech recognition," 2006 International Conference on Computing & Informatics, 2006, pp. 1-5, doi: 10.1109/ICOCI.2006.5276486.

[23] Kingma, D. and Ba, J., 2021. Adam: A Method for Stochastic Optimization. [online] arXiv.org. Available at: <https://arxiv.org/abs/1412.6980> [Accessed 18 May 2021].

[24] Bengio, Y., 2021. Practical recommendations for gradient-based training of deep architectures. [online] arXiv.org. Available at: <https://arxiv.org/abs/1206.5533> [Accessed 18 May 2021].

[25] S. Siami-Namini, N. Tavakoli and A. S. Namin, "The Performance of LSTM and BiLSTM in Forecasting Time Series," 2019 IEEE International Conference on Big Data (Big Data), 2019, pp. 3285-3292, doi: 10.1109/BigData47090.2019.9005997.

[26] Hymel, S., 2021. ShawnHymel/tflite-speech-recognition. [online] GitHub. Available at: <https://github.com/ShawnHymel/tflite-speech-recognition> [Accessed 18 May 2021].

[27] Young, Andrew T. "Distance to the Horizon". San Diego State University Department of Astronomy. Archived from the original on October 18, 2003. Retrieved April 16, 2011.

[28] “IP Rating Chart,” DSMT.com. [Online]. Available: https://www.dsmt.com/resources/ip-rating-chart/. [Accessed: 24-May-2021].

[29] “Maker Buoy,” Makerbuoy. [Online]. Available: https://www.makerbuoy.com/about-1. [Accessed: 24-May-2021].

Appendices

a. Table of Modern Tools for Acoustic Monitoring

This table provides a summary compiled by [18] of the current hardware and software used for Acoustic Monitoring and Analysis.

Table 1

Company	Summary	Species/ habitat	Example models and uses	Website
ARBIMON / Sieve Analytics (Puerto Rico)	Hardware, software and data analysis tools for acoustic biodiversity monitoring, including a cloud-computing platform for data storage and analysis.	Terrestrial, audible range	ARBIMON Portable Recorders; ARBIMON Permanent Monitoring Stations (with solar panels and ethernet/wi-fi data transfer capacity); ARBIMON II web-based analysis platform.	www.sieve-analytics.com /#!arbimon/cjg9
AudioMoth (UK)	Low-cost open-source environmental acoustic sensor, which can record both audible range and full spectrum ultrasound	Terrestrial, audible range and ultrasonic	AudioMoth sensor details discussed in Case Study 3 of this report, with further details provided at the website.	www openacousticdevices.info
Chelonia (UK)	Marine passive acoustic monitoring equipment for odontocete monitoring.	Aquatic, odontocetes	C-POD and DeepC-PO, detect odontocete echolocation clicks and record several call parameters (e.g. time, centre frequency, duration) for later analysis. Can record for 4+ months.	www.chelonia.co.uk
Dodotronic (Italy)	Bioacoustic sensor manufacturer producing a range of recorders and microphones (ultrasonic and audible range)	Terrestrial, ultrasonic and audible range	Ultramic, Ultramic384K (ultrasonic microphone/recorder); MOMIMIC (miniature microphone electronics component); Hydromic (ultrasonic hydrophone preamplifier)	www.dodotronic.com
Elekon (Switzerland)	Batlogger range of full-spectrum bat recorders and detectors.	Terrestrial, bats	Batlogger C and A/A+ (for passive monitoring); Batlogger M (for transects); Batscanner (heterodyne detectors). BatExplorer analysis software.	www.batlogger.com/en
Frontier Labs (Australia)	Recording equipment for bioacoustic and ecological research.	Terrestrial, audible range	Bioacoustic Audio Recorder, includes omnidirectional microphone, integrated GPS unit and sampling rate up to 96kHz. Handheld Audio Recorder to attach to smartphone.	www.frontierlabs.com.au
High Tech Inc (USA)	Hydrophones and recording systems for marine environments. Many models are for industrial/military uses but also produce marine mammal hydrophones.	Aquatic	HTI's Marine Mammal hydrophone, maximum operating depth of >3000m. Produce a range of hydrophones that can be customised to order.	www.hightechincusa.com
Ocean Instruments (New Zealand)	Produce self-contained underwater autonomous recorders for ocean acoustic research.	Aquatic	SoundTrap 300 STD and HF models can record constantly for 13 days and weigh approximately 0.5 kg.	www.oceaninstruments.co.nz
Petterson Elektronik (Sweden)	Ultrasonic bat detectors and analysis softwares, with heterodyne, frequency-division and full-spectrum models.	Terrestrial, bats	D230 (frequency division); D500X (full-spectrum static detector); D1000X (frequency-division/time expansion). Also produce M500-384 ultrasonic microphone that can be attached to a smartphone.	www.batsound.com
Solo (UK)	Open-source, low-cost audio recorder built around a Raspberry Pi microcomputer, with customisable specifications	Terrestial, audible range	Customisable: example configuration provided in publication is costed at UK£167 (Whytock & Christie 2016).	solo-system.github.io/home.html
Teledyne Reson (Denmark)	Marine acoustic monitoring equipment, mainly for industrial and military purposes, but also produce a range of hydrophones useful for bioacoustic research.	Aquatic	Various hydrophone models; website includes a useful hydrophone look-up table.	www.teledyne-reson.com
Titley Scientific (UK)	Anabat Systems range of bat detectors. Mainly use zero-crossing/ frequency-division detection, but some also include heterodyne and full-spectrum/time-expansion modes.	Terrestrial, bats	Anabat SD2 (frequency-division); Anabat Walkabout (zero-crossing/time-expansion/ heterodyne) bat detector; Anabat Express (zero-crossing, waterproof, static sensor). Analook analysis software.	www.titley-scientific.com

