UNIVERSITI TEKNOLOGI MARA

SOUND EVENT DETECTION OF WILDLIFE RESERVE INTRUSION DETECTION USING HYBRID CONVOLUTIONAL NEURAL NETWORK AND RANDOM FOREST

MUHAMAD AMIRUL SADIKIN BIN MD AFENDI

MSc

February 2022

UNIVERSITI TEKNOLOGI MARA

SOUND EVENT DETECTION OF WILDLIFE RESERVE INTRUSION DETECTION USING HYBRID CONVOLUTIONAL NEURAL NETWORK AND RANDOM FOREST

MUHAMAD AMIRUL SADIKIN BIN MD AFENDI

Thesis submitted in fulfilment
of the requirements for the degree of
Master of Sciences
(Computer Science)

Faculty of Computer and Mathematical Sciences

February 2022

CONFIRMATION BY PANEL OF EXAMINERS

I certify that a Panel of Examiners has met to conduct the final examination of Muhamad Amirul Sadikin Bin MD Afendi on his **Master of Science** thesis entitled "Sound Event Detection of Wildlife Reserve Intrusion Detection Using Hybrid Convolutional Neural Network - Random Forest" in accordance with Universiti Teknologi MARA Act 1976 (Akta 173). The Panel of Examiner recommends that the student be awarded the relevant degree. The Panel of Examiners was as follows:

Ahmad Zia Ul-Saufie Mohamad Japeri, PhD
Associate Professor
Faculty Of Computer and Mathematical Sciences
Universiti Teknologi MARA
(Chairman)

Azlin Ahmad, PhD
Faculty Of Computer and Mathematical Sciences
Universiti Teknologi MARA
(Internal Examiner)

Sharifah Sakinah Syed Ahmad, PhD
Associate Professor
Faculty Of Computer and Mathematical Sciences
Universiti Teknikal Malaysia Melaka
(External Examiner)

PROFESSOR IR DR ZUHAINA HAJI ZAKARIA
Dean
Institute of Graduates Studies
Universiti Teknologi MARA
Date: 18 February 2022

AUTHOR'S DECLARATION

I declare that the work in this thesis was carried out in accordance with the regulations of Universiti Teknologi MARA. It is original and is the results of my own work, unless otherwise indicated or acknowledged as referenced work. This thesis has not been submitted to any other academic institution or non-academic institution for any degree or qualification.

I, hereby, acknowledge that I have been supplied with the Academic Rules and Regulations for Postgraduate, Universiti Teknologi MARA, regulating the conduct of my study and research.

Name of Student	:	Muhamad Amirul Sadikin Bin MD Afendi
Student I.D. No.	:	2019342591
Programme	:	Master of Sciences (Computer Sciences) – CS750
Faculty	:	Computer Science and Mathematics
Thesis Title	:	Sound Event Detection Of Wildlife Reserve Intrusion Detection Using Hybrid Convolutional Neural Network And Random Forest
Signature of Student	:	……
Date	:	February 2022

ABSTRACT

The wildlife reserve is a sanctuary for many endangered species with a high value. Therefore, wildlife reserve intrusion by poachers and illegal loggers needs to be stopped before they cause a disturbance. Commonly available security solutions require a high cost to area coverage ratio for the vast forest environment. Implementing modern technology might allow more cost-efficient solutions to this problem. Intruders often emit sounds from their very distinctive forest activities. Sound Event Detection (SED) is expected to be able to assist in the detainment of intruders. This research should act as an extension to increase the performance of wildlife security in Malaysia. To the best of our knowledge, the application of SED in the Malaysian Forest environment for security purposes has not yet been explored. Machine Learning (ML) method for surveillance in forest environments seems viable. The use of Mel-log Energies are sound features formidable for the task and Convolutional Neural Networks (CNN) have shown good performance with SED in urban environments. However, sound events frequency overlapping leads to a high correlation of features between SED classes, leading to a high false positive rate. Hence, this study embarks on research on ML capabilities and sound features methods for SED. The CNN model was tested on first-hand forest environment sound data to measure its true performance. Multiple models including a hybrid CNN - Random Forest (RF) hybrid model were formulated to find the best performance. Several parameters were also used to tune all models to achieve the best outcome. In addition, a post-processing layer was applied to cater false alarms to an acceptable level by using threshold decision making. The research finding showed that SED can detect intruders within 100m radius. It demonstrated that the CNN-RF model outperformed by a small margin compared to other models. It produced up to 0.8215 F1-Score while having a false prediction rate of approximately 10%. The study aimed to help more advanced research for SED application in the forest for wildlife reserve security. Hence, it may lead to improvements in protecting wildlife from danger by providing effective surveillance solutions

ACKNOWLEDGEMENTS

Firstly, I would like to thank my primary supervisor, Associate Professor Dr. Marina Yusoff, for her resourceful advises that helped me through my research up till my thesis writing. Her involvement and encouragement were gratefully appreciated.

Next, my special thanks to my co-supervisor, Dr Zaki Zakaria for his insights on the study. I am blessed with good minds around me throughout completing this study. Thanks for his continuously active support on this research.

I would also like to thank, Associate Professor Dr Megawati Omar for encouraging me to pursue my studies. Her support helped throughout my journey and would not be the same without.

Finally, my utmost thanks to Malaysian Technical Standards Forum Berhad (MTFSB) in providing a financed project, supporting a proof-of-concept for this research's interest that allowed data collection in Malaysian forests essential to the study. This advanced the research with a variety of data to tinker with.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

Abbreviations

AAC	Augmentative and Alternative Communication
ADAM	Adaptive Moment Estimation
ADC	Analog-to-digital Converter
Bi-LSTM	Bidirectional-LSTM
CNN	Convolutional Neural Network
CNN-GRU	Convolution Recurrent Neural Network-gated Recurrent Unit
CNN-KNN	Convolution Recurrent Neural Network K-Nearest Neighbour
CNN-RF	Convolutional Neural Network Random Forest
CRNN	Convolutional Recurrent Neural Network
CRNN	Convolution Recurrent Neural Network
DCASE	Detection and Classification of Acoustic Scenes and Events
DL	Deep Learning
DNN	Deep Neural Network
DT	Decision Tree
DWT	Discrete Wavelet Tempo
ER	Error Rate
FC	Fully Connected
FCN	Fully Convolutional Network
FFT	Fast Fourier Transform

FN	False Negative
FP	False Positive
GMM	Gaussian Mixture Model
GPS	Global Positioning System
HMM	Hidden Markov Model
JPSM	Jabatan Perhutanan Semenanjung Malaysia
KNN	K-Nearest Neighbor
LSTM	Long Short-term Memory
MBEON	Malaysia Biodiversity Enforcement Operation Network
MEL	Mel-log Energies
MFCC	Mel-frequency Cepstral Coefficients
ML	Machine Learning
MLE	Mel-log Energies
MLP	Multilayer Perceptron
MP3	Moving Picture Experts Group (MPEG) -1 Audio Layer III
NMF	Non-negative Factorization Matrix
NN	Neural Network
PCA	Principal Component Analysis
PERHILITAN	Perlindungan Hidupan Liar dan Taman Negara Wilayah Persekutuan
PlantCLEF	Plant Identification Challenge
PTNJ	Perbadanan Taman Negeri Johor

PTNP	Perbadanan Negeri Perak
RF	Random Forest
RF	Random Forest
RNN	Recurrent Neural Network
ROC	Receiver Operating Characteristics
SBC	Single Board Computers
SED	Sound Event Detection
SVM	Support Vector Machine
TP	True Positive
TP	True Positives
TUT	Tampere University Technology
VGG	Visual Geometry Group
WAV	Waveform Audio File
WCS	Wildlife Conservation and Science
WMA	Windows Media Audio

CHAPTER ONE
INTRODUCTION

1.1 Research Background

Wildlife all around the world are being hunted rampantly which reduces them to the point of extinction. Wildlife sanctuaries were made in efforts to protect and preserve them, but many still slip through the cracks. Recently, wildlife law enforcement in Malaysia and Southeast Asia regions have confiscated various wildlife from poachers (Rivera, Knight, & McCulloch, 2021). Despite the efforts, perpetrators' illegal logging and poaching activities are still due to the demand (Shepherd, Gomez, & Nijman, 2020). The protection of wildlife has become of concern as their numbers grow smaller every year (Zainuddin, 2020). Unfortunately, detecting poaching activities is tough due to the forest environment. The forests in Malaysia consist of tropical rainforest with thick foliage and bushes that cause huge difficulty in surveillance. There are many people who lost their way in these thick forests (Middleton, 2020). According to Saw Leng Guan from the Forest Research Institute Malaysia, "Malaysia has one of the most complex ecosystems in the world" (Saw, 1992).

The current available solutions using human patrol and cameras are costly to facilitate the security. This is due to the vast area such as the Johor National Park which spans more than 400 km^2. The amount of surveillance area to be covered by the rangers are immensely outnumbered. Therefore, the use of camera traps has been the current solution to improve the surveillance (Dasgupta, 2018). But the nature of the Malaysian rainforest makes it more difficult to conduct surveillance. The rainforest trees are dense which limit the field of view for cameras.

It was reported that poachers live in the forest. They emit noise at several locations, but they can successfully flee before the law enforcement can act (WCS, 2018). The noises are being made when they cut trees to make traps to capture the animals. These noises can be heard up to kilometers away due to the vast forest area which is more than 14km wide. This leads to the chances of reacting on time is almost impossible (WCS, 2018). The illegal logging activity is very loud but conducted deep within the forest. This makes it difficult even though its sound is very detectable.

Sound Event Detection (SED) to recognize a sound event's presence using artificial intelligence methods. SED in detecting the intrusions for wildlife reserves as the poacher's activities sound very distinctive within the natural ambience. Previous work on the SED uses various types of Machine Learning (ML) and Deep Learning (DL) algorithms such as random forest (RF) and Convolutional Neural Network (CNN). A review of the SED solutions was conducted on an artificial dataset from Detection and Classification of Acoustic Scenes and Events (DCASE 2017). The researchers used Convolutional Recurrent Neural Network - Long Short-Term Memory (CRNN - LSTM) to obtain 93% F1 Score (Lim, Park & Han, 2017). In the meantime, the CNN achieved 91% (Cakir & Virtanen, 2017), Multilayer Layer Perceptron (MLP - CNN) with 84% (Ravichandran & Das, 2017), while ensemble with 78% (Dang, Vu, & Wang, 2017). Although these experiments used artificial dataset, it still shows how the SED is becoming ready for industrial applications.

The real-world environment sounds are far from noise-free (Chung, 2020). Unlike the DCASE 2017 dataset, the SED done on noisy data can be most challenging. Sound event's effective detection rate also varies on recent SED experiments and is somewhat biased to the context (Serizel, Turpault, Shah & Salamon, 2020). Noisy sound data is inevitable as sound travels through air shared by many other sounds that may interfere with different sounds in detection. Recent work on the detection in the felling of trees using the SED with real-world data on the domain of a dense forest employed an improved distance-based algorithm and achieved 76% accuracy with 21% False positive (FP) outperforming in the efficiency of other methods such as the Gaussian mixture model (GMM), K-Means Clustering and Principal Component Analysis (PCA) (Ahmad & Singh, 2019). In most of the SED cases in real-world scenarios, noise can be unpredictable where no collection can sum up all existing sounds in the domain.

In the SED application, features need to be extracted from the raw audio data to a piece of tailored information to better distinguish between sound events more effectively. A common sound feature extraction method used in the recent SED studies uses Mel-Log Energies (MLE). The features are rich with essential values that contribute to class recognition. The methods explored by the previous research frequently reported that using MLE features with the CNN has produced good results (Turpault & Serizel, 2020). These features significantly support the model's performance (Tian, Xu & Zuo, 2020). MLE features an extraction process that includes

Fast Fourier Transform (FFT) for frequency separation is useful as different sounds has unique frequencies. The process would increase the features distinguish-ability between sound events. The MLE features processes are inspired by the hearing biological system perceiving loudness of frequencies. It lets the ML model perceive how humans perceive sound.

The MLE sound feature extraction is observed as a graph of frequency intensity changes between a specified time frame. The model detects these pattern changes to predict a sound event occurrence. Hence this problem can be solved using a pattern recognition model such as image object detection algorithms. CNN is well known for its excellent detection of images (Otter, Medina, & Kalita, 2018; Hershey et al., 2017). Therefore, the CNN is believed to be reliable for SED. The CNN requires large amount of data to work best and overcome challenges such as overfitting, exploding gradient, and class imbalance on the training process (Joshi, Verma, Saxena, & Paraye, 2019; Brousseau, Rose, & Eizenman, 2020; Tian, Xu & Zuo, 2020). The same argument was stated by (Barz & Denzler, 2020) and (Chan and Chin ,2020). Recent research was established to solve these issues, one of the solutions was by using data augmentation in urban environments (Mushtaq, Su & Tran, 2021; Jung, Liao, Wu, Yuan & Sun, 2021; Lella and Pja, 2021). However, limited research was done within the forest environment. Some of the solutions are using RF, distance-based and Deep Learning (DL) methods. The overall performance of these solutions is less than 80% accuracy and has high false alarms rate.

Several solutions were established mainly in urban environments with hybrid approaches such as Convolution Recurrent Neural Network (CRNN), LSTM-CNN, CNN- Support Vector Machine (SVM), and CNN-RF. Whereas a study shown using ensemble methods obtained 85% accuracy on urban rare sound event detection (Phan, Krawczyk-Becker, Gerkmann, & Mertins, 2017). Recent works on SED with hybrid approaches using CRNN with ensembles achieved up to 91% accuracy on the artificial data and urban dataset (Wang et al., 2021). A CNN can act as a feature extractor that improves the feature input for the classifier by processing the original input first. The improved features need to be coupled with a suitable classifier.

Therefore, this research aims to employ a hybrid solution for the SED in the Malaysian forest environment. Implementing the SED with recent technology might improve the effectiveness of wildlife surveillance and protection task.

1.2 Problem Statement

Wildlife reserves are targeted by poachers and loggers to plunder. Preventing them is no easy task due to rough terrains and vast areas (WCS, 2018). The Endau Rompin National Park spans about 450km^2 wide, like many other wildlife sanctuaries. To protect these sanctuaries requires large amounts of resources due to their magnitude. Conventional urban camera-based security systems are limited and will be costly in order to cover the whole large area. The budget allocated for protection is not enough to apply adequate security to every sanctuary (Rahana, 2017). It is evident that loggers too cause destruction of habitats and the deaths of animals. Poaching is a serious threat to national security that requires immediate attention (Miwil, 2017). Detection of intruders is vital in coping with the problem. The sanctuary is a wide area to locate poachers where many often manage to escape. Patrolling the sanctuary is also an expensive and a weary task due to its envireonment and magnitude. Constant monitoring of the sanctuary is unlikely with just camera equipment available in the market. The drawback of using cameras instead of sound is that cameras are limited to a limited direction of sight. At the same time, sound can be more informative as it is not limited to a single direction of sight. To add, camera traps are less effective during bad weather than acoustic monitoring (Browning, Gibb, Glover-Kapfer & Jones, 2017; Crunchant, Borchers, Kühl and Piel, 2020).

Security in vast reserve forests also requires automation to detect poachers intruding on the lands. Sound is a good indicator of poacher's intrusion. SED has shown reliability on isolated sound classification. The SED can detect poachers applying in forest environments. But challenges with the SED approach include the ambient noise and frequency overlapping coming from the forest naturally (Wang et al., 2021; Wisdom et al., 2021). The sound data recorded using Analog-Digital Converter in mics saves data in amplitudes, and these features are highly correlated between classes (Hoshen, Weiss & Wilson, 2015). The sound features required must be tailored to Machine Learning (ML) with high correlation features between classes (Wisdom et al., 2021). It also requires having a distinct pattern between classes. This is evident in producing false positives (FP) predictions. The FP rate of the prediction is also known as the false alarms rate. A low FP rate is required for obtaining a low false alarm rate. Hence, surveillance tasks can avoid wasting enforcement resources on responding to false detection of intrusions. SED in forest environments obtains high accuracy but

faces a high FP rate of more than 20% (Selman & Demir, 2019; Ahmad & Singh, 2019). It was reported that the occurrence of high false alarm rate is due to the variety of sounds with similar features (Selman & Demir, 2019; Lim et al., 2017).

Most of the recent studies show that SED on forest environment employs distance-based such as K-Nearest Neighbour (KNN) and Deep Learning (DL) methods such as CNN (Hershey et al.., 2017; Heittola & Mesaros, 2017; Otter et al., 2018; Ahmad & Singh, 2019; Demir, Turkoglu, Aslan & Sengur, 2020; Mushtaq t al., 2021). Distance-based methods work on a specific sound, but it has less prediction ability compared to the DL methods (Alom et al., 2019; Ganaie & Hu, 2021). However, the DL solutions require large amounts of data for best performance (Joshi et al., 2019; Brousseau et al., 2020; Tian et al., 2020). The lack of specific data available in the forest environments would lead to less prediction performance. The DL can also be overfitted due to limited amount of data and fail to generalize the sound event (Joshi et al., 2019; Brousseau et al., 2020; Tian et al., 2020).

Overall, the issues are as follows:

i. Less efficiency of surveillance with current approaches leads to wildlife reserve intrusion in a forest environment.
ii. Sound events frequency overlapping leads to a high correlation of features between classes (Wang et al., 2021; Wisdom et al., 2021).
iii. High FP rate indicates high false alarm in a forest environment (Selman & Demir, 2019; Ahmad & Singh, 2019).
iv. Lack of sound data in forest environment may impact performance in CNN (Brousseau et al., 2020)
v. The overfitting and failure to generalize occur in CNN (Joshi et al., 2019; Tian et al., 2020).

1.3 Research Questions

The research questions are as listed below:

i. What features extraction method can produce distinct features between classes?
ii. What is the SED technique that is suited for surveillance in the forest environment?
iii. What is the SED approach best suit for surveillance in a forest environment?
iv. Can a hybrid solution provide good performance SED in the forest environment?

1.4 Objectives

The research aims to construct a solution for SED in the wildlife environment for Malaysian forest. In achieving the objective of this research, the specific objectives are:

i. To identify significant patterns of features in sound between classes.
ii. To determine the sound events detection technique that is suitable for surveillance in the forest environment.
iii. To propose a hybrid solution for sound event detection using a real forest environment condition of surveillance.
iv. To evaluate the performance of the proposed solution.

1.5 Scope

The scope of the study is SED on a forest environment within distances within 100 meters between the source of sound event and the recording unit. The sound events data were collected in an environment of a reserved forest area in Taman Negara Endau-Rompin Johor, Peninsular Malaysia. It is located a few kilometres away from the vicinity of the Nature Education & Research Centre located at latitude and longitude of 2.5294807 103.3700159. The time of day when the data was collected is between 9:00 AM to 5:00 PM Malaysia Standard Time GMT+8. The weather during collection is sunny, and the recording areas are under the shades of trees inside the forest. Intrusion sounds collected in the study are limited to three types of sound like the vehicle activity, chainsaw activity and wood cutting activity.

The sound activities can be heard by a typical person within the 100 meters radius. The distances were calculated using a global positioning system with precision

flaws of positive or negative 5 meters. The vehicle activity consists of just a single type of 4x4 pickup truck. The vehicle activity sound includes the vehicle revving and idle. The chainsaw activity source is a 14-inch petrol chainsaw 1000W with an engine capacity of 25 cm^3. The chainsaw activity consists of three states that is revving, cutting wood, and idling. The next activity is wood cutting and it consists of one man cutting a 10-inch-wide stump on the ground. The activity is carried out by an average man between the age of 25-30 striking the stump with a single hand 12-inch hatchet.

The recording device used is limited to a single Zoom H6 hand-held recorder attached with four different microphones recording simultaneously for each sound event session. The microphones are supplied with 12 Volts of phantom power from the recorder. The recording file format is saved as .wav file with 44.1Khz sampling rate. The amount of data used in the study was 31,792 of 5-second activities samples

1.6 Significance

The research is important because it concerns the security of wildlife. Current security are costly, and the study expects to improve surveillance cost effectiveness. SED implementations in intrusion detection allow microphones to be placed instead of cameras, which will save many funds in the future as microphones are significantly cheaper compared to cameras. The system's effectiveness will also bring reliable results to the benefits of sound waves detection instead of camera views of light waves capturing that is ineffective in the dark and camouflage requirements. Lowering the maintenance frequency will also bring a significant change in maintaining costs. After a successful assessment of this approach, the product can be applied in the natural forest area. This study will be able to demonstrate sound-based surveillance on wildlife protection in the forest and that employment of the SED solution assists wildlife conservation agencies and nature conservation authorities with an efficient system. Leading to the protection of wildlife from extinction caused by poachers. Figure 1.0 shows a quadruple helix diagram of the significance.

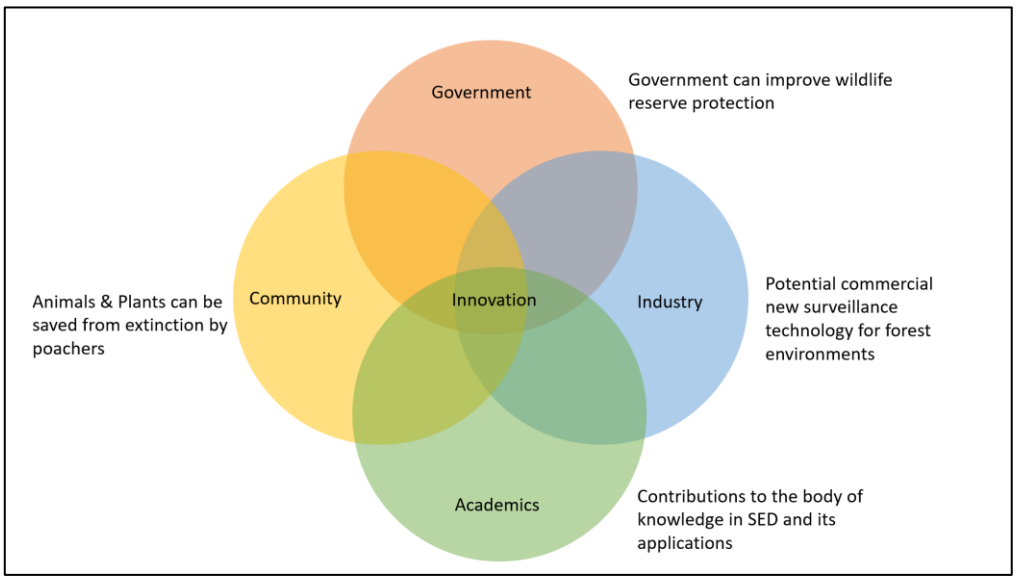


Figure 1.1 Quadruple helix diagram of the significance of the study

1.7 Outline of the thesis

The thesis consists of five chapters including chapter one introduction, chapter two literature review, chapter three research methodology, chapter four results and discussion and chapter five conclusion and recommendation. First, chapter one introduces the research background and problem that is wildlife were hunted by poachers to the point of extinction and they need to be stopped. Hence the intrusion detection using SED was proposed to improve security in wildlife sanctuaries. Next, chapter two literature review covers the literature review of wildlife intrusions in Malaysia, SED approaches using machine learning, Sound and its features, suitable machine learning classifiers for SED and post-processing predictions. Furthermore, Chapter three, elaborates the methodology in four phases consists of data collection, feature extraction, feature analysis, and outline experiments on audio classification. Chapter four illustrates the results produced by feature analysis using heatmaps. Then audio classification models performance for SVM, RF, CNN, and CNN-RF. Finally, chapter five, contains the conclusion of the research and future recommendations.

CHAPTER TWO
LITERATURE REVIEW

2.1 Introduction

This chapter contains the exploration of literature to help understand the depth of the problem and the solution available in recent studies. Firstly, Malaysia's conservation efforts are in practice with the magnitude of the problem and challenges. Next, the basics of SED along with a review on approaches applied in the past. Then, touching on the sound representation by computers and ML features that is suitable. Afterward, on the ML models suitable for SED including SVM, CNN, RF and hybrid models. Finally, the post-processing of prediction to refine the prediction of models.

2.2 Malaysian Wildlife Conservation Efforts

There are several organizations playing roles in preserving wildlife conservation in Malaysia. Those include Wildlife Conservative and Science (WCS) Malaysia and Jabatan Perlindungan Hidupan Liar dan Taman (PERHILITAN). Both government and non-government organizations play their part in conserving wildlife. Some of the efforts done are to maintain population growth in a few aspects such as listing endangered species, counting the animal population, stopping deforestation, habitat protection, creating forest reserves, creating laws favouring wildlife and providing law enforcement. WCS Malaysia is a non-profit organization. It is incorporated under Section 14(2)b of the Companies Act 1965 as a "Company Limited by Guarantee" in Malaysia (WCS, 2020). Such a firm has no share capital and therefore cannot pay dividends. It is a method used by many organizations and operating on a non-profit basis, where any income surplus overspending is returned to charities.

The Department of Wildlife and National Parks is known as PERHILITAN. In 1896, the first law on wildlife was created by the government. Then in 1902, the first wildlife reserve, Chior Wildlife Reserve, was gazetted. They set up as the central agency to coordinate habitat conservation activities and wildlife species.

2.2.1 Malaysia's Wildlife Reserve Recent Issues and Solutions

Wildlife protection enforcement in Malaysia and Southeast Asia regions have confiscated many stolen wildlife in the illegal wildlife trade (Rivera et al., 2021). Wildlife sanctuaries were established to isolate vast amounts of land from human activities that could easily cause ruin to the land (Pei, 2017). Hence, sanctuaries are present for the protection of wildlife. It raises the stake and increases in their selling value. Thus, luring poachers to accept the risk of stealing protected wildlife (Povera, 2019). The detection of intrusion in a wildlife reserve is important to protect the flora and fauna. Hence, there have been numerous efforts to counter intruding poachers (Povera, 2019). Despite many efforts, the perpetrators' illegal logging and poaching activities are still rampant based on the illegal wildlife trade (Shepherd et al., 2020). It has been reported that illegal loggers are still active. Poachers are devastating protected wildlife which is endangering their existence (Pei, 2017). Then in Semporna, Sabah, the wildlife department officers are still hunting those involved in turtle poaching activities (Miwil, 2017). Some countries do not have the resources to stop these types of illegal activities in their forest. According to Davis (2018), some governments in Southeast Asia are offering low investments resulting in the lack of wildlife protection. These intruders destroy and take all they want from the sanctuaries as they please. The protection initiatives are more attentive as the numbers of wildlife species grows lower and even near extinction for some species that reside in the sanctuary. The Sabah Forestry Department favours setting up a dedicated wildlife enforcement team as intruders became more daring in forests and reserve areas (Povera, 2019). The issues address many approaches, but the resources required are high.

There are several solutions for wildlife protection in Malaysia. One of many efforts is to train squads and firearms at priority areas such as Tabin wildlife reserve, Kinabatangan, and Ulu Segama to focus on endangered species like Borneo Pygmy Elephants and Sumatran Rhinos (Inus, 2017). Next, by the PERHILITAN in the operation of (MBEON) in collaboration with the Angkatan Tentera Malaysia, Perbadanan Taman Negara Johor (PTNJ), Perbadanan Taman Negara Perak (PTNP), and Jabatan Perhutanan Semenanjung Malaysia (JPSM) has shown massive intrusion with a total trace of 1,504. Table 2.1 shows a summary of intruders caught from PERHILITAN'S operation. The type of activity were identified into two categories that are poaching and intrusion. Poaching related activities are findings of bullets, snares,

bones (animal remains), traps, and animal organs. Intrusion activities found include tents, food waste, vehicle tracks, tree markings, felling of trees, weapons, logging, local and immigrant trespassers.

Table 2.1
Operation MBEON 2016 Report

Type of Activity	Item	Number of Observations	Total Traces
Poaching	Bullet Shells	1	12
	Snares	49	84
	Bones/Animal Remains	4	8
	Traps	1	2
	Animal Organs	1	2
	Others	1	10
	Total of Poaching	57	118
Intrusion	Tents	186	523
	Tree Marking	198	631
	Food Waste	20	50
	Felling of trees	7	27
	Left Laundry	5	7
	Cutting of small trees	49	82
	Other Equipment	17	31
	Weapons	1	2
	Vehicle Tracks	1	3
	Human Tracks	3	9
	Logging	4	5
	Detaining Thailand Immigrant or locals	1	2
	Stumbling upon Locals	6	14
	Total Intrusion	498	1,386
	GRAND TOTAL	555	1,504

Note: data.gov.my, 2019

2.2.2 Challenges of Intrusion Detection in Wildlife

Challenges in implementing security in remote areas require special equipment and designs to endure the conditions of a rainforest (WCS, 2018). Mainly, the power supply is no issue in urban areas as power is supplied directly from the power grid. In remote areas, it would be much more costly to conduct surveillance and communications. Lowering the cost required for the operation is imperative. Electricity source is limited from batteries in these scenarios. Thus, reducing power requirements in devices for operation can improve reliability. Figure 2.1 shows a Google map highlighting Taman Negara Endau Rompin.

Figure 2.1 Google Map View of Endau Rompin National Park

The Endau Rompin National Park is located with the Endau and Rompin rivers flowing through the park. It is the second biggest national park in West Malaysia, covering about 489 km^2 with approximately 26 km of jungle trails (Nordin, 2019). To conduct surveillance over 489 km^2 of land is very expensive and challenging, especially in the forest environment.

Wildlife reserve intrusion sound detection variables include illegal activities done by poachers. Poachers' equipment detained are mostly axes, machetes, and tools for extracting the loot (Zolkepli, 2019). Thus, it can be assumed that the sound of unauthorized vehicles and cutting trees could indicate an intruder. These sounds can be the solution to detect incoming threats. The activity of cutting down a tree with a large axe can be heard a kilometre away (WCS, 2018). Poachers commonly produce these sounds as they are constantly collecting resources for their camping needs. Reported that there were many abandoned poacher campsites in Malaysian forest reserves (WCS,

2018). These campsites were abandoned as poachers are always on the run from getting caught by authorities. These sounds could be a good indicator in detecting the presence of poachers in a reserve forest. But the forest is a huge piece of land to be on the lookout for these sounds constantly.

2.3 Sound and its Features

Sound is the waves that vibrate through a medium commonly as air from one place to the other. Humans perceive sound using ears, by frequencies of 20Hz to 20kHz. The difference in frequency changes within a certain time can give a meaning such as the pronunciation of words. Understanding word pronunciation is a more challenging task compared to detecting how a truck sounds. Truck sounds are very common to make engine rattling like sounds that stay within a specific frequency range and are sustained longer and do not require accurate timing to understand that a truck is making noise, proving its presence. The truck scenario is like SED, while words are speech recognition tasks. SED is considered easier compared to the complexity of speech recognition. SED can also be complex based on the domain and specific sounds to detect, especially when noise interference comes into play. Computers are not humans; hence, we need to understand the perception of sound on computers to teach an artificial intelligence model effectively. The following section explains the sound representation from a computer's point of view.

2.3.1 Computer Sound Representation

Sound is a variety of changing vibrations on a medium such as air pressure in changes transmiting sound waves. To detect sound, humans can hear using their ears detecting different frequencies. For computers, recording uses an Analog-to-digital Converter (ADC) to detect changes of sound waves into a digital waveform on microphones. Sound waves are represented in a digital recording in an array of numbers that is in amplitudes over time. These amplitudes convert into the frequency domain by using FFT and separating the frequencies within the digital sound amplitudes. On the frequency domain, we can observe the frequency intensity of the sound data.

Figure 2.2 shows the illustration of sound representation in three dimensions, time, frequency and amplitude (Doshi, 2019).

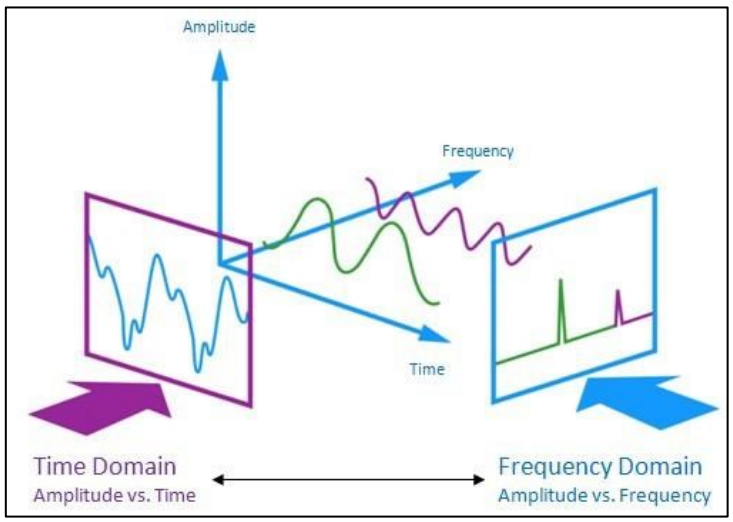


Figure 2.2 Sound Representation in Three Dimensions

The frequency domain, known as the spectrum, contains information on every frequency intensity that belongs to different sounds within the sound time frame. This effort improves the dimensionality and quality of the input sound data for ML purposes by perceiving sound by the frequency components. But even frequency tends to overlap on similar sound types leading to a high correlation which is bad for pattern recognition. The human ear perceives sound frequency in a non-linear manner. There is evidence that using sound representation mimicking the human perception of sound can improve pattern recognition. A recent study on SED employs the Mel-scale filter bank energies, also known as the Mel-log Energies (MEL), demonstrated good results (Yan, Song, Dai & McLoughlin, 2020; Wu & Lee, 2019; Lu, Duan & Zhang, 2018). The Mel-scale perceives more discriminative features in the lower frequencies while less focus on a higher frequency (Roberts, 2020). Applying the Mel-scale filter banks converts the frequency domain into human-centric perceived sound data. Hence the use of MEL is highly recommended for SED tasks.

2.3.2 Mel-log Energy Feature

MLE is a type of audio feature extracted from an audio file (Dilber, 2016). The features are also called log Mel-scaled spectrogram or log Mel filter bank energies and MLE by different researchers. It is one of the many sound feature extraction methods in the second step in SED (See Figure 2.2). I t is considered that MLE is effective in detecting rare sound events by a previous study involving human activity detection in a typical house for emergency response with up to 90% accuracy in detecting emergency

sound events (Kim, Min, Jung & Chi, 2020). The use of MLE features in SED has shown 93% accuracy on rare SED on the urban scene, such as glass breaking, baby crying, and gunshots (Lim et al., 2017). MLE produces a good visual representation of the sound as the feature goes through the FFT, which decomposes the signal frequencies. Sounds of intrusions have their respective frequency as musical instruments produce different frequencies. The Mel-scale filter banks help the frequency representation as humans perceive sound. Log at the powers of each Mel scaled frequency increases the visibility of changes in the cestrum.

Sound feature extraction has two forms that are spectral features and rhythm features. Spectral features are commonly used in SED for its frequency consciousness. The cepstral features are computed by taking the FFT of the warped logarithmic spectrum. They contain each spectrum band's rate of change (Rao, Kim & Hwang, 2010). Due to their ability to distinguish the influence of source and filter in a speech signal, the cepstral characteristics are advantageous (Gupta, Bansal & Choudhary, 2019). In other words, in the cepstral domain, the source and filter of sound are separable. Formant filters help in the detection of vowel speech sound for speech recognition tasks.

The commonly used in ML spectral features are MLE and MFCC (Gupta et al., 2019). MFCC is popular in speech recognition, but the MLE is popular in SED. The MLE is selected as it has been shown viable when paired with the DL techniques for SED. Next, MLE has variables to optimized to improve quality. The MLE feature extraction parameters:

- Input Sound signal rate (Quality in kHz)
- Frequency range (Min & max Frequency)
- Window type
- Window or Frame Time length (milliseconds)
- Hop Time 0.01 (milliseconds)
- Window Size
- Hop Size
- Number Filter Banks

The signal sampling rate originates from the sound recording source. The sampling rate consist of how fine the sound is captured. In theory, the higher the kHz value used, the better the sound quality will be. It is due to more data chunks used to

describe the analogue waveform. The sampling rate for typical consumer applications are:

- At 8 kHz, Talking, audio books, etc.
- At 22 kHz, digital-analogue mono recordings like vinyl record and cassettes
- At 32 kHz, music or radio station
- At 44.1 kHz, commonly used at Audio Compact discs and most consumer-level songs with file formats such as Moving Picture Experts Group (MPEG) -1 Audio Layer III (MP3), Augmentative and Alternative Communication (AAC) and Windows Media Audio (WMA)
- At 48 or 96 kHz, used on high-definition professional recording tools.

