A COMPREHENSIVE SOLAR POWERED REMOTE MONITORING AND IDENTIFICATION OF HOUSTON TOAD CALL AUTOMATIC RECOGNIZING DEVICE SYSTEM DESIGN

by

Abdullah Al Bashit, B.Sc.

A thesis submitted to the Graduate Council of
Texas State University in partial fulfillment
of the requirements for the degree of
Master of Science
with a Major in Engineering
August 2019

Committee Members:

Damian Valles, Chair

Michael Forstner

Semih Aslan

FAIR USE AND AUTHOR'S PERMISSION STATEMENT

Fair Use

Duplication Permission

DEDICATION

To the researchers in the field of signal processing, embedded system and machine learning who dedicated their time to the advancement of these technologies.

ACKNOWLEDGEMENTS

I thank Dr. Michael Forstner, the Department of Biology, Texas State University who shared knowledge, insights, and logistics required for this project that assisted the research. I thank Andrew MacLaren, Department of Biology for his assistance in collecting database and his experience on toad call frequency spectrum.

It is always a blessing to have a supervisor like Dr. Damian Valles for his in-depth thoughts, guidance, time management, and helpfulness. As a signal processing expert, Dr. Vishu Viswanathan's direction was very fruitful in producing the desired outcome. I have to appreciate my friend Rezwan Matin, MS in Engineering for his dexterous drawing for Automatic Recognizing Device (ARD). I give full credit to the designer of the ARD model Mr. Stephan Ronsonette, Amazon Robotics Maintenance, and Engineering Tech. He was the one who thoroughly built and designed the ARD. Saunders Drukker, Ph.D. Student in Dept. of Biology was the guy who always drove me over to the Bastrop where ARD was deployed.

Last but not least, the family is very important to me. In the journey to my master's degree, my parents, brother, and wife supported me all along. San Marcos Masjid brother's room is the place where I used to study and did most of my research work. This place gave me all the comfort and necessities required to conduct this research smoothly.

TABLE OF CONTENTS

Page

LIST OF TABLES

Table **Page**

LIST OF FIGURES

Figure **Page**

ABSTRACT

The Houston Toad is an endangered amphibian living in the edge of extinction. To save from annihilation, localization of their mating calls needs to be determined in order to protect the eggs from being hunted by predators. The current method of monitoring their vocalization lacks real-time and onboard voice recognition capability. In this research, a solar-based battery powered Raspberry Pi is designed along with a microphone that records environmental sound at prescribed intervals triggered by a Witty Pi module. This thesis proposes a naive approach to build a predictor model to detect the Houston Toad mating call signature through recorded audio files from the embedded design. These prerecorded several audio files have been analyzed to determine the unique characteristics of Houston toad call for their identification. The audio file is bandpass filtered, and then preprocessed by multiplying every frame with the hamming window into segments. Next, the Mel-Filterbank and Mel-Frequency Spectral Coefficient (MFCC) are used for feature extraction, and the Support Vector Machine (SVM) and Multi-layer Perceptron (MLP) neural networks are utilized as classifiers to determine the best fit. This experimental result reflects the higher accuracy of the MLP neural network over SVM showing the best potential of classification. The trained neural network predictor model is deployed in the Raspberry Pi to identify Houston Toad, and if there exists trailing in the call, it sends notification of time stamp via Email and SMS over the GSM and GPRS modules to the researchers.

1. INTRODUCTION

Introduction

The Houston Toad is an endangered amphibian that is endemic to Texas in the United States. It is estimated that only a few thousand adult Houston Toads are left in the world. The Department of Biology at Texas State University is leading a population supplementation project in which Houston Toads of early life-stages (eggs, tadpoles, etc.) are protected in the field or raised to a larger size in captivity before release into the wild. This effort seeks to increase the wild Houston Toad population. In addition to enhancing wild Houston Toad populations, the real-time identification of chorusing toads, subsequent confirmation of successful breeding events and partial egg collections is critical to population management. The Ingram School of Engineering is working in association with Department of Biology on the Toad Phone development project. The Toad Phone is a project which develops engineering solutions to monitor range-wide breeding activity using Automated Recognizing Devices (ARD's) with microclimate sensors, microphones, and development of over cellular network audio data transmission systems to enhance the efficiency and geographic coverage of the wild population.