Titley Scientific (UK)	Anabat Systems range of bat detectors. Mainly use zero-crossing/ frequency-division detection, but some also include heterodyne and full-spectrum/time-expansion modes.	Terrestrial, bats	Anabat SD2 (frequency-division); Anabat Walkabout (zero-crossing/time-expansion/ heterodyne) bat detector; Anabat Express (zero-crossing, waterproof, static sensor). Analook analysis software.	www.titley-scientific.com
Wildlife Acoustics (USA)	Wide range of bioacoustic recorders, microphones, hydrophones and analysis softwares, suitable for a range of taxonomic groups and habitat deployments.	Terrestrial and aquatic, all taxonomic groups	Song Meter audible range (e.g. SM3, SM4) and full-spectrum and zero-crossing ultrasonic (e.g. SM3BAT, SM4BAT) recorders for terrestrial habitats. Song Meter SM3M Deep Water and Submersible for marine habitats at a range of depths. Analysis softwares include Kaleidoscope (bats) and Song Scope.	www.wildlifeacoustics.com

Software	Availability	Summary	Website
ARBIMON II	Free initially, charges apply for larger quantities of data	Cloud-computing based bioacoustics storage and analysis platform (Aide 2013); features include visualising and annotated recordings, soundscape analysis and automated call detection via pattern matching.	arbimon.sieve-analytics.com
Audacity	Free, open source	Intuitive, general-use audio software that enables listening, viewing spectrograms, subsetting and annotating files.	www.audacityteam.org
AudioTagger	Free	Free audio software for listening, viewing and manually annotating large volumes of audio files.	github.com/groakat/AudioTagger
AviSoft	Proprietary (AviSoft-SASLab Pro); free (Lite)	Bioacoustic analysis software, functions include visualisation and annotation, automated classification tools (spectrogram cross-correlation), geo-referencing tools and noise analysis.	www.avisoft.com
BatScope	Free	Free software for visualising and analysing full-spectrum bat recordings, including automatic species call classifiers.	www.wsl.ch/dienstleistungen/produkte/software/batscope/index_EN
iBatsID	Free	Free software tool for classifying European bat call recordings to genus and species (Walters *et al.* 2012); requires call parameters extracted by SonoBat (see below).	ibatsid.eu-west-1.elasticbeanstalk.com
CPOD.exe	Free	Free software for analysing data collected by T-PODs and C-PODs. Reads raw data from the C-POD SD cards and detects trains of cetacean clicks and classifies into groups (e.g. narrow-band high frequency clicks and other cetaceans).	www.chelonia.co.uk/cpod_downloads.htm
Ishmael	Free	Specialised bioacoustics software from CIMRS, with a marine emphasis. Includes visualisation and annotation tools and functions for aquatic sound localisation and automated call recognition.	www.bioacoustics.us/ishmael.html
Kaleidoscope	Proprietary (Kaleidoscope Pro), however spectrogram viewer tool is free	Software package by Wildlife Acoustics for bioacoustic analysis, including methods to visualise and annotate files, tools for cluster analysis and classifier training, batch processing, and bat call classifiers.	www.wildlifeacoustics.com/products/kaleidoscope-software