Higher frequencies may not define the quality of sound as higher frequency ranges are undetectable by human ears. Under normal circumstances, human hearing is in the frequency range between 20-20kHz (Smith, 1997). It is suggested that inaudible frequencies can negatively affect sound quality. The frequency range is the minimum to a maximum frequency range which can be tuned to the relevant target sounds. Humans commonly can interpret sound in the frequency range of 20Hz to 20 kHz. Humans can detect an intrusion well enough within this range of frequency. Thus, the range can emulate the same result (Reynolds, Kinard, Degraff, Leverage & Norton, 2010).

To target a sound source specifically, searching in a specific range for the desired target is imperative. The selection of voice frequency of 85 Hz to 180Hz used by (Re, O'Connor, Bennett & Feinberg, 2012) demonstrates a specific target. Windows type is how each frame can be processed before the FFT step. When performing a Fourier Transform on-time data and transforming it into the frequency domain, several different windows are used to minimize spectral leakage. Each window designs with a specific purpose. The windows used are Hamming, Flattop, and Uniform. Hamming windows are recommended in this experiment for the properties it provides for frequency-selective analysis (SIEMENS DSP Community, 2019).

The window or frame length in milliseconds determines how finely the features should be extracted. Smaller windows will give more features in a second of sound input. Best practices recommend 20-40ms for 16kHz source resulting in 320-640 per second. Sampling in much smaller windows will not give more meaningful features and will waste more resources. The window needs to have a minimum duration to include lower frequencies and that the maximum duration is limited to have stable sounds. The window or frame length in milliseconds determines how finely the features

overlap between features. Overlapping is important, events near or at the boundaries will be severely impaired, and the risk that transients are between windows or not isolated in a single window is increased, thus decreasing how well the MLE can classify different inputs. A traditional window feature, such as Von Hann or Hamming, can be very lossy at both ends of each window. (Liu & Zagzebski, 2010). Equation 1 is used to calculate the overlapping percentage of a frame with its succession. The percentage should indicate how much data merges in between frames. The parameter adjustment will allow fine or coarse data extraction that could affect the performance of the predictions.

$$Overlapping\ (\%)\ = \frac{Window\ Length}{\text{Window Hop}} \tag{1}$$

2.4 Sound Event Detection (SED)

Sound event detection or also known as SED is the recognition of a sound event's presence using artificial intelligence. It seems appropriate to apply SED to detect the intrusions for wildlife reserves as the poacher's activities sounds are very distinctive. Figure 2.3 describes the block diagram of the most basic form of SED known as audio recognition (Subramanian, 2004). The diagram shows the basic structure of a SED pipeline or processes.

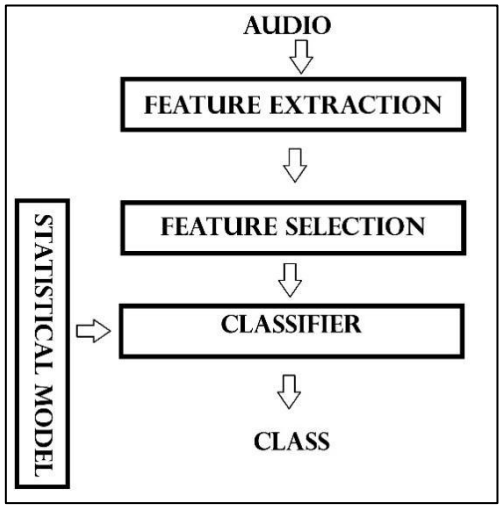


Figure 2.3 Basic Block Diagram Audio Recognition
(Subramanian, 2004)

The audio in Figure 2.3 is sound input which is next applied a feature extraction process to acquire sound features that can be done in many ways including frequency domain features or time domain features depending on the requirements. The features selection step is to allow fine tuning before it is to be used as an input for the classifier. The classifier can be of any statistical model or a Neural Network (NN) to infer the features (Subramanian, 2004). The final step is the output of classification for the input signal.

2.4.1 Sound Event Detection Types

There are two common types of SED approaches namely monomorphic and polymorphic. In the present study, monomorphic SED will be employed as it is suitable to detect sound events for security. Figure 2.4 shows Monomorphic Sound Events Detection Approach (Heittola & Mesaros, 2017). Monomorphic audio events detection is an approach to finding the most prominent audio event from each instance. This approach ignores the insignificant events and showing only the main event that is the loudest and clearest.

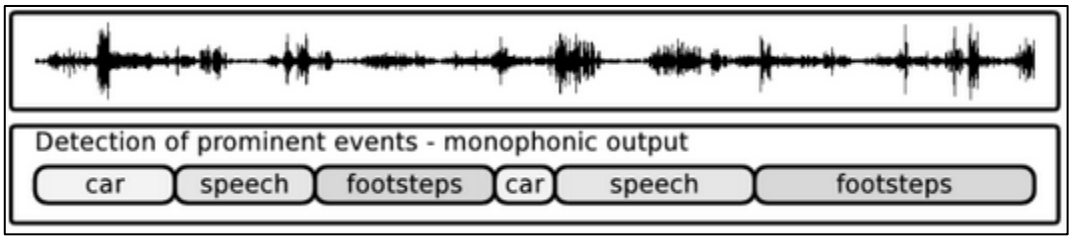


Figure 2.4 Monomorphic SED (Heittola & Mesaros, 2017)

Figure 2.5 shows Polymorphic SED (Heittola & Mesaros, 2017). Polymorphic detection is described as the detection of overlapping audio events. The output of this approach will produce a multi-event sequence output for each instance. This approach is more in-depth and detailed looking into the background of the events to identify simultaneous events. Hence details could be used for applications to determine the situation of such a combination of audio events.

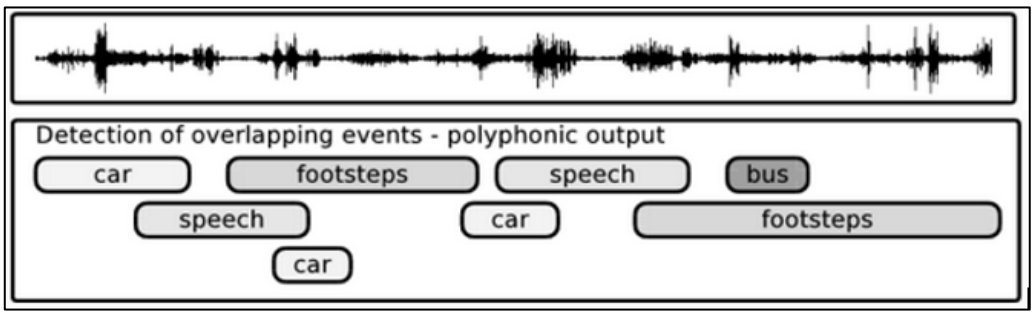


Figure 2.5 Polymorphic SED (Heittola & Mesaros, 2017).

The monomorphic SED approach detects the most prominent sound event while polymorphic SED finds the overlapping events. Therefore, monomorphic SED is adequate for surveillance between the two SED approaches since a single existing event can identify the intrusion sounds. For polymorphic SED, multiple events overlapping is unnecessary complexity for the research goal. Hence, the study believes that employing a monomorphic SED approach should be adequate for the research.

2.4.2 Previous Forest Environment SED Approaches

SED applications in the forest environment are found to be limited, recently implemented methods are distance-based algorithms such as K-Means Clustering, SVM, CRNN, CNN and RF(Singh et al., 2020; Tzirakis et al., 2020; Selman & Demir, 2019; Ahmad & Singh, 2019; Liu et al., 2019). The research was to cater the detection of a specific sound event including elephant calls, Borneo gibbon sounds, cutting of trees and gunshot. The common ground of the studies done was to detect a specific event within a forest environment scene. Sound events vary in length for detection from 1 second to 10 seconds and are specified to ideally contain the events within the time frame for detection (Selman & Demir, 2019). The features used were mostly tailored to the specific sounds nature in frequency range and applies the common FFT later into Mel spectrograms, MLE and Mel-frequency Cepstral Coefficients (MFCC). Table 2.2 shows a list of recent studies that were conducted with sound events in the environment of a forest. The Sound Event Length column states the observed sound length to detect their respective event. The Features column indicates what sound features were extracted for the study. The Task column explains the objective target detection task. Finally, the Results column shows the obtained performance.

The survey of recent research has indicated the practicality use of Mel scaled features that are MLE, Mel Spectrogram and MFCC. The methods that were most

successful were found to be closely related to the task at hand. The nature of long sound events with lengths above 1 second are mostly solved using a DL technique including CRNN, CNN, and FCN. While short sound events of lengths below or equal to 5 second applied with SVM and KNN.

The recent review of recent approaches of SED in a forest environment is found to be using spectrograms that are a frequency domain feature extracted from audio. These features prove to be useful for classification. Hence the study believes that frequency-based features including the MLE are suitable for this niche environment. Next, the classification method used previously shown DL methods, SVM and Random Forest SED compared to distance-based methods considering the overall performance with FP rates. Therefore, these methods should be considered as a possible solution for forest environment SED. Thresholding can reduce FP rates and may also result in reduced performance scores that show the actual performance ((Ahmad & Singh, 2019). Thresholding should be applied to improve FP for specific surveillance applications for this research purpose

Table 2.2
Forest Environment Research Summary

Author	Sound Event Length (Seconds)	Features	Task	Method	Results
Selman & Demir, 2019	5-10	Tailored targeted frequency spectrogram features.	Elephant Call presence	Pre-segmentation + Conv1D Layers With RNN-LSTM	Accuracy 90% F1 Score 0.69 Due to over predicting detections. (High FP rate)
Ahmad & Singh, 2019	5	Tailored targeted frequency spectrogram features.	Axe Tree Cutting in between ballon pop hammer, digging, and clap sound in dense forest	Gaussian Mixture Model GMM K-Means clustering PCA (80% matching threshold)	Accuracy 92% and 5% FP
			Axe Tree Cutting in between ballon pop hammer, digging, and clap sound in an open forest		Accuracy 76% and 21% FP
Liu et al., 2019	1	MLE	Chainsaw presence	CNN (VGG16)	Accuracy 81.3% F1 Score 0.806
				Fully Convolutional Networks FCN (VGG16)	90.1% F1 Score 0.898
Tzirakis et al., 2020	5	Mel Spectrogram	Borneo Gibbon Presence	CRNN	Accuracy 93.3%
				RF better than SVM and KNN	Accuracy 84.8%
Singh et al., 2020	0.1 – 1	low-level Discrete Wavelet Tempo (WVT) - based features with bag-of-words approach	Gunshots and burst shots.	SVM	Accuracy 96.04% Receiver Operating Characteristics (ROC) 0.9866 (Low FP rate)

2.4.3 Previous Urban Environment SED Approaches

Several researchers performed SED with a variety of approaches. SED can be applied in many fields for automation and will soon be ready for application (Heittola & Mesaros, 2017). Research outcomes previously showed the use of a decision filter helps reducing false positives and improving accuracy. SED approaches are organized by methods covering SED tasks into three components, feature extraction method, classification algorithm and final decision filter. SED approaches are important to understand by comparing a collection of approaches to solve a particular case based on a specific dataset. The idea of a surveillance system is the detection of unwanted events in a specific area. Surveillance using audio is believed to be possible as recent study implies. The DCASE 2017 Tampere University Technology (TUT) Rare Sound Events 2017 set up the development of a dataset to challenge researchers from all over the world (Heittola & Mesaros, 2017). The challenge was to detect three rare sound events within the audio. Many researchers are taking part in solving this challenge and producing different results with their respective configurations on solving the matter.

The rare sounds were three classes of babies crying, breaking glass, and gunshots. The datasets for training and validation uses software to combine the recorded isolated audio and recorded background noise from many unique sources of each class. Table 2.3 shows the statistical summary for the datasets. The methods in solving this challenge were collected to compare configurations done to find the best practices for SED. Composition of audio materials sound events which are known are as the following classes:

- Baby crying (106 instances for training, 42 instances for the test)
- Glass breaking (96 instances for training, 43 instances for the test)
- Gunshot (134 instances for training, 53 instances for the test)

Table 2.3
DCASE 2017 Dataset Statistics

Class	**Baby crying**		**Glass Break**		**Gun Shot**	
Usage	Train	Test	Train	Test	Train	Test
Mean	2.41	1.85	1.36	0.72	1.43	1.04
Max	5.10	4.24	4.54	0.18	4.40	3.68
Min	0.66	0.78	0.26	0.30	0.24	0.30
Median	2.33	1.67	1.29	0.70	1.21	0.76
Standard Deviation	0.98	0.83	0.75	0.30	0.86	0.83

Note: Sound Event Detection in the DCASE 2017 Challenge (A. Mesaros et al,2019)

The length of the target scenario is generally brief relative to the 30-second background noise (Mesaros, Heittola, & Virtanen, 2016). Training and testing datasets are produced, respectively. Combinations are produced by software package with source information, including ambient noise and specified occurrences (Wu, Mao, & Li, 2017). The official metric in the DCASE 2017 Challenge on Error Rate (ER) is an insertion, deletion, and substitution rates (Jayalakshmi, Chandrakala & Nedunchelian, 2018). The evaluation involves the calculation of true positive (TP), FP, and false negatives (FN). If the system's output correctly predicts the presence and onset of a case, it is calculated as TP. Only when it is predicted within 500 milliseconds(ms) of the actual starting time is it detected as accurate (Lim et al., 2017). FP demonstrates that if there is no event, the system wrongly detects the existence of an event. If the output of the strategy misses the event, an FN can be considered.

Traditionally, SED's popular approaches were Gaussian Mixture Model (GMM) and Hidden Markov Model (HMM), but they have moved to the new deep learning-based methods (Ahmad & Singh, 2019). The modern SED approach includes Deep Neural Network (DNN), Recurrent Neural Network (RNN) and CRNN (Kaiwu, Liping, & Bin, 2017). The observation on configurations of past research based on the DCASE 2017 Challenge compares their respective performance, feature extraction method, classification algorithm and decision filter composition in table 2.4. The features extraction method is important as it chooses sound features that load into the classification algorithm. Destructive features will inevitably lead to bad performance. The column Classification method denotes the detection technique used by the

researcher. The column Decision-making refers to the verification step as in the SED Pipeline that determines to accept or reject the predictions. Additionally, F1 multi-class inconsistency shows the deviation of each class performance, a higher deviation indicates that the model may be having trouble detecting on certain sound type than the rest.

It is found that the method of a 1-Dimensional CNN, RNN and LSTM units with an average accuracy of 93.1% was found suitable to work with (Lim et al., 2017). However, the other methods used in SED were CRNN, LSTM, DNN, MLP, RNN, CNN, non-negative Factorization Matrix (NMF) and Bidirectional-LSTM (Bi-LSTM). They have shown less F1-Score, as demonstrated in Table 2.4.

Table 2.4
Summary of SED Composition Conducted on DCASE 2017

References	F1-Score	F1 Multi-class Inconsistency	Feature Extraction method	Classification method	Decision Method
(Lim et al., 2017)	93.1	4.1	MLE	CRNN-LSTM	thresholding
(Cakir & Virtanen, 2017)	91	3.7	spectrogram	CNN	majority voting
(Phan et al., 2017)	85.3	3.6	MLE	CRNN	median filtering ensemble
(Ravichandran & Das, 2017)	84.2	10.6	MLE	MLP CNN	thresholding
(Zhou & Feng, 2017)	82	0.9	Log-gammatone cepstral coefficients	Tailored-loss DNN with CNN	filtering by median
(Kaiwu et al., 2017)	83.9	13	MLE	CRNN	majority voting
(Vesperini et al., 2017)	79.1	10.2	MLE and MFCC	CNN, MLP, and RNN	filtering by median ensemble and hard thresholding
(Dang et al., 2017)	78.6	8.3	Energy spectral centroid pitch, MFCC and Zero Crossing Rate	ensemble	threshold
(Kaiwu et al., 2017)	73.4	20	MLE	DNN	filtering by median
(Ghaffarzadegan, Salekin, Das, & Feng, 2017)	74.2	17.9	spectrogram	NMF	filtering by moving average
(Cakir & Virtanen, 2017)	69.8	11.6	DNN(MFCC)	Bi-LSTM	top output probability
(Heittola & Mesaros, 2017)	64.1	14.2	MLE	MLP	filtering by median
(Jeon & Kim, 2017)	65.8	16.5	NMF source separation extracted MLE	MLP	filtering by median

*Note. Adapted from DCASE 2017 Participations Results (DCASE, 2017)

In conclusion, the review finds that the configurations in solving SED include feature extraction, classifiers and decision-making used on SED tasks. It was found that a viable approach consists of using MLE features in a CNN Model with decision processing by thresholding is considered to work well on SED. Although these are the results done on an artificial dataset, they can steer the study on SED for a real dataset.

2.5 Machine Learning

Machine learning (ML) is a subset of artificial intelligence algorithms capable of learning autonomously by feeding data (Bertolini et al., 2021). Recently ML has been developed to address challenges in many subdomains to improve quality of life (Mehrabi et al., 2021). ML identifies patterns within data based on their similarities toward a specific group of data to produce a rule for predicting an expected classification or outcome. Basic ML algorithm types include supervised and unsupervised (Sarker, 2021). Supervised learning maps an input to output as pairs to create rules for unique inputs to guess the output (Carcillo et al., 2021). Unsupervised learning finds similarities between data and creates rules to identify common patterns input (Gutiérrez et al.,2021). Supervised learning methods are suitable for the research tasks. Supervised learning techniques that are prominent in the niche area of study in SED by previous recent research were SVM, CNN, RF, and hybrid algorithms. Further review of each individual technique was explored in the study

2.5.1 Support Vector Machine

Support vector machine, SVM is a supervised ML technique that has been well established for binary problems. The premise of SVM prediction uses a separation boundary or space in which the common features are located for each input feature individually. SVM maximizes separation boundaries of the data points or features on the defined labels. The boundaries are used to separate the class and create a prediction with all the input features. Figure 2.6 shows the separation of boundaries via the kernel function of linear. The common kernel function determines the effectiveness of class separation in SVM kernels are linear. X_1 AND X_2 represents a feature of an input. Figure

2.6 (a) is a binary separation meanwhile Figure 2.6 (b) is a multi-class separation.

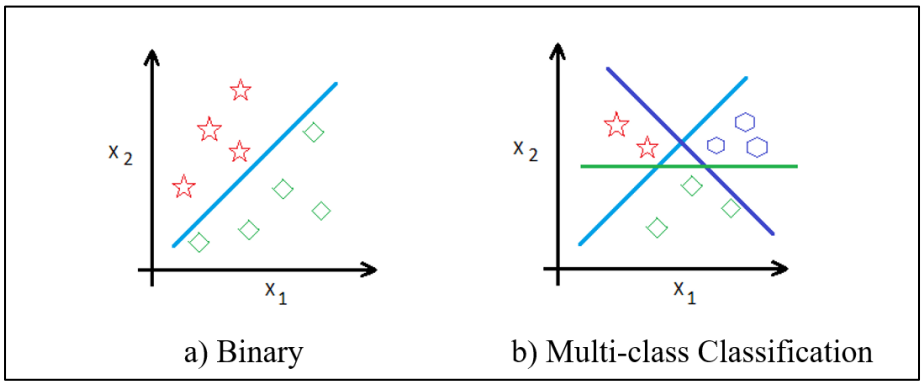


Figure 2.6 SVM Class Separation

The use of SVM in SED or sound classification is still relevant in 2021 based on recently researchers having implemented SVM in audio classification for construction sites (TMaccagno et al., 2021) and detecting violin bowing technique and emotion (Alar, Mamaril, Villegas & Cabarrubias, 2021). It shows the growing interest of SVM in SED implies the consideration of SVM qualities in SED (Liu & Li, 2020). SVM applied on rare event detection of whales using noisy hydrophone data has achieved up to 98% accuracy by using tailored cestrum data done by (Sattar, Driessen, Tzanetakis & Page, 2020). It was still considered to be relevant on SED based on a recent review on SED approaches (Chandrakala, Venkatraman, Shreyas & Jayalakshmi, 2021). The common distinct patterns in MLE features of each class individually can be applied to SVM. The pattern distribution is believed to be able to spot the major differences between classes effectively.

SVM strengths include a low risk of overfitting even with high dimensionality data (Pisner and Schnyer, 2020). In recent studies, involving balanced and imbalanced data, SVM has performed well with high accuracy (Thanh and Kappas, 2018). SVM weakness that should be considered is the training time for SVM is based on the data and can be very challenging for huge amounts of data (Cervantes et al., 2021). Then, SVM is a computer intensive algorithm that requires dynamic hardware for implementation (Cervantes et al., 2021).

2.5.2 Convolutional Neural Network

CNN is a type of Artificial NN. CNN is widely used in image recognition for its good performance (Otter et al., 2018; Hershey et al., 2017). The use of CNN on audio classification is still relevant in 2021 as researchers achieved great results (TMaccagno et al., 2021). CNN on SED for surveillance purposes is most relevant based on (Pandya & Ghayyat, 2021) as applied to recognize unknown acoustics events of a residence. Hence the application of CNN must be considered to allow impactful results for SED growth.

CNN were inspired by animals' biological visual cortex design to replicate the visual understanding of the restricted region of the visual segment overlapping in covering the entire view. The components that make up a CNN architecture can be customized to suit a certain problem. The CNN components to be tuned are the layer size, layer amount and kernel size. Conducting numerous CNN designs could allow proper optimization for the intended model. A more prominent CNN will take up more computation power. The CNN called VGG16 proposed by (Simonyan & Zisserman, 2015) from the University of Oxford trained millions of images from the ImageNet database. ImageNet can identify 1,000 objects in an image with 92.7% accuracy. The VGG16 uses 13 layers of CNN and 2 Fully Connected (FC) or Dense layers and 1 SoftMax layer. Figure 2.7 and Figure 2.8 show the visualized CNN architecture (Simonyan & Zisserman, 2015). This design was intended for detecting 1,000 objects with an input of 224 x 224 with three channels of the colour image. The design can be used for reference in building a custom CNN model. The common design implemented in the VGG16 model uses 4,096 dense layers (fully connected layers) before the output layer of 1,000. The dense layers for smaller classification output should be around the Dense layers and output layers ratio. Another reference can be made on linearly smaller convolution starting from 112, 56, 28, and 14. These commonalities can be used as a baseline for custom build CNN models. The study can use best yielding practices to produce a similar effective model.

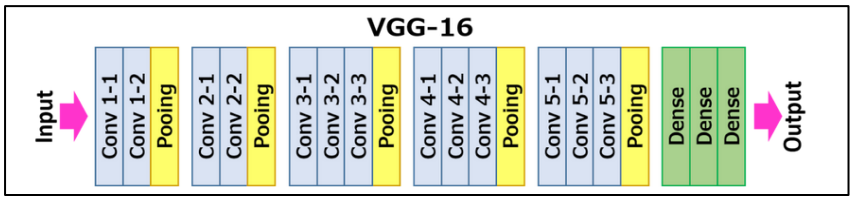


Figure 2.7 VGG16 CNN Flow Design

Note: (Muneeb ul Hassan,2018) VGG16 – Convolutional Network for Classification and Detection Retrieved from https://neurohive.io/en/popular-networks/vgg16/

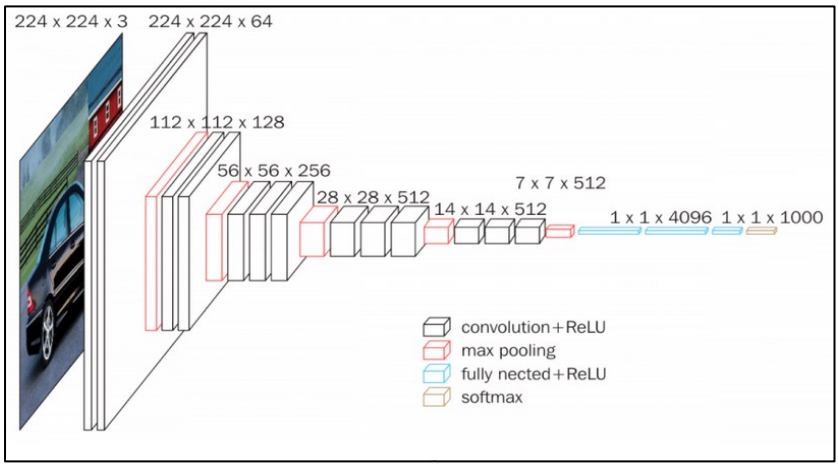


Figure 2.8 VGG16 Illustration of Layers

Note: (Muneeb ul Hassan,2018) VGG16 – Convolutional Network for Classification and Detection Retrieved from https://neurohive.io/en/popular-networks/vgg16/

The creation and training of a CNN network take a lot of effort and data. CNN has stability issues if not trained with enough data (Tian et al., 2020). As great as the results can be, it also requires good data labelled in the millions to achieve great results in a certain domain. The scarcity of data is the main issue when it comes to specific domains. As an alternative, transfer learning is a way to use a pre-trained model on other tasks. The trained and proven VGG16 model can be used without the inferencing segment and just as a feature extractor layer. The convolutional layers act as feature extractors to find meaningful data from the input layer. Transfer learning is compelling and has produced good results in improving input features.

The idea of recognizing the patterns for image identification can apply to sound representations (Hershey et al., 2017). Input image uses convolution layers for in-depth feature extraction (Khoshdeli, Chong & Parvin, 2017). Another CNN strength is offered good pre-processing layer as it can fine-tune input features into more meaningful input features. Previous studies used CNN feature extraction showing

significant improvement in performance from their predecessor (Brousseau et al., 2020).

The state-of-the-art CNN Architecture Visual Geometry Group (VGG) for image classification can identify overlapping images effectively with high accuracy. It seems that the approach of detecting images can be used for detecting overlapping sounds. Hence, MLE as the image input is expected to produce good results. Sound representation using MLE features results in a visual representation that is very distinctive to their respective sound. By representing the sound as an image, the CNN can learn the image of such an event we intend to find. The CNN is found very suitable in the classification step in SED (see Table 2.3). The application of CNN in SED produces good results in previous studies (Cakir & Virtanen, 2017). However, CNN results can vary between designs of layers, a proper design of CNN will result in a better model. Hence the various CNN construction should be tuned for better performance.

CNN is powerful, but it requires a well-optimized model to perform well. Overfitting can always occur when the training dataset is small, and typically all focused domain problem faced with data scarcity. But overfitting can be overcome using a method of early stopping, in which the model stops training before it overfits the current training data. It can be said that tuning hyperparameters leads to significant improvements and success and is also very time-consuming (Chen and Kyrillidis, 2019). The Adaptive Moment Estimation (ADAM) is a stochastic optimizer and has worked well with empirical results with little tuning (Kingma and Ba, 2014; Bock and Weiß, 2019; Jiang, Hu, Chandra, Wang & Zhang, 2020). Performing a hyperparameter tuning would improve the model performance and reducing overfitting.

Regularization by dropout is normally around 0.20% to 0.50%, a higher dropout rate may yield under learning issues by the model (Radhakrishnan, 2017). The gradient descent defines where the trained model and induced by the hyperparameter learning rate in which, if too big, may lead to not converging and missing intermediary learning points as seen in the Figure 2.9 (Radhakrishnan, 2017). Figure 2.9 shows the points where the training session will point in the loss between training epochs. High values in learning rate might miss the lowest point of the descent and end up with an overfitting model. The well fit model can be seen in the descent of the loss function computed on each training epoch. The commonly used loss function on the deep learning technique

is the categorical cross-entropy. The equation (2) describes the Cross entropy, CE.

$$CE = -\sum_{i}^{C} t_i log(s_i) \tag{2}$$

Where ti and si are the ground truth distribution and the CNN prediction confidence for each class i in CNN,

The lower the loss indicates a better model by large learning rate value will result in skipping the optimal model training point with the lowest loss.

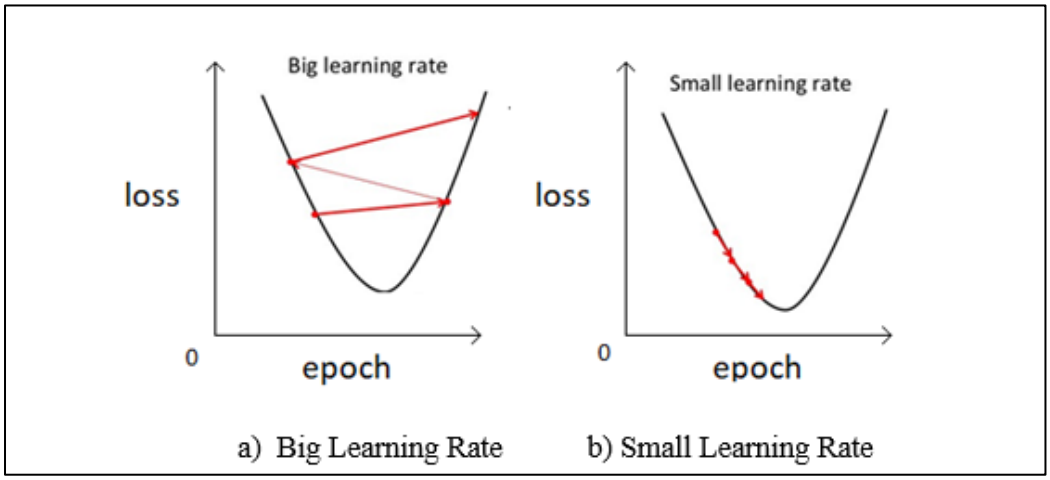


Figure 2.9 Epoch Loss Over Time for a) Big Learning Rate and b) Small Learning Rate

A small learning rate is relevant to find the lowest loss, but it takes longer to train due to smaller steps to converge the model. Decaying the learning rate can reduce the time taken for training for the early stages with a more significant learning rate. The learning rate will reduce exponentially in each successive epoch. It is recommended to use a decaying learning rate to improve training efficiency (Radhakrishnan, 2017). The decaying rate, also known as momentum, can drastically change the outcome of a model (Chen & Kyrillidis, 2019).

2.5.3 Random Forest

A random forest implements a collection of independent prediction trees on random vectors. The benefits are high accuracy, good generalization, and swift classification time. The RF algorithm is considered among the best classification algorithms (Liu & Li, 2020). Figure 2.10 shows the flow chart of a random forest model. The instance in Figure 2.10 refers to the input data for a certain prediction (Koehrsen, 2017). The model consists of multiple DT that will take a partial segment of the input to produce an output class result. The collection of trees will each individually produce a predictions. The Majority-Voting phase will count the most class prediction from all the trees. The Final-Class is the prediction of the most voted class from the Majority-Vote as the RF prediction.

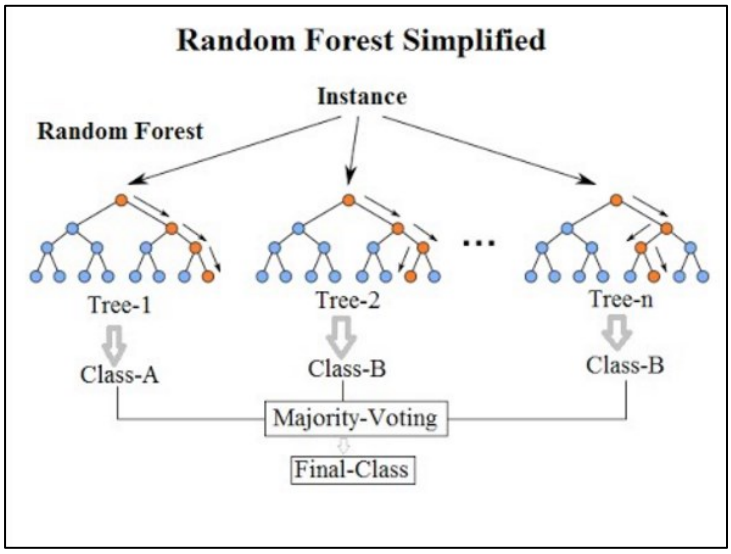


Figure 2.10 Random Forest Simplified

Note: (Venkata Jagannath ,2017) Random forest diagram Retrieved from https://upload.wikimedia.org/wikipedia/commons/7/76/Random_forest_diagram_complete.png

The model has been repeatedly proven successful in the ML task and is considered the algorithm most important in the field (Yeşilkanat, 2020). The model can understand deep feature relationships in complex data (Seifert, 2020). The RF model is known for its high accuracy on ML on image classification tasks (Sarica, Cerasa & Quattrone, 2017). RF employs on SED involving underground pipeline damage detection cause by construction has been successful with up to 95% accuracy (Liu & Li, 2020). This method uses random subset samples, creating unique individual trees

to cope with unstable data, noise, and outliers expected to exist in sound data (Kumar, 2019). RF compared with NN, Naive Bayes and DT on sound-based transportation mode detection has achieved the best performance of 82% accuracy and getting 92% with a mixture of sound and motion features (Richoz, Wang, Birch & Roggen, 2020). Another fact, RF can handle big data with large feature counts (Subudhi, Dash & Sabut, 2020). The previous studies done implies RF is suitable for SED in forest environments in a multitude of scenarios.

However, the RF algorithm can perform worse if not well optimized (Brieuc et al., 2018). The required number of decision trees in RF is different depending on the type of data. A more significant number of trees need more computational time to process (Singh, Halgamuge, & Lakshmiganthan, 2017). RF may not be suitable for real-time or time-critical tasks if the computational time required was found to be high for the specific task (Chen et al.,2021). Although RF has some limitations, it can still be beneficial due to its good performance in the previous ML solution. Considering both the strength and limitations of RF, the study believes the algorithm is suitable.

2.5.4 Hybrid Convolutional Neural Network

A hybrid model is a combination of two or more artificial intelligence techniques to improve the performance of the system. A hybrid model can include statistical and artificial intelligence techniques to predict more effectively (Carlos, Silvana, Juan & Florentino, 2013). Each method has its strengths and weaknesses. The combination of strengths between models will improve and balance out weaknesses to produce better results. CNN has been integrated into many hybrid models that have proven its strong feature expression capacity to improve the input data quality before classification (Yu, Zhang, Xu, Dong & Zhangzhong, 2021). Hybrid CNN are becoming a common approach for many types of solutions showing improved results based on recent studies, including CNN-RF, CNN-KNN and CNN network-gated recurrent unit (CNN-GRU) (Bayoudh, Hamdaoui & Mtibaa, 2021; Nigam & Srivastava, 2021; Geetha, Thilagam & Padmavathy, 2021; Yu et al, 2021).

CNN has been proven many times before to produce promising results on image classification tasks (Otter et al., 2018; Hershey et al., 2017). Meanwhile, RF is used in

many pattern recognition detection solutions (Sarica et al., 2017). The methods are expected to significantly improve SED in the context of this study and its limitations. The CNN portion of the hybrid model acts as a feature extraction layer improving the input data quality. The features extracted automatically optimized to be the most useful in classification. Previous studies used CNN feature extraction showing significant improvement in performance (Brousseau et al., 2020). Hybrid versions of CNN have been proven to improve the results compared to the standalone CNN model (Yu et al, 2021).

Table 2.5
Forest Environment Research Summary

Author	Task	Technique	Results
Maria, et al., 2022	Cauliflower Disease Recognition detection	VGG16 Transfer Learning	Best with RF compared to other ML techniques
Kumar et al., 2021	Wink-based EEG detection	VGG16+RF	From 89% to 94%
Subramanian et al.,2021	Corn Disease detection	VGG 16	Up to 97%
Singh et al.,2021	diagnosis for melanoma detection	VGG16	99.1%

2.6 Algorithm Performance Evaluation

The learning algorithm or classification model requires to be evaluated to measure its accuracy in prediction. Hence, to evaluate classification model's prediction performance, it is commonly employed based on the goal of the study (Chicco and Jurman, 2020). The most obvious measure is accuracy, accounting for all the correctly predicted classes as shown in equation 3. Where TP, True positives is the total predictions correctly predicted and True Negative TN is vice versa. FP False positives is the total predictions failed to predict correctly and False Negative FN is vice versa.

$$A\ , Accuracy\ = \frac{TP+TN}{TP+FP+TN+FN} \tag{3}$$

Accuracy focuses on the correctly predicted or TP over the False Positives that is not acceptable as it does not include missed predicted metric in the equation. To acquire a better measure of performance the F1 Score as it takes harmonic mean precision and recall (Opitz and Burst, 2019). Precision is to measure the correctly predicted true cases over the total true cases. While recall, measure the TP over the actual true cases. These metrics are used in the final phase to calculate the precision, recall the F1-score. Equation 4, 5, and 6 is the precision, recall, and F1 respectively.

$$P, precision = \frac{TP}{TP+FP} \tag{4}$$

$$R, recall = \frac{TP}{TP+FN} \tag{5}$$

$$F1 = \frac{2PR}{P+R} \tag{6}$$

F1 score considers the false prediction in which in a surveillance perspective is false alarm. False alarms are not fit for surveillance tasks and should always be taken into consideration. Therefore, the study finds it is adequate to include the metric of precision, recall and F1 score to evaluate performance.