Currently, in order to monitor potential breeding ponds, more than 80 Song Meter units sold by Wildlife Acoustics (Figure 1.1) are deployed at different toad breeding sites to record acoustical data. However, it requires researchers to travel to the locations of each data logger and extract data from the audio loggers every month, then process those data onto a proprietary desktop application by Wildlife Acoustics. This application has been configured to help in identifying the existence of toad calls but lacks all field continuity for

real-time notification transmission capabilities. To overcome some of the limitations, this thesis research work proposes the ARD system design that centralizes a solar-powered Raspberry Pi that signal processes, classifies, and transmits recorded data when a Houston Toad call is detected. The design needs to be rugged to withstand harsh outdoor environments and robust to stay operational in remote sites without requiring maintenance for long extended amounts of time.

Figure 1.1 Wildlife Acoustics Data Logger

A speech processing technique to find a solution for the existence of Houston Toad in a recorded audio file has been developed with certain constrains and limitations. Observations on the frequency spectrum of captured audio files collected by the Department of Biology for several ideal toad calls have been tested by the software Praat.

It can be quantified that ideal toad calls spectrogram lies from 1,600 Hz to 2,500 Hz in its frequency range. In order to develop the toad recognition technique, an audio file is preprocessed, framed, bandpass-filtered, feature extracted, and classified using neural network model. However, the developed signal processing method performs better while there is relative less interference with other species of the same kind in the toad's frequency range.

The thesis objective is to build an ARD system that records sounds at prescribed intervals and durations, as well as, to process the signal processing techniques to detect Houston Toad calls and operate uninterruptedly between service periods. The ARD handles captured data and transmits real-time notifications of toad calls signature with time stamps to the Department of Biology. The temperature, barometric pressure, and humidity are also identified as beneficial readings to record where ARD will be located. An ARD prototype is developed that executes neural network predictor model to detect toad call, stores environmental sensor information as well as recording data and send notifications in real time to meet the expectations of the Department of Biology for the conservation stewardship of the Houston Toad.

The current conditions of the Houston Toad

The Houston Toad, *Bufo houstonensis*, is endemic to southeast-central Texas and is listed as endangered by the State of Texas and Federally. It was among the first amphibians to be placed on the Federal Endangered Species list in 1970. The Houston Toad has undergone several significant reductions in its overall population numbers since its

description 60 years ago. Main drivers of the species decline have been habitat loss, degradation, and drought.

Head-starting is a management practice in which wild individuals of early life-stages (eggs, tadpoles, etc.) are protected in the field or raised to a larger size in captivity. Renewed head-start releases of Houston Toads began in 2013 in partnership with the Houston Zoo, Texas Parks and Wildlife Department (TPWD), Texas State University, and United States Fish and Wildlife Service (USFWS). In 2013, 19,000 eggs were released, 97,500 in 2014, and ~500,000 eggs released in 2015. The research thesis work seeks an engineering embedded solution in support of the Houston Toad recovery by: 1) using ARD's with microclimate sensors and audio recorders, and 2) development of data transmission systems over cellular network for real-time identification of chorusing Houston Toads allowing real-time confirmation of breeding events and thus enable subsequent detection and retrieval of partial egg collections to bolster genetic diversity in the assurance colonies.

History of ARD Development

The Department of Biology has been using data logger, a proprietary device, to capture the audio sound at different locations in Texas where the Houston Toad is present. After a few weeks of data collection, the research team drives to each location where data loggers were placed and collect the recorded data to process it on a proprietary desktop application to identify possible toad vocalizations in each file. This thesis proposes an embedded solution to this problem and provides a real-time notification structure of a toad

call signature detection system that facilitates in minimizing human labor, resources and time.