PAMGUARD	Free, open-source	Open-source package for bioacoustic research, with an emphasis on marine mammals. In addition to core acoustic analysis functionality, a variety of plugins are available for more complex signal processing and analysis, including detection, classification and localisation.	www.pamguard.org
Pumilio	Free, open-source	Open-source application for managing bioacoustic recordings, including visualising, annotating and manipulating sound files (see Villanueva-Rivera *et al* 2012).	ljvillanueva.github.io/pumilio
Raven	Proprietary (Raven Pro); free (Raven Lite)	Sound analysis software from Cornell Lab of Ornithology, with functions including visualisation/annotation, call detection and spectrogram correlation.	www.birds.cornell.edu/brp/raven/RavenOverview.html

Software	Availability	Summary	Website
Seewave (R package)	Free, open source	Package for the open-source statistical environment R, providing a range of tools for bioacoustic analysis including visualisation, annotation and calculating acoustic indices.	rug.mnhn.fr/seewave
Song Scope	Proprietary	Software by Wildlife Acoustics for spectrogram visualisation, including over long timescales.	www.wildlifeacoustics.com/products/song-scope-overview
Sonobat	Proprietary	Software for analysis of full-spectrum bat recordings; features include visualisation, call detection, parameter extraction and species classification.	www.sonobat.com
SonoChiro	Proprietary	Software for automated analysis of full-spectrum bat recordings, designed for use with large volumes of data. Includes automated species classifiers for Europe and the Neotropics.	www.biotope.fr/fr/accueil-innovation/sonochiro
Soundecology (R package)	Free, open source	Package for R providing functions to calculate soundscape indices from spectrograms.	cran.r-project.org/web/packages/soundecology/index.html
Tadarida	Free	Open software and code for developing and applying an acoustic classifier.	github.com/YvesBas (Bas *et al.* 2017).
WarbleR (R package)	Free, open-source	Package for R providing a range of functions for batch processing of bioacoustic signals, including spectrogram visualisation, feature extraction, cross-correlation functions and recording quality assessment	cran.r-project.org/web/packages/warbleR/index.html

b. Gantt chart

		Task Name	3/8 -9/8	10/8 - 16/8	17/8 - 23/8	24/8 -30/8	31/8 - 6/9	7/9 - 13/9	14/9 - 20/9	21/9 - 27/9	28/9 - 4/10	5/10 - 11/10	12/10 - 18/10	19/10 - 25/10	26/10 - 1/11	2/11 - 8/11
			1	2	3	4	5	6	7	MSB	8	9	10	11	12	13
SEMESTER 2 2020	1	Literature review														
	2	Feasibility report														
	3	Develop hydrophone														
	4	Test hydrophone and record audio														
	5	Recreate bomb sound														
	6	Develop fish bomb detection hardware or purchase off-shelf components														
	7	Progress Report														
	8	Progress Presentation Video														
	9	Progress Presentation Q&A Session														