2.7 Post-processing Predictions

Post-processing is a process that takes place after the main process to filter the overall outcome. Post-processing in ML is the refinement of the main model's prediction to reduce errors (Frame, Nearing, Kratzert & Rahman, 2020). In this study case, it is crucial to reduce false alarms in a security surveillance system. Hence False Positives FP should be lowered or if possible avoided. The ML models produced in labs will require tuning to adapt to real-world conditions (Bode, Thul, Baranski & Müller, 2020). The stricter decision filter condition should also contribute to the performance of the model. Thus, post-processing is considered mandatory if such a model is prone to FP in the study's interest. FP are the occurrences where the prediction is positive or true when it is negative or false. False alarms are no different, so the main priority of SED for surveillance to detect FN being at the lowest. It is better to detect some true intrusions than all intrusions with most false alarms. The typical range for

confidence thresholding is between 50% and 75% has produced improved accuracy (Peixeiro, Naji, and Charton, 2021). A recent study implemented a threshold of 80% to accept a certain prediction of an axe striking a tree detection and produce FP of 5% (Ahmad & Singh, 2019). Thresholding improves the quality of prediction performance by reducing false predictions.

Each model suffers a degree of FP rate confusion between ambience events with intruders' events. The high FP are not aligned with the intended purpose of the research that is to be used as a security surveillance system. Hence, a post-processing layer is considered mandatory, and the raw result is considered too loose with false prediction rates. Thresholding post-processing is applied to the point of approximately 10% false prediction on any intruder events. When a prediction has a low degree of confidence favouring 51% over the other 49%, it cannot be considered a good prediction (TaheriNejad and Jantsch, 2019). Hence, by increasing the threshold can reduce FP. The variable threshold level is optimized until the 10% false detection rate target is acceptable for the SED surveillance task.

2.8 Summary

There have been numerous attempts to conserve wildlife in Malaysia. The effort was not fully accomplished due to insufficient resources such as finances, equipment, people, and safety. It was found that the problem at hand is important for the preservation of wildlife. The application of artificial intelligence in solving the problem by using SED is expected to help save wildlife. The use of ML and DL techniques directed to the path of the industrial revolution 4.0 vision. Literature review has pointed out the SED in a forest environment has implemented various methods found suitable in the approach of SED by using RF, CNN, and SVM models with MLE feature extraction. The proposed method of hybrid CNN-RF is expected to improve performance based on the individual strengths and weaknesses that complement each method. The RF is less susceptible to overfitting while CNN deep features could improve performance by its deep features. Next, the problem of false alarms is not acceptable for surveillance purposes. Therefore, post-processing of thresholding is found to handle this matter. Finally, to measure performance a practical method is the F1 score, precision and recall due to its consideration of FP in the equation.

CHAPTER THREE
RESEARCH METHODOLOGY

3.1 Introduction

This chapter describes the methodology of the research. It explains the phases including data collection, feature extraction, analysis of features and the classification models of CNN, SVM, RF and CNN-RF used with their respective parameters and configurations. The methodology has four phases and explains in detail with its specifications in the methodology framework.

3.2 Methodology Framework

The overview of the methodology framework has four phases described in Figure 3.1. It covers the flow of work and how the research was conducted. All the phases are necessary to complete the study. Figure 3.1 shows the four phases of the research. The first phase is a data collection process that includes the location environment, data classes, and recording equipment. The data collection was assisted by a non-profit organization, namely WCS Malaysia. The data collection was done in 2019 at Endau-Rompin National Park, Malaysia.

In Phase 2, feature extraction is done on the sound data in phase 1. The feature extraction parameter is tweaked to cater for sound event detection. The extracted features were analysed in Phase 3. If the features prove to be irrelevant, feature extraction parameters were re-tweaked (See 3.4). Once the parameters produce optimal features, the features proceed to Phase 4 to be employed in sound classification models. In Phase 4, evaluation of the SED is concluded and compared.

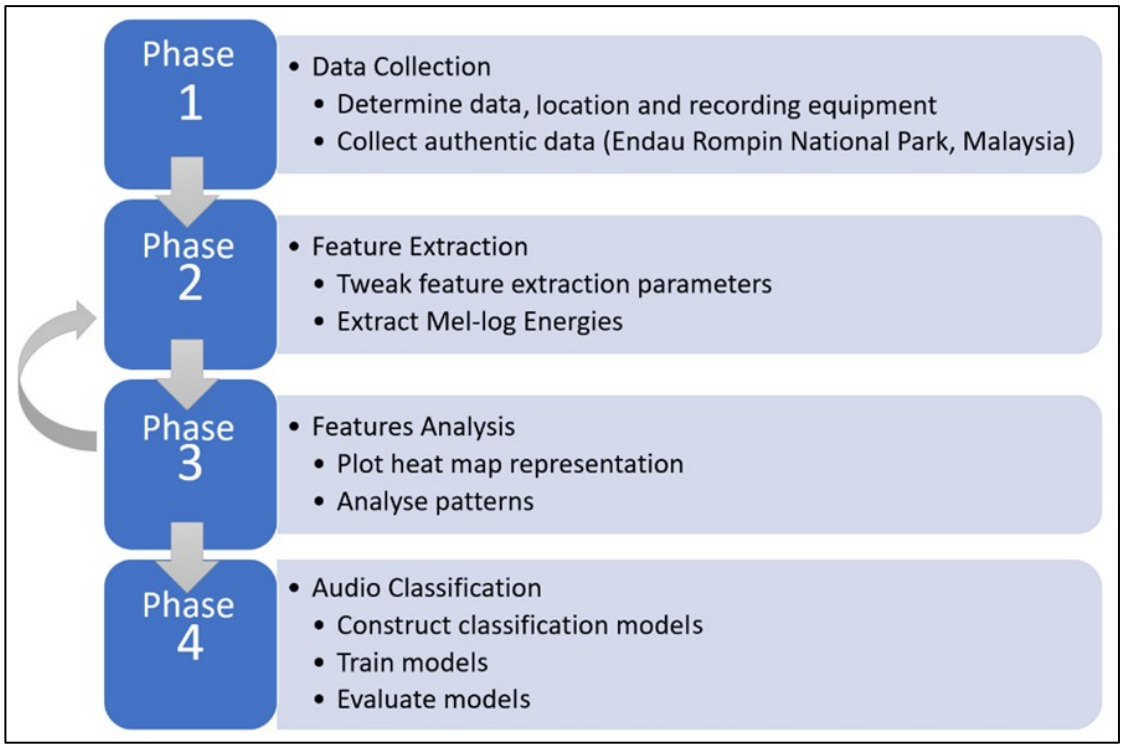


Figure 3.1 Methodology Framework Phases

3.2.1 Phase 1: Sound Data Collection

This phase is the specifications conducted to collect sound data as Waveform Audio File (WAV) Format. Steps include sourcing of data, processing the data and extraction of features from the data. In collaboration with WCS and PERHILITAN, data collection is done with the support and supervision of wildlife law enforcement advisors. Next the data collection detail is elaborated in five sections including the location of data collection, the collection of intrusion sounds, environment sounds, the distances from the sound source and the equipment used for recording.

i. Location of data collection

The datasets were collected at Taman Negara Endau Rompin at three locations for hatchet, chainsaw, and vehicle sound events as shown in Figure 3.2. The three types of environment locations consist of thick forest, riverside, and roadside. Thick Forest is a place having higher density of trees and vegetation. This location has many obstacles for sound to travel. The obstacles present here are tree trunk, thick foliage and woods growing under the big trees. The second location is the riverside, a place

located near a flowing river. Here the river flowing produces noise in the background which is believed to disrupt the flow of the sound. The third location is the roadside in the said jungle. The road here is a logging road. It was not surface by bitumen but just plain laterite soil road.

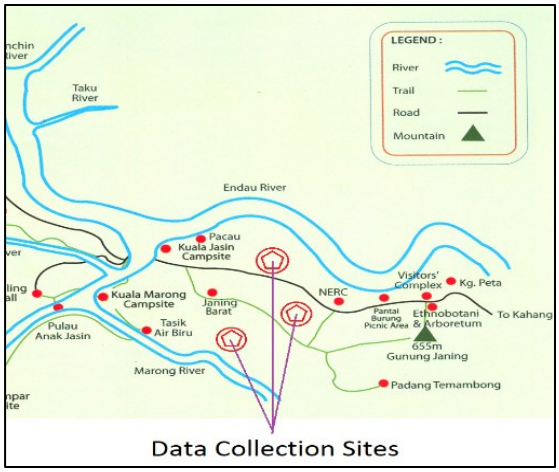


Figure 3.2 Sites of Data Collection in Endau Rompin

ii. Collection of intrusion sounds

The sounds considered was suggested by the WCS Malaysia on 14th July 2019, which was an active player on the anti-poaching effort. The sound collected were of three types obtained from the following activities:

1) Hatchet or Machete hitting a tree trunk
   - A person cutting a tree using a single-handed hatchet or machete
2) Chainsaw revving & cuts a tree trunk
   - A person operates a chainsaw at idle, revving and cutting a tree trunk.
3) 4x4 Vehicle engine idle and revving
   - A vehicle operates in a static position while idle and revving.

The machete and hatchet were also considered because the poachers often use them to cut trees to build their camps (Zolkepli, 2019). Then the chainsaw is a common tool of illegal loggers. Vehicles that used in jungles 4x4 are used to transport the goods out of the forest. Vehicle engines are considered rare sounds in forest reserves.

iii. Environment sounds

The next element considered in the data collection is the environmental sound or ambience of each location. This sound data consists of the forest natural environment.

It is important because the sound is considered as the background noise. Simulation or emulations of sound events were done to get sound in the jungle with its current condition and environment. Table 3.1 is the combination of locations, intrusion sound events that was collected. Locations include three types of forest environment: thick forest or deep forest, roadside forest nearby a dirt vehicle pathway or road, and riverside forest where we can hear the river nearby. Each location was recorded with sound emulations such as tree cutting, chainsaw and vehicle activity.

Table 3.1
Location and Emulations of Data Collection

Location	**Sound Emulation**
Thick forest	Tree Cutting Activity
	Chainsaw Activity
	Vehicle Activity
	Ambience (No emulation)
Roadside	Tree Cutting Activity
	Chainsaw Activity
	Vehicle Activity
	Ambience (No emulation)
Riverside	Tree Cutting Activity
	Chainsaw Activity
	Vehicle Activity
	Ambience (No emulation)

iv. Distances from Sound Source

Distances from sound source is the distance from the source emission point to the recording point in meters. The aim of the study is to at least be able to detect an intrusion sound as a human ranger would recognize in the forest. The sounds collected were repeated at three distances which are 30m, 60m and 100m. The distance was measured between the recording device and the sound emulation. The distance was calculated accurately using a Global Position System (GPS). It was scattered with random directions and different distances approximately where the distances are considered based on a normal person's hearing capability (Reynolds et al., 2010) since SED is assumed able to at least mimic a human ability to hear in normal circumstances. Figure 3.3 shows the distance of 30m, 60m and 100m from the point of recording to the sound event emulation. The emulation sound is not moving when the emulation point is at the centre. The recorder moves from the center in random directions within a radius

of 30m, 60m, and 100m.

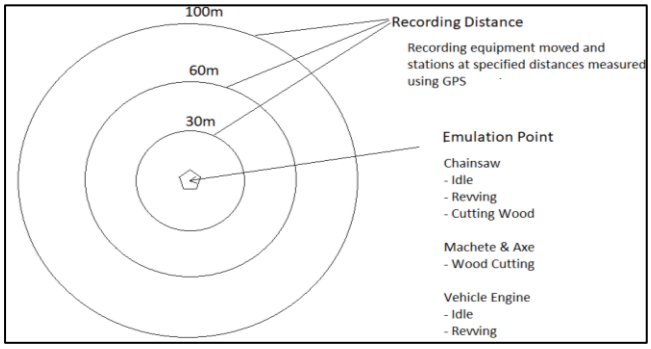


Figure 3.3 Illustration of Distances of Data Collection Process

v. Recording Equipment

Recording data is heavily dependent on the recording tools used in the acquisitions of it. The tools used specifically for this research is as Recorder Zoom H6 Handy Recorder professional grade recording tool as shown in Figure 3.4. The sound recording settings of 22.05 kHz and 12V phantom power enabled. Recorder recording with three microphones simultaneously. The four microphones consist of an unbranded low-end LR stereo mic, mid-range BM800 condenser mic and a high-end XYH Zoom H6 capsule mic. The reason behind using more microphones is to capture more data simultaneously. The diverse microphone types are to capture all variations of recordings between mics. Zoom H6 with 4 external microphone XLR Input with phantom power allows data collection with different microphones without any hardware level inconsistency and distortions. The use of different microphones are efforts to reduce inconsistency in recording quality. It helps the ML/DL to generalize between different input microphones making them more robust for future implementations.

Figure 3.4 Zoom H6 Handheld Recorder

3.2.2 Phase 2: Sound Data Feature Extraction

Mel -log energies known as MLE is selected as the feature extraction method on its high reliability on rare sound detection in past studies. The feature was extracted from the sound files collected on 4th-6th August 2019 at Endau Rompin National Park. The feature extraction of waveform sound files was done using python libraries, including Librosa, SoundLazy, Speechy and PySound. The features are visualized using a heatmap representation to find the patterns that exist with the current MLE extraction parameters. If the feature patterns are not visible or insufficient, the feature extraction parameters require tweaking.

i. MLE Feature Extraction parameters

A suitable parameter will produce reliable data for SED. The MLE feature extraction parameters in Table 3.2 are from common practices in previous SED solutions that were used as a final configuration for feature extraction. The parameters are frame length, frame overlap percentage, the FFT size, and the number of filter bank, lowest band edge and highest band edge. The parameters are defined based on standard practices from previous work in SED.

Table 3.2
MLE Feature Extraction Parameters

Frame Length	Frame overlap	FFT	Number of Filter Banks	Lowest band edge	Highest band edge
100 milliseconds	50%	512	40	20Hz	20kHz

ii. Data Augmentation Shifting

Sound shifting is a popular technique to increase samples for CNN applications without interfering with the nature of the original sound. Sound events vary from 2 to 10 seconds of observation to establish a good estimation for detecting the activity's existence (Pandya & Ghayvat, 2021). The sound segment of five seconds was used for an activity presence time frame required for detection. Data augmentation improves model training in CNN as more samples contribute to CNN performance (Inoue et al., 2018; Mushtaq & Su, 2020, Mushtaq et al., 2021). Figure 3.5 demonstrates how shifting

was implemented from a) original clip to b) augmented clips up to five times. From the original clip into 5-second sound clips extracted at the start of the clip to the end. The next iteration is done with a hop of 1 second, starting at one second from the start of the sound clip. So, in the present research, the sound was shifted by 1 seconds to produce about five times more sound samples. The one second hop represents the skipping of 20% of the 5 second's clip. This will produce the next clip to be unique having 20% difference between previous samples.

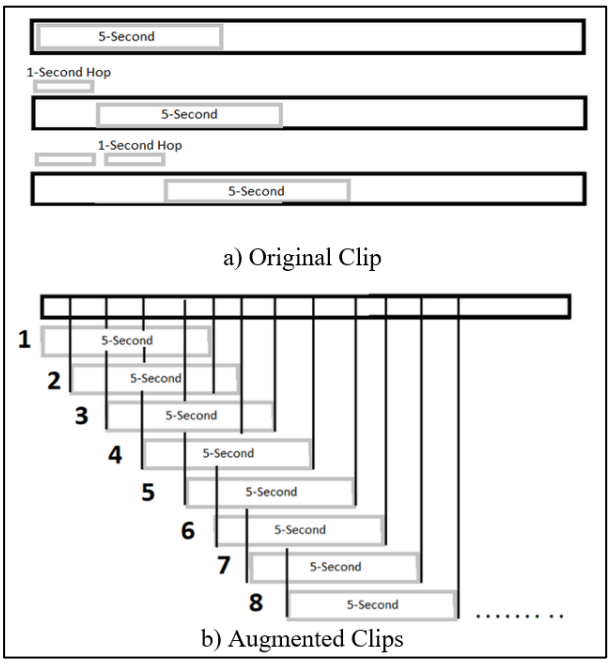


Figure 3.5 Illustration of Augmentation from a) Original Clip to b) Augmented Clips

The tools used were created using python code and audio libraries such as audio segmentation

3.2.3 Phase 3: Analysis of Extracted Feature

Analysis of extracted features using a heat map representation will help in locating significant patterns in between classes. The feature matrix of two-dimensional floating numbers can be interpreted easier as a heat map. A collection of heat maps representing of features was made to identify patterns. The heat maps were constructed using python libraries numpy, Speechpy and Pyplot. An observation of the heat maps was conducted to find the significant patterns of the feature with their respective labels. This step was crucial to indicate a hypothesis for the model's efficiency to learn from

the features. A good feature would produce a significant pattern for a specific class. Figure 3.6 shows the code snippet that was used to extract and generate heatmap representation for analysis for a batch of files. The code includes taking a partial of the sound file by taking the amplitues by seconds times the sampling rate to cut off the section of the audio file. Next, emphasizing and extracting MLE using speechpy library.

```
for f in files[:]:
    file_name = f
    #1 Read AUDIO Signal File
    fs, signal = wav.read(file_name,1)
    signal = signal[0:44100*10,0]

    #2 Define Overelapping (50%) and Hop length (100ms)
    para_fl = 0.1
    para_ovlp = 0.5

    #3 Audio pre-emphasizing.
    signal_preemphasized = speechpy.processing.preemphasis(signal, cof=0.98)
    #3 Extract mel-log energies MLE
    logenergy = speechpy.feature.lmfe(signal, sampling_frequency=fs, frame_length=para_fl, frame_stride=para_fl*0.5,
                num_filters=40, fft_length=512, low_frequency=0, high_frequency=20000)

    print('logenergy features=', logenergy.shape)
    #4 Plot MLE feature heatmap
    fig = plt.figure()
    plt.figure(figsize=(20,4))
    plt.ylabel("Mel-log filterbank")
    plt.xlabel("Frame")
    first_image = np.array(logenergy.transpose((1, 0)), dtype='float')
    plt.imshow(first_image, interpolation='nearest', aspect='auto')
    plt.colorbar()
    plt.gca().invert_yaxis()
    #5 Save Image to Folder
    saveas = file_name.split("/")[7].split("\\")[1].split(".")[0]+'AR19-4_'
    print(saveas)
    plt.savefig(saveas, dpi = 100)
```

Figure 3.6 MLE Feature heatmap code snippet

3.2.4 Phase 4 (a): Classification of Intrusion Sound

The classification of intrusion techniques employed are CNN, SVM, RF and CNN-RF variants. The first step in Phase 4 is to prepare the data so that the data is in the same format for all classification models to be tested. Then the standardized data is then used in training and testing for comparison.

iii. Data Preparation

Data preparation is needed for training and testing. The data split is shown in Table 3.3. The data preparation uses the best practices in recent studies and is found suitable for CNN training. The data are divided into two subsets: in-sample and out-of-sample data. In-sample data are used for training and testing of the CNN model. Meanwhile, the out-of-sample data used for the evaluation of the trained model.

Table 3.3
Data Split for training and testing

Collected Data	Out-of-Sample Data (Testing)	In-sample Data (Training)	
		70%	
100%	30%	Training 70% of In-sample Data	Validation 30% of In-sample Data
		49%	21%

The model creation process will require training and validation data to assess its learning progress and to observe the model performance. The model will utilize the In-sample data for the training and validation data. After the model was trained then the model will be evaluated using the out-of-sample data. The out-of-sample data is based on the thick forest and forest roadside environment mixture. Table 3.4 and Table 3.5 show the statistics of the datasets including percentage to overall data and a total time of the actual sound data duration for a combination of each class.

Table 3.4
In-sample Data for training

Class	Samples	Percentage (%)	Total Time (Mins)
Ambience	4878	19.9	406.50
Hatchet	8694	35.4	724.50
Chainsaw	7479	30.5	623.25
Vehicle	3483	14.2	290.25
Grand Total	24534	100.00	2044.50

Table 3.5
Out-of-sample Data for validation

Class	Samples	Percentage (%)	Total Time (Mins)
Ambience	1330	18.3	110.83
Hatchet	2622	36.1	218.50
Chainsaw	1311	18.1	109.25
Vehicle	1995	27.5	166.25
Grand Total	7258	100.0	604.83

iv. CNN Model

The CNN is constructed using Python libraries such as Tensorflow, Numpy, and Keras. The CNN consists of four layers which are the input, Convolutional, FC and the output. The Convolutional and the FC layers can be customized to increase their efficiency. A suitable CNN model is assumed to be a less complicated model. The minimal-sized model requires less computational power, suitable for low-powered devices. The devices include Single Board Computers (SBC) that consume less than 400 milliamps per hour. The design of the CNN requires testing until an optimal design for the present research. The layers which should be adjusted to get the best fit are Convolutional and FC layers.

The variables of a Convolutional layer can be adjusted in the present research. The amount and size of layers contribute to the overall efficiency of the model. It is assumed that more and larger layers may allow a better result but too many and larger layers might disrupt power consumption. On the other hand, optimizing the amount avoids unnecessary computation requirements. Model X in Figure 3.5 is the variable model design of CNN that will be modified for different results. The N in Figure 3.7 is the number of epoches or iterations to train the model before stopping for out-of-sample evaluation. The sound features from Phase 2: Sound Data Feature Extraction will be split for in-sample and out-of-sample data. Model X in Figure 3.7 is a variable Model that can be constructed in many forms and iterates the entire process for different results to find the optimal model design.

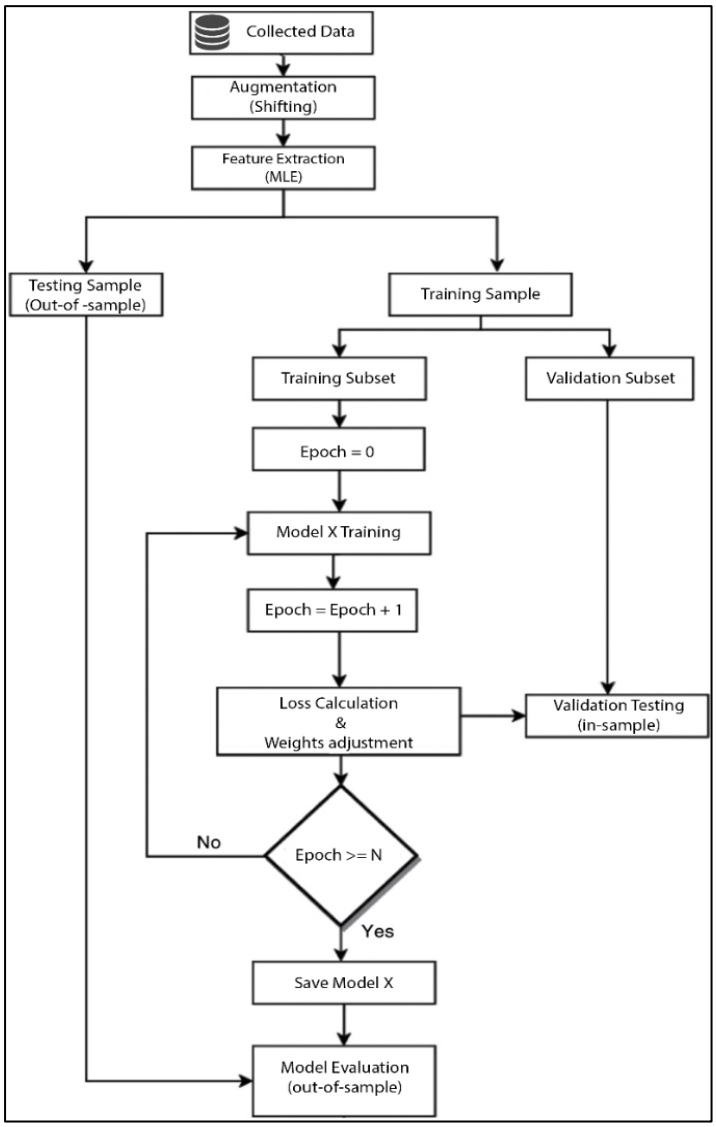


Figure 3.7 CNN Model Evaluation Flow Chart

To monitor the learning progress of the current epoch the model performance was calculated using the prediction results from the validation subset. The prediction results were used to calculate the loss. A loss closer to zero is desired indicating good performance. The loss calculation will be done on each epoch to monitor the models learning progress. The loss function used in the study is the categorical cross entropy. The weights adjustments optimizer on the NN uses ADAM, as it is still relevant in recent studies. The loss for all models and on each epoch will be plotted to observe performance for comparison. Figure 3.8 shows the Python code snippet to construct a CNN. The code snippet in Jupyter notebook shows the process of defining architecture, call-backs and including the ADAM optimizer. The parameters in model architecture

includes lines 1 to 17 the code model.add(layers.Conv2D) is the CNN layer and model.add(Dense()) fully connected layer. Lines 4-8 define the 1st CNN layer and to make the 2nd,3rd and so on can be done by replicating the lines making as many layers required for the study. Next, compile the model with an ADAM optimizer in lines 20-23 with the learning rate, loss function, and accuracy metrics. The training log of each epoch is done by call-back functions that will occur on each epoch after training is done defined in lines 25-38. The call-back function includes the loss calculations to find the best loss model from all epochs. The training and validation are done automatically on each epoch using the ModelCheckpoint call-back to find the best loss model. Finally, line 43 initiates the training process using the defined model and call-backs from lines 1 to 38. This code will produce the best model file (BestLossModel.h5) and log files, including each epoch loss and accuracy of the model trained.

```
1  #Creating a Custom CNN Model
2  model = Sequential()
3  #Define Convolutional Layer (2D) = 32 x 32 with kernel size 3x3 and input shabe MLE 90*48,+1 Dimension
4  model.add(layers.Conv2D(32,
5                          (3, 3),
6                          activation='relu',
7                          input_shape=(90, 48, 1)))
8  model.add(layers.MaxPooling2D((2, 2)))
9  #Flatten Outpur into 1-Dimension Array
10 model.add(Flatten())
11 #Define Dense of Fully Connected Layers 32
12 model.add(Dense(32,
13           activation = "relu"))
14 #Define Dropout 0.5
15 model.add(Dropout(0.5))
16 #Define Output Layer 4 = 4 Classes adn Softmax activation typical for multi-class problems
17 model.add(Dense(4, activation = "softmax"))
18 #Define Compile the model and set ADAM as the optimizer,
19 #loss function using categorical_crossentropy and calculate accuracy
20 model.compile(Adam(learning_rate=0.0001),
21               loss='categorical_crossentropy',
22               metrics=['accuracy'])
23
24
25 #Define where to save the logs (Loss and Accuracy each epoch)
26 TrainingLogDir = 'TrainLog-'+ datetime.datetime.now().strftime("%Y%m%d-%H%M%S")
27 tensorboard_callback = tf.keras.callbacks.TensorBoard(log_dir=log_dir, histogram_freq=1)
28
29
30 #Create a callback to save the Best Model.
31 #When (monitor='val_loss') validation loss on epoch is at most minimum (mode='min')
32 #Define Where to save the Best Model
33 BestLossModelDIR = TrainingLogDir+'/BestLossModel.h5'
34 checkpoint = ModelCheckpoint(BestLossModelDIR, monitor='val_loss',
35                              verbose=1, patience=2,save_best_only=True, mode='min')
36
37
38 callbacks_list = [checkpoint,tensorboard_callback]
39
40 #Begin Training Session
41 #saving information to history(loss/accuracy during trainign on all epoch saved)
42 #validation_generator = input validation data based on a folder
43 history = model.fit(train_generator,
44                     steps_per_epoch=train_steps,
45                     validation_data=validation_generator,
46                     validation_steps=val_steps,
47                     epochs=100,
48                     verbose=1,
49                     callbacks=callbacks_list,
50                    )
```

Figure 3.8 CNN python Code snippet

The log files can be viewed using tensorboard by running the command on a python environment (Anaconda prompt) “tensorboard --logdir TrainLog-Time/”. The command will open a tensorboard webserver with user interface and allow viewing the logfiles created by callbacks during the training session. Once the best performing model is found it will be further investigated for fine tuning. Finally, after the best model is found, the model will be evaluated with the out-of-sample data to observe how it performs with unique data.

The CNN model requires a random search on viable models, observation on overfitting and hyperparameter tuning. The hyperparameter including the CNN and dense layers. The search for an optimal CNN model will be done using three stages preliminary model search, in-depth model search. Preliminary model search stage is the initial search on an optimal model design. The initial convolutional sizes are inspired based on the VGG16 design for the 1st and 2nd layer are between 16 and 64 paired with a fully connected layer of 64 or 128. Table 3.5 shows the model structures established for preliminary model search.

Table 3.6
Initial model structures for preliminary model search

Model Structure		
Convolutional Layers (2D)		Fully Connected Layer
1st	2nd	
16	32	64
32	32	64
32	32	128
32	64	64
32	**64**	**128**
64	64	64
64	64	128
64	128	128

Next, in-depth model search stage observes the similar best model obtained from the previous stage and finally in hyperparameter optimization stage efforts on tuning hyperparameters to find the optimal training point early stopping to avoid overfitting. The preliminary model search focuses on getting the right model design in by training a batch of models with a wide array of sizes. The model construction is the sequential

component of the model. Those variables are the size of the Convolutional layer, size of the fully connected layers and the amount of each layer. The Figure 3.9 was established based on the nature of the VGG16 design of CNN. Figure 3.9 CNN Model Construction Components with the position of each layer and the necessary layers needed in between.

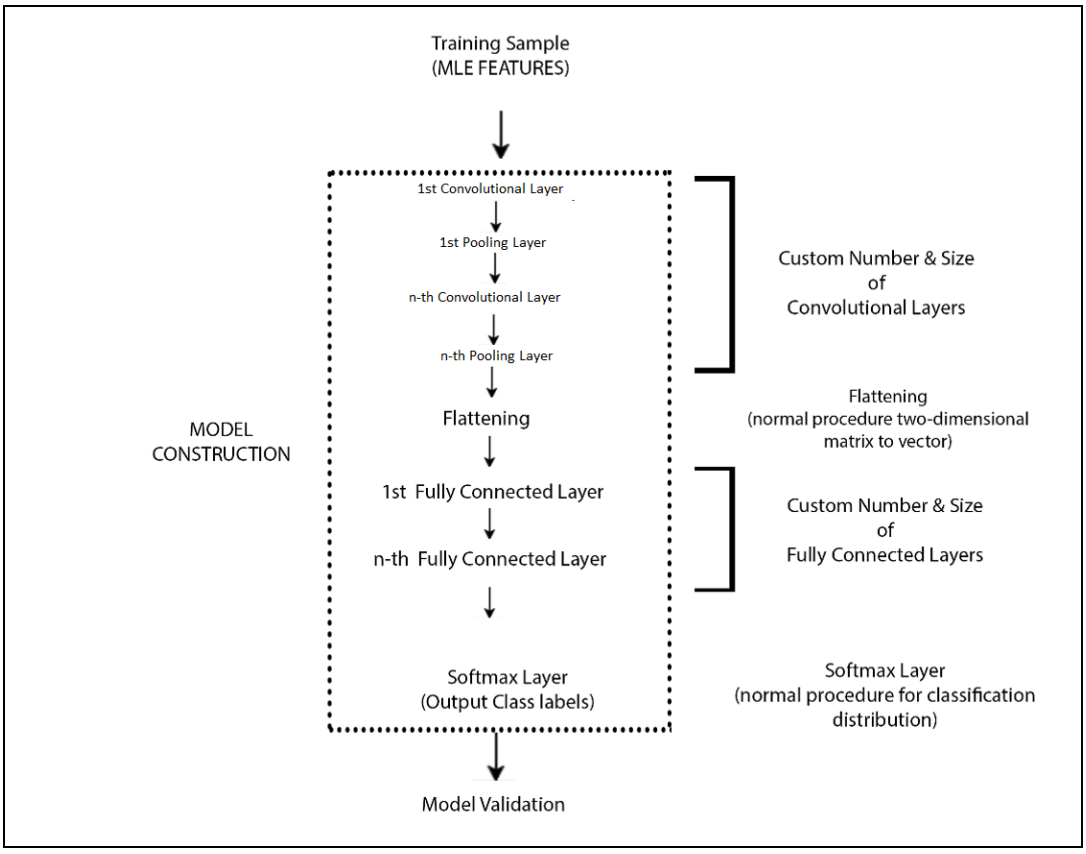


Figure 3.9 CNN Model Construction Components

A series of CNN models will be tested and all models in the present research will follow the model construction framework. The first layer of the model will be the convolution layer with its very own customizable size. The model then can either add more Convolutional layer(s) or proceed to the flattening procedure. Afterward, the fully connected layers can be included into customizable sizes. The model can either add more fully connected layer(s) or proceed to the softmax layer. The softmax layer is a common procedure to produce probability distribution output for each respective class.

After the models are designed and constructed, all will be trained using the in-sample dataset split of 70% training and 30% validation. The loss function used is the

multi-class cross-entropy loss will be calculated at the end of each epoch for every model. The accuracy will also be computed on each epoch to observe the overall results. This loss metric will be the main indicator for a better model. The hyperparameters used are fixed with ADAM optimizer with a learning rate of 0.001, batch size of 250 and no dropout on any layers.

In the in-depth model search, the study investigates the best models from the experiments for further exploration to find overfitting indicators. The indication of overfitting lies in the gap of training loss and validation loss. A bigger gap will suggest the occurrence of overfitting. The model deemed best will be tested on unique out-of-sample data from the collected data and next, the application of threshold post-processing, if necessary.

Finally, tuning the hyperparameters to avoid overfitting and increasing the overall performance. Hyperparameters include the learning rate momentum, batch size, dropout rate and model complexity. The aim is to get low loss without overfitting the training set to create a reliable model.

v. Support Vector Machine

SVM implemented in the study is a linear SVM. The experiment is done using the python library of Scikit SVM. The input features are flattened to train the SVM model, from 2-Dimensional of 98,40 to 1-Dimensional 3920 array of floating number. This procedure is required as the SVM requires a 1-Dimensional input. The parameters implemented are default values considered standard best practices: (Kensert et al., 2018; Kramer, 2016; Pedregosa et al.,2011).

- C, regularization parameter = 1.0
- kernel, kernel type parameter = rbf
- gamma, kernel coefficient = scale
- max_iter = 1
- probability, probability output = True
- shrinking, use shrinking heuristic = True
- tol, tolerance stopping value = 0.001
- cache_size, kernel cache = 200
- decision_function_shape, return one vs rest decision = ovr

Figure 3.10 shows the SVM code snippet to train the SVM model with the settings. The code snippet shows the parameters based on standard best practices defined in line 12-

20 imported SVM implementation from sklearn. Line 23 executes the training session using the features *FlatenX* and labels *y* on the function (*clf.fit*)

```
12 from sklearn.pipeline import make_pipeline
13 from sklearn.preprocessing import StandardScaler
14 from sklearn.svm import SVC
15 #Define SVM Pipeline based on best practices
16 clf = make_pipeline(StandardScaler(), SVC(C=1.0, break_ties=False, cache_size=200, class_weight=None,
17                     coef0=0.0, decision_function_shape='ovr', degree=3,
18                     gamma='scale', kernel='rbf', max_iter=-1,
19                     probability=True, random_state=42, shrinking=True,
20                     tol=0.001, verbose=True))
23 #Train Model on FLatten Features
24 clf.fit(FlatenX,y)
```

Figure 3.10 SVM python code snippet

vi. Random Forest Model

The random forest implemented in the study uses an ensemble of the common Classification and Regression Tree (CART) type of decision tree. The RF model can be tuned to the get optimized results by using parameters that are tuned. The input features are flattened before modelling the RF model, from 2-Dimensional of 98,40 to 1-Dimensional 3920 array of floating number. This procedure is required as the RF model requires a 1-Dimensional input. The tuned parameters are the ensemble size, the number of DTs that are used to produce the final decision. But the relationship for amount trees and performance is not linear. Hence a series of the model were trained to reach the best model. The list of ensembles size that will be trained to achieve the optimized model will be 10, 20, 50, 100, 200, 300, 400, 500 and 1000. Thresholding post-processing method is expected to reduce the FP rate (Ahmad & Singh, 2019). The result shows the changes in the F1, precision, and recall scores. Once post-processing is done, the performance can be measured more accurately for a real-world basis as a detection system achieves a minimum of 10% false alarm rate to be deemed usable.