A collaborative relationship had been established with a team of faculties and students in the Ingram School of Engineering at Texas State University to develop the hardware and software to achieve the goal of real-time remote identification of calling Houston Toads and potentially other vocalizing organisms. During the Spring-2017 and Fall-2017 semester, three student teams in the Electrical Engineering Senior Design (EESD) program were sponsored to develop and build the prototype of the ARD's. The designs were based on the Raspberry Pi microcomputer platform equipped with a compatible sound card and Internet of Things (IoT) LTE cellular add-on board. The device was also equipped with a solar-battery power management system for continuous power. The EESD proposed design is shown in Figure 1.2. However, Houston Toad call detection signal processing was not accomplished with the prototype. Also, solar panel, battery charger, battery, power management board, IoT module and the whole frame are resigned to accomplish the signal processing requirement and the stability of the device. This thesis objective is to design the toad call detection signal processing phase, enhance the requirements of the hardware by developing software triggers for audio recording functionalities, while managing energy consumption through the solar power management system.

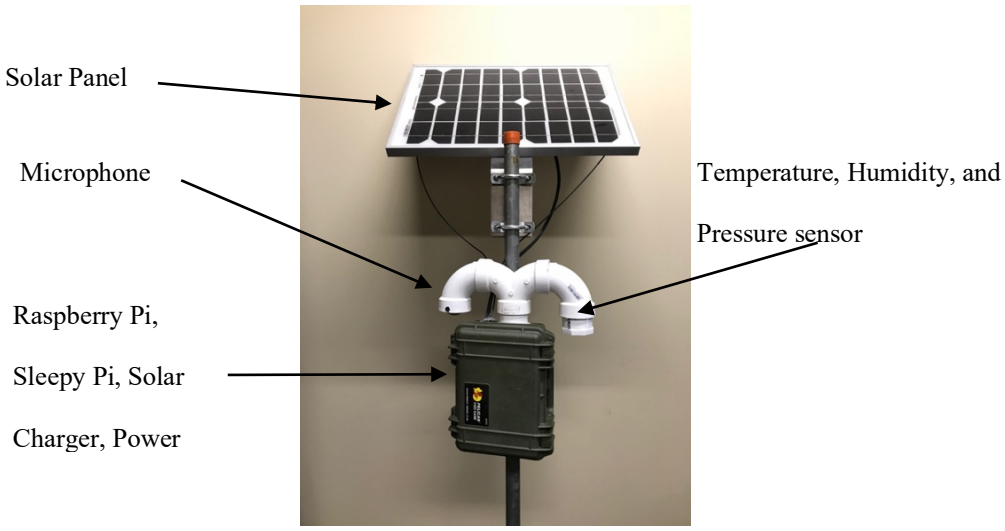


Figure 1.2: EESD Proposed ARD Detection System

Motivation towards building ARD

The Houston Toad male mating call is distinctive and can be used to identify whether the toad is in a particular area. Current efforts to locate and study them involves utilizing the Wildlife Acoustics Song Meters which is a proprietary device placed in remote sites that records audio of environment sounds. This is an inefficient method due, but not limited to, frequent extraction of data from audio loggers, routine maintenance of audio loggers, changing batteries on location, moderate to high cost, the lack of internet transmission capabilities, and power consumption management. The Toad Phone is an integrated project that involves resources analysis of the Houston Toad vocalizations, and engineering design that facilitates efficient research efforts in designing an ARD solution.

This research work is designed to help in the conservation effort of the Houston Toad by gathering audio data, easing the mode of data mining from audio recorders, and enabling real-time notification to the researchers for toad presence in a particular area toward an objective to the preservation of laid eggstrands. Currently, the Wildlife

Acoustics Song Meter is used by the Department of Biology to record audio sound without analyzing it on-board. The target ARD operation of the device will be able to record audio, environmental data (temperature, pressure, and humidity), and transmits the toad call signature real-time notification with the status of the device to a remote user. Due to the nature of audio data, the size and number of files can become large as the recorded intervals increase. This can potentially become a large cost of the research project due to cellular network data transfer rates for large files, and cloud data storage solutions. Researchers from the Department of Biology have requested that the desired device functionalities including building a cost-effective hardware embedded solution and developing efficient algorithms that will automatically recorded audio sounds, use an internet module to transmit the audio of the identified toad signature, and the overnight detection summary to the prescribed research personnel.