Gantt chart for Semester 2 2020

		Task Name	1	2	3	4	5	6	7	MSB	8	9	10	11	12	13
SEMESTER 1 2021	1	Develop hardware														
	2	Develop or modify existing localization algorithms														
	3	Test entire system														
	4	Build app interface														
	5	Final Report Draft														
	6	Research Paper														
	7	Final Seminar Presentation and Demo Videos														
	8	Final Seminar Presentation Q&A Session														
	9	Final Report														
	10	Student Reflection														
	11	Best FYP Competition														

Gantt chart for Semester 1 2021

c. Risk Analysis

No.	Critical Aspect	Risk Level	Countermeasure
1	MCO Lockdown	Med	Build parts in modular fashion. Once access is available to lab, join everything together
2	Cost of LoRa gateway too expensive	Low	Try using Raspberry Pi gateway
3	Unable to find lakes/swimming pool to test devices	Med	Test in plastic water tank.
4	Unable to find audio files on fish bombing sound.	Med	Recreate sound in test environment by comparing signal with literature.

5	System does not function as intended after integrating all the parts	Low	Analyse subsystems into smaller sections. Break down activities into smaller pieces.
6	Project too complex and beyond capabilities	Med	Ensure objectives are specific, measurable, achievable, realistic and time-bound.

Risk Analysis

d. Deep Learning Network Trials

Trial	Train Dataset	Validation Dataset	Neural Network Settings
1	18469 of gunshot and augmented Reef Check (heavy augmentation like adding noise)	5040 augmented Reef Check	Layers = sequenceInputLayer(numFeatures) lstmLayer(50,"OutputMode","last") fullyConnectedLayer(numel(unique(Labels))) softmaxLayer classificationLayer]; miniBatchSize = 27; options = trainingOptions("adam", ... 'ExecutionEnvironment','cpu', ... "Shuffle","every-epoch", ... 'MiniBatchSize',miniBatchSize, ... 'GradientThreshold',1, ... "ValidationData",{featuresValidation,LabelsVal}, ... "Plots","training-progress", ... "Verbose",false);
2	same	same	Layers = sequenceInputLayer(numFeatures) **bilstmLayer(50,"OutputMode","last")** fullyConnectedLayer(numel(unique(Labels))) reluLayer softmaxLayer classificationLayer]; **miniBatchSize = 27;** options = trainingOptions("adam", ... 'ExecutionEnvironment','cpu', ... "Shuffle","every-epoch", ...

			'MiniBatchSize',miniBatchSize, ... 'GradientThreshold',1, ... "ValidationData",{featuresValidation,LabelsVal}, ... "Plots","training-progress", ... "Verbose",false); (Changed lstmlayer -> bilstmlayer, Added relulayer, miniBatchsize changed from 32 to 64, 128, 1024)
3	1300 Train with augmented reef check audio (small augmentation like reducing volume) and also train with white, pink ,brown noise Did not train using gunshot	1300 of augmented reef check and also train with white, pink ,brown noise	Layers = sequenceInputLayer(numFeatures) lstmLayer(50,"OutputMode","last") fullyConnectedLayer(numel(unique(Labels))) softmaxLayer classificationLayer]; miniBatchSize = 27; options = trainingOptions("adam", ... 'ExecutionEnvironment','cpu', ... "Shuffle","every-epoch", ... 'MiniBatchSize',miniBatchSize, ... 'GradientThreshold',1, ... "ValidationData",{featuresValidation,LabelsVal}, ... "Plots","training-progress", ... "Verbose",false);
4	10,000 augmented (small) Reef Check 1600 Aug UW Noise 300 Combined white, pink. Brown noise (03/04/2021)	5000 augmented (small) Reef Check 1100 of noise & UW noise	Layers = sequenceInputLayer(numFeatures) lstmLayer(50,"OutputMode","last") fullyConnectedLayer(numel(unique(Labels))) softmaxLayer classificationLayer]; miniBatchSize = 27; options = trainingOptions("adam", ... 'ExecutionEnvironment','cpu', ... "Shuffle","every-epoch", ... 'MiniBatchSize',miniBatchSize, ... 'GradientThreshold',1, ...