Figure 3.11 shows a code snippet to prepare RF Model. RF used imported from sklearn.ensemble library. Line 14 defines the RF model parameters where n_estimators are the number of trees inside the RF ensemble.

```
12 from sklearn.ensemble import RandomForestClassifier
13 #DEFINE n_estimators = numbers of trees
14 clf=RandomForestClassifier(n_estimators=500,
15                            bootstrap = True,
16                            max_features = 'sqrt',random_state=42,verbose=1)
```

Figure 3.11 RF python code snippet

vii. CNN-RF Hybrid Model

The CNN-RF model implementation is a combination of the CNN and RF model. The CNN model is used from a pre-trained model that has proved to provide performance for image recognition and the custom CNN produced by the study. The CNN model is inspired based on the VGG16 architecture (mentioned in Section 2.5.2). A lot of resources require to optimized weights. Thus, the use of the trained weights can reduce training time. This method also known as transfer learning. The transfer learning is also used in a CNN-RF model. The CNN portion of the hybrid model acts as a feature extraction layer. The features extracted will be optimized to be the most useful in classification. The RF portion will perform the prediction based on the CNN output. Finally, the process can be applied for post-processing to optimize the results FP rate. The following are the brief steps of CNN-RF model. It is illustrated in Figure 3.12.

1. The MLE Features with dimensions of (90 x 48) are all pre-processed by feeding it through the CNN (VGG16) as an input for a prediction task.
2. The CNN layer will output a set of features with dimensions of (2 x 1 x 512) in which then later flatten to a 1-Dimensional array with dimensions of (1024 x 1) for RF model compatibility requirements.
3. The flatten features are used as a typical input to the random forest model.
4. The random forest will operate as usual and generate the results as a class prediction with each class confidence level.
5. Next the Post-processing layer of thresholding is applied to the results acquired from the RF prediction to help reduce FP rate by rejecting the low confidence intrusion predictions as non-intrusion.

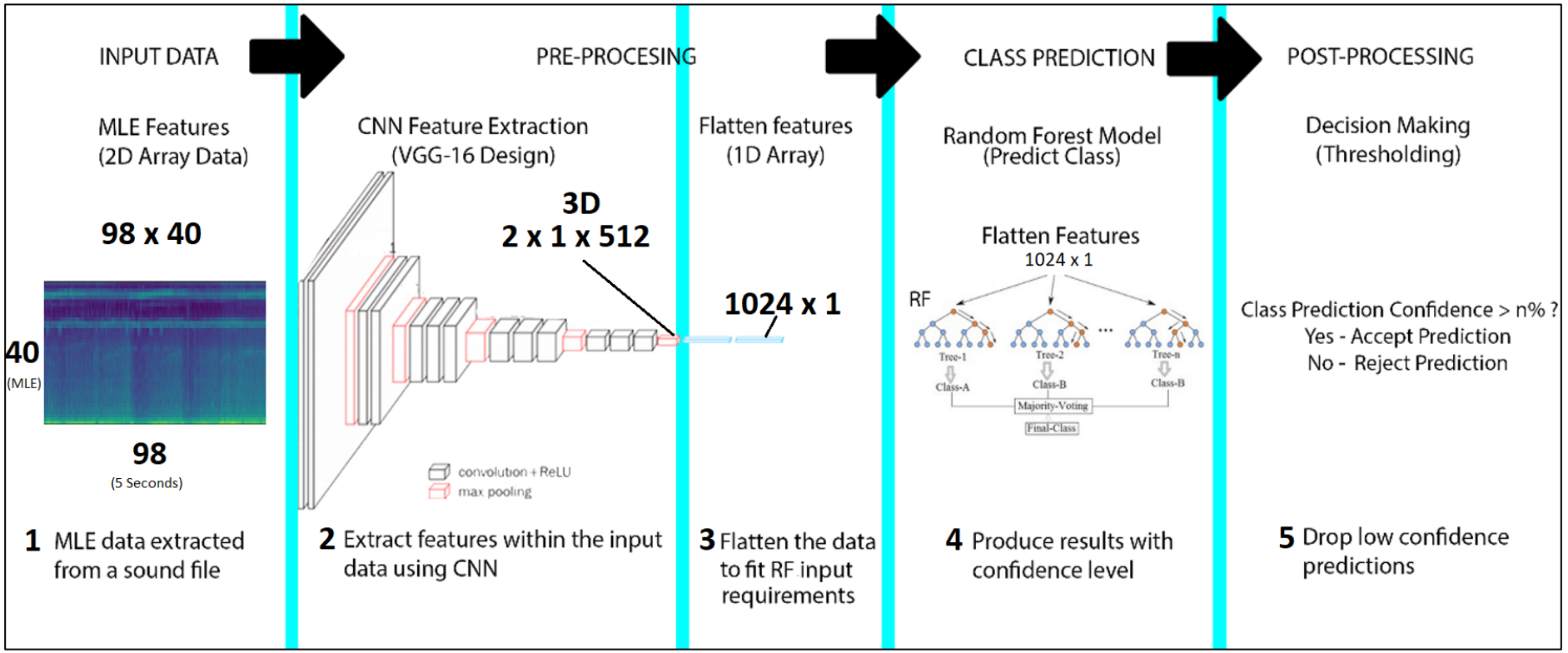


Figure 3.12 CNN-RF Hybrid Model Flowchart

The CNN model of VGG16 was used as a feature extraction method to get meaningful features from the MLE features. The CNN output will be used as the input for the RF model to compute the prediction. The RF ensemble size will be tested in ranges 10 to 500 ensembles to find the best producing model. Once the best model is found, the post-processing is done. The performances between models are compared to analyse the improvements of the hybrid method.

Figure 3.13 is the model VGG16 that was modified at the input layer to fit the input of the MLE Features. The input layer shape modification from 224 x 224 x 3 input to 98, 40, 3 was applied and no other modifications were done. The rest of the model is the original VGG16 design with the pre-trained weights from ImageNet.

```
Model: "VGG16 input 90 48"
_________________________________________________________________
Layer (type)
Output Shape                                          Param #
=================================================================
input (Input Layer)
[(None, 98, 40, 3)]                                       0
_________________________________________________________________
block1_conv1 (Conv2D)
(None, 98, 40, 64)                                     1792
_________________________________________________________________
block1_conv2 (Conv2D)
(None, 98, 40, 64)                                    36928
_________________________________________________________________
block1_pool (MaxPooling2D)
(None, 45, 24, 64)                                        0
_________________________________________________________________
block2_conv1 (Conv2D)
(None, 45, 24, 128)                                    73856
_________________________________________________________________
block2_conv2 (Conv2D)
(None, 45, 24, 128)                                   147584
_________________________________________________________________
block2_pool (MaxPooling2D)
(None, 22, 12, 128)                                       0
_________________________________________________________________
block3_conv1 (Conv2D)
(None, 22, 12, 256)                                   295168
_________________________________________________________________
block3_conv2 (Conv2D)
(None, 22, 12, 256)                                   590080
_________________________________________________________________
block3_conv3 (Conv2D)
(None, 22, 12, 256)                                   590080
_________________________________________________________________
block3_pool (MaxPooling2D)
(None, 11, 6, 256)                                        0
_________________________________________________________________
block4_conv1 (Conv2D)
(None, 11, 6, 512)                                   1180160
_________________________________________________________________
block4_conv2 (Conv2D)
(None, 11, 6, 512)                                   2359808
_________________________________________________________________
block4_conv3 (Conv2D)
(None, 11, 6, 512)                                   2359808
_________________________________________________________________
block4_pool (MaxPooling2D)
(None, 5, 3, 512)                                         0
_________________________________________________________________
block5_conv1 (Conv2D)
(None, 5, 3, 512)                                    2359808
_________________________________________________________________
block5_conv2 (Conv2D)
(None, 5, 3, 512)                                    2359808
_________________________________________________________________
block5_conv3 (Conv2D)
(None, 5, 3, 512)                                    2359808
_________________________________________________________________
block5_pool (MaxPooling2D)                                 0
(None, 2, 1, 512)
=================================================================
Total params: 14,714,688
Trainable params: 0
Non-trainable params: 14,714,688
```

Figure 3.13 VGG16 Model Adapted to MLE Input Shape

Figure 3.14 is the code snippet shows how the VGG16 Model from imagenet and reshaped the input layer from 224,224,3 to 40,98,3 instead. The line 8-13 show a sample of audio MLE Heatmap on form of PNG loaded and fed into the VGG16 mode to output a 3-Dimension tensor or deep features as 2,1,512. All data features were extracted using the same method and then flattened to be trained in the RF Model. The Deep features extracted will be an array of 1024 floating numbers fed into the RF classifier.

```
5  #Import VGG16 FROM imagenet and change the inputshape fitting 2DMLE from 40x98
6  model = VGG16(weights="imagenet", include_top=False, input_tensor=Input(shape=(40, 98, 3)))
7
8  img_path = 'Test.png'
9  img = image.load_img(img_path, target_size=(40, 98))
10 img_data = image.img_to_array(img)
11 img_data = np.expand_dims(img_data, axis=0)
12 img_data = preprocess_input(img_data)
13 vgg16_MLE_feature = extractionModel.predict(img_data)
```

Figure 3.14 CNN-RF (VGG16) code snippet with weights from imagenet

3.2.5 Phase 4 (b) Thresholding Prediction Post-processing

Thresholding post-processing on prediction is done to reduce the false alarm by rejecting the lower confidence predictions as an additional effort. Lower confidence prediction can be measured in probability distribution of a class prediction with 50% being the common threshold and is considered low. The determination of the predicted class is based on the confidence probability value of each class. By default, the class with the highest value of confidence or 50% is selected. The thresholding method is expected to improve performance (Ahmad & Singh, 2019; Seccia et al., 2020). In a real-world scenario, false alarms are not suitable for surveillance. Since the objective of this study is to detect sound events for surveillance, false alarms must be taken into consideration. Figure 3.10 shows the confusion matrix of the surveillance perception includes True Positives (TP), miss alarm, false alarm/FP and miss class. TP are the main concerns that should have the highest priority while false alarms must ideally be at its minimum. The miss alarm indicates the prediction of hatchet, chainsaw and vehicle as ambience missing the intruder detection. The miss classification is still an alarm, but it classifies the wrong intruder sound event.

The inaccurate detection of a class might not have too much of an impact on

surveillance because it still detects an intruder but misclassifies them as the other. The Figure 3.10 shows a labelled confidence matrix from a surveillance standpoint where it emphasizes the false detection when the model wrongly predicts only the ambience class to be an intruder of any type, chainsaw, vehicle, or hatchet. While the inaccurate detection between classes of intruders can be misleading it still can help in the detection of intruders.

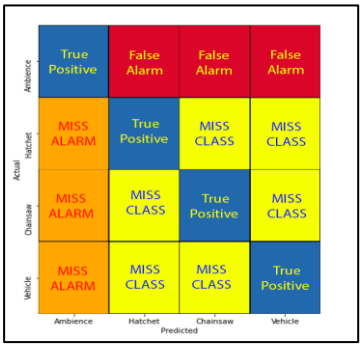


Figure 3.15 Confusion Matrix for Surveillance Perception

To reduce the FP rate on a specific class the threshold for confidence rate will be increased. The prediction with low confidence under the threshold should be assumed as Ambience class, also known as the environment noise. For a binary classification problem, 50% is the default threshold value for deciding the predicted class. To reduce the FP rate, a higher threshold value is required. Hence, we proposed threshold value is between 51% to 99%. In this case the highest threshold value is 99% is chosen because a prediction probability of 100% is often produced by an overfitted model. This statement is supported by (Fong and Tyler, 2021; Park et al., 2021; Kristiadi, Hein, and Hennig, 2020). The suggested range of threshold value is expected to give a better performance on reducing the FP rate. The pseudo code for Prediction Probability Thresholding is constructed as the follows:

Pseudo Code: Prediction Probability Thresholding

```
1   Initialize Threshold = 0.51
2   WHILE (Threshold < 1.00)
3       IF prediction confidence is greater or equal to threshold
            Class prediction is accepted.
4       ELSE
            Class prediction is rejected, prediction redirected as ambience noise (background)
5       Threshold = Threshold + 0.01
```

The pseudo code starts with initializing the minimum value of threshold. Then

it will apply to all the prediction confidence of each class until the maximum threshold value 0.99. The class prediction is in the form of probability distribution for the respective classes. For instance, scenario A the model prediction output of [0.2, 0.3, 0.5, 0.0] are 0.2 Ambience, 0.3 Hatchet, 0.5 Chainsaw and 0.0 vehicle class. Prediction confidence or probability of a specific class made by a model from a specific sound will output a probability that ranges from 0.0 to 1.0 for each class. Threshold of acceptable confidence and probability set between 0.50 and 0.99. The probability distribution will always total up to 1.0. Then the F1, precision and recall will be computed on each threshold setting.

It represents the confidence of prediction, is used to take the highest confidence level achieved as the prominent prediction probability between the classes. Two scenarios are established as demonstrated scenario A and B in Table 3.6 and Table 3.7, respectively. Table 3.6 shows the prediction output will be by default Chainsaw as it is the highest probability between classes. Table 3.7 shows Scenario B, where 0.39, 39% confidence of vehicle class is accepted when not applying thresholding while thresholding will reject the prediction due to low confidence.

Table 3.7
Scenario A for Comparison of Thresholding Results

Items	Default Prediction	Thresholding
Prediction Probability Output	[0.20, 0.30, 0.50, 0.0]	
Threshold Value	Max	60%
Prediction Result	0.5	None Accepted
Predicted Class	Chainsaw	Ambience, by default

Table 3.8
Scenario B for Comparison of Thresholding Results

Items	Default Prediction	Thresholding Applied
Prediction Probability Output	[0.25, 0.15, 0.31, 0.39]	
Threshold Value	Max	60%
Prediction Result	0.39	None Accepted
Predicted Class	Vehicle	Ambience, by default

By increasing the threshold, we can avoid misclassification and FP. The threshold effect will be applied to the class with the FP to reduce the total false alarms of the overall system prediction. The threshold can be adjusted for each class specifically to cater to class prediction difficulties and minimize false alarm predictions. The threshold filter is not mandatory if class performance is already achieving low false

alarm class performance. It should be applied when the confusion between the specific class is high between the noise or ambience class causing a high rate of false alarms.

3.2.6 Phase 4 (c) Evaluation Technique and Consistency Effort

This section explains the efforts to obtain consistent input of sound then classify events. Hence, measuring the accuracy comparing between techniques and types of detection sound relationship in different environments. Each model will be evaluated by using a constant dataset. The algorithms run on the same machine to avoid any hardware performance inconsistency. The model evaluation metric used are F1, precision and recall score performed on the out-of-sample data. The prediction results of a ML model will be in the form of a collection of TP, TN, FP and FN

- TP, True Positive is a correct prediction
- TN, True Negative is a correct not prediction
- FP, False Positive is a wrong prediction
- FN, False Negative is a wrong not prediction

The formula used for evaluation are precision, recall and F1 scores are listed in equations (4, (5), (6) Section 2.6. The efforts to maintain consistency were to control the hardware and software configurations. Hence, all evaluations were tested on a single computer equipped with Intel Xeon E5 v4 8 Core 16 Threads, Geforce GTX 1080 Graphics Card 8GB GDDRX5, 16GB DDR4 2666 MHz and 1TB SATA SSD. The software used was running on the Windows 10 operating system with Python 3.7 based Jupyter notebook. The python libraries included were TensorFlow, Scipy, Keras, Numpy Pyaudio, Librosa, and PIL. Audio data is processed from importing raw wav files directly for extraction of MLE features using only Python, no other software was used beforehand to avoid any unwanted or unrecognized software manipulation of the original sound file.

3.3 Summary

This chapter discussed the four phases of research methodology. The phases are data collection, feature extraction, analysis of features, and the classification models. A brief explanation has been established for each phase. For the classification models phase an additional task was done specifically on thresholding post-processing and evaluation of models. In this respect four models were evaluated thoroughly including CNN, SVM, RF and CNN-RF. All the phases are expected to accomplish the aim of the research.

CHAPTER FOUR
RESULTS AND DISCUSSION

4.1 Introduction

This chapter describes the result obtained by the study based on the methodology in Chapter 3. The features extraction visualization analysis of the MLE for each intrusion sound activity, the preliminary experiment of the CNN and tailored optimization of model design, followed by the SVM, RF and CNN-RF results with and without thresholding post-processing. The total samples acquired from augmentation were 31,792, consisting of ambient, hatchet, chainsaw, and vehicles with 6,208, 11,319, 8808, and 5478, respectively. These samples were used in each model training and testing experiments. The results were tabulated in a confusion matrix with F1, precision and recall scores to demonstrate the performance of all methods. Finally, the overall comparison of the model's performances was discussed.

4.2 Analysis of Audio features – Mel-log Energies (MLE)

The features shown in the following Figures are heat maps representation of the MLE features extracted from the collected audio. The audio was cut into 5-second segments of an activity occurrence and the MLE was extracted on 198 frames with 40 MLE features. The yellow indicates saturation of high numbered values in the specific MLE feature. The heat map chart label Mel-log Banks, are the MLE features extracted 40 each frame and the Frame, indicates the frames in the 5-second audio segment. Figures 4.1 show the ambience forest sound. The features showed a very distinctive pattern on the top side of 31-35 MLE Banks.

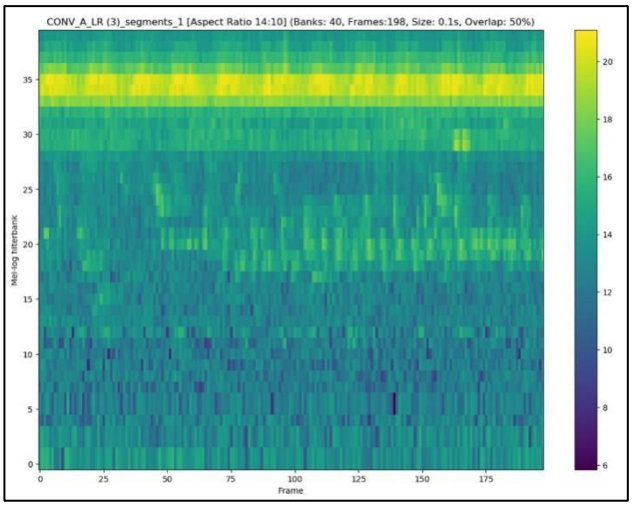


Figure 4.1 Audio Features MLE of Natural Forest Ambience

Figure 4.2 shows more forest ambience MLE features, the observation has found that MLE features of forest ambience have a common higher intensity of MLE on the 31st to 36th constant throughout the clip. These can be an indicators of ambience sound events.

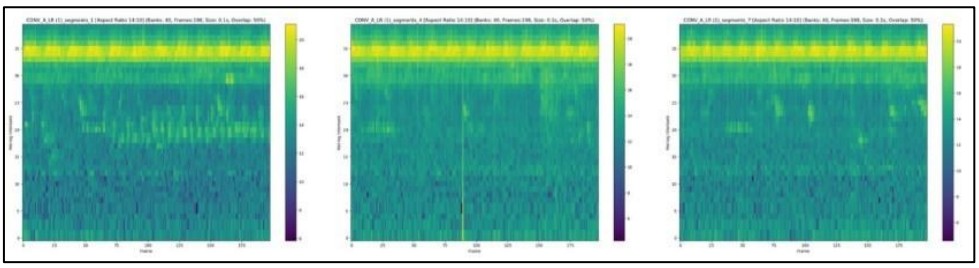


Figure 4.2 MLE Features of Different Natural Forest Ambience

Figure 4.3 and Figure 4.4 show the ambience forest sound. The features show distinctive patterns on the top side of 20-35 and 0-5 MLE. The MLE Feature patterns for chainsaw activity presence is very distinctive in the distribution of intense MLE in the regions 1st to 5th and 20th to 35th MLE. This is very distinctive between the other activities. It can be associated that chainsaw sounds are concentrated more on the lower bands MLE but having bigger spreads instead of natural ambience where its highly concentrated on the higher bands MLE.

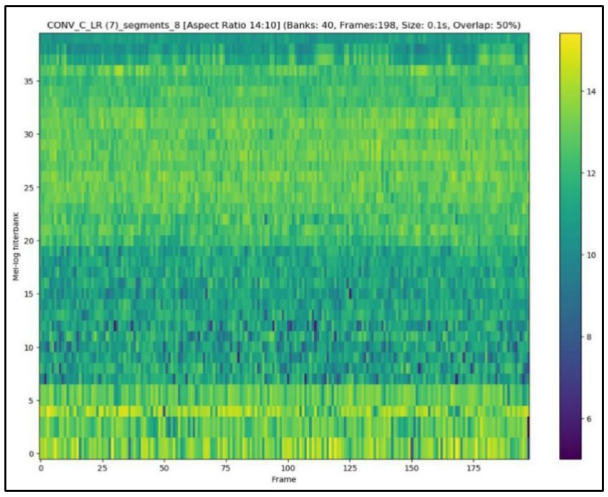


Figure 4.3 MLE Features of Chainsaw Activity

Figure 4.4 shows more chainsaw instances of MLE features, the observation has found that MLE features of forest ambience have a common higher intensity of MLE on the 1st to 6th consistent and medium intensity in 7th to 20th MLE throughout the clip. These can be an indicators of ambience sound events.

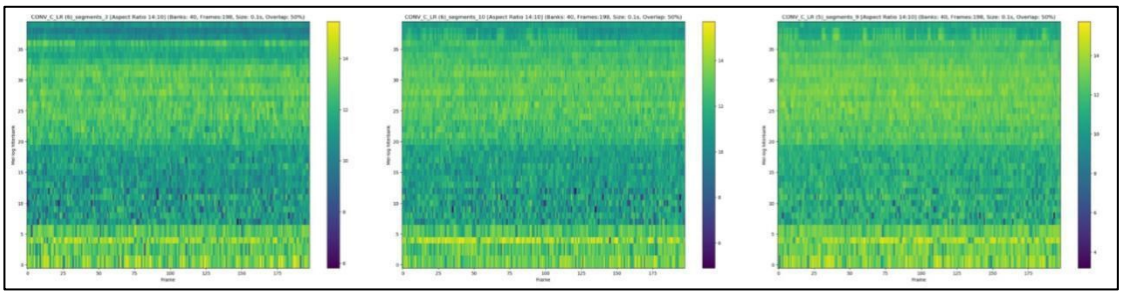


Figure 4.4 MLE Features of Different Chainsaw Activities

Figures 4.5 show the ambience forest sound while Figure 4.6 shows different instances of the samples side by side. The features show distinctive patterns on the top side of 0-10 and occasionally on 31-35 MLE Banks. The Hatchet class is like the natural ambience class MLE since the sound of the hatchet comes in intervals of about a second. These intervals can be seen as high intensity MLE in the 33rd to 36th MLE but only 1 or 4 instances can be observed in the MLE Heatmap. This might cause confusion and bring difficulty in differentiating ambient and hatchet sounds.

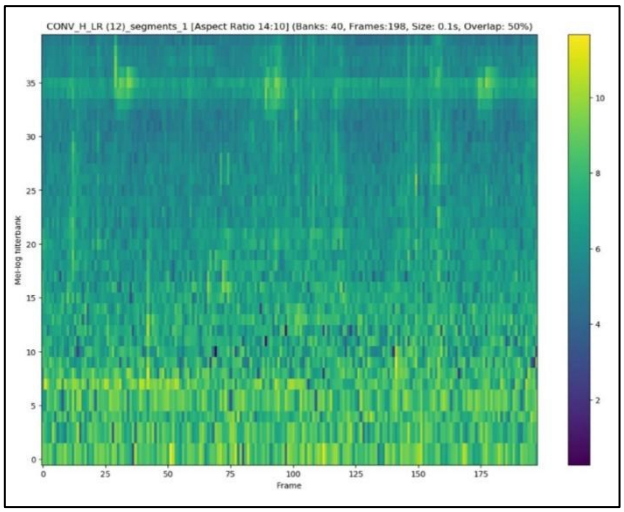


Figure 4.5 MLE Features of Hatchet Activity

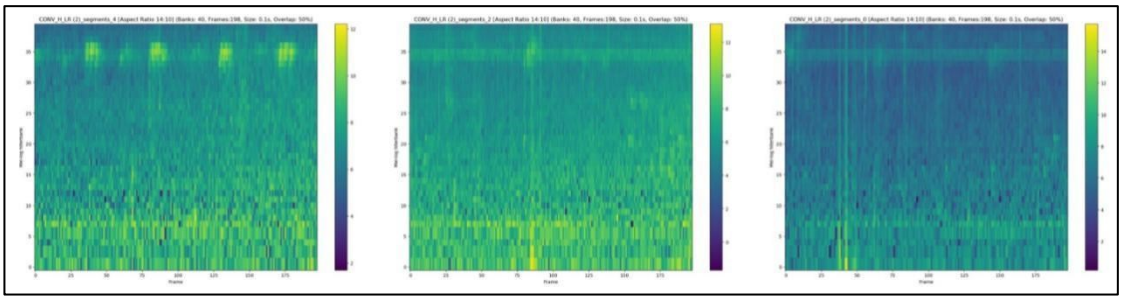


Figure 4.6 MLE Features of Different Hatchet Activities

The observation of the MLE features extracted showed distinctive patterns between sound event occurrences and was believed to be adequate for the ML class. It was found that some correlation between hatchet and ambience class is expected to cause detection difficulties. The MLE features were now ready to be used in model training and testing.

4.3 CNN Model Results

The CNN results consist of a series of models trained and tested using the datasets collected. The datasets for training and validation are a split of 70-30 from the in-sample datasets. The CNN is then tested with out-of-sample data and then the hyperparameter tuning to optimize the CNN model training.

4.3.1 Preliminary Model Search

The preliminary model search consists of a random model search where models were created and trained with standard best practices in search of an initial model design. The CNN results consist of a series of models trained and tested using the in-sample segment of the datasets collected. The datasets for training and validation are a split of 70-30 from the in-sample datasets. The out-of-sample testing were done on the best model amongst them. The out-of-sample data is unique compared to the in-sample data to produce a near real-world testing data. The training runs for 50 epochs on all models to find the most viable performing model before any tuning. The accuracy and loss were calculated based on the validation data (30% of in-sample data) for every epoch used as the training performance metric. The metrics were logged and plotted to observe the performance over each training epoch. The performance metrics of accuracy can be seen in Figure 4.7 meanwhile loss is illustrated in Figure 4.8. Both figures demonstrate the performance metric for the eight model which are 16-32-Conv-64, 32-32-Conv-64, 32-32-Conv-128, 32-64-Conv-64, 32-64-Conv-128, 64-64-Conv-64, 64-64-Conv-128 and 64-128-Conv-128. Figure 4.7 shows that the model 32-64-Conv-128 achieved the highest accuracy 99.32%. The other model obtained between 85% and 99%. As can be seen in the Figure 4.8, model 64-64-conv-128 has the most noticeable spikes jumping in validation loss from 28th to 29th epoch from 0.05491 to 0.5494 but it was not considered insignificant. For an exact number Table 4.1 was created to compare model results. The best conditions for validation loss were the lowest closest to 0.00 in Figure 4.8, while the best score for accuracy was closer to 1.00 in Figure 4.7.

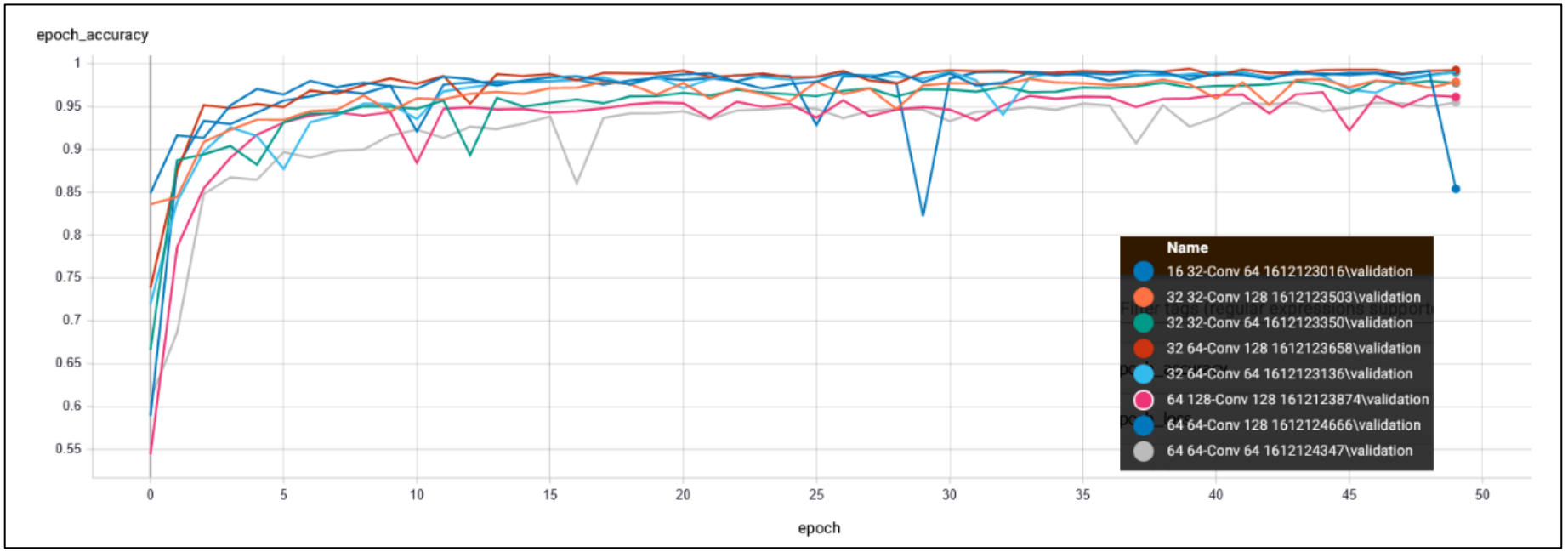


Figure 4.7 Accuracy on Validation of CNN Model

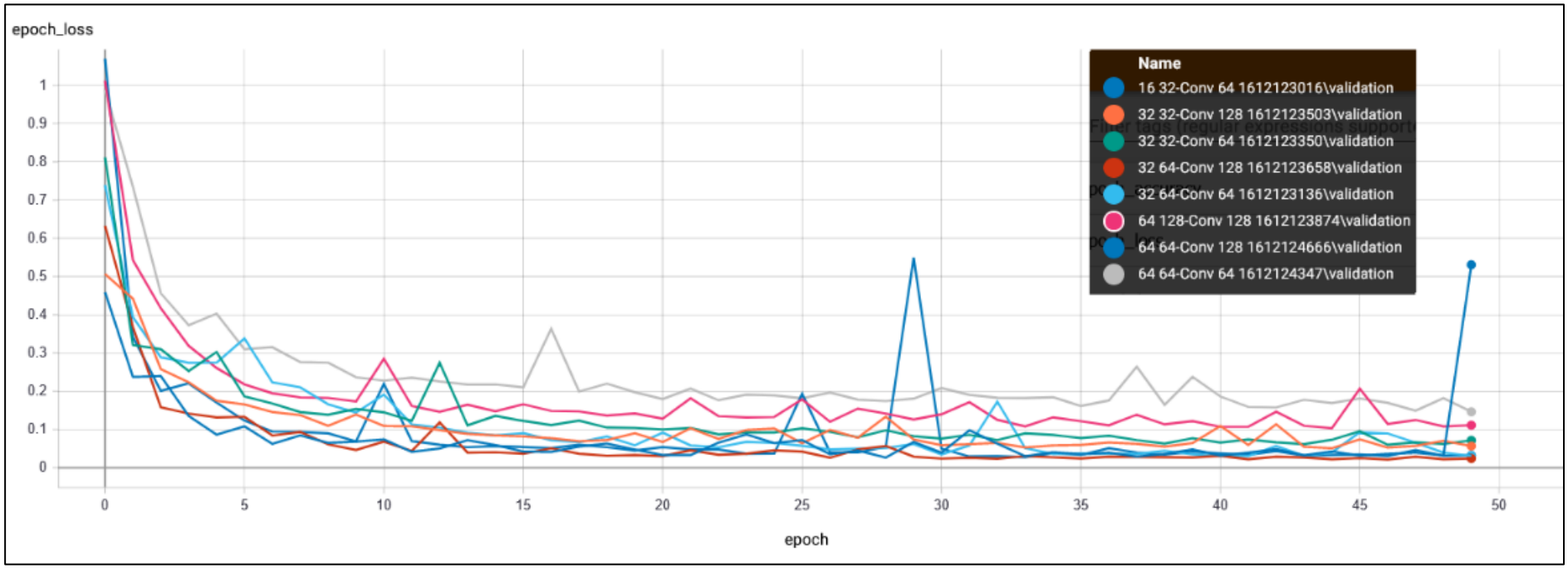


Figure 4.8 Epoch on Validation Data Loss of All Mode

It was found that exist large spikes on accuracy in Figure 4.7 and loss in Figure 4.7. The model 64-64-conv-128 has the most noticeable spikes jumping in validation loss from 28th to 29th epoch from 0.05491 to 0.5494 but it was not considered insignificant. This can be explained as the model was currently in a bad local minimum, but it immediately recovered on the 30th epoch showing that it was still training well. Table 4.1 shows the list of the CNN architectures trained with its best training performance metric of validation loss, accuracy and which epoch achieved those best metrics. The best loss obtained was the CNN model 32-64-Conv-128, with the validation loss of 0.0215 and accuracy of 99.32% at the 46th epoch. This model showed a very stable learning curve as observed in its accuracy in Figure 4.7 and loss in Figure 4.8 where there are smaller spikes found.

Table 4.1
CNN Model Optimization Results

Model Structure			Lowest Validation Loss Achieved	No of Epoch of lowest validation loss	Best evaluation accuracy achieved (%)
Convolutional Layers (2D)		Fully Connected Layer			
1st	2nd				
16	32	64	0.0266	28	95.52
32	32	64	0.0602	46	98.00
32	32	128	0.0509	44	98.22
32	64	64	0.0307	41	99.02
32	**64**	**128**	**0.0215**	**46**	**99.32**
64	64	64	0.1463	49	95.49
64	64	128	0.0277	33	99.04
64	128	128	0.1034	44	96.69

The model 32-64-Conv-128 was considered the best since it achieved a best loss of 0.02215 and was further investigated for overfitting. The observation on comparing the training and validation loss implied that it is not overfitting the training data from 70% in-sample data. The rule of thumb in detecting overfitting was the training loss and validation loss difference. The closer they are together shows that it is not overfitting. It was observed that in Table 4.1 there was no significant gap between training and validation loss. Hence, it was considered there was no overfitting in-sample dataset. In addition, Figure 4.9 is used to further investigate the model 32-64-Conv-128 to see the gap between training and validation loss. The model produces a small gap hence it is not considered overfitting. Validation loss is the same from Figure 4.8,

but each training loss was also recorded along the experiment and now placed together to observe the differences between them. Figure 4.9 shows the model 32-64-Conv-128 training and validation loss.