Scope and Emphasis

This thesis aims to develop a standalone embedded solution with signal processing and pattern recognition techniques to automate audio recognition and classification that targets the Houston Toad call. The voice recognition or speaker detection algorithms available in the field of natural signal processing are closely related to this research work. Analyzing and deploying these signal processing algorithms into a solar-powered hardware solution to detect the toad call in that device recorded file is the objective of the proposed thesis research work.

Outline of Thesis

The coordination of this thesis is as follows: Chapter 2 presents a review of some previous works related to signal propagation techniques for Houston Toad detection system. Next, chapter 3 and 4 presents a description of the proposed system which contains a detailed description of signal processing and ARD Hardware building procedures respectively. Result analysis are shown in chapters 5. Finally, chapter 6 presents conclusions and prospects for future work.

2. LITERATURE REVIEW

In this section, audio signal processing techniques used by several researchers have been scrutinized by the target vision. The majority of those conference and journal papers are for human voice or speaker recognition applications while only a few are for animal signal identification. It can be stated that the signal detection algorithms work similarly for human voice processing as well as animals. The human voice frequency spectrum and toad frequency spectrum exists at different frequency levels. Moreover, most of the human voice detection algorithms work well for low noise environments and decrease in efficiency within noisy environments. In this case, it is observed that as the recorder is located close to the ground near potential breeding ponds, many types of noise overlaps with the desired toad signal. It is suspected that some of the other animal species have frequencies overlapping decreasing the efficiency of the desired Houston Toad call. However, processes involving the extraction of the existence of any specific signature in a signal is similar for any voice source. Some of the techniques used by the researchers in different applications that are related to this thesis are discussed in the subsequent paragraphs.

Signal Propagation Technique for Houston Toad Detection

In this paper [1], the animal voice pattern recognition algorithm has been developed. The developed animal voice recognition system uses the zero-cross-rate (ZCR), Mel-Frequency Cepstral Coefficients (MFCC) and Dynamic Time Warping (DTW) joint algorithms as the tools for recognizing the voice of the particular animal. The ZCR

approach was used for the endpoint detection of input voice such that the silenced voice can be removed. The MFCC was utilized for feature extraction while the voice pattern classification was done by using the Dynamic Time Warping (DTW) algorithm. After the feature extraction using the MFCC technique, the next process was the matching of reference and tested signals in order to undergo a verification process. However, the voice input signal varies in terms of speed or time when compared with the reference voice signal. Therefore, these two signals must be aligned in order to get the optimal match between them. By using DTW, calculating the minimal distance between the reference and input voice particular animal was detected. However, this paper used DTW classifier to recognize the same trained and test data, which is not the case for an effective system installed in the Houston Toad natal breeding sites. This is due to each time the designed system audio recording is different from the trained classifier model with the previously collected audio file.

In the research found in [2], gender identification process for speech signals using three different features namely pitch using autocorrelation, signal energy, and MFCCs and Linear Support Vector Machine (SVM) classifier is used for classification of features extracted from above three methods. Two sets of experiments were performed - in the first experiment, one speech file was tested against one training file as a one-on-one experiment. In the second experiment, one speech file was tested against three training files. The authors used the Texas Instruments Massachusetts Institute of Technology database for their work. It is discussed a classification model called Hidden Markov Model (HMM) for the classification process. However, their observation was that the HMM had depicted certain limitations when dealing with multiple observations. Also, it was discussed of

another study that utilizes the Gaussian Mixture Model (GMM) for gender classification for pitch related features combined with Relative Spectral Perceptual Linear Predictive (RASTA-PLP) coefficients. The GMM, which was trained with different covariance matrices and components, achieved above 98% accuracy for clean speech and 95% accuracy for noisy speech. Finally, the work showed the efficiency and high-level accuracy of utilizing three extractions methods with a single classifier for gender identification speech audio signals.