			"ValidationData",{featuresValidation,LabelsVal}, ... "Plots","training-progress", ... "Verbose",false);

e. Link for the repository for all codes

https://drive.google.com/drive/folders/1ys60eOGeCvJEktGU6hApB_AmqPmrM0XY?usp=sharing

f. Reflection on Program Outcomes (PO) Achievement

Program Outcomes	
1	Apply knowledge of science and engineering fundamentals and achieve specialization in solving complex Mechatronics Engineering problems (Engineering knowledge)
	Student's Reflection: Throughout the final year project I have applied the knowledge learnt throughout my 4 years from programming, software, mathematics and science and signal processing
2	Identify, formulate and analyses complex engineering problems, and reach substantiated conclusions (Problem analysis)
	Student's Reflection: I am able to analyze the pros and cons of each system and determine the possible improvements made by writing the literature review to identifying the problem statement
3	Design solutions to complex engineering problems (Design/Development of solutions)
	Student's Reflection: The project aims to develop and design a microcontroller system that can detect explosions in real-time.
4	Analyses complex engineering problems and systems using research-based knowledge and methods (Investigation)
	Student's Reflection: Plenty of research was done to understand the current solutions used to detect fish bombings, what the signal characteristics are for explosions, and what software is suitable to detect it.
5	Create, select and apply appropriate techniques, resources and modern engineering and IT tools to complex engineering activities with an understanding of their limitations (Modern tool usage)
	Student's Reflection: I have used Audacity, Matlab, Jupyter notebook, python.

6	Assess social, public health and safety, cultural and legal consequences of complex engineering solutions and relate them to the responsibilities of a professional engineer (Engineer and society)
	Student's Reflection: It is an engineer's responsibility to follow the code of ethics. Also, in order to simulate fish bombing sound, it was not possible to test using a real bomb so the pool experiment was conducted.
7	Demonstrate knowledge of and need for sustainable development and understand the environmental impacts of complex engineering solutions (Environment and sustainability)
	Student's Reflection: To achieve this PO, minimal testing materials sucha s balloons/firecrackers was used. Experiments were conducted in a controlled area where the environment was not affected.
8	Apply ethical principles and commit to professional ethics and responsibilities and norms of engineering practice (Ethics)
	Student's Reflection: Knowing that this prototype is to be deployed at sea, safety factors are included during simulations. Robust materials that are environmentally friendly are considered for ocean deployment.
9	Communicate effectively both in oral and written forms (Communication)
	Student's Reflection: Weekly meetings with the supervisor is done to communicate ideas and exchange thoughts and to report the progress. Proposals and Reports such as this current one are written and submitted
10	Function effectively as an individual and in multi-disciplinary and multicultural teams (Individual and team work)
	Student's Reflection: The FYP is a solo project however involving 3 parties, me, Mr Khoo my supervisor, and Mr Adzmin from Reef Check Malaysia. We communicate effectively using whatsapp. Mr Khoo and I have weekly meetings where we set weekly goals and other deadlines.
11	Recognize the need for independent and lifelong learning, and possess the capacity to do so (Lifelong learning)
	Student's Reflection: Throughout the project, I realized that there are a lot of interesting initiatives currently used by the Sabah reef check team which are not in my area of expertise and I realized that we have to keep learning.
12	Manage an engineering project systematically (Project management and finance)
	Student's Reflection: By completing this FYP and other projects before, I have learnt to manage a project systematically and have achieved the objectives I have set for my project.