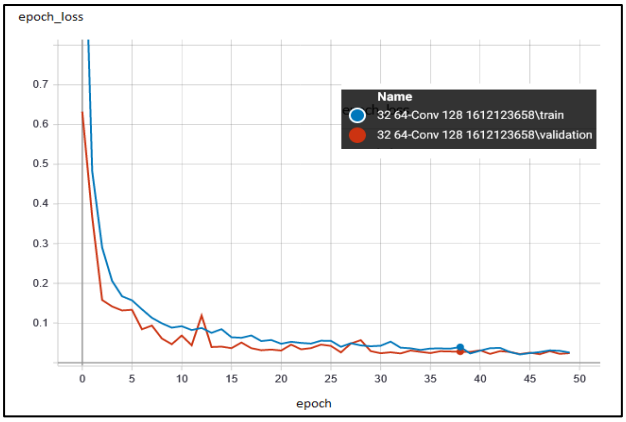
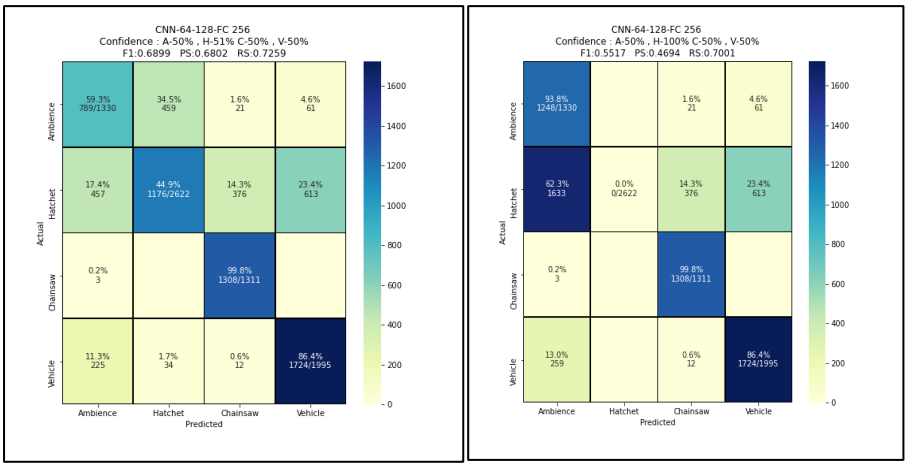


Figure 4.9 32-64-Conv-128 Training Versus Validation Loss

Then the CNN model of 32-64Conv128 was tested using out-of-sample data. The results are shown in Figure 4.10 (a) and (b)Figure 4.10 (a) shows the CNN results in a confusion matrix produced before any post-processing. The results produced from the training were considered unreliable based on the high false alarm rate of 34.5% on the Hatchet detection. Hence post-processing was applied to reduce false alarms. The Hatchet class detection was at 44.9% detection rate while having some confusion, misclassified classes to 17.4% as Ambience, 14.3% as Chainsaw and 23.4% as Vehicle.

Figure 4.10 (b) shows the confusion matrix after thresholding post-processing efforts to reduce false alarms on the Hatchet class only started to make changes at 100%. But F1 score was affected severely and the prediction on the Hatchet class was performing at its worst. The model was clearly overfitted to the in-sample data due to its gap between in-sample and out-of-sample accuracy of 99% and 69% respectively is a big gap. This shows that CNN does not work with a small dataset and will fail to generalize well. The CNN overfitting issue was expected beforehand. Therefore, data augmentation of shifting was employed to increase data amount and transfer learning was considered.



a) Before threshold b) After threshold

Figure 4.10 CNN-64-128-Conv-256 Results a) Before threshold and b) After threshold

4.3.2 In-depth Model Search

For in-depth Model Search, in-sample data was included with the training and validation was the out-of-sample data. Thus, the CNN could have more learning material for the training and observed the differences. The study conducted 250 epochs trained using all in-sample data and evaluated on all out-of-sample, seven models were trained and recorded each epoch evaluation. To get the best result, it requires high accuracy and lowest loss. The graphs in Figures 4.11 and 4.12 show the graph of accuracy and validation from each model. Each Figure contains two graphs, top and bottom. The top graph is the primary graph while the latter shows a smoothed and focused area of interest from the primary graph. The bottom graphs were smoothed using an exponential moving average by tensor board. The smoother graph view allows better clarity on overlapping models.

Figure 4.11 shows the epoch validation accuracy computed at the end of each epoch for all models. The bottom graph was enlarged on the area of interest between 0.7 and 9.5 accuracy to allow better observation. Figure 4.12 shows the epoch validation loss computed at the end of each epoch for all models. The results were focused on the best working models with the highest validation score. There are four models with more than 88% accuracy, namely the 16-32-Conv-64, 16-32-Conv-128, 32-32-Conv-64 and 32-32-Conv-128.

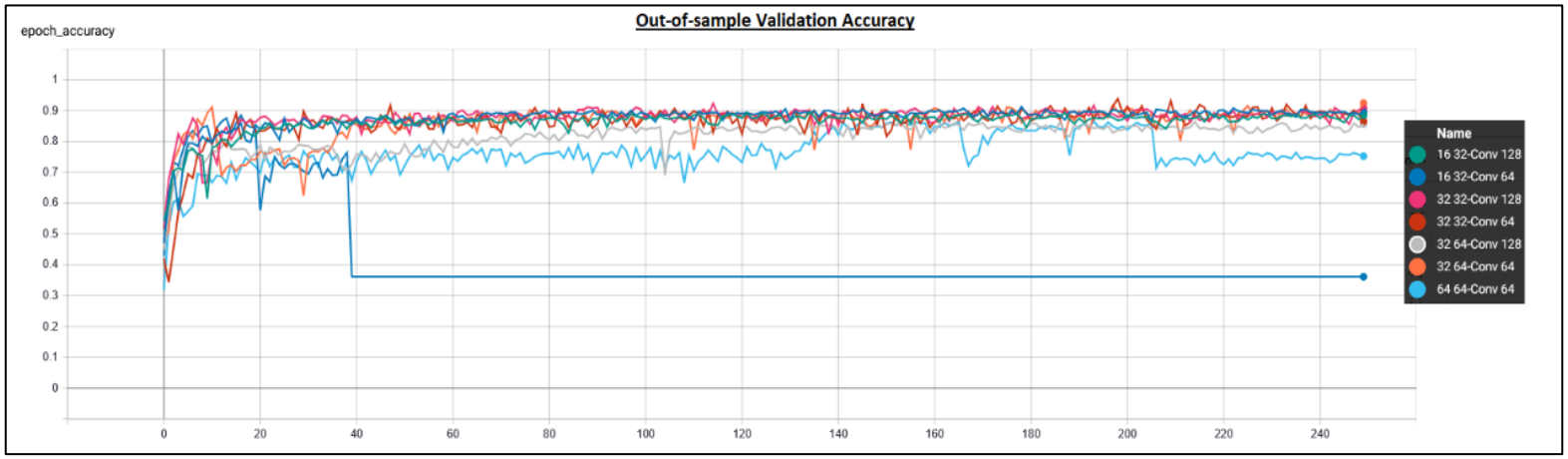


Figure 4.11 Out-of-sample Epoch Validation Accuracy Smoothed

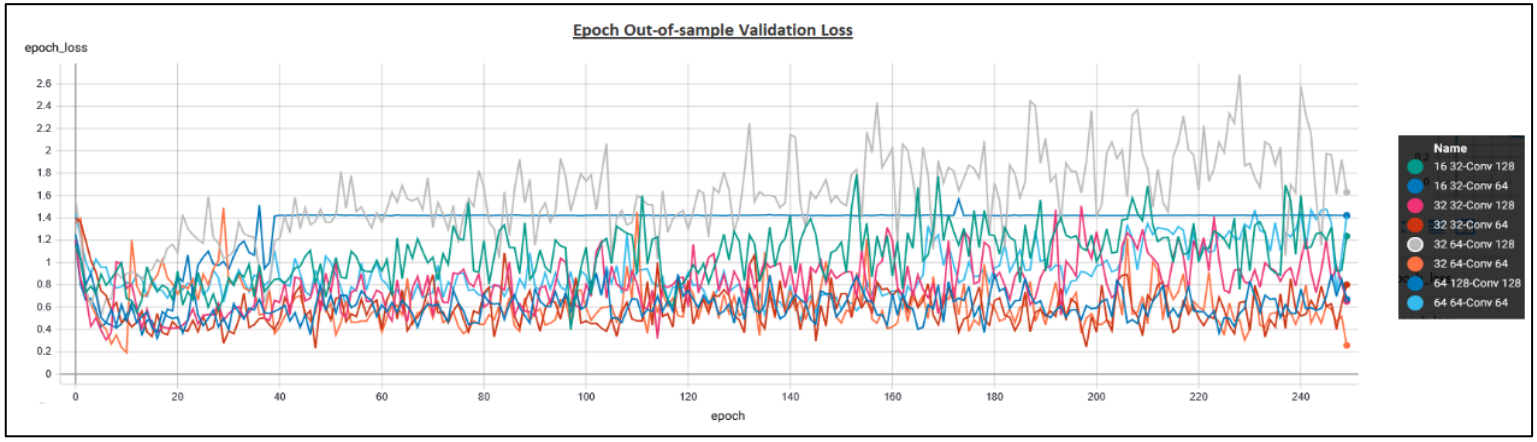


Figure 4.12 Out-of-sample Epoch Validation Loss Smoothed

Next, the validation loss was observed closer. Figure 4.13 shows the top-4 model validation loss graph of each training epoch smoothed graph by exponential moving average. The graph shows a higher loss of over 0.7 on the 16-32-Conv-128 and 32-32-Conv-128, indicating that the other two models performed better with loss reaching under 0.5. This is because lower loss produces a more quality prediction. The top-2 model observed were 16-32-Conv-64 and 32-32-Conv-64. However, the best model was 32-32-Conv-64 with both best scored lowest at 0.2425 loss and highest at 93.88% accuracy.

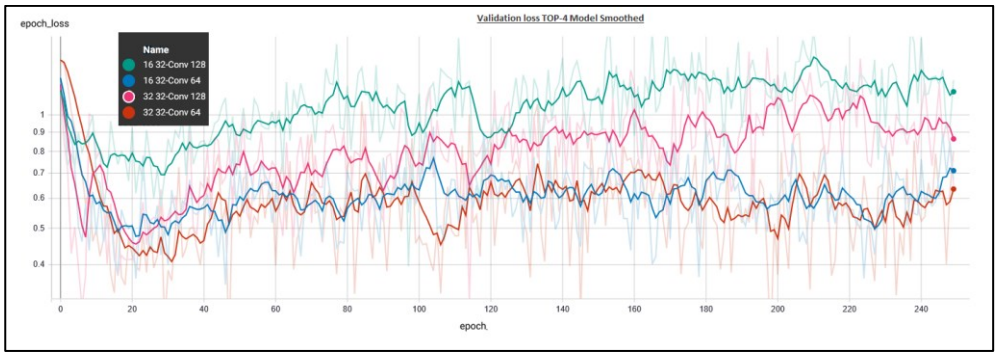


Figure 4.13 Epoch Validation Loss of Top-4 Model

As important as having high accuracy, it is also crucial to not to overfit the training datasets. Overfitting occurrences can be identified by comparing loss between training and validation. This gap existence implies an overfitted model. The model has learned the features in the training data too well but not improving validation loss is overfitted by the training data (Albert, 2019). The situation is called overfitting when the model over generalizes the training set, assuming no other pattern for a certain class exists outside of the training data. The results with higher accuracy after the overfitting epoch were considered as having low confidence and less reliable. The divergence of the training loss and validation loss shows if a certain model is overfitting.

Figure 4.14 shows the validation versus training loss and accuracy. Since the gap after the 10^{th} epoch showed a dramatic increase between training and validation loss, it was believed that the model had overfitted the training data. The models beyond this epoch cannot be considered reliable since it has overfitted the training data. The early stopping technique were applied to cope with overfitting. Hence, the models perceived as good when the training and validation loss gap conditions are at their minimum. The value of accuracy beyond the overfitting gap cannot be considered to

see how good the model is.

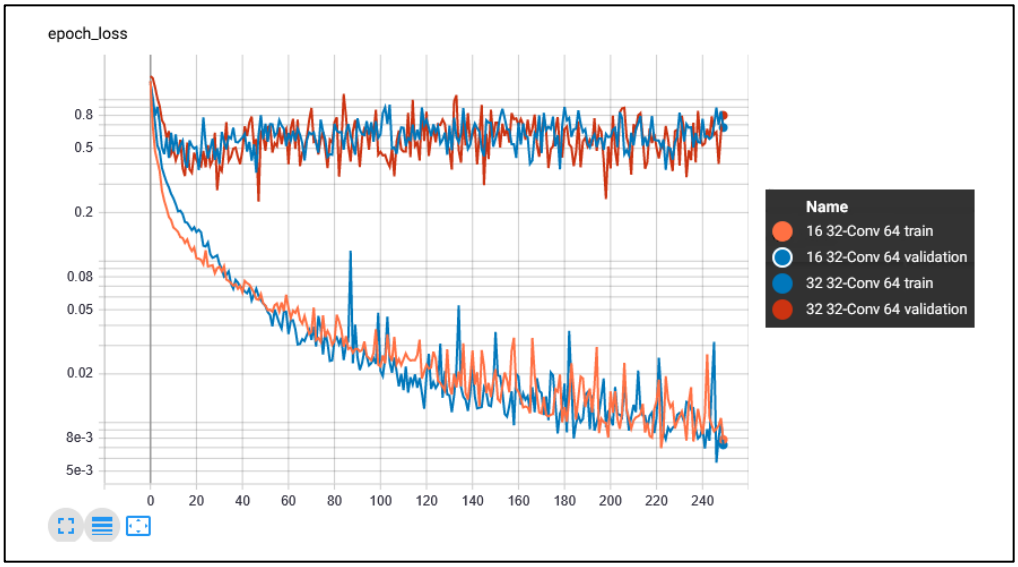


Figure 4.14 Epoch on Top-2 Model Training and Validation Loss Graph

The accuracy validation and training showed an improvement. However, since the loss graph has a gap in validation and training, it takes within a certain epoch before the epoch begins to overfit the datasets. The accuracy improvements on validation after the overfitting of training data are possible because the model was overfitting to the validation data. Figure 4.15 shows the training and validation accuracy comparison on top-2 models. The validation accuracy and training accuracy were seen as a significant gap for the models average of 88% to 99% training accuracy. This was another indication of overfitting when the training accuracy reaches about 98% at 30th epoch. It implies that the learning rate is high, and it converges fast.

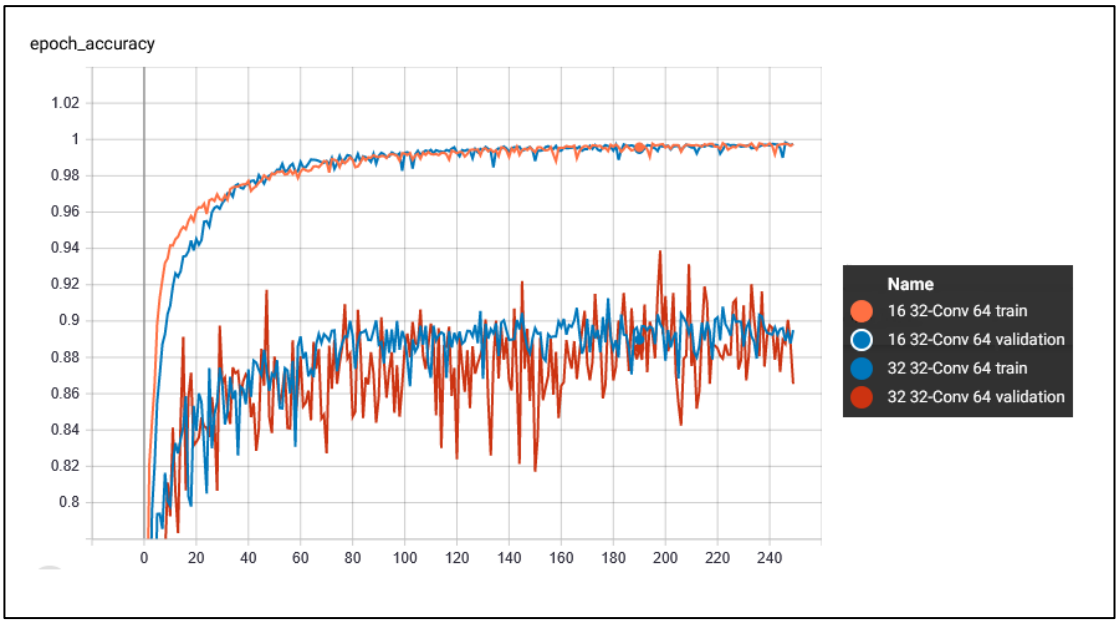


Figure 4.15 Epoch on Top-2 Training and Validation Accuracy Graph

In efforts to avoid overfitting models, this study believed an early stopping on training the model will help. Figure 4.16 shows the early epochs model loss graph comparison between train loss and validation loss of the top-2 models. To determine the epoch stopping condition of the training session, an observation on the early results of the training and validation loss were inspected. It was found that at the first 15th epoch, the validation loss increases and stops improving. At the same time, the training and validation loss gap was minimal. Figure 4.16 shows the early epochs model accuracy graph comparison between train loss and validation loss of the top-2 models. After close inspection of the top-2 model, it was found that the most relevant epoch to stop was different between the top-2 model.

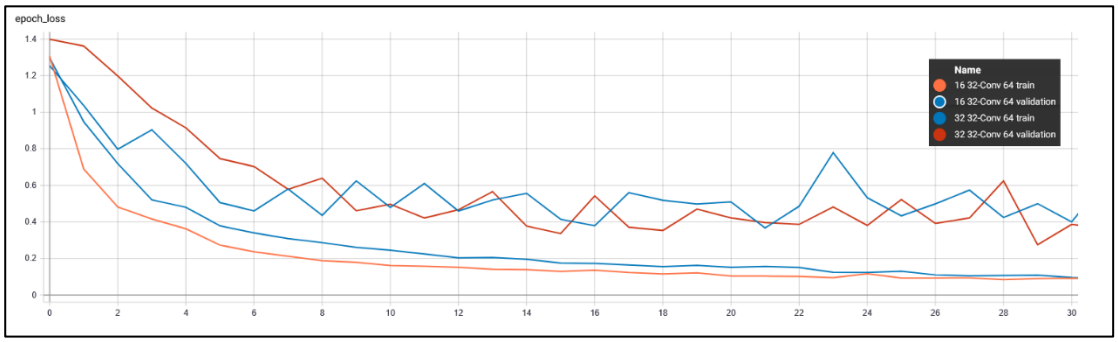


Figure 4.16 Early Epoch Loss of Top-2 Model Training Versus Validation Loss

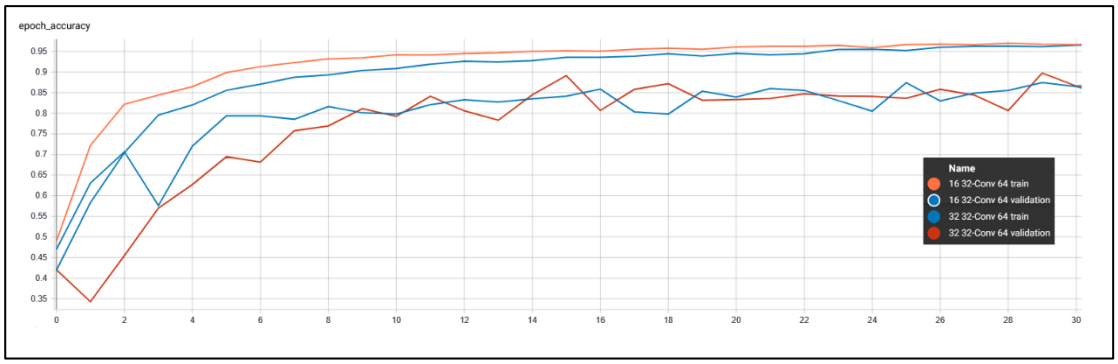


Figure 4.17 Early Epoch Accuracy of Top-2 Model Training Versus Validation Loss

A detailed observation of the results was tabulated in Table 4.2. The validation accuracy, validation loss, loss gap, training loss and early epoch stopping points of the models were stated as in Table 4.2. The loss gap is the gap between the validation loss and the training loss. Typically, a bigger loss gap indicates a worse case of overfitting. The model with the convolution of 16-32 and 32-32 was found to have a loss gap considered significant where the next effort was to reduce the loss gap further, in which 32-16 was proposed to reduce the parameters before the FC layers by half. It was considered to improve as lesser features reduced unwanted correlations. Hence a follow-up experiment was conducted to minimize the loss gap by lowering the FC layers as these layers were a probable cause of overfitting. Reducing memory capacity will enforce learning and avoid overfitting the datasets in the FC layers. The reduction of 64 to 32 FC layers was applied to the model design.

The new model designs were trained and tested, and it was compared in Table 4.2 where the loss gap was found considerably improved. The performing model was 32-16-Conv-32, 32-32-Conv-32 with early stopping at the 26th epoch with the lowest validation loss while having a considerable loss gap. The models found performing best were next applied for hyperparameter tuning in the following experiment.

Table 4.2
Performance of CNN Models

Model	Stop Epoch	Validation Accuracy (%)	Validation Loss	Training Loss	Loss Gap
16-32-Conv-32	26	72.98	0.7487	0.7155	0.0332
32-32-Conv-32	**26**	**87.05**	**0.3824**	**0.2257**	**0.1567**
32-16-Conv-32	**26**	**85.26**	**0.4659**	**0.5236**	**0.0577**
16-32-Conv-64	21	86.02	0.367	0.1036	0.2634
32-32-Conv-64	29	89.75	0.2748	0.1087	0.1661
32-16-Conv-64	28	86.77	0.3066	0.1759	0.1307

Next, this study observed the results individually to identify the class prediction performance. It is shown in Figure 4.18, the confusion matrix at epoch 2 for 32-16-Conv-32. Since the model is overfitted, this study must address the problem by tuning hyperparameters to avoid overfitting and stop training before it occurs. The next point was to find the correct early stopping point before overfitting occurs. Then once not overfitting and generalized well, the results were considered more reliable.

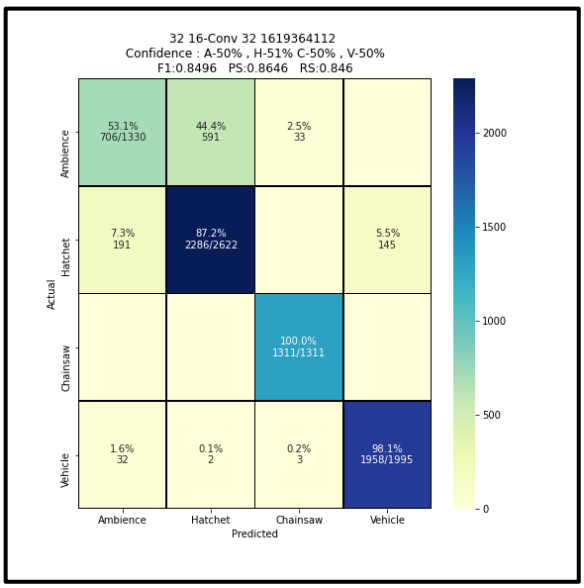
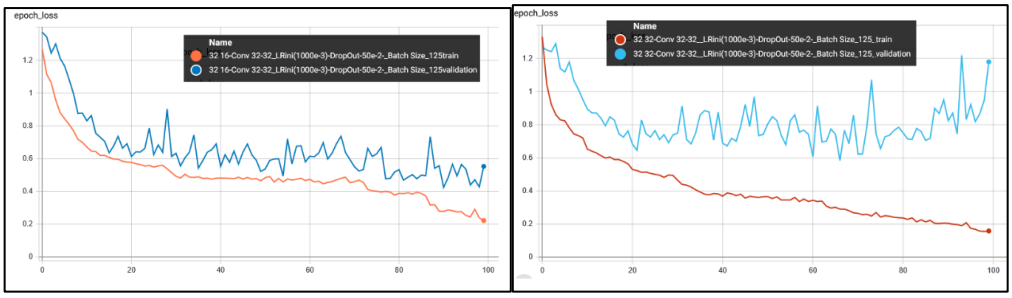


Figure 4.18 32-16-Conv-32 Results

4.3.3 Model Hyperparameters Optimization

From the results of experiment Two, the model cannot be not considered the most reliable since it has overfitted the training data. A few strategies were used to avoid overfitting. This study used hyperparameters by reducing batch size from 250 to 125, including a drop out layer of 0.5 on the dense layer, and adjusted the learning rate by decay staircase starting at 1.0 and reduces 90% at each epoch to reduce time to reach early convergence of training data on the early epochs. Reducing the kernel size of the final max pooling layer to 1 x 1 from 3 x 3 to minimize the complexity of convolution layers before the FC layer.

The architecture of 32-32-Conv-32 was still found overfitting on the update hyperparameters leading to a design change by reducing the 2nd convolutional layer 32 to 16 to reduce model capacity. This design was inspired by the VGG16 model (Simonyan & Zisserman, 2015) that has half the size of its previous convolutional layer. The new model 32-16-Conv-32 was compared with the previous model and has shown less overfitting. Figure 4.19 shows the comparison of both models on the gap between training and validation loss. Two graphs, left Figure 4.19 (a) 32-16-Conv-32 and Figure 4.19 (b) right 32-32-Conv-32, were compared for overfitting. The model of 32-16-Conv-32 was a clear success with less gap between training loss and validation indicating a less overfitted model compared to the other.



a) 32-16-Conv-32 b) 32-32-Conv-32

Figure 4.19 Loss Gap on Model a) 32-16-Conv-32 and b) 32-32-Conv-32

The model 32-16-Conv-32 has achieved validation loss at minimum of 0.42 at 54th epoch compared to the 32-32-Conv-32 model with just 0.58 at 66th epoch and continues to increase loss on further epochs. The 32-16-Conv-32 was the best performing model, and the confusion matrix was observed to see individual class prediction performance. Figure 4.20 shows the confusion matrix of the best model without any post-processing. The false alarm rate of 50% Ambience wrongly predicted as Hatchet class was very impractical. The post-processing of thresholding was 50% to 99% and found that 70% thresholding was applied to reduce false alarm to 10%.

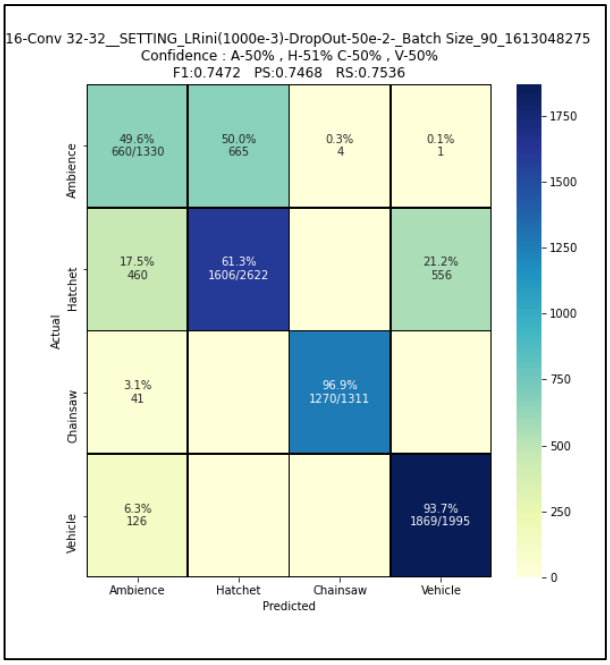


Figure 4.20 32-16-Conv-32 Confusion Matrix at 54th Epoch.

Figure 4.21 shows the confusion matrix of the model after thresholding at 81% on the Hatchet class. The results were improved with a score of F1 0.7472, precision 0.7468 and recall of 0.7536. The class performance after thresholding reduced the detection of the Hatchet class to 45.7% from 61.3%.

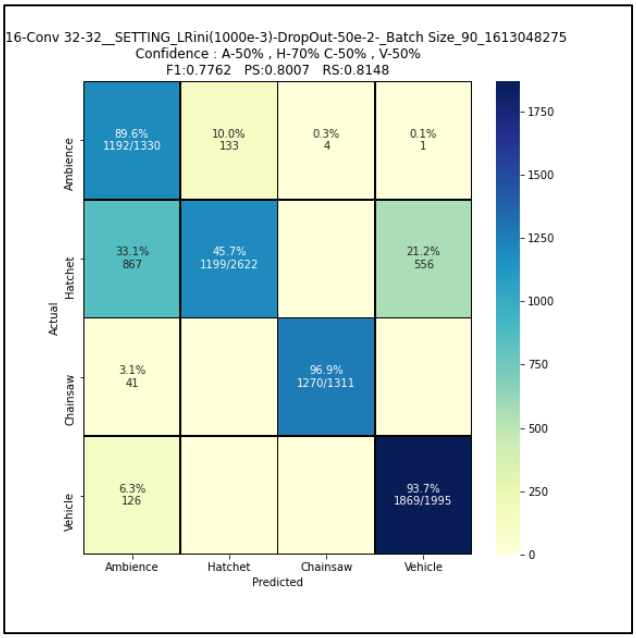


Figure 4.21 32-16-Conv-32 Confusion Matrix at 54th Epoch.

The best model CNN were concluded with the architecture of 32 convolutional layers with 16 convolutional layers and 32 fully connected dense layers yielding F1 Score of 0.7762.

4.4 Support Vector Machine Results

An experiment was executed using the recommended parameters and produced results of 79% accuracy, F1 Score 0.774, Precision 0.766, Recall 0.7966 and 36% FP rate. Next the post-processing was applied to reduce FP rate and achieved at 94% threshold 10% FP rate with the results of F1, precision, recall as 0.759, 0.768, 0.814, respectively. Table 4.3 shows the post-processing threshold results from 55 to 99% with their respective performance scores

Table 4.3
RF SVM Post Processing Results

Hatchet Threshold (%)	F1	Precision	Recall	Hatchet Threshold (%)	F1	Precision	Recall
50	0.775	0.767	0.797	75	0.778	0.768	0.811
51	0.775	0.767	0.798	76	0.778	0.768	0.812
52	0.776	0.768	0.799	77	0.778	0.769	0.814
53	0.777	0.768	0.800	78	0.777	0.769	0.814
54	0.777	0.768	0.800	79	0.777	0.769	0.814
55	0.777	0.768	0.801	80	0.777	0.769	0.815
56	0.778	0.769	0.802	81	0.778	0.770	0.816
57	0.779	0.769	0.803	82	0.777	0.770	0.816
58	0.778	0.769	0.804	83	0.777	0.771	0.817
59	0.780	0.770	0.806	84	0.777	0.771	0.818
60	0.780	0.770	0.806	85	0.775	0.770	0.817
61	0.780	0.769	0.806	86	0.776	0.771	0.818
62	0.780	0.769	0.807	87	0.775	0.771	0.818
63	0.778	0.768	0.806	88	0.775	0.772	0.819
64	0.778	0.767	0.806	89	0.775	0.773	0.820
65	0.779	0.768	0.807	90	0.774	0.773	0.820
66	0.778	0.767	0.807	91	0.773	0.774	0.821
67	0.779	0.768	0.809	92	0.771	0.773	0.820
68	0.779	0.768	0.809	93	0.765	0.770	0.817
69	0.778	0.768	0.809	**94**	**0.759**	**0.768**	**0.814**
70	0.778	0.768	0.810	95	0.757	0.769	0.815
71	0.778	0.767	0.810	96	0.754	0.769	0.814
72	0.777	0.767	0.810	97	0.753	0.773	0.816
73	0.777	0.767	0.810	98	0.748	0.773	0.814
74	0.775	0.767	0.797	99	0.736	0.771	0.808

The confusion matrix was computed to compare before and after post

processing. The FP was found to reduce FP from 36.1% to 10.1% at the cost of a reduction of Hatchet accuracy from 60.9% to 42.1%. Figure 4.22 shows the confusion matrix before and after the post processing the results.

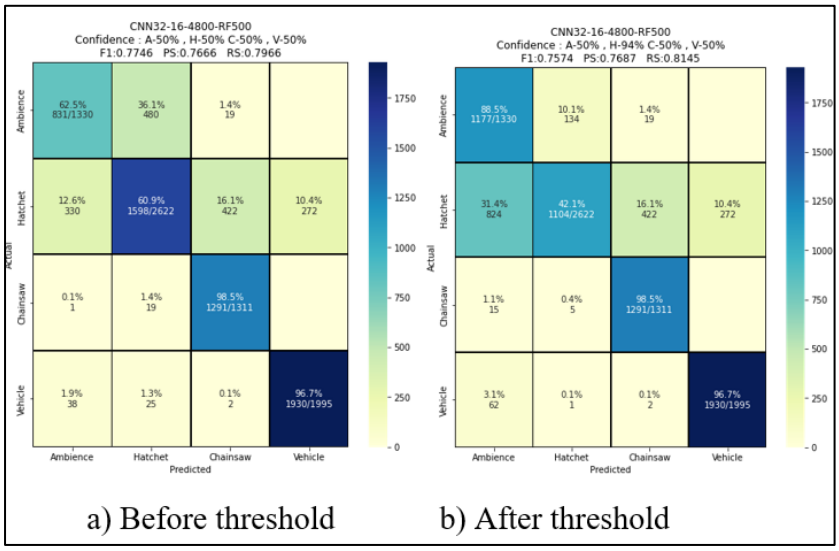


Figure 4.22 SVM Result a) Before threshold and b) After threshold

4.5 Random Forest Model Results

A series of experiments were executed in search of the best parameters for an optimized model with the best performing results. The parameters adjusted were the size of the ensemble, which is the number of DTs on the ensemble. Table 4.4 shows the performance result of multiple random forest models varying in ensemble size, from 10 to 1000. The performance of the model was tested by its accuracy, F1, precision and recall. The peak performance was achieved at the ensemble size of 500 with an accuracy of 72.08%, F1 of 0.7117, precision of 0.7137 and recall of 0.7202. The results were saturated at 72% at the ensemble size of 300 to 500. Performance started dropping at 1000 ensembles, where it was seemed that more ensembles were not always better but may cause it to perform worse. This research found that the model of 500 ensembles was the optimized model. The best model will be referred to as RF-500.

Table 4.4
RF Results for Accuracy, F1, Precision, and Recall between Ensemble Sizes

Ensemble Size	Accuracy (%)	F1	Precision	Recall
10	0.7074	0.6909	0.6954	0.7041
20	0.7167	0.7077	0.7095	0.7214
50	0.7159	0.7114	0.7119	0.7232
100	0.7180	0.7088	0.7097	0.7194
200	0.7202	0.7088	0.7097	0.7194
300	0.7195	0.7105	0.7119	0.7197
400	0.7200	0.7106	0.7121	0.7197
500	**0.7209**	**0.7117**	**0.7137**	**0.7202**
1000	0.7189	0.7104	0.7121	0.7188

The best performing model RF-500 was then placed through the decision-making step, also known as post-processing of thresholding. This step was to reduce the false alarm rate by increasing the threshold. The threshold is the minimum acceptance rate of the prediction confidence of a class. Instead of the typical 50% rate, this research executed a series of experiments as illustrated in Table 4.5. The cost of increasing the threshold will degrade the performance of the model to avoid the false detection problem. Hence the threshold should also be balanced with the performance reduction. It was found that the Hatchet class is the most confusing class with a high false alarm rate between the Ambience classes. Since the main objective of the model was to detect intruders, the false alarm rates can only be accepted at about 10%. Afterward, thresholding was done on the best model to reduce false alarm while keeping high accuracy. It was found that at 81% threshold, the model produced about 10% false alarm rate. The model suffered a reduction in performance due to post-processing.

Table 4.5
RF Post-processing Results

Hatchet Threshold (%)	F1	Precision	Recall	Hatchet Threshold (%)	F1	Precision	Recall
50	0.7116	0.7136	0.7203	75	0.7125	0.7430	0.7486
51	0.7116	0.7138	0.7206	76	0.7103	0.7420	0.7474
52	0.7110	0.7135	0.7204	77	0.7065	0.7408	0.7456
53	0.7114	0.7143	0.7213	78	0.7038	0.7403	0.7445
54	0.7129	0.7162	0.7234	79	0.7016	0.7404	0.7440
55	0.7122	0.7161	0.7234	80	0.6988	0.7396	0.7427
56	0.7137	0.7185	0.7260	81	0.6964	0.7406	0.7428
57	0.7148	0.7205	0.7283	82	0.6954	0.7433	0.7441
58	0.7149	0.7215	0.7295	83	0.6943	0.7464	0.7457
59	0.7169	0.7243	0.7325	84	0.6953	0.7522	0.7493
60	0.7183	0.7267	0.7351	**85**	**0.6961**	**0.7605**	**0.7537**
61	0.7197	0.7291	0.7377	86	0.6956	0.7694	0.7576
62	0.7215	0.7322	0.7409	87	0.6948	0.7772	0.7603
63	0.7227	0.7347	0.7435	88	0.6910	0.7844	0.7611
64	0.7249	0.7380	0.7469	89	0.6851	0.7903	0.7600
65	0.7262	0.7408	0.7497	90	0.6777	0.7918	0.7567
66	0.7272	0.7434	0.7522	91	0.6678	0.7918	0.7516
67	0.7274	0.7446	0.7533	92	0.6550	0.7925	0.7453
68	0.7265	0.7451	0.7535	93	0.6417	0.7923	0.7386
69	0.7263	0.7467	0.7548	94	0.6320	0.7927	0.7338
70	0.7248	0.7464	0.7542	95	0.6205	0.7912	0.7281
71	0.7222	0.7457	0.7531	96	0.6107	0.7900	0.7233
72	0.7196	0.7448	0.7517	97	0.5998	0.7887	0.7182
73	0.7173	0.7440	0.7505	98	0.5894	0.7875	0.7134
74	0.7152	0.7439	0.7499	99	0.5843	0.7870	0.7111

Table 4.6 shows the comparison between the model's performance before 50% and after post-processing 85%. The post-processing step has improved the model's F1 score by -1.55%, precision by +4.68% and recall by +3.35%. In other words, it has improved the RF-500 model's F1 score by 2.46%, precision by 0.30% and recall by 0.73%. The post-processing has improved the precision and recall but reduced the F1 score.