For the work in [3], experiments have been performed on the database obtained from IIIT- Hyderabad without separating voiced and unvoiced frames of randomly taken from 200 male and female speech signals. The accuracy of gender identification claimed in the literature reached 99% of true-positives. Using pitch and MFCCs that are calculated from extracted voiced frames, the accuracy of gender identification has gone up to 99.5% using SVM according to this paper. Also, the work in [4] stated that for certain types of sound, in which essential characteristics are reflected in the high-frequency range, a linear-scale in frequency provide more information than the Mel-scale approach. Along with that, it was proposed a deep neural network (DNN)-based voice activity detection system using a linear-scale feature.

As in [4], in order to train and test the developed automatic person identification system, an in-house voice database was created that contains recordings from 100 people from 25 different countries usernames (50 males and 50 females). Each person had to utter his/her voice 30 times whereby the training data set contains 20 repetitions per person, whereas the testing data set contains the remaining ten repetitions per person. Therefore, a total of 3,000 utterances were collected. To extract features from the voice signals, the

MFCC technique was applied, producing a set of feature vectors. These feature matrices were fed into MATLAB that uses Vector Quantization (VQ) for features training and classification. Their analysis was in an objective to classify gender classification of male and female in clean speech.

In [5], the voice filtering by attenuating human voice's frequencies within an instrumental song's frequencies using MATLAB band-stop filter design with Butterworth windowing. The specified stop-band is based on the human voice's frequencies. The chosen one stop-band is used for male voice filtering and female voice filtering to see if it is suitable for both kinds of voices. The purpose of voice filtering includes to clean the voice itself from a set of sound. The method for the optimization by FFT the signal, and the results are to show its frequency spectrum and repeat the process until the heard voice has the optimal result. Optimal stop-band specifications are chosen by varying F_{pass} and F_{stop} and hearing filtered instrumental sounds from the song. When the clear instrumental male voice was found, then that stop-band applied for a female voice filtering and obtained an almost clear instrument sound. In this case, a few instrument sounds were chopped, and a female voice was heard as a whisper and kept as a successful experiment. This paper depicts the idea of using band-stop filter to separating undesired signal what can be a suitable concept to separate noisy signal.

In [6], the research work consisted of performing simultaneous segmentation and classification of bird species using a Convolutional Neural Network (CNN) with encoder-decoder architecture and working on developing a Recurrent Neural Network (RNN) model which considers the temporal relation between bird syllables. In [7], the Karnuhen-Loeve Transform noise reduction algorithm as an additional pre-processing sub-stage with

Fundamental Frequency detection as a voiced/unvoiced segmentation method was implemented with the Voice Analysis software called Praat. In [8], [9], [10], the MFCC-based human auditory filtering model was applied for audio signal feature extraction and Artificial Neural Network (ANN) was evaluated for automatic speaker recognition system. However, in [11], a text independent speaker recognition system based on MFCC and GMM in the design of speaker models with HMM Toolkit was used which aimed to use it as a biometric authentication system. In these papers [8], [9], [10], and [11], male and female voices were taken as an input, on the contrary, this thesis research focuses on the identification of Houston Toad call voice activity detection with exact time stamp and duration.

ARD Hardware design

Signal processing techniques play a key role in identifying Houston Toad of an audio file. These detection methodologies are programmed in a microcomputer, in this research Raspberry Pi. Considering the fact that the device would operate in an outdoor environment, an ARD has to build that records sound at specified intervals and save the data to local storage. In this thesis, a Raspberry Pi is powered by one 12-Volt (V) / 20-Amps (A) parallel rechargeable batteries that are charged by a 12V/20A solar charger connected with a 35W solar panel. It also collects environmental information such as temperature, humidity and atmospheric pressure. Powered with a rechargeable and portable battery and a solar panel, it can operate for long durations without the need to change batteries. This is accomplished by integrating a Global System for Mobile Communications (GSM)- based General Packet Radio Service (GPRS) module, the Short Messaging Service

(SMS) functionality, and Email notification of the Houston Toad call in a file to the Department of Biology.

In [12], a Raspberry Pi was utilized for the real-time remote solar monitoring system using LabVIEW and in [13] and [14], crossover of certain paraments i.e. temperature humidity and real-time clock was triggered by an SMS alert system. In this paper, the Raspberry Pi also needed to be integrated with a GPRS and GSM module.

In[15], a solar panel and battery sizing requirements were estimated. Recording audio of the surroundings at prescribed intervals using a Witty Pi 2 module are configured using official documentation [16], and detection of toad call is performed in Python [17] and MATLAB. In addition to recording, configuration settings for temperature, atmospheric pressure and humidity, three sensors combined in BME280 module, is illustrated with example in the Adafruit website [18].

Conclusion

It is observed from the literature reviews that speaker recognition undergoes stages of preprocessing, framing, windowing, feature extraction, and classification. To achieve the best performance for this particular application of Houston Toad detection, several signal processing algorithms were tested and applied. In regard to approaching Houston Toad identification, the recorded audio signal has been preprocessed first, and then a bandpass filter was applied to the preprocessed signal before applying the framing and windowing. For feature extraction out of several available methods mentioned earlier, the MFCC and Mel filter bank was implemented. Whereas for classification, it was observed from the papers that ANN reflects better performance than other available algorithms. The

initial progress on implementing some of these techniques are analyzed in the methodology and initial results section. However, building an efficient Houston Toad signal detector software requires testing different available algorithms and compare their performance to find the best fit.

3. PROPOSED SYSTEM

The goal of this thesis is to build an ARD solution that records audio sound and performs data signal processing for the detection of the HT calls on that device. A computational and power-efficient low-cost device and embedded components in the ARD has been redesigned. The ARD has the capability of recording sound files of ambient audio every ten minutes at each hour overnight generating an overall of 130 minutes each day from 6 PM to 7 AM. This design was placed in known breeding sites, and storing the recorded sound data, as well as, transmitting the ARD toad call signature information to the Department of Biology when HT calls are detected. The device utilizes a solar-powered battery as its power source and capable of storing data for at least 180 days without maintenance. The device collects and stores other environmental parameters such as temperature, atmospheric pressure, and humidity. The signal processing design for the ARD was developed using Python, and the frequency spectrum observation Audacity and Praat applications were utilized. The purpose of the signal processing component design is to feature extract from the HT calls and execute classifier algorithms. In this prototype, Mel Filterbank and MFCC were implemented for the feature extraction phase. The Support Vector Machine (SVM) and the Multi-Layerd Perceptron (MLP) were designed to perform the classification of the HT calls.

Generic Signal Processing Techniques

Through the literature review, it is observed that the Voice Activity Detection (VAD) and Automatic Speech Recognition (ASR) signal processing stages are very similar to the HT call detection system, which is illustrated in Figure 3.1.

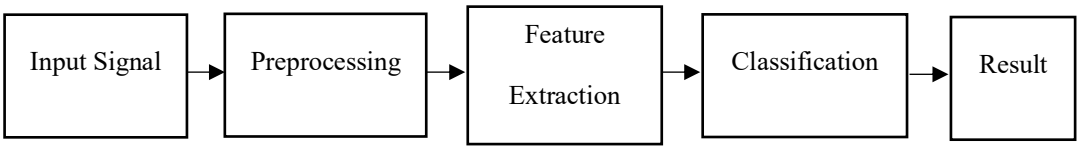


Figure 3.1 Basic Signal Processing Stages in a block diagram

The input audio signal pre-processing techniques are essential for optimal recognition. Typically, speech or audio segments must be separated from non-speech or undesired portions to achieve high recognition accuracy. The conventional pre-preprocessing techniques include Zero Crossing Detector (ZCR), signal energy, pitch detection, thresholding, and other techniques that help to find a voiced frame in a signal. However, the environment background sounds can be noisy that using these techniques cannot always be separated. In contrast, if the desired signal spectrogram lies in a particular frequency band, then the bandpass filter in this range can eliminate the unnecessary frequency in each audio file to be analyzed.