Table 4.6
RF Post-processing Before and After Results Comparison

Results	**F1**	**Precision**	**Recall**
Before Post-processing	0.7117	0.7137	0.7202
After Post-processing	0.69618	0.76045	0.75374
Comparison	-1.55%	+4.68%	+3.35%

Figure 4.23 shows the confusion matrix produced from the model after post-processing with a threshold of 85%. The 10.1% confusion between Hatchet and Ambience class was considered a false alarm. False alarm is when the prediction outcome is Hatchet, Chainsaw or Vehicles class but it is actually the Ambience class. The Hatchet class prediction rate was very low at 27.2% while the other class Chainsaw and Vehicle has very good results of above 80%. There was also some confusion between the Chainsaw and Vehicles class of 14%. The main concern was to avoid confusion on predicting the Ambience class wrongly as the Hatchet, Chainsaw and Vehicles class when false alarm in the prediction concludes the Ambience class as the Hatchet, Chainsaw and Vehicles class. False alarms must be minimized for the detection practicality, especially in surveillance work.

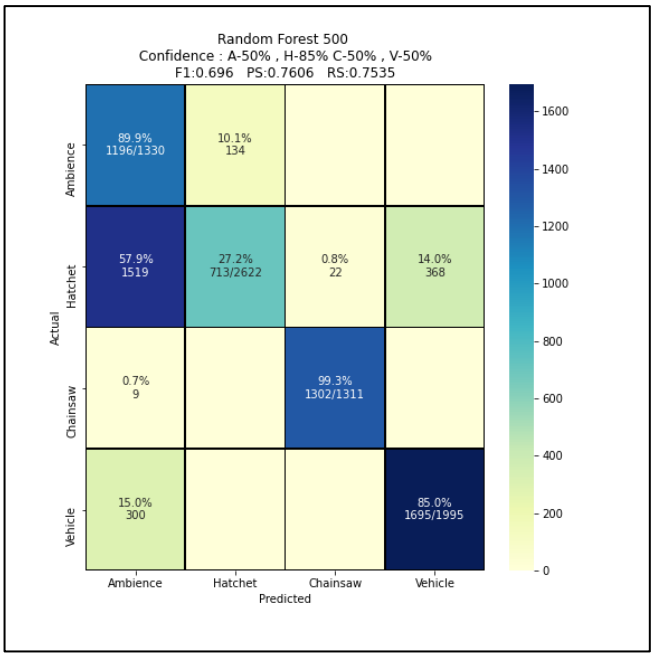


Figure 4.23 RF After Post-processing Confusion Matrix

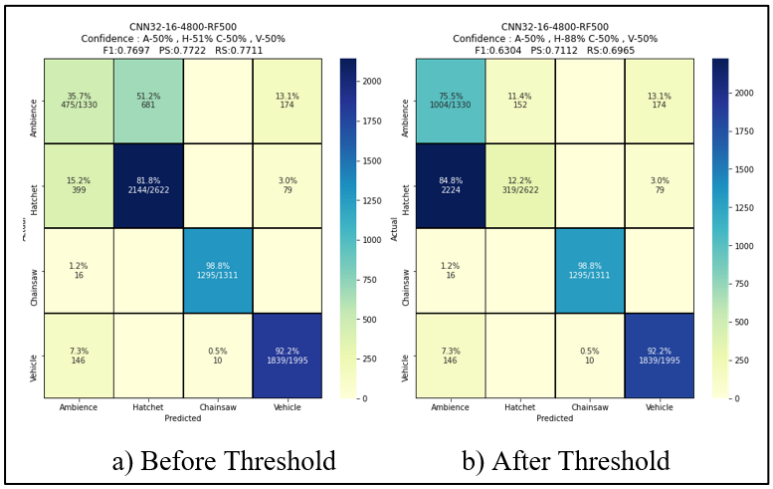
4.6 Selected Model CNN-RF Model Results

The CNN model 32-16-Conv-32 implemented in this experiment is the model produced from tuning hyperparameters in Subtopic 4.3.3. The weights trained on this model in the layer 32-16 convolutional layers were extracted to be used as a feature extraction layer for the hybrid CNN-RF model. The MLE features were computed using the 2 convolutional layers and fed as an input to the RF model. A series of experiments were executed in search of the best parameters for an optimized model with the best performing results. The parameters adjusted were the size of the ensemble, which was the number of the DTs on the ensemble. Table 4.7 shows the performance result of the multiple RF models varying in ensemble size, from 10 to 1000. The performance of the model was the accuracy, F1, precision and recall. The peak performance was achieved at the ensemble size of 500 with an accuracy of 78.12%, F1 of 0.7696, precision 0.7722 and recall of 0.7711. The results saturated at 0.7812 and the ensemble size of 300 to 500. Performance started dropping at 1000 ensembles, where it was seemed that more ensembles were not always better but may cause it to perform otherwise. This research found that the model of 500 ensembles was the optimized model.

Table 4.7
32-16-CNN-RF Results on Different Ensemble Sizes

Ensemble Size	Accuracy (%)	F1	Precision	Recall
10	77.05	0.7187	0.7220	0.7156
20	77.61	0.7454	0.7548	0.7460
50	77.59	0.7596	0.7610	0.7627
100	77.55	0.7614	0.7626	0.7643
200	77.93	0.7700	0.7726	0.7717
300	77.84	0.7693	0.7716	0.7708
400	78.02	0.7692	0.7715	0.7708
500	**78.12**	**0.7696**	**0.7722**	**0.7711**
1000	77.91	0.7698	0.7723	0.7710

Figure 4.24 is the confusion matrix of the best performing RF ensemble of 500. It was found that post-processing did not improve the overall performance. The major confusion was between the Hatchet and Ambience class as can be seen in the Figure below. Further illustrations demonstrate the thresholding value at each level in the experiments. Figure 4.24 (a) demonstrates thresholding results for 32-16 CNN-RF (a) before and (b) after. The FP rate 10% was achieved at 88% threshold but resulted in a significant reduction in the Hatchet class accuracy and F1 score. Overall performance of F1 Score and accuracy dropped to 0.6304 and 12.2%, respectively. Table 4.8 shows the thresholding done at 88%, reducing FP to 11.4% but dramatically reduced the Hatchet class accuracy from 81.8% to 12.2%.



a) Before Threshold b) After Threshold

Figure 4.24 32-16-CNN-RF Model results a) Before Threshold and b) After Threshold

Table 4.8
32-16-CNN-RF Thresholding Results

Hatchet Threshold (%)	F1	Precision	Recall	Hatchet Threshold (%)	F1	Precision	Recall
51	0.7697	0.7697	0.7697	75	0.6787	0.6787	0.6787
52	0.7673	0.7673	0.7673	76	0.6739	0.6739	0.6739
53	0.7648	0.7648	0.7648	77	0.6697	0.6697	0.6697
54	0.7633	0.7633	0.7633	78	0.6649	0.6649	0.6649
55	0.7604	0.7604	0.7604	79	0.6601	0.6601	0.6601
56	0.7580	0.7580	0.7580	80	0.6568	0.6568	0.6568
57	0.7542	0.7542	0.7542	81	0.6540	0.6540	0.6540
58	0.7496	0.7496	0.7496	82	0.6521	0.6521	0.6521
59	0.7463	0.7463	0.7463	83	0.6482	0.6482	0.6482
60	0.7436	0.7436	0.7436	84	0.6454	0.6454	0.6454
61	0.7399	0.7399	0.7399	85	0.6427	0.6427	0.6427
62	0.7363	0.7363	0.7363	86	0.6371	0.6371	0.6371
63	0.7328	0.7328	0.7328	87	0.6325	0.6325	0.6325
64	0.7282	0.7282	0.7282	88	0.6304	0.6304	0.6304
65	0.7248	0.7248	0.7248	89	0.6297	0.6297	0.6297
66	0.7208	0.7208	0.7208	90	0.6262	0.6262	0.6262
67	0.7175	0.7175	0.7175	91	0.6236	0.6236	0.6236
68	0.7135	0.7135	0.7135	92	0.6206	0.6206	0.6206
69	0.7085	0.7085	0.7085	93	0.6137	0.6137	0.6137
70	0.7017	0.7017	0.7017	94	0.6054	0.6054	0.6054
71	0.6968	0.6968	0.6968	95	0.5983	0.5983	0.5983
72	0.6943	0.6943	0.6943	96	0.5905	0.5905	0.5905
73	0.6900	0.6900	0.6900	97	0.5862	0.5862	0.5862
74	0.6855	0.6855	0.6855	98	0.5838	0.5838	0.5838
				99	0.5838	0.5838	0.5838

4.7 VGG16 CNN-RF Model Results

The CNN model VGG16 implemented in this experiment was the model acquired from ImageNet. The weights trained on this model in the convolutional layers were extracted to be used as a feature extraction layer for the hybrid CNN-RF model. The MLE features were computed using the 5 convolutional layers and fed as an input to the RF model. A series of experiments were executed in search of the best parameters for the best model with the best results. The parameters adjusted were the size of the ensemble and the number of DTs on the ensemble. The CNN architecture for hybrid design uses a fixed design that is based on the VGG16 model with the weights that are pre-trained. The results were listed on Table 4.8 VGG16 CNN-RF Results, it is found that at the ensemble size of 400 trees has produced best results F1 score of 0.8250, precision 0.8370 and recall 0.8187.

Table 4.9
VGG16 CNN-RF Results on Different Ensembles

Ensemble Size	Accuracy (%)	F1	Precision Score	Recall
10	77.25	0.7781	0.7885	0.7724
20	78.55	0.7982	0.8072	0.7931
50	79.33	0.8158	0.8251	0.8109
100	79.64	0.8194	0.8299	0.8138
200	79.62	0.8200	0.8312	0.8143
300	79.65	0.8218	0.8331	0.8159
400	**79.69**	**0.8250**	**0.8370**	**0.8187**
500	79.68	0.8240	0.8358	0.8180
1000	79.76	0.8229	0.8332	0.8177

The Figure 4.25 is the confusion matrix obtained by the 400 ensemble RF that has performed optimally compared to the other ensemble values. The FP rate on the Hatchet class was found to be high at 30.8%. This displays a 30% misclassified of the Ambience class as the Hatchet class detection false alarm. Even though with the accuracy of 94.6%, the Hatchet class was considered very good, the FP rate of 30% was not ideal for surveillance purposes. Post-processing of thresholding was applied to reduce FP.

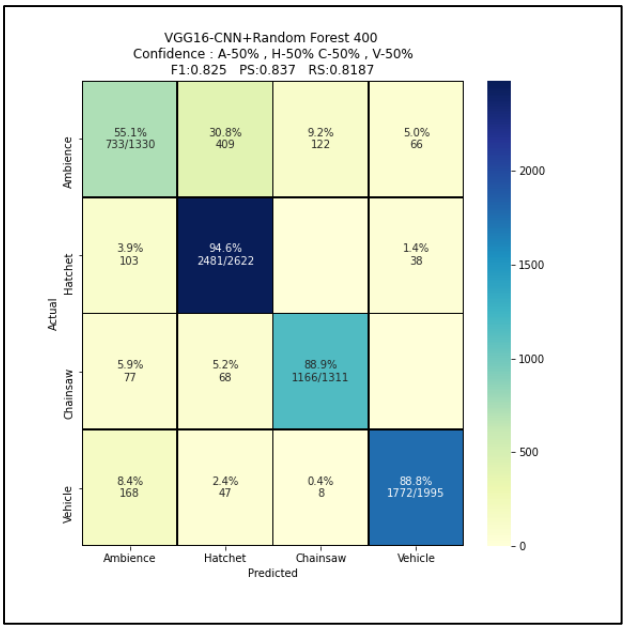


Figure 4.25VGG16 Based CNN-RF results with 400 Ensembles

After post-processing, the best model maintained a good performance while reducing false alarm. It was found that at 75% threshold the model produced under 10% false alarm rates while maintaining its performance. The threshold reduced all false alarm rate to about 10% while maintaining performance. All thresholds were computed and listed in Table 4.10 to find the optimal solution of under 10% false alarm rate.

Table 4.10
VGG16 CNN-RF Thresholding Results

Hatchet Threshold (%)	F1	Precision	Recall	Hatchet Threshold (%)	F1	Precision	Recall
50	0.8250	0.8370	0.8187	**0.75**	**82.15**	**0.8280**	**0.8297**
51	0.8254	0.8362	0.8195	0.76	82.07	0.8284	0.8298
52	0.8249	0.8340	0.8196	0.77	81.97	0.8285	0.8294
53	0.8262	0.8341	0.8212	0.78	81.88	0.8292	0.8296
54	0.8278	0.8348	0.8233	0.79	81.75	0.8299	0.8294
0.55	82.80	0.8340	0.8238	0.80	81.49	0.8287	0.8272
0.56	82.91	0.8342	0.8254	0.81	81.33	0.8290	0.8266
0.57	83.05	0.8348	0.8273	0.82	81.00	0.8281	0.8242
0.58	83.04	0.8339	0.8276	0.83	80.71	0.8276	0.8223
0.59	83.18	0.8347	0.8295	0.84	80.33	0.8263	0.8193
0.60	83.20	0.8344	0.8303	0.85	80.00	0.8250	0.8167
0.61	83.21	0.8339	0.8310	0.86	79.51	0.8239	0.8130
0.62	83.10	0.8325	0.8304	0.87	79.15	0.8228	0.8102
0.63	83.11	0.8324	0.8310	0.88	78.69	0.8216	0.8066
0.64	83.05	0.8317	0.8308	0.89	78.18	0.8201	0.8025
0.65	82.93	0.8305	0.8301	0.90	77.53	0.8176	0.7971
0.66	83.03	0.8314	0.8316	0.91	76.98	0.8164	0.7929
0.67	83.10	0.8321	0.8331	0.92	76.57	0.8154	0.7897
0.68	83.08	0.8321	0.8336	0.93	75.92	0.8134	0.7845
0.69	82.98	0.8314	0.8333	0.94	75.11	0.8113	0.7782
0.7	82.88	0.8308	0.8329	0.95	74.22	0.8092	0.7714
0.71	82.67	0.8293	0.8315	0.96	72.79	0.8055	0.7604
0.72	82.65	0.8298	0.8321	0.97	71.00	0.8014	0.7473
0.73	82.38	0.8281	0.8302	0.98	68.01	0.7956	0.7266
0.74	82.22	0.8276	0.8294	0.99	0.6282	0.7880	0.6947

Table 4.11 shows the comparison between the model's performance before and after post-processing. The post-processing step has improved the model's F1 score by 2.46%, precision by 0.30% and recall reduced by 0.73%. The post-processing has improved the overall performance without affecting the performance on F1, precision and recall scores.

Table 4.11
Comparison of RF performance Before and After Thresholding Results

Results	F1	Precision	Recall
Before Post-processing	0.7969	0.825	0.837
After Post-processing	0.8215	0.828	0.8297
Comparison	+2.46%	+0.30%	-0.73%

Figure 4.26 shows the confusion matrix results of the CNN-RF model. The confusion matrix shows the performance of each class. The results after post-processing reduced the false alarm rate to about 10% and maintained the prediction rate of the Hatchet class at 77.5%, Chainsaw class at 88.9% and Vehicle class at 88.8%. The CNN-RF results have improved greatly compared to RF results on Figure 4.25.

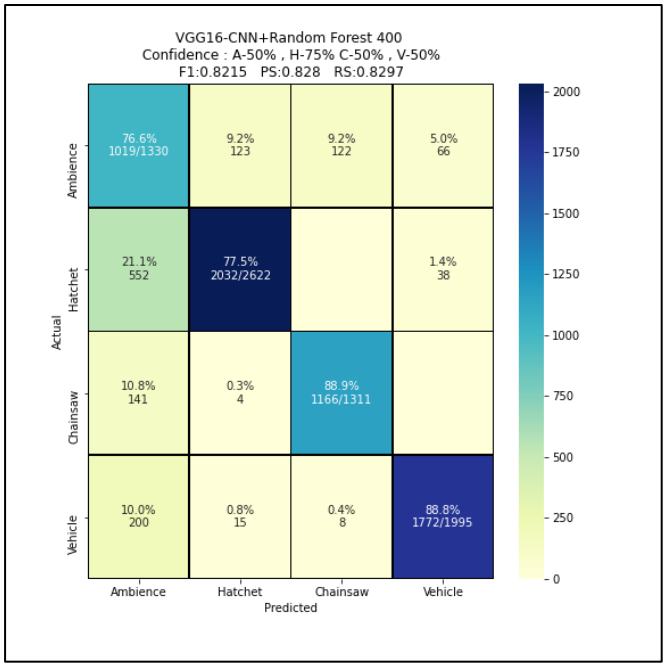


Figure 4.26 VGG16 CNN-RF after threshold

The collective results showed a significant improvement compared between the RF and the CNN-RF algorithm. Table 4.12 shows the comparison of approaches and performance improvement between them. The CNN-RF has 12.55% more F1 score compared to the RF. The ensemble size lowered afterward to produce best results shows that it was more efficient as it uses smaller ensemble size. The CNN-RF achieved less than 10% FP rate at 75% the Hatchet class threshold compared to 85% on the RF.

Table 4.12
Comparison Between VGG16 CNN-RF and RF

Algorithm	Key performance scores			Threshold (%)	Individual class performance		
	F1	Precision	Recall		Hatchet	Chainsaw	Vehicle
RF	0.696	0.761	0.754	85	27.2	99.3	85
VGG16 based CNN-RF	0.822	0.828	0.83	75	77.5	88.9	88.8
Difference	+0.126	+0.067	+0.076	-10	+50.3	-11.6	+3.0

The accepted threshold value setting was the value when the FP rates reached about 10%. The FP rate was essential for practicality, without it the algorithm cannot be considered reliable. Another aspect to consider was the ensemble size contributes to more computational time requirements. Small ensembles size should be relatively more efficient. However, this study does not include the in-depth evaluation in this aspect.

4.8 Model Performance Comparison

The performance of each model was compared before post-processing thresholding and after. The result before thresholding indicates the raw performance on detection while after thresholding shows how it is adequate in a real-world scenario where false alarms are damaging to the task. Table 4.3 shows the raw prediction results without post-processing. The best performing model seen was the CNN-RF with VGG16 and the runner up being the selected CNN-RF followed by the CNN.

Table 4.13 shows that the CNN-RF was a clear winner with 0.8250 F1 score in terms of overall performances, this method adapts from the VGG16 pre-trained model that is well optimized with millions of images and re-purposed into this study. The transfer learning approach requires less effort compared to custom CNN. However, this approach can be adapted for a quicker setup environment if needed as it performs better on some classes but has reduced accuracy on other classes. It was noted that the efforts to optimize a custom CNN was more difficult to optimize due to overfitting issues and lack of data. The selected CNN-RF was second best with 0.7698 F1 Score and was expected due to the amount of data it was trained on was beyond comparison of the VGG16. The RF approach achieved 0.6960 F1 score which is better compared to the

SVM, and it also requires less intensive configuration and tuning to optimize. The ML approaches were expected to yield lower results due to its limited capabilities compared to the DL approach.

Table 4.13
Comparison Between Models Without Thresholding

Algorithm	Key performance scores		
	F1	Precision	Recall
VGG16 CNN-RF	**0.825**	**0.837**	**0.819**
RF-500	0.712	0.714	0.720
32-16 CNN-RF	0.770	0.772	0.771
32-16-32 CNN	0.770	0.772	0.771
SVM	0.774	0.766	0.796

The prediction accuracy was the performance of the model. The reliability of the model needs to be proven. Hypothetically this study believed the model should perform well with a minimal false alarm rate. An assessment to see the reliability of the model in a surveillance perspective was required. Table 4.14 shows the comparison in performance between models after post-processing the prediction. In accordance with this, this study listed out the scores where the thresholding effort produced lower than 10% false alarm rate as seen in Table 4.14. Based on the FP, the misclassification of classes between the Ambience class can be identified.

Table 4.14
Comparison Between Models with Post-processing

Algorithm	Key performance scores			False alarm rate (approx. 10%)	Individual class performance Accuracy (%)		
	F1	Precision	Recall	Reliable Threshold (%)	Hatchet	Chainsaw	Vehicle
VGG16 CNN-RF	**0.8215**	**0.8280**	**0.8297**	**75**	77.5	88.9	88.8
32-16-conv CNN-RF	0.6304	0.7112	0.6965	88	12.2	**98.8**	92.2
RF-500	0.6960	0.7606	0.7535	85	27.2	99.3	85.0
(32-16-conv-32) CNN	0.7762	0.8007	0.8018	80	**80.0**	**99.8**	86.4
SVM	0.7594	0.7679	0.8144	94	43.2	98.5	**96.7**

In the result observation, it was found that on average, the VGG16 CNN-RF model was performing best with F1-Score 0.8215, but it also reduced the performance of individual classes accuracy, the Chainsaw and Hatchet class detections to 88.9% and 88.8%. The 32-16-Conv CNN-RF model acquired the best results in detecting the Chainsaw and Vehicle class with 98.8% and 92.2% accuracy. The combination of the CNN and the RF can improve the individual class accuracy of the Hatchet class. Its relation seemed to improve and reduce the performance of individual class prediction, ultimately improving overall performance, which was very important since inconsistent performance of models for different classes was a known problem. This study believed that a properly well-trained CNN on sound data specifically can improve detection methods as the feature extraction layer as used in this study. The VGG16 was well-trained, but within the domain of images, perhaps a well-trained CNN for the SED can be established using more variety of real datasets in the future.

4.9 Discussion

The goal of the study was to explore the SED for the application of wildlife reserve security in a forest environment. It was found through the study that the SED can soon be use in the field as it produced up to 0.8215 F1 Score with individual class prediction accuracy of the Hatchet class at 77.5%, the Chainsaw class at 88.9% and the Vehicle class at 88.8%. It was noted that a specific sound event can produce different performance of detection in many techniques' due to the nature of sound (Serizel et al., 2020). On the model performance, the application of the technology to be implemented for the industry might be a reality soon. It should be noted that this study has examined a limited number of three types of sound events and sources taken in a controlled manner (Explained in Chapter 3, Phase 1: Sound Data Collection). There are possibilities of unknown factors that could contribute to the prediction performance. The data were real-world data collected that simulated the intrusions well that a typical human can tell the event by listening to the audio clip. However, the main concern was to detect an intrusion and not the actual intrusion on a specific class if it is seen from a different perspective emphasizing surveillance. The Ambience class is the most crucial class as the sound of the ambient forest should not be considered an intruder of any sort.

This research explored three methods namely the CNN, the RF and the SVM. A proposed method of the CNN-RF was exclusively established and explored for the SED solution in a forest environment. In these experiments, training a CNN was a complex process with many variables. The results found on the CNN-RF model shown considerable improvements from other models with results of 0.7762 F1 Score. However, the loss function of multi-class cross-entropy was 0.42, which shows that it needed more data to improve prediction quality. In contrast, the RF showed better results and further improved with the CNN-RF model. The CNN-RF achieved an 82% F1 score outperformed all other models by about 5% to 9% more. This is due to the CNN capable of a good feature extractor and RF can assist in avoiding overfitting. Besides that, MLE feature used in the study is a 2-Dimension (2D) array of floating numbers. This can be said that it is an image of hotspots based on the MLE pattern analysis done on the collected sounds. The CNN 2D layer extracts more spatial features allowing the RF to improve performance with the tuned feature by the hybrid portion

of CNN.

Observations found an anomaly of difference in sound event class performance, especially at the Hatchet class. It always shows a lower detection regardless of any models. The nature hatchet sound was different compared to others, instead of a long sustaining event like the others it was in multiple bursts. The results show that the model can detect the hatchet event at the rate of 77% accuracy and high FP rate at 30%. The other sound events provide better results of more than 85% and low FP under 10%. Thresholding reduced the FP of the Hatchet class from 30% to 9% as demonstrated in Section 4.6 (Figure 4.26). The thresholding methods were optimized on individual classes tailored to their respective difficulty helping to reduce FP.

Each model suffered a high degree of false prediction rate, confusing between ambience's events with intruders' events. The false prediction was not aligned with the intended purpose of this research, that is to be used as a security surveillance system. Hence, a post-processing layer was considered mandatory. The raw result was considered too loose with false prediction rates. Thresholding post-processing was applied to the point of approximately 10% false prediction on any intruder events. When a prediction has a low degree of confidence favouring 51% over the other 49%, it cannot be considered a good prediction (TaheriNejad and Jantsch, 2019). Hence, increasing the threshold will avoid this problem. The variable threshold level was optimized until the target of 10% false detection rate was achieved.

With the best-performing model's results, the (VGG16) CNN-RF can be considered reliable for security in wildlife reserves. However, the findings did not imply for all cases of the SED. The lack of diversity in the data for the training meant that we cannot for certain assume it would work on other uncontrolled scenarios. Further research is required to set the foundations to implement the use of the SED for industrial needs. The research on the SED applications for the industry is growing, with many researchers exploring this section of the SED. The most noticeable movement includes the DCASE Challenge 2021 by the TUT. With more efforts on the SED research, the future will allow for many practical use cases for many industry sectors.

CHAPTER FIVE
CONCLUSION AND RECOMMENDATIONS

5.1 Introduction

This chapter concludes the research, the significant contributions that were successfully achieved and the important issues addressed for future research.

5.2 Thesis Summary

This dissertation has investigated the solution to the SED in the wildlife environment problem. In this research, the aim was to solve the SED solution in wildlife environments with the ML technologies. This research has successfully achieved its aim and the derived objectives.

Four objectives of the research were presented. The first objective was to identify significant patterns of features in sound between classes. Literature review was done to find a suitable feature extraction method for the SED. Previous studies demonstrated the MLE feature extraction method is most suitable. This study involves the MLE features extraction and its analysis. It was discovered and confirmed that the MLE features of collected sound data have distinct patterns with low correlation between classes. Hence, the features were significantly viable for the ML/DL methods for the SED solution in the forest environment.

The second objective was to determine the sound events detection technique that is suitable for surveillance in the forest environment. The RF, the CNN, the SVM and the CNN-RF models were found suitable for the SED solution in forest environments by literature review.

The third objective was to propose a hybrid solution for sound event detection using a real forest environment condition for surveillance. Four phases of methodology were done namely data collection, feature extraction, features analysis, and audio classification. Data collection was performed in a forest environment. Sound features of the MLE was extracted. The MLE features were analysed and used for classification. The classification of the sound event using a solution of the CNN-RF was proposed for

four class labels of chainsaw, vehicle, hatchet, and ambience activity.

The fourth objective was to evaluate the performance of the proposed solution. The final phase of the methodology evaluated models recommended by recent studies were created and evaluated. The models, including the CNN, the RF, the SVM and the CNN-RF were evaluated based on the F1-Score, precision and recall scores hence compared. Additionally, the models were also evaluated with the post-processing of thresholding to investigate another perspective. The CNN technique was believed to be relevant having superior pattern analysis with spatial awareness. Sound features that bring and impact the SED were the MLE because it contains the frequency domain information with a human-like perspective of sound.

This study also discovered that many algorithms and techniques in solving the SED had shown possible industrial applications' reliability. The CNN-RF has been proven to show an overall improvement in performance. It was also found that it also requires less configuration and optimization efforts due to the capabilities of the CNN transfer learning. The RF was recommended to be a suitable classifier in the SED task for forest environment compared to the other as mentioned in literature review. It indicates the benefits of a hybrid approach in tackling the SED in the domain. The models were all measured using the F1, precision and recall evaluation metrics. The CNN-RF model shows the best performance that obtained 82% F1 score. It was discovered that high FP rates of about 30% on the Hatchet class.

A post-processing method of thresholding was applied on the prediction results to assist in reducing the FP rate. Thresholding managed to reduce FP rates from 30% to 9% with an accuracy penalty from 94.6% to 77.5% on the Hatchet class. The VGG16 based on the CNN-RF model and thresholding combination has achieved considerably reliable performance for surveillance applications with 80% accuracy average and less than 10% FP rate. This study suggests that more research is required in SED with real-world sound samples to understand the contributing factors such as different sound events, environment, noise, location and weather. There are many paths to improve the SED for industrial application that can be considered.

5.3 Contributions

The contributions emerged from this study are as follows:

i. An enhanced solution has been produced to contribute to the resolution of the SED in the Malaysian forest environment. The solution is the CNN-RF for real-world intrusion SED.

ii. Establishment of intrusion sound event dataset in Malaysian forest environment.

5.4 Limitation of the research

The limitation of the research are as follows:

i. Data collection is a costly and resource-intensive task, and the total amount of data collected is based on the available time available, the time, and the funding allocated.

ii. It is challenging to collect data variations in forest environments due to access restrictions to deeper forest areas as it raises safety concerns.

iii. In a forest environment, recording equipment must be portable and run on batteries. The recording quality may not be the highest quality.

5.5 Recommendations

This research has brought up many issues that need further investigations. It is recommended that further research should be undertaken in the following areas:

i. Investigate the use of noise cancellation to isolate the featured sound events. The environmental sounds sometimes have more noise resulting in interference in detection performance. Sound filtering technologies such as low-pass filters could be used to improve feature quality.

ii. Another suggestion is to apply an optimization algorithm to acquire a more optimal thresholding degree on the post-processing threshold. It would be practical when dealing with more classes. The thresholding degree can solve the optimal degree for each class while maintaining accuracy.

iii. Add more sound datasets such as gunshots, motorcycles, SUVs, drones, helicopters, and aircraft. The sounds that can be applied as non-ambience sounds are helpful in security measures.

REFERENCES

Ahmad, S. F., & Singh, D. K. (2019). Automatic detection of tree cutting in forests using acoustic properties. *Journal of King Saud University - Computer and Information Sciences*, 1–7. https://doi.org/10.1016/j.jksuci.2019.01.016.

Alar, H. S., Mamaril, R. O., Villegas, L. P., & Cabarrubias, J. R. D. (2021). Audio classification of violin bowing techniques: An aid for beginners. *Machine Learning with Applications*, 4, 100028. https://doi.org/10.1016/j.mlwa.2021.100028

Albert, B., (2019). Why could an overfitted CNN model have a higher validation accuracy?. *Data Science Stack Exchange*. Retrieved February 3, 2021, from https://datascience.stackexchange.com/questions/53857/why-could-an-overfitted-cnn-model-have-a-higher-validation-accuracy

Alom, M. Z., Taha, T. M., Yakopcic, C., Westberg, S., Sidike, P., Nasrin, M. S., … & Asari, V. K. (2019). A state-of-the-art survey on deep learning theory and architectures. *Electronics*, 8(3), 292. https://doi.org/10.3390/electronics8030292

Banerjee, S., & Pamula, R. (2020). Random Forest Boosted CNN: An Empirical Technique for Plant Classification. *In Proceedings of the Global AI Congress 2019* (pp. 251-261). Springer, Singapore. https://doi.org/10.1007/978-981-15-2188-1_20

Barz, B., & Denzler, J. (2020). Deep learning on small datasets without pre-training using cosine loss. *In The IEEE Winter Conference on Applications of Computer Vision* (pp. 1371-1380). https://doi.org/10.1109/WACV45572.2020.9093286

Bayoudh, K., Hamdaoui, F., & Mtibaa, A. (2021). Transfer learning based hybrid 2D-3D CNN for traffic sign recognition and semantic road detection applied in advanced driver assistance systems. *Applied Intelligence*, *51*(1), 124-142. https://doi.org/10.1007/s10489-020-01801-5

Brieuc, M. S., Waters, C. D., Drinan, D. P., & Naish, K. A. (2018). A practical introduction to Random Forest for genetic association studies in ecology and evolution. *Molecular ecology resources, 18*(4), 755-766. https://doi.org/10.1111/1755-0998.12773

Bertolini, M., Mezzogori, D., Neroni, M., & Zammori, F. (2021). Machine Learning for industrial applications: a comprehensive literature review. *Expert Systems with Applications*, 175, 114820. https://doi.org/10.1016/J.ESWA.2021.114820

Bock, S., & Weiß, M. (2019, July). A proof of local convergence for the Adam optimizer. *In 2019 International Joint Conference on Neural Networks (IJCNN)* (pp. 1-8). IEEE. https://doi.org/10.1109/IJCNN.2019.8852239

Bode, G., Thul, S., Baranski, M., & Müller, D. (2020). Real-world application of machine-learning-based fault detection trained with experimental data. *Energy*, 198, 117-323. https://doi.org/10.1016/j.energy.2020.117323.

Brousseau, B., Rose, J., & Eizenman, M. (2020). Hybrid Eye-Tracking on a Smartphone with CNN Feature Extraction and an Infrared 3D Model. *Sensors*, *20*(2), 543. https://doi.org/10.3390/s20020543

Browning, E., Gibb, R., Glover-Kapfer, P., & Jones, K. E. (2017). Passive acoustic monitoring in ecology and conservation. http://dx.doi.org/10.25607/OBP-876

Carcillo, F., Le Borgne, Y. A., Caelen, O., Kessaci, Y., Oblé, F., & Bontempi, G. (2021). Combining unsupervised and supervised learning in credit card fraud detection. *Information sciences*, 557, 317-331. https://doi.org/10.1016/j.ins.2019.05.042

Cakir, E., & Virtanen, T. (2017). Convolutional Recurrent Neural Networks for Rare Sound Event Detection. https://doi.org/10.1109/TASLP.2017.2690575

Carlos, H. F. T., Silvana, G. M., Juan, F. G., Florentino, F.-R. (2013) A Hybrid artificial intelligence model for river flow forecasting. *Applied Soft Computing*, 13 (8), 3449-3458. https://doi.org/10.1016/j.asoc.2013.04.014.

Cervantes, J., Garcia-Lamont, F., Rodríguez-Mazahua, L., & Lopez, A. (2020). A comprehensive survey on support vector machine classification: Applications, challenges and trends. *Neurocomputing*, 408, 189-215. https://doi.org/10.1016/j.neucom.2019.10.118

Chan, T. K., & Chin, C. S., (2020) A Comprehensive Review of Polyphonic Sound Event Detection. *IEEE Access*, 8, 103339-103373. https://doi.org/10.1109/ACCESS.2020.2999388.

Chandrakala, S., Venkatraman, M., Shreyas, N., & Jayalakshmi, S. L. (2021). Multi-view representation for sound event recognition. *Signal, Image and Video Processing*, 1-9. https://doi.org/10.1007/S11760-020-01851-9

Chen, G., Zhang, X., Wu, Z., Su, J., & Cai, G. (2021). An efficient tea quality classification algorithm based on near infrared spectroscopy and random Forest. *Journal of Food Process Engineering*, *44*(1), e13604. https://doi.org/10.1111/jfpe.13604

Chen, J., & Kyrillidis, A. (2019). Decaying momentum helps neural network training, 1910-04952. Retrieved from the arXiv database. https://arxiv.org/abs/1910.04952v4

Chicco, D., & Jurman, G. (2020). The advantages of the Matthews correlation coefficient (MCC) over F1 score and accuracy in binary classification evaluation. *BMC genomics*, *21*(1), 1-13. https://doi.org/10.1186/s12864-019-6413-7

Chung K. (2020). Perceived sound quality of different signal processing algorithms by cochlear implant listeners in real-world acoustic environments,*Journal of Communication Disorders*,Vol. 83, 105973. https://doi.org/10.1016/j.jcomdis.2019.105973

Crunchant, A. S., Borchers, D., Kühl, H., & Piel, A. (2020). Listening and watching: Do camera traps or acoustic sensors more efficiently detect wild chimpanzees in an open habitat?. *Methods in Ecology and Evolution*, *11*(4), 542-552. https://doi.org/10.1111/2041-210X.13362

Dang, A., Vu, T., & Wang, J.-C. (2017). Deep Learning for {DCASE2017} Challenge. Retrieved January 20, 2021 from http://dcase.community/documents/challenge2017/technical_reports/DCASE2017_Toan_209.pdf

Darst, B.F., Malecki, K.C., & Engelman, C.D. (2018) Using recursive feature elimination in random forest to account for correlated variables in high dimensional data. *BMC Genet 19*, 65. https://doi.org/10.1186/s12863-018-0633-8

Dasgupta, S., (2018, June 01). This Tiny Camera Aims To Catch Poachers — Before They Kill. *Mongabay Environmental News*. Retrieved January 20, 2021 from https://news.mongabay.com/2018/06/this-tiny-camera-aims-to-catch-poachers-before-they-kill/

Davis, E. (2018, February 28). New Study Shows Over a Third of Protected Areas Surveyed are Severely at Risk of Losing Tigers. Retrieved January 20, 2021 from https://www.worldwildlife.org/press-releases/new-study-shows-over-a-third-of-protected-areas-surveyed-are-severely-at-risk-of-losing-tigers

Demir, F., Turkoglu, M., Aslan, M., & Sengur, A. (2020). A new pyramidal concatenated CNN approach for environmental sound classification. *Applied Acoustics*, 170, 107520. https://doi.org/10.1016/j.apacoust.2020.107520

Dilber, D. (2016). Feature Selection and Extraction of Audio. 3148–3155. https://doi.org/10.15680/IJIRSET.2016.0503064.