After the preprocessing stage, feature extraction aims to obtain a feature vector that generates a matrix of unique voice representation of the HT call in a more compact, less redundant, and more suitable way for statistical modeling. Therefore, the essential features of the speech signal are extracted and used for further signal processing. The widely used techniques for feature extraction are Linear Predictive Cepstral Coefficients (LPCC),

Perceptual Linear Predictive Coefficients (PLPC), Cepstrum Analysis, Linear Frequency Cepstral Coefficients (LFCC), and the Mel Frequency Cepstral Coefficients (MFCC). For feature extract for the HT call, the MFCC and Mel Filterbank methods are considered for the ARD design.

To recognize these feature vectors as a toad or non-toad, a voice predictor model is generated by using classifier algorithms. A template of features needs to be created from a HT call and use it as a reference for the recognition process. It means each frame of the feature vector will be compared to a reference model to discriminate a toad or non-toad call. The output of the classifier should provide a final result that shows the recognition of a HT. Table 3.1 shows a list of conventional classifiers for the VAD and ASR process.

Table 3.1 List of Potential Classifiers

Sl. No.	**Classifier**	**Algorithm Type**
1.	Dynamic Time Warping (DTW)	Unsupervised
2.	Hidden Markov Model (HMM)	Unsupervised
3.	Gaussian Mixture Model (GMM)	Unsupervised
4.	Support Vector Machines (SVM)	Supervised
5.	Multi-layer Perceptron (MLP)	Supervised
6.	Vector Quantization (VQ)	Unsupervised

The classifier algorithms can also be supervised or unsupervised. In supervised learning, labels are assigned to data with known outcomes that help model the classifier to recognize desired outcomes. For unsupervised learning, labels are not attached to the data, and the classifier model is developed through error and correction as data is being acquired and processed. Due to the availability of recorded audio files that contain HT calls, the development approach is to use a supervised classifying model that can provide a high level of confidence in recognizing a HT call and not be computational-intensive for the ARD.

Houston Toad Call Detection Signal Processing Techniques

The Department of Biology had set up Wildlife Acoustics Song Meters (Figure 1.1) at potential toad breeding zones to collect sound files of potential toad calls in the last few years. The audio files that contain HT calls can be visualized by looking over the frequency spectrum using the Praat application. Praat is a software application that utilizes audio files and helps to visualize frequency spectrums. After importing an audio file to the Praat application, the frequency spectrum is observed for the dark black shaded region at 2 kHz as a signature of the potential HT call as seen in Figure 3.2. To firmly ensure the presence of HT call, this portion of the spectrum is heard utilizing headphones. Subsequently, going over this process on several files, a new database of audio file names with the duration of toad and non-toad calls have been generated to train and test this detection system.

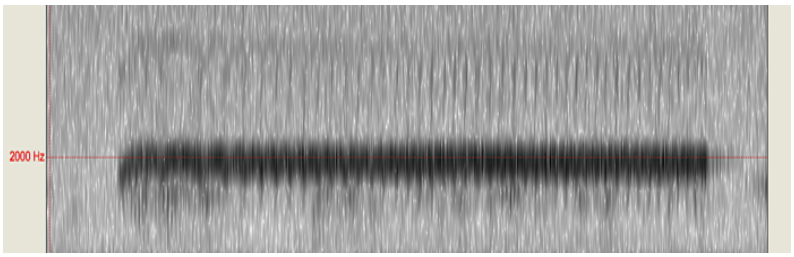


Figure 3.2 Praat tool Houston Toad frequency spectrum

The HT call detection signal processing phases are shown in Figure 3.3. Figure 3.3 entails the stages of audio file reading, processing, analyzing, and evaluation steps applied in Python to process and classify the presence of HT calls.