Doshi, S. (2019, April 4). Extract features of Music. *Towards Data Science*. https://towardsdatascience.com/extract-featuresof-music-75a3f9bc265d.

Frame, J., Nearing, G., Kratzert, F., & Rahman, M. (2020). Post-processing the US National Water Model with a Long Short-Term Memory network. https://eartharxiv.org/repository/object/124/download/253

Fong, C., & Tyler, M. (2021). Machine Learning Predictions as Regression Covariates. Political Analysis, 29(4), 467-484. doi:10.1017/pan.2020.38

Ganaie, M. A., & Hu, M. (2021). Ensemble deep learning: A review. 2104-02395, from arXiv preprint arXi. https://arxiv.org/pdf/2104.02395

Geetha, R., Thilagam, T., & Padmavathy, T. (2021). Effective offline handwritten text recognition model based on a sequence-to-sequence approach with CNN–RNN networks. *Neural Computing and Applications*, 1-12. https://doi.org/10.1007/s00521-020-05556-5

Ghaffarzadegan, S., Salekin, A., Das, S., & Feng, Z. (2017). Bosch Rare Sound Events Detection Systems for {DCASE2017} Challenge. http://dcase.community/documents/challenge2017/technical_reports/DCASE2017_Ravichandran_184.pdf

Gutiérrez, L., Patiño, J., & Duque-Grisales, E. (2021). A Comparison of the Performance of Supervised Learning Algorithms for Solar Power Prediction. *Energies, 14*(15), 4424. https://doi.org/10.3390/en14154424

Gupta, D., Bansal, P., & Choudhary, K. (2019). Performance Comparison of Robust Speech Recognition Using Different Feature, *8*(1), 125–130. https://www.academia.edu/download/58564402/V8I1201923.pdf

Heittola, T., & Mesaros, A. (2017). {DCASE} 2017 Challenge Setup: Tasks, Datasets and Baseline System. https://hal.inria.fr/hal-01627981/

Hershey, S., Chaudhuri, S., Ellis, D. P., Gemmeke, J. F., Jansen, A., Moore, R. C., … & Slaney, M. (2017, March). CNN architectures for large-scale audio classification. *In 2017 IEEE international conference on acoustics, speech and signal processing (ICASSP)* (pp. 131-135). IEEE. https://doi.org/10.1109/ICASSP.2017.7952132

Hoshen, Y., Weiss R. J., & Wilson, K. W. (2015). "Speech acoustic modeling from raw multichannel waveforms," *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 4624-4628. https://doi.org/10.1109/ICASSP.2015.7178847.

Inoue, T., Vinayavekhin, P., Wang, S., Wood, D., Greco, N., & Tachibana, R. (2018). Domestic activities classification based on CNN using shuffling and mixing data augmentation. *DCASE 2018 Challenge*. http://dcase.community/documents/challenge2018/technical_reports/DCASE2018_Inoue_14.pdf

Inus, K. (2017, October 24). *Special armed wildlife enforcement team to be set up to counter poachers*. New Straits Times Online. https://www.nst.com.my/news/nation/2017/10/294584/special-armed-wildlife-enforcement-team-be-set-counter-poachers

Jayalakshmi, S. L., Chandrakala, S., & Nedunchelian, R. (2018). Global statistical features-based approach for Acoustic Event Detection. *Applied Acoustics*, 139(April), 113–118. https://doi.org/10.1016/j.apacoust.2018.04.026.

Jiang, X., Hu, B., Chandra S. S., Wang, S. H., & Zhang, Y. D. (2020). Fingerspelling identification for Chinese sign language via AlexNet-based transfer learning and Adam optimizer. *Scientific Programming*, 2020. https://doi.org/10.1155/2020/3291426

Joshi S., Verma D. K., Saxena G., & Paraye A. (2019) Issues in Training a Convolutional Neural Network Model for Image Classification, *Communications in Computer and Information Science*, vol 1046. https://doi.org/10.1007/978-981-13-9942-8_27.

Jung, S. Y., Liao, C. H., Wu, Y. S., Yuan, S. M., & Sun, C. T. (2021). Efficiently Classifying Lung Sounds through Depthwise Separable CNN Models with Fused STFT and MFCC Features. Diagnostics, *11*(4), 732. https://doi.org/10.3390/diagnostics11040732

Kaiwu, W., Liping, Y., & Bin, Y. (2017). Audio Events Detection and Classification Using Extended {R-FCN} Approach. http://dcase.community/documents/workshop2017/proceedings/DCASE2017Workshop_Wang_121.pdf

Kensert, A., Alvarsson, J., Norinder, U., & Spjuth, O. (2018). Evaluating parameters for ligand-based modeling with random forest on sparse data sets. *Journal of Cheminformatics, 10*(1), 1-10. https://doi.org/10.1186/s13321-018-0304-9

Khoshdeli, M., Cong, R., & Parvin, B. (2017). Detection of nuclei in H&E stained sections using convolutional neural networks. 2*017 IEEE EMBS International*

Conference on Biomedical and Health Informatics, BHI 2017, 105–108. https://doi.org/10.1109/BHI.2017.7897216.

Kim, J., Min, K., Jung, M., & Chi, S. (2020). Occupant behavior monitoring and emergency event detection in single-person households using deep learning-based sound recognition. *Building and Environment*, 181, 107092. Retrieved from https://doi.org/10.1016/j.buildenv.2020.107092.

Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization, 1412-6980. Retrieved from the arXiv database. https://arxiv.org/abs/1412.6980

Koehrsen, W., (2017). Random Forest Simple Explanation. *Medium.com*. Retrieved Januay 16, 2021 from https://williamkoehrsen.medium.com/random-forest-simple-explanation-377895a60d2d.

Kramer, O. (2016). Scikit-learn. In Machine learning for evolution strategies (pp. 45-53). Springer, Cham. https://doi.org/10.1007/978-3-319-33383-0_5

Kristiadi, A., Hein, M., & Hennig, P. (2020, November). Being bayesian, even just a bit, fixes overconfidence in relu networks. In International Conference on Machine Learning (pp. 5436-5446). PMLR. https://proceedings.mlr.press/v119/kristiadi20a.html.

Kumar, J. L. M., Rashid, M., Musa, R. M., Razman, M. A. M., Sulaiman, N., Jailani, R., & Majeed, A. P. A. (2021). The classification of EEG-based winking signals: a transfer learning and random forest pipeline. *PeerJ*, 9, e11182. https://peerj.com/articles/11182/

Kumar, N., (2019). Advantages And Disadvantages Of Random Forest Algorithm In ML. *Theprofessionalspoint.blogspot.com*. Retrieved January 16, 2021 from https://theprofessionalspoint.blogspot.com/2019/02/advantages-and-disadvantages-of-random.html.

Lella, K. K., & Pja, A. (2021). Automatic COVID-19 disease diagnosis using 1D convolutional neural network and augmentation with human respiratory sound based on parameters: cough, breath, and voice. *AIMS Public Health, 8*(2), 240. https://dx.doi.org/10.3934%2Fpublichealth.2021019

Lim, H., Park, J., & Han, Y. (2017). Rare Sound Event Detection Using {1D} Convolutional Recurrent Neural Networks. http://dcase.community/documents/challenge2017/technical_reports/DCASE2017_Lim_204.pdf

Liu, W., & Zagzebski, J. A. (2010). Trade-offs in data acquisition and processing parameters for backscatter and scatterer size estimations. *IEEE transactions on ultrasonics, ferroelectrics, and frequency control*, 57(2), 340–352. doi:10.1109/TUFFC.2010.1414

Liu, Y., Cheng, Z., Liu, J., Yassin, B., Nan, Z., & Luo, J. (2019). AI for Earth: Rainforest Conservation by Acoustic Surveillance, 1908-07517. https://arxiv.org/pdf/1908.07517

Liu, Z., & Li, S. (2020). A sound monitoring system for prevention of underground pipeline damage caused by construction. Automation in Construction, 113, 103125. https://doi.org/10.1016/j.autcon.2020.103125

Lu, R., Duan, Z., & Zhang, C. (2018, April). Multi-scale recurrent neural network for sound event detection. *In 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (pp. 131-135). IEEE. https://doi.org/10.1109/ICASSP.2018.8462006

Maria, S. K., Taki, S. S., Mia, M., Biswas, A. A., Majumder, A., & Hasan, F. (2022). Cauliflower Disease Recognition Using Machine Learning and Transfer Learning. *In Smart Systems: Innovations in Computing* (pp. 359-375). Springer, Singapore. https://doi.org/10.1007/978-981-16-2877-1_33

Mehrabi, N., Morstatter, F., Saxena, N., Lerman, K., & Galstyan, A. (2021). A survey on bias and fairness in machine learning. *ACM Computing Surveys (CSUR), 54*(6), 1-35. https://dl.acm.org/doi/abs/10.1145/3457607

Mesaros, A., Diment, A., Elizalde, B., Heittola, T., Vincent, E., Raj, B., & Virtanen, T. (2019). Sound event detection in the DCASE 2017 challenge. IEEE/ACM Transactions on Audio, Speech, and Language Processing, 27(6), 992-1006. https://doi.org/10.1109/TASLP.2019.2907016

Mesaros, A., Heittola, T., & Virtanen, T. (2016). Metrics for Polyphonic Sound Event Detection. *Applied Sciences*, 6(6), 162. https://doi.org/10.3390/app6060162.

Miwil, O. (2017, October 3). *Sabah Wildlife Department looking for at least 3 suspects in turtle poaching case*. New Straits Times Online. https://www.nst.com.my/news/nation/2017/10/287055/sabah-wildlife-department-looking-least-3-suspects-turtle-poaching-case.

Mushtaq, Z., & Su, S. F. (2020). Environmental sound classification using a regularized deep convolutional neural network with data augmentation. *Applied Acoustics*, 167, 107389. https://doi.org/10.1016/j.apacoust.2020.107389

Mushtaq, Z., Su, S. F., & Tran, Q. V. (2021). Spectral images based environmental sound classification using CNN with meaningful data augmentation. *Applied Acoustics*, 172, 107581. https://doi.org/10.1016/j.apacoust.2020.107581

Nigam, A., & Srivastava, S. (2021, January). Macroscopic Traffic Stream Variables Prediction with Weather Impact Using Hybrid CNN-LSTM model. *In Adjunct Proceedings of the 2021 International Conference on Distributed Computing and Networking* (pp. 1-6). https://doi.org/10.1145/3427477.3429780

Nordin, R. (2019, June 13). *Showcasing the wonders of Endau-Rompin park*. The Star Online. https://www.thestar.com.my/metro/metro-news/2019/06/13/showcasing-the-wonders-of-endaurompin-park.

Opitz, J., & Burst, S. (2019). Macro f1 and macro f1, 1911-03347. Retrieved from the arXiv database. https://arxiv.org/abs/1911.03347v1

Otter, D.W., Medina, J.R., & Kalita, J.K. (2018). A Survey of the Usages of Deep Learning in Natural Language Processing, 1807-10854. From ArXiv database. https://doi.org/10.1109/TNNLS.2020.2979670

Pandya, S., & Ghayvat, H. (2021). Ambient acoustic event assistive framework for identification, detection, and recognition of unknown acoustic events of a residence. *Advanced Engineering Informatics*, 47(January), 101238. https://doi.org/10.1016/j.aei.2020.101238.

Park, C., Awadalla, A., Kohno, T., & Patel, S. (2021). Reliable and Trustworthy Machine Learning for Health Using Dataset Shift Detection. Advances in Neural Information Processing Systems, 34. https://proceedings.neurips.cc/paper/2021/file/17e23e50bedc63b4095e3d8204ce063b-Paper.pdf

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... & Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12, 2825-2830. https://www.jmlr.org/papers/volume12/pedregosa11a/pedregosa11a.pdf

Pei, G. P. (2017, September 18). *Sarawak Forestry Department sending out strong message to timber thieves*. New Straits Times Online. https://www.nst.com.my/news/nation/2017/09/281224/sarawak-forestry-department-sending-out-strong-message-timber-thieves.

Peixeiro, M., Naji, N., & Charton, E. (2021). Direct Answer Threshold Optimization in Dialogue Systems. In Proceedings of the 34th Canadian Conference on Artificial Intelligence. https://assets.pubpub.org/thbtnq2u/31621609391750.pdf

Phan, H., Krawczyk-Becker, M., Gerkmann, T., & Mertins, A. (2017). {DNN} and {CNN} with Weighted and Multi-Task Loss Functions for Audio Event Detection. https://arxiv.org/abs/1708.03211

Pisner, D. A., & Schnyer, D. M. (2020). Support vector machine. *In Machine Learning* (pp. 101-121). Academic Press. https://doi.org/10.1016/B978-0-12-815739-8.00006-7

Povera, A., (2019, September 29). *MACC cracks down on illegal logging and mining*. New Straits Times Online. https://www.nst.com.my/news/nation/2019/09/525563/macc-cracks-down-illegal-logging-and-mining.

Radhakrishnan, P., (2017). What are Hyperparameters? and how to tune the Hyperparameters in a Deep Neural Network? *towardsdatascience.com*. Retrieved February 6, 2021 from https://towardsdatascience.com/what-are-hyperparameters-and-how-to-tune-the-hyperparameters-in-a-deep-neural-network-d0604917584a.

Rahana, H. (2017). Malaysia's Statement on Budget 2018. *WWF-Malaysia*. Retrieved February 6, 2021, from http://www.wwf.org.my/?24845/WWF-Malaysias-Statement-on-Budget-2018.

Rao K. R., Kim, D. N., & Hwang, J.-J. (2010). Fast Fourier Transform - Algorithms and Applications (1st. ed.). Springer Publishing Company, Incorporated. https://danylastchild07.files.wordpress.com/2016/05/fft_algorithms-and-applications.pdf

Ravichandran, A., & Das, S. (2017). Bosch Rare Sound Events Detection Systems for {DCASE2017} Challenge. http://dcase.community/documents/challenge2017/technical_reports/DCASE2017_Ravichandran_184.pdf

Re, D. E., O'Connor, J. J., Bennett, P. J., & Feinberg, D. R. (2012). Preferences for very low and very high voice pitch in humans. *PloS one*, 7(3), e32719. https://doi.org/10.1371/journal.pone.0032719

Reynolds, R. P., Kinard, W. L., Degraff, J. J., Leverage, N., & Norton, J. N. (2010). Noise in a Laboratory Animal Facility from the Human and Mouse Perspectives. *49*(5). https://www.ingentaconnect.com/contentone/aalas/jaalas/2010/00000049/00000005

Richoz, S., Wang, L., Birch, P., & Roggen, D. (2020). Transportation Mode Recognition Fusing Wearable Motion, Sound, and Vision Sensors. *IEEE Sensors Journal*, *20*(16), 9314-9328. https://doi.org/10.1109/JSEN.2020.2987306

Rivera, S. N., Knight, A., & McCulloch, S. P. (2021). Surviving the Wildlife Trade in Southeast Asia: Reforming the 'Disposal'of Confiscated Live Animals under CITES. Animals, *11*(2), 439. https://doi.org/10.3390/ani11020439

Roberts, L. (2020). Understanding the Mel Spectrogram. *Analytics Vidhya*. Retrieved January 27, 2021 from https://medium.com/analytics-vidhya/understanding-the-mel-spectrogram-fca2afa2ce53.

Sarica, A., Cerasa, A., & Quattrone, A. (2017). Random forest algorithm for the classification of neuroimaging data in Alzheimer's disease: A systematic review. *Frontiers in Aging Neuroscience*, 9(OCT), 1–12. https://doi.org/10.3389/fnagi.2017.00329.

Sattar, F., Driessen, P. F., Tzanetakis, G., & Page, W. H. (2020). A new event detection method for noisy hydrophone data. *Applied Acoustics*, *159*, 107056. https://doi.org/10.1016/j.apacoust.2019.107056

Seccia, R., Gammelli, D., Dominici, F., Romano, S., Landi, A. C., Salvetti, M., ... & Palagi, L. (2020). Considering patient clinical history impacts performance of machine learning models in predicting course of multiple sclerosis. *PloS one*, *15*(3), e0230219. https://doi.org/10.1371/journal.pone.0230219

Seifert, S. (2020). Application of random forest based approaches to surface-enhanced Raman scattering data. *Scientific Reports*, 10(1), 1–11. https://doi.org/10.1038/s41598-020-62338-8.

Selman, J. G., & Demir, N. (2019). Automatic Detection for Acoustic Monitering of Wild Animals. Stanford InfoLab. http://ilpubs.stanford.edu:8090/1166/

Serizel, R., Turpault, N., Shah, A, & Salamon, J., (2020). "Sound Event Detection in Synthetic Domestic Environments," ICASSP 2020 - 2020 IEEE International Conference on Acoustics. *Speech and Signal Processing (ICASSP)*, Spain(2020), 86-90.https://doi.org/10.1109/ICASSP40776.2020.9054478.

Shepherd, C. R., Gomez, L., & Nijman, V. (2020). Illegal wildlife trade, seizures and prosecutions: A 7.5-year analysis of trade in pig-nosed turtles Carettochelys insculpta in and from Indonesia. *Global Ecology and Conservation*, 24, e01249. https://doi.org/10.1016/j.gecco.2020.e01249

Singh, A., Halgamuge, M. N., & Lakshmiganthan, R. (2017). Impact of different data types on classifier performance of random forest, naive bayes, and k-nearest

neighbors algorithms. https://minerva-access.unimelb.edu.au/bitstream/handle/11343/216910/2017_Asmita_Different_Data.pdf

SIEMENS DSP Community (2019, November 1). *Fundamentals of Digital Signal Processing*. https://www.plm.automation.siemens.com/global/en/topic/digital-signal-processing/29851.

Singh, V., Ray, K. C., & Tripathy, S. (2020). Robust Gunshot Features and Its Classification Using Support Vector Machine for Wildlife Protection. *Electronic Systems and Intelligent Computing* (pp. 939-948). Springer, Singapore. https://doi.org/10.1007/978-981-15-7031-5_89

Simonyan, K., & Zisserman, A. (2015). Very deep convolutional networks for large-scale image recognition. *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*, 1–14.

A. (2021). A CNN approach for audio classification in construction sites. *Progresses in Artificial Intelligence and Neural Systems* (pp. 371-381). Springer, Singapore. https://arxiv.org/abs/1409.1556

Smith SW. (1997). Audio processing, (pp. 351–372 : Smith SW, editor. The scientist and engineer's guide to digital signal processing. San Diego (CA), *California Technical Publishing* . www.dspguide.com/CH28.PDF

Subramanian, H. P. P. R. D. S. D. R. (2004). Audio signal classification. M.Tech. Credit Seminar Report, *Electronic Systems Group*, *1*(1), 1–17. EE. Dept, IIT Bombay. http://www.ee.iitb.ac.in/~esgroup/es_mtech04_sem/es_sem04_paper_04307909.pdf

Subudhi, A., Dash, M., & Sabut, S. (2020). Automated segmentation and classification of brain stroke using expectation-maximization and random forest classifier. *Biocybernetics and Biomedical Engineering*, *40*(1), 277–289. https://doi.org/10.1016/j.bbe.2019.04.004.

Thanh Noi, P., & Kappas, M. (2018). Comparison of random forest, k-nearest neighbor, and support vector machine classifiers for land cover classification using Sentinel-2 imagery. *Sensors, 18*(1), 18. https://doi.org/10.3390/s18010018

TaheriNejad, N., & Jantsch, A. (2019, May). Improved machine learning using confidence. In 2019 IEEE Canadian Conference of Electrical and Computer Engineering (CCECE) (pp. 1-5). IEEE. Jeon, K. M., & Kim, H. K. (2017). Nonnegative Matrix Factorization-Based Source Separation with Online Noise Learning for Detection of Rare Sound Events. https://doi.org/10.1109/CCECE.2019.8861962

Tian, C., Xu, Y., & Zuo, W. (2020). Image denoising using deep CNN with batch renormalization. *Neural Networks*, 121, 461-473. https://doi.org/10.1016/j.neunet.2019.08.022

TMaccagno, A., Mastropietro, A., Mazziotta, U., Scarpiniti, M., Lee, Y. C., Uncini,

Turpault, N., & Serizel, R. (2020). Training sound event detection on a heterogeneous dataset, 2007-03931. Retrieved from the arXiv database. https://arxiv.org/abs/2007.03931

Tzirakis, P., Shiarella, A., Ewers, R., & Schuller, B. W. (2020). Computer Audition for Continuous Rainforest Occupancy Monitoring: The Case of Bornean Gibbons' Call Detection. Proc. Interspeech 2020, 1211-1215. http://www.interspeech2020.org/uploadfile/pdf/Mon-3-4-9.pdf

Vesperini, F., Droghini, D., Ferretti, D., Principi, E., Gabrielli, L., Squartini, S., & Piazza, F. (2017). A Hierarchic Multi-Scaled Approach for Rare Sound Event Detection. https://www.researchgate.net/profile/Fabio-Vesperini/publication/321425618_A_Hierarchic_Multi-scaled_Approach_for_Rare_Sound_Event_Detection/links/5a2164a90f7e9b71dd0310cb/A-Hierarchic-Multi-scaled-Approach-for-Rare-Sound-Event-Detection.pdf

Wang, Q., Du, J., Wu, H. X., Pan, J., Ma, F., & Lee, C. H. (2021). A Four-Stage Data Augmentation Approach to ResNet-Conformer Based Acoustic Modeling for Sound Event Localization and Detection, 2101-02919. Retrieved from the arXiv database. https://arxiv.org/abs/2101.02919

Wildlife Conservation & Science (WCS) Malaysia. (2018). Wildlife Tigers and prey Retrieved January 20, 2021 from https://malaysia.wcs.org/Wildlife/Tigers-and-prey-program.aspx

Wisdom, S., Erdogan, H., Ellis, D. P., Serizel, R., Turpault, N., Fonseca, E., ... & Hershey, J. R. (2021, June). What's all the fuss about free universal sound separation data?. *In ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (pp. 186-190). IEEE. https://doi.org/10.1109/ICASSP39728.2021.9414774

Wu, Y., & Lee, T. (2019, May). Enhancing sound texture in CNN-based acoustic scene classification. *In ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (pp. 815-819). IEEE. https://doi.org/10.1109/ICASSP.2019.8683490

Wu, Y.-P., Mao, J.-M., & Li, W.-F. (2017). Robust speech recognition by selecting mel-filter banks. *EEEIS 2016*, 117. https://doi.org/10.2991/eeeis-16.2017.52

Yan, J., Song, Y., Dai, L. R., & McLoughlin, I. (2020, May). Task-aware mean teacher method for large scale weakly labeled semi-supervised sound event detection. *In ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (pp. 326-330). IEEE. https://doi.org/10.1109/ICASSP40776.2020.9053073

Yeşilkanat, C. M. (2020). Spatio-temporal estimation of the daily cases of COVID-19 in worldwide using random forest ML algorithm. *Chaos, Solitons & Fractals*, *140*(110210). https://doi.org/10.1016/j.chaos.2020.110210

Yu J., Zhang X., Xu L., Dong J.,Zhangzhong L. (2021). A hybrid CNN-GRU model for predicting soil moisture in maize root zone. *Agricultural Water Management*, 245(2021), 106649, 0378-3774. https://doi.org/10.1016/j.agwat.2020.106649.

Zainuddin, Z. (2020, October 04). *Extinction Of The Malayan Large Mammals*. The Star Online. https://www.thestar.com.my/news/focus/2020/10/04/extinction-of-the-malayan-large-mammals.

Zhou, Q., & Feng, Z. (2017). Robust Sound Event Detection Through Noise Estimation and Source Separation Using {NMF}. http://dcase.community/documents/workshop2017/proceedings/DCASE2017Workshop_Zhou_113.pdf

Zolkepli, F. (2019, October 22). *Ten poachers detained, almost RM1mil worth of wildlife parts seized*. The Star Online. https://www.thestar.com.my/news/nation/2019/10/22/10-poachers-detained-almost-rm1mil-worth-of-wildlife-parts-seized

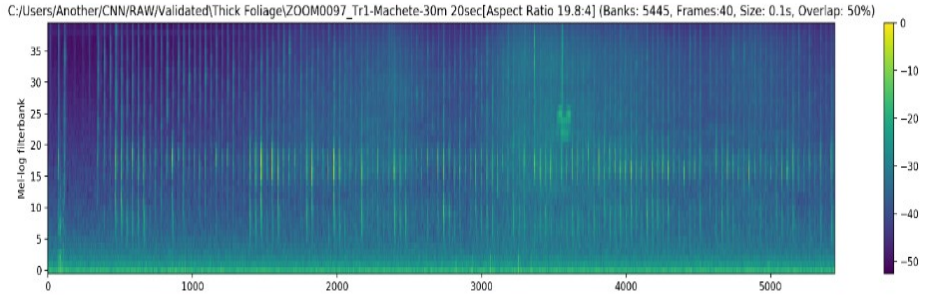
APPENDICES

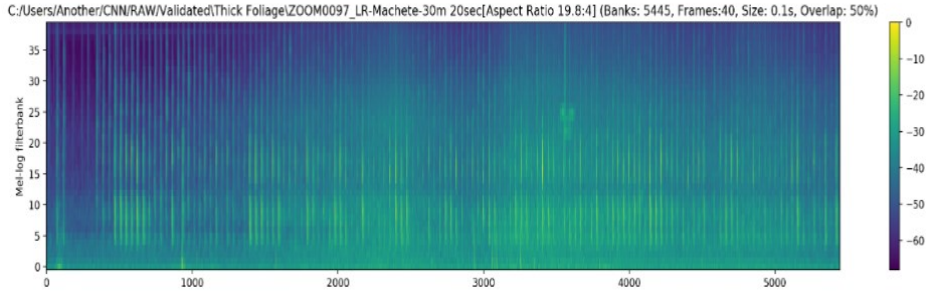
APPENDIX 1

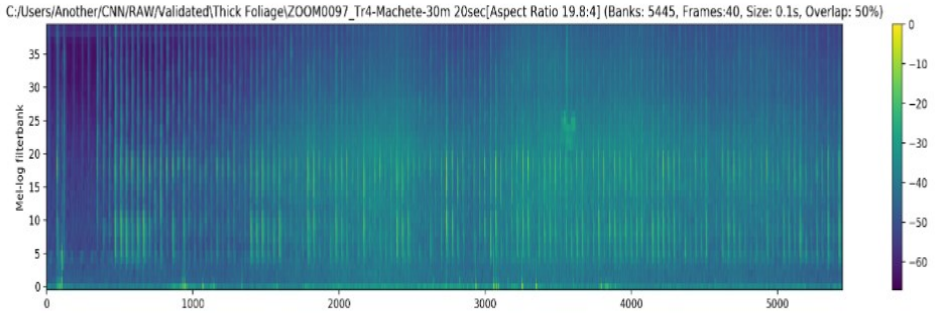
MLE Feature visualization code snippet

```
for f in files[:]:
    file_name = f
    #1 Read AUDIO Signal File
    fs, signal = wav.read(file_name,1)
    signal = signal[0:44100*10,0]

    #2 Define Overelapping (50%) and Hop length (100ms)
    para_fl = 0.1
    para_ovlp = 0.5

    #3 Audio pre-emphasizing.
    signal_preemphasized = speechpy.processing.preemphasis(signal, cof=0.98)
    #3 Extract mel-log energies MLE
    logenergy = speechpy.feature.lmfe(signal, sampling_frequency=fs, frame_length=para_fl, frame_stride=para_fl*0.5,
                num_filters=40, fft_length=512, low_frequency=0, high_frequency=20000)

    print('logenergy features=', logenergy.shape)
    #4 Plot MLE feature heatmap
    fig = plt.figure()
    plt.figure(figsize=(20,4))
    plt.ylabel("Mel-log filterbank")
    plt.xlabel("Frame")
    first_image = np.array(logenergy.transpose((1, 0)), dtype='float')
    plt.imshow(first_image, interpolation='nearest', aspect='auto')
    plt.colorbar()
    plt.gca().invert_yaxis()
    #5 Save Image to Folder
    saveas = file_name.split("/")[7].split("\\")[1].split(".")[0]+'AR19-4_'
    print(saveas)
    plt.savefig(saveas, dpi = 100)
```

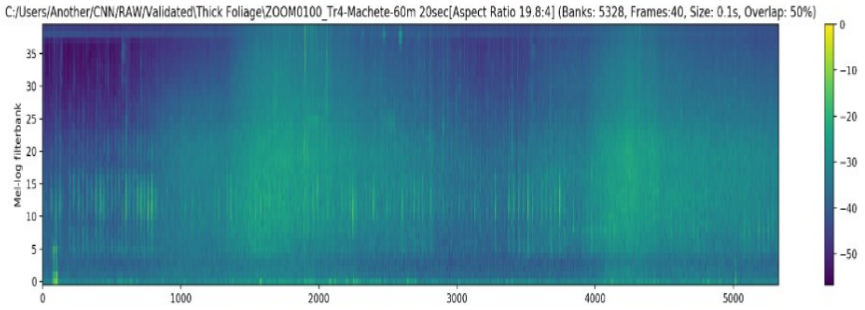
Hatchet MLE Audio Feature 30m 6000 Frames each 100ms x 40 MLE (3 Minutes Span)
3 Different Microphones TR1, TR4 and LR

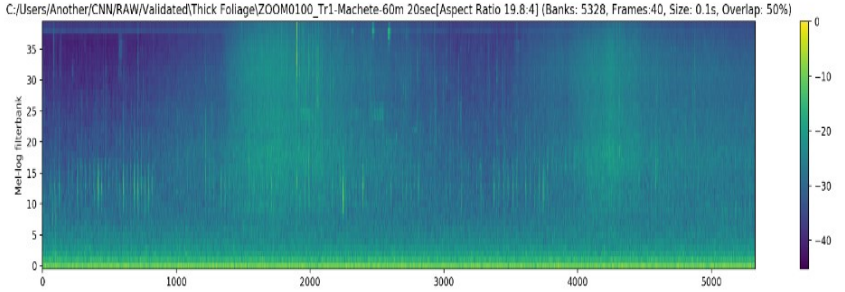


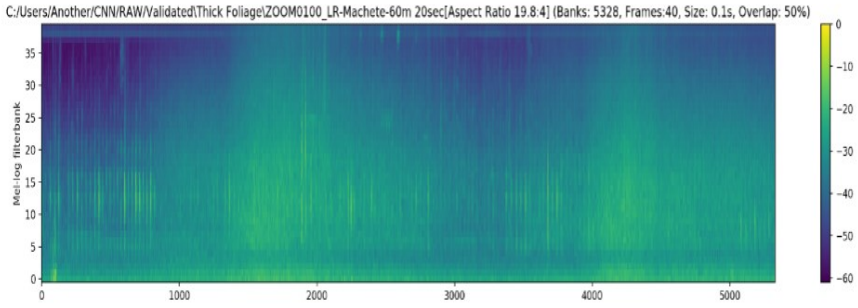




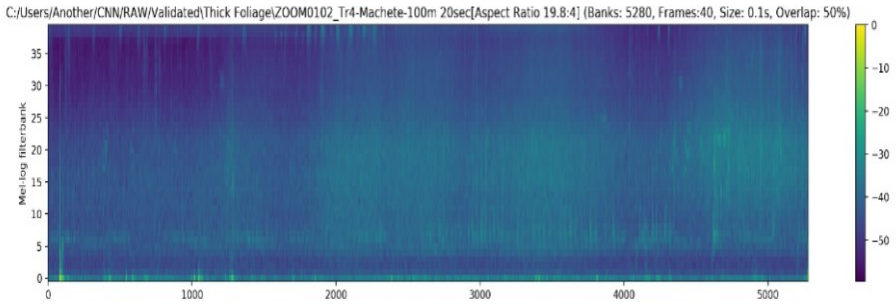
Hatchet MLE Audio Feature 60m 6000 Frames each 100ms x 40 MLE (3 Minutes Span)
3 Different Microphones TR1, TR4 and LR

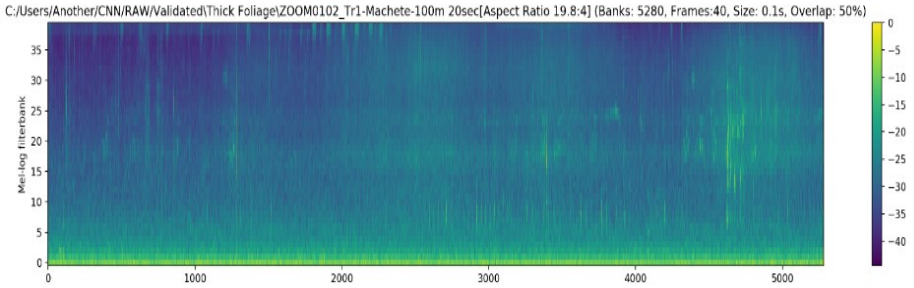


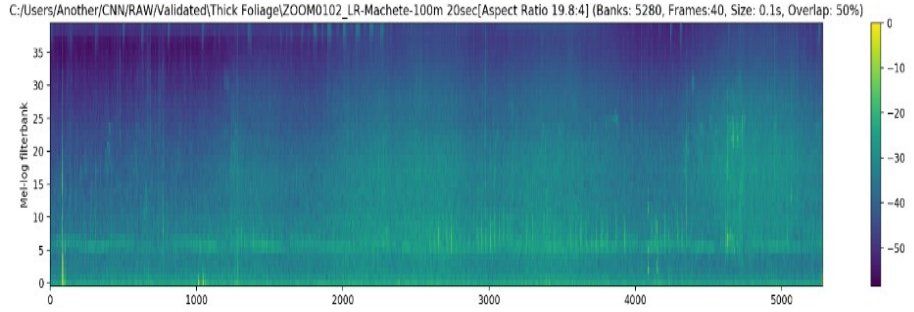




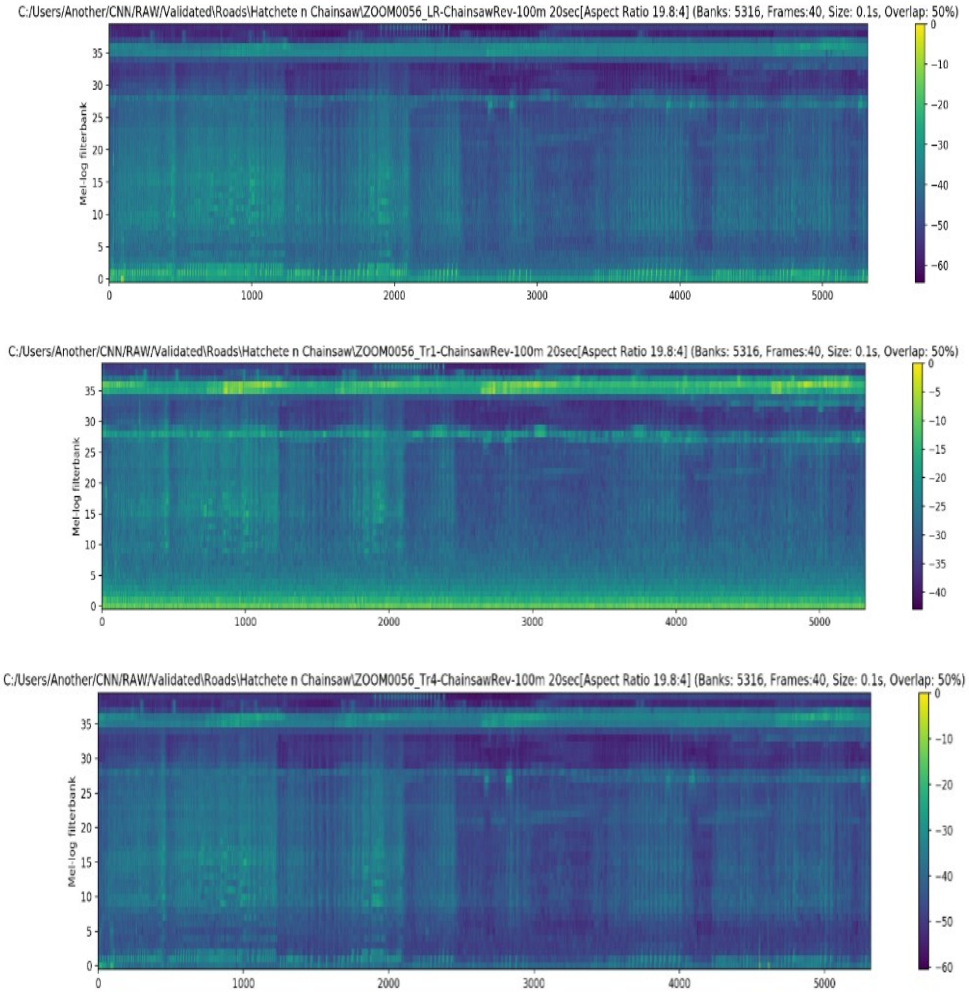
Hatchet MLE Audio Feature 100 meter 6000 Frames each 100ms x 40 MLE (3 Minutes Span)
3 Different Microphones TR1, TR4, and LR



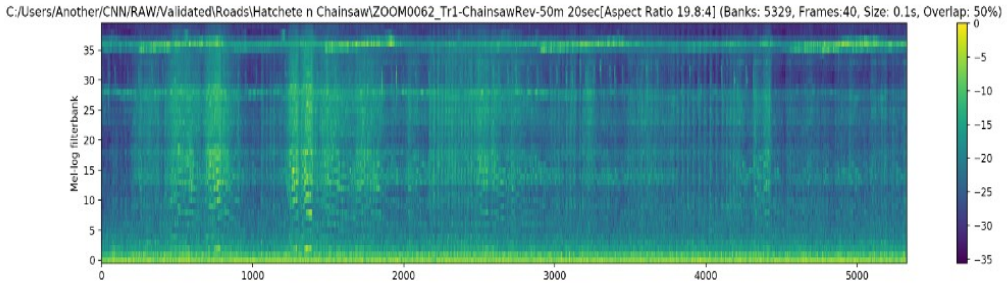


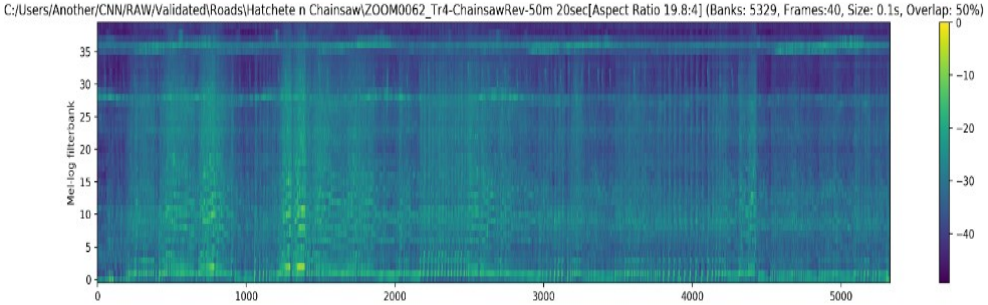


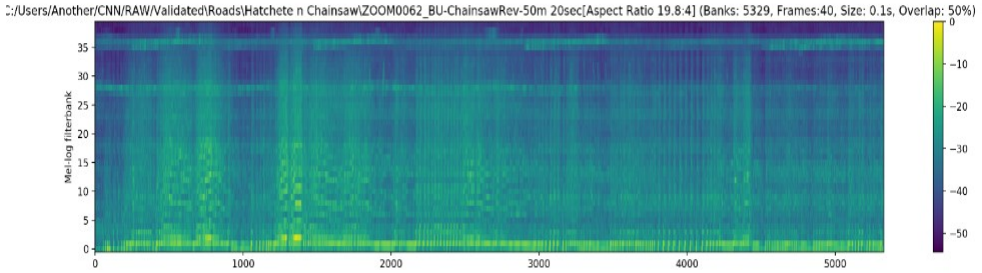
Chainsaw MLE Audio Feature 100 meter 6000 Frames each 100ms x 40 MLE (3 Minutes Span)
3 Different Microphones TR1, TR4 and LR



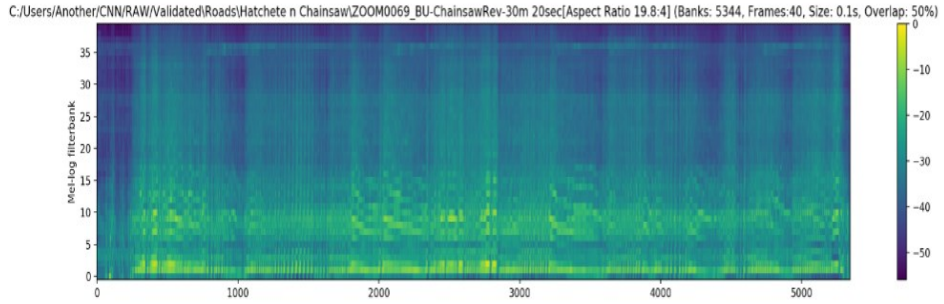
Chainsaw MLE Audio Feature 60m 6000 Frames each 100ms x 40 MLE (3 Minutes Span)
3 Different Microphones TR1, TR4 and LR

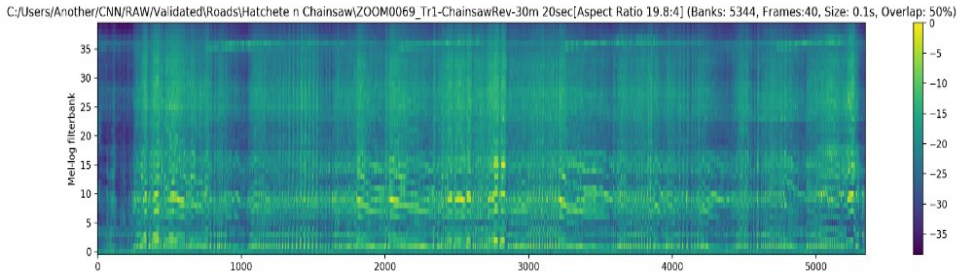


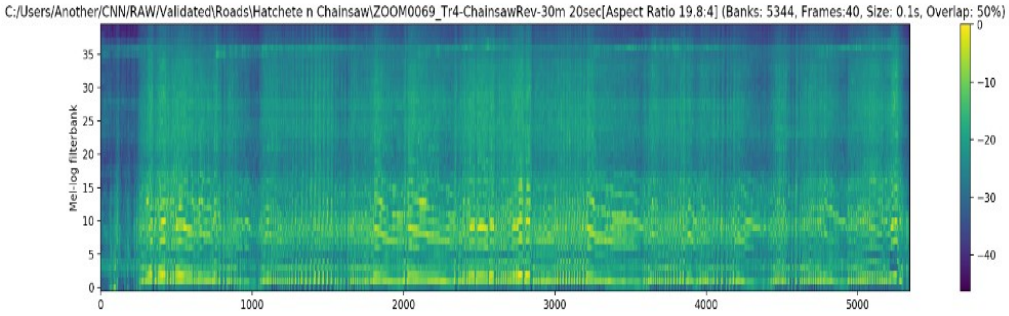




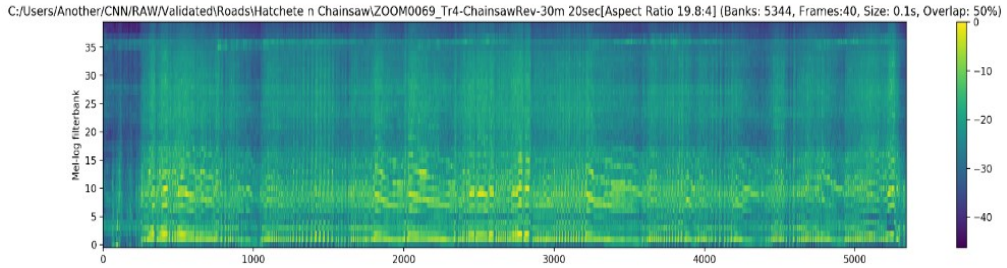
Chainsaw MLE Audio Feature 30m 6000 Frames each 100ms x 40 MLE (3 Minutes Span)
3 Different Microphones TR1, TR4 and LR

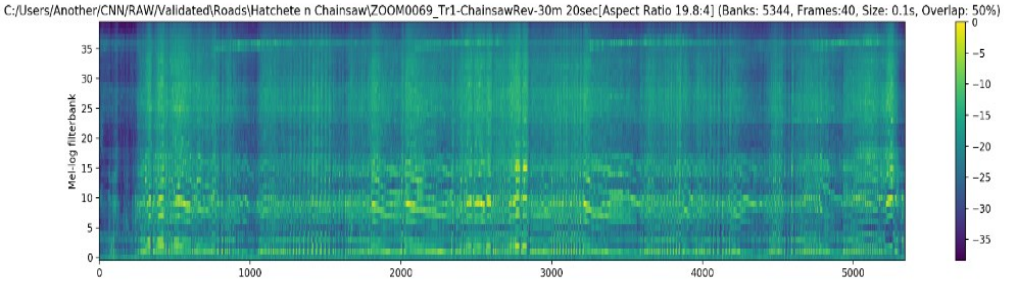


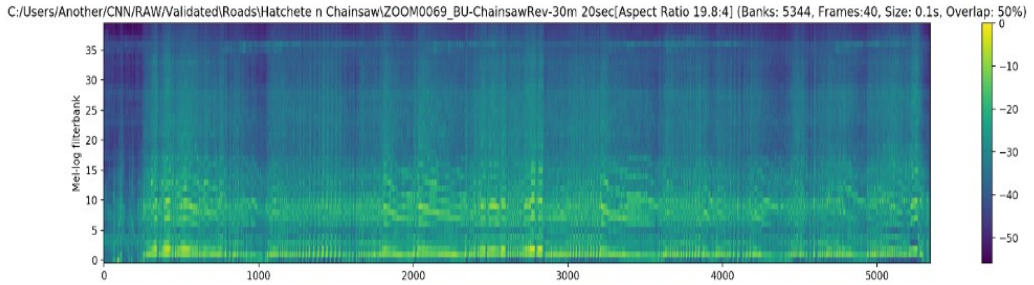




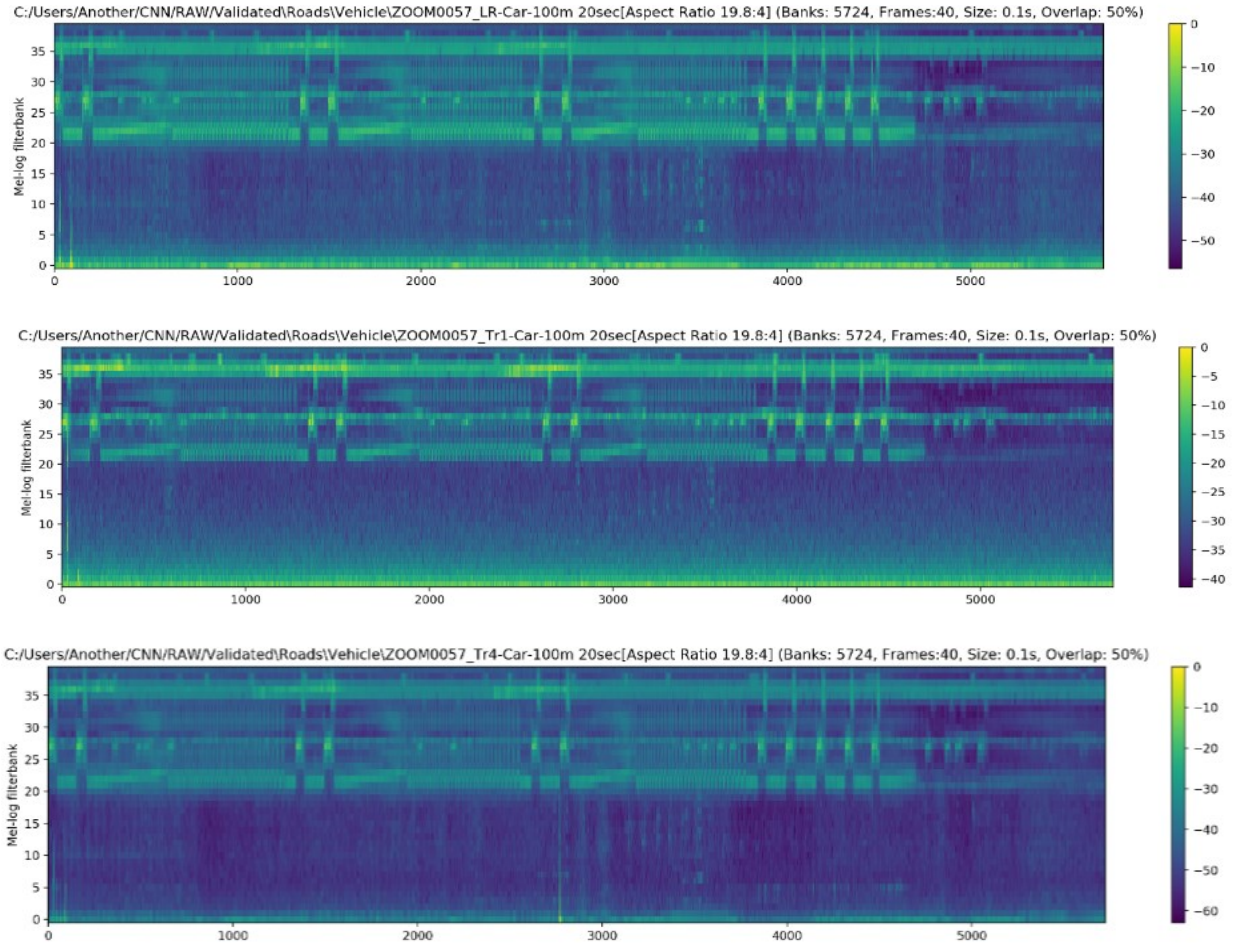
CAR MLE Audio Feature 100 meter 6000 Frames each 100ms x 40 MLE (3 Minutes Span)
3 Different Microphones TR1, TR4 and LR



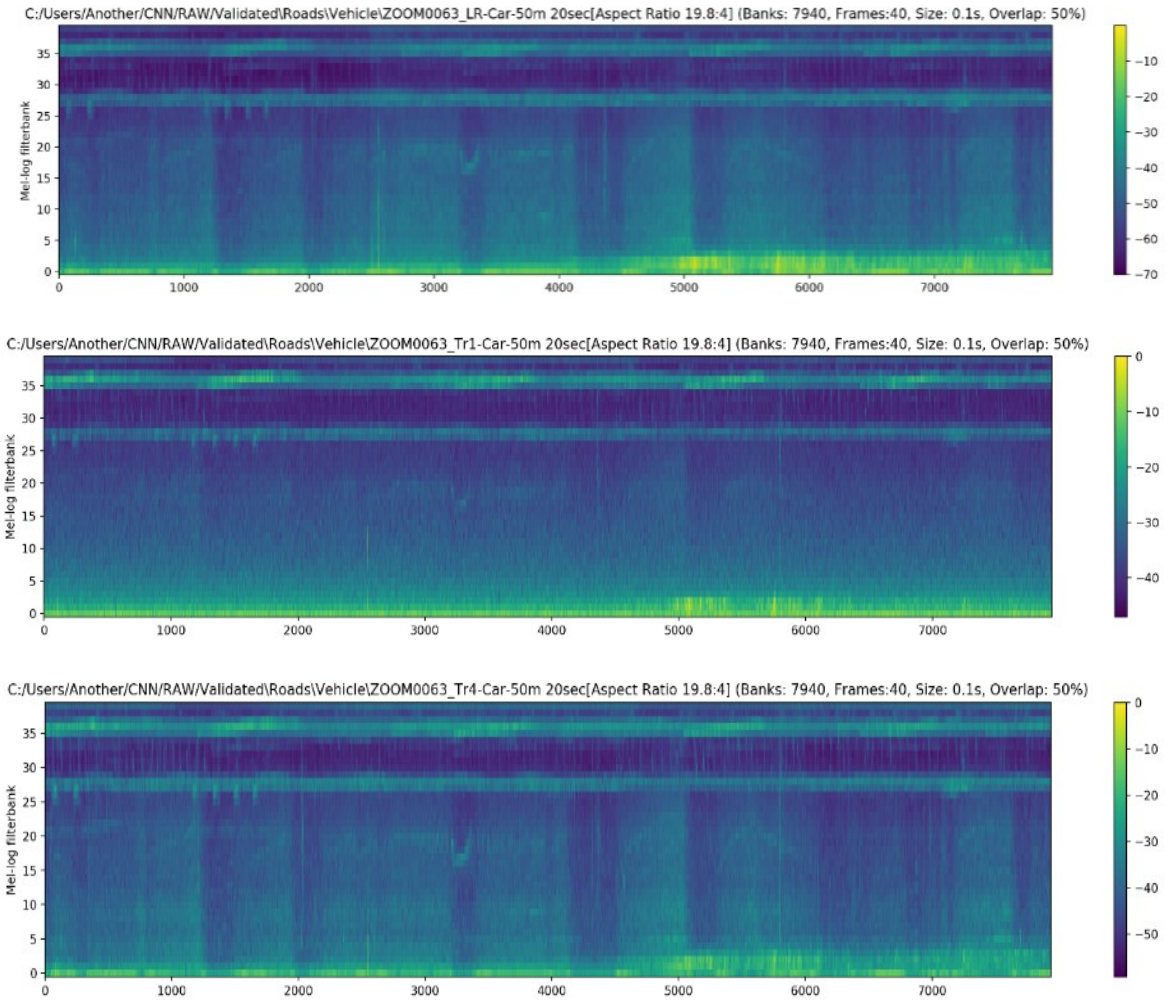




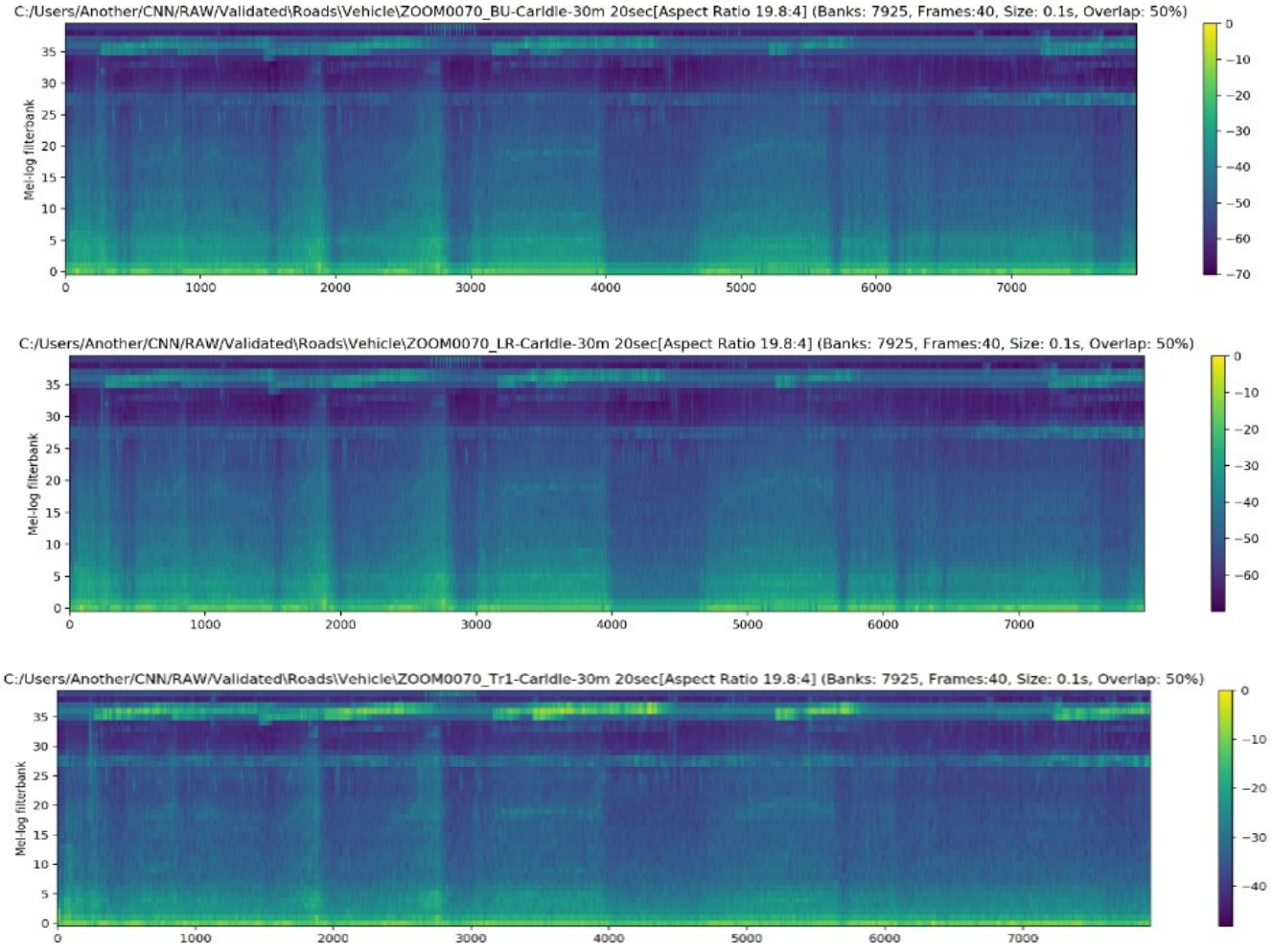
CAR MLE Audio Feature 100 meter 6000 Frames each 100ms x 40 MLE (3 Minutes Span)
3 Different Microphones TR1, TR4 and LR



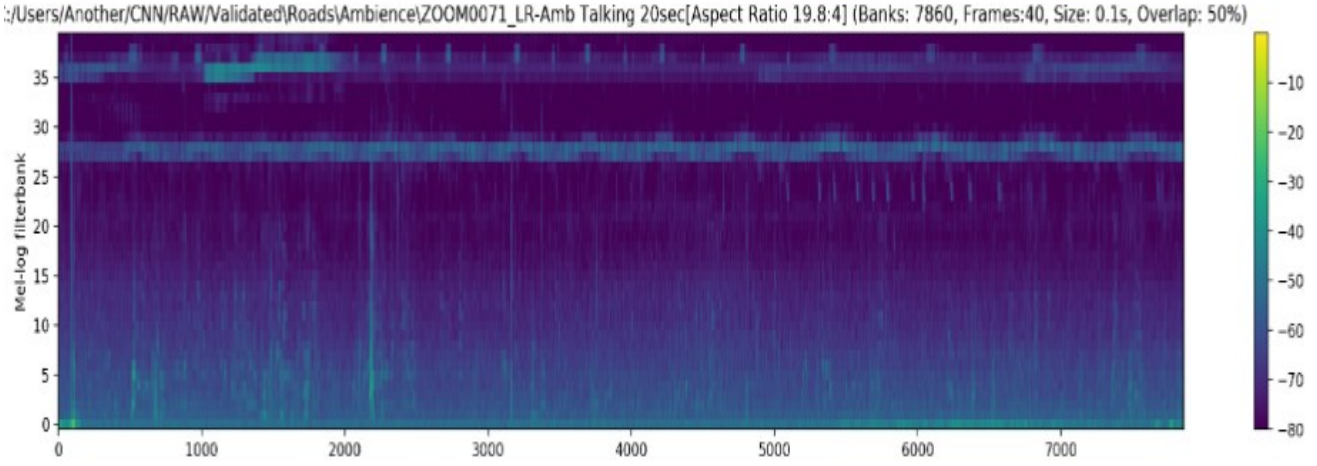
CAR MLE Audio Feature 60m 9000 Frames each 100ms x 40 MLE (3 Minutes Span)
3 Different Microphones TR1, TR4 and LR

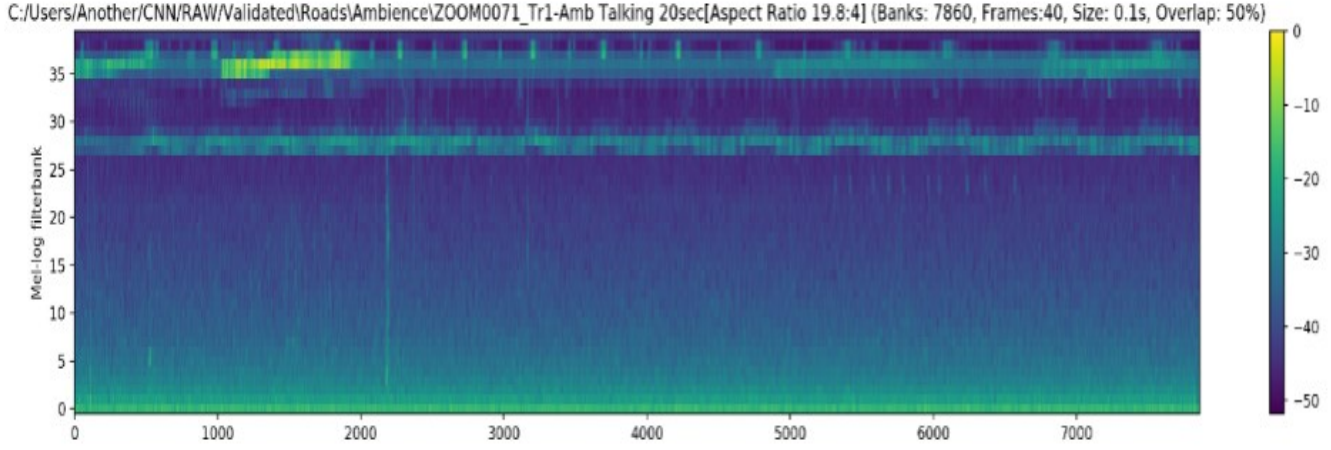


CAR MLE Audio Feature 30m 9000 Frames each 100ms x 40 MLE (3 Minutes Span)
3 Different Microphones TR1, TR4 and LR



Natural Ambience Audio Feature 9000 Frames each 100ms x 40 MLE (3 Minutes Span)
3 Different Microphones TR1, TR4 and LR





APPENDIX 2

Data Collection Recording Tools

Zoom H6 with 4 external microphone XLR Input with phantom power allows data collection with different microphones without any hardware level inconsistency and distortions. Attached on top is the XY stereo microphones that are very good professional grade quality for recording

Data Collection Trip Endau Rompin, Johor, Malaysia

Application of GPS to track distances from source and recording positions.

Carrying chainsaw into deeper parts of the forest assisted by WCS and Endau Rompin National Park Ranger.

Using multiple Microphones recording on one recorder to capture hardware inconsistency.

ASIDS – Acoustic Surveillance Intrusion Detection system

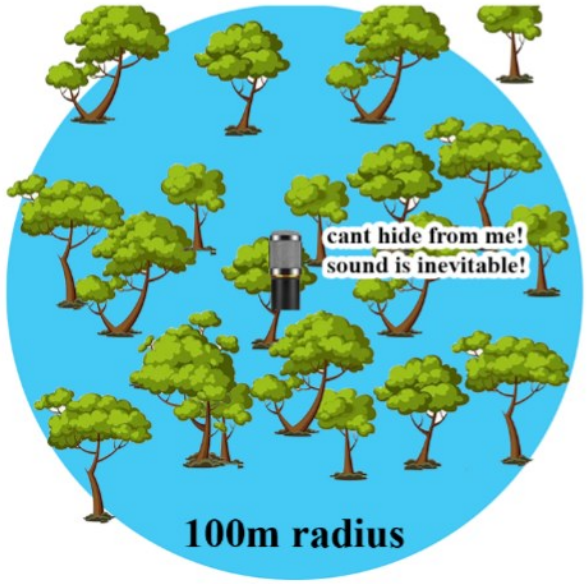
Asids Node is an Arm based computer with a microphone powered on batteries recording surrounding sound and running the SED algorithm using on board CPU arm v7 Raspberry pi 3B. Sends out a signal detection to the gateway using LoRa RF 413 Mhz. The whole machine is fitted in an IP68 weatherproof enclosure for the forest environment.

ASIDS Node Left ASIDS Gateway (Right) ASIDS Gateway, the receiver module that collections detection reports and display it on a dashboard.

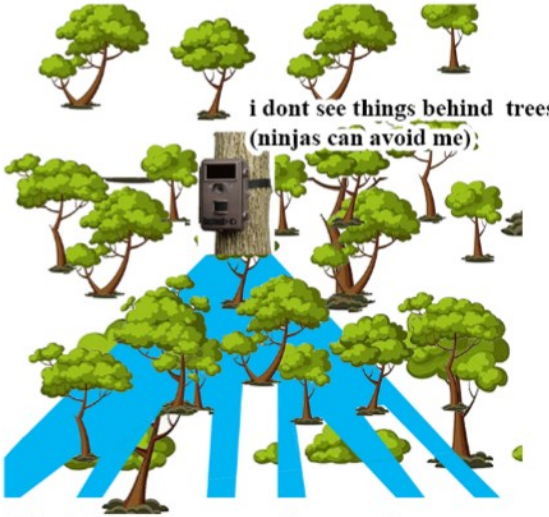
ASIDS Vs Cameras

ASIDS

Wireless Communication included
Selling Price RM300 each node

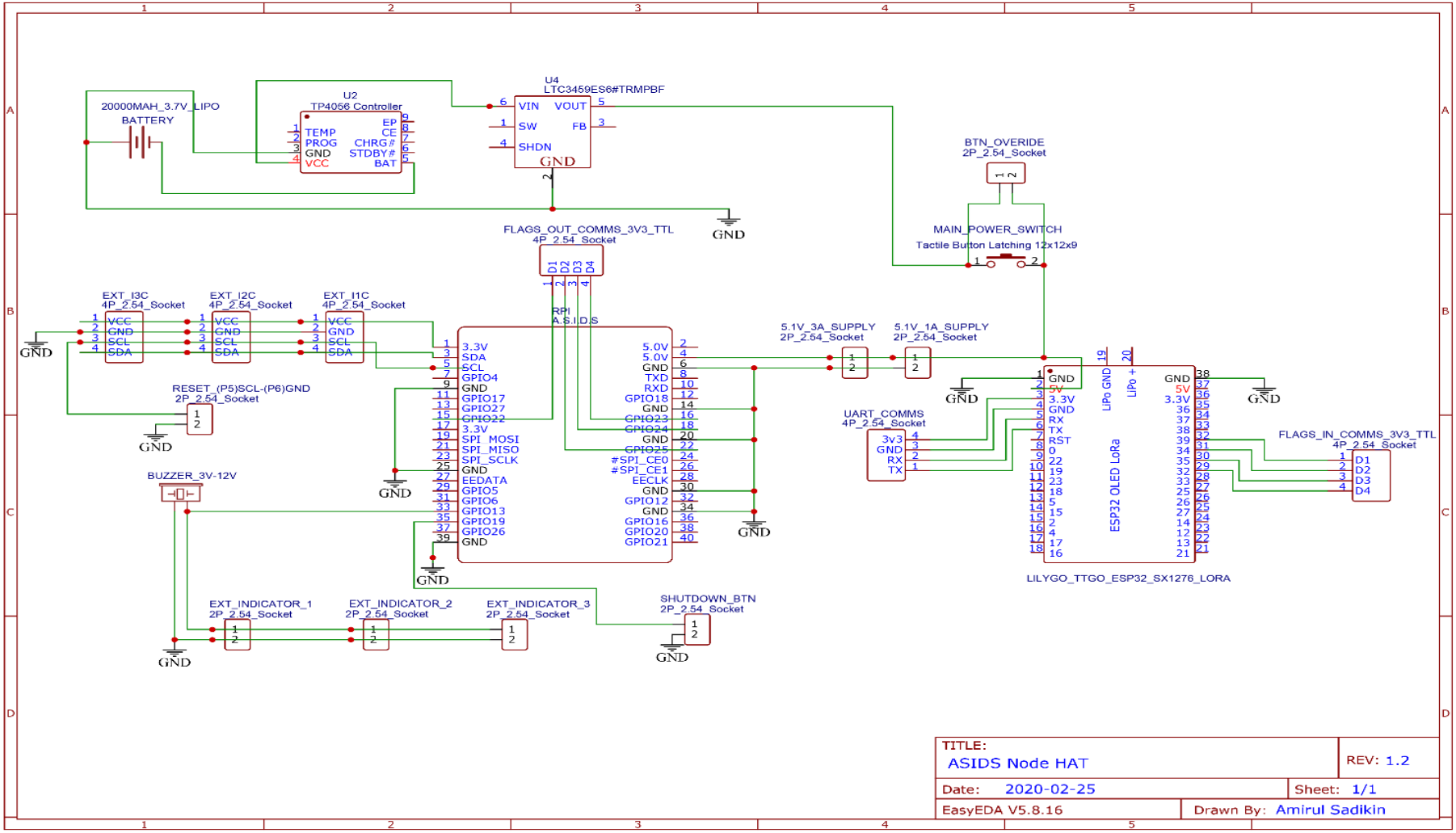


CAMERA TRAPS
with Wireless Communication
RM1000+ each
(COMPETITION)



20-100 Meters depends on trees

ASIDS NODE Schematics



AUTHOR'S PROFILE

Muhamad Amirul Sadikin Bin Md Afendi obtained Bachelor of Information Technology (Hons.) Intelligent System Engineering 5th July 2019. Also, a freelance full stack web-system developer since 2018, working on various projects creating workflow management software. Projects previously ventures into microcontroller programming, applying ML with microcomputers, electronics hardware prototyping, and webserver cloud computing. Recently completed a funded project, Acoustic Surveillance Intrusion Detection System (ASIDS) by Dec 2020 with Dr Marina Yusoff and Dr Zaki Zakaria. ASIDs is a standalone micro-computer that detects intrusion in wildlife reserve using on audio input and ML to monitor deep forest reserves autonomously to prevent poachers. The project is a proof-of-concept venture funded by MTFSB pitched on Nov 2018 and awarded RM 69,000 on May 2019.

LIST OF PUBLICATION

Awards

Competition: Invention Innovation and Design Competition 2020, UiTM, Malaysia

Members: Marina Yusoff, Amirul Sadikin and Zaki Zakaria

Project Title: ASIDs (Acoustic Surveillance Intrusion Detection System)

Medal: Gold

Participation Video: https://www.youtube.com/watch?v=RPdQ0rHnWNY

Published Articles

1. Afendi, A. S., & Yusoff, M. (2019). Review of anomalous sound event detection approaches. 8(3), 264–269. https://doi.org/10.11591/ijai.v8.i3.pp264-269

2. Afendi, A. S. M., Yusoff, M., & Omar, M. (2020). Mel-log energies analysis of authentic audible intrusion activities in a Malaysian forest. Bulletin of Electrical Engineering and Informatics, 9(2), 582–587. https://doi.org/10.11591/eei.v9i2.2091

Accepted Article

1. Afendi, A. S. M., Yusoff, M., & Zakaria M. Z., (2021). Sound Event Detection Based on Hybrid Convolution Neural Network and Random Forest Using Mel Log Energies Features IAES International Journal of Artificial Intelligence (IJ-AI) (Scopus Q2) ISSN/e-ISSN 2089-4872/2252-8938

Other Publications

Experiments & Source Code on Github ID:aiamirul

Bounded Report to Grant Provider (validation stage)

MALAYSIAN TECHNICAL STANDARDS FORUM BERHAD (MTFSB) GREEN ICT GRANT 2018 – Title: ASIDs – Acoustic Surveillance Intrusion Detection System. Duration: May 2019 to October 2021. Cost of Project: RM69,000.00

The ASIDs Project was broken into three milestones in which we created a short video of each milestone to be shown to MTSFB for their technical validation team monitoring the progression.

Milestone 1 – Data Collection Video:

https://www.youtube.com/watch?v=TN7kLvERBEc

Milestone 2 – Algorithm Developmnent

Video:https://www.youtube.com/watch?v=ILYfwK2NayI

Milestone 3 – Physical Prototype Development Video:

https://www.youtube.com/watch?v=fB1ZQQw7xTQ