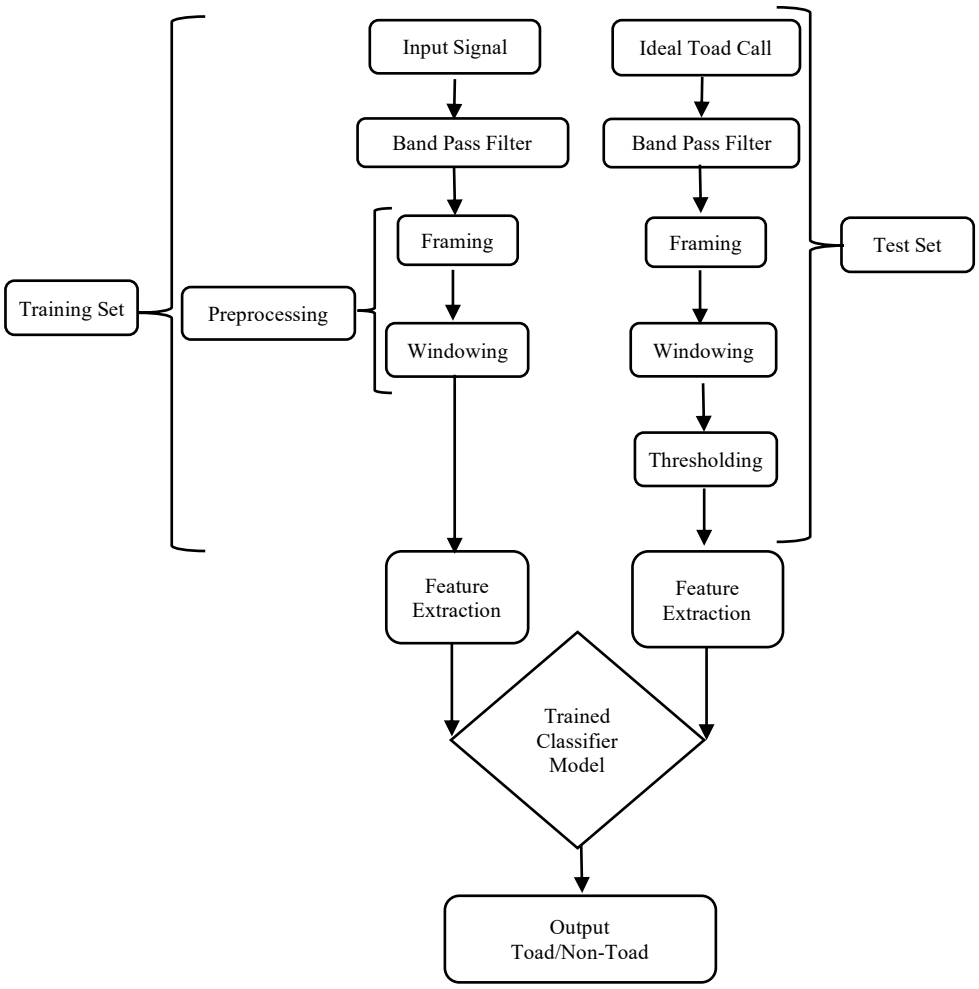


Figure 3.3 Houston Toad call Recognition Signal Process block diagram

The signal processing begins with reading recorded audio data. This data is bandpass filtered and broken into smaller frames by multiplying them with Hamming windows. Several ideal HT calls have been observed, and those characteristics will be compared with these smaller frames and identified as potential frames. These frames are then feature extracted and trained with the classifier to make a final decision to whether the file has a HT call or not. In an unknown audio file or testing the accuracy of the model, similarly, it is preprocessed, framed, windowed, feature extracted, and the best candidate model is used to identify a HT call. It is worth considering that the model that shows the highest accuracy is the candidate model, and its parameters are exported for future prediction.

The audio files containing potential toad calls were collected by the Department of Biology at Texas State University. Out of the terabytes of the database, files that have a toad and non-toad data in it have been randomly selected to evaluate and analyze the HT detection system. Thus, a list of files with time location of HT calls has been created to create a database. In this thesis, a database of 3,426 frames, a frame is the one-second duration of 16,000 Hz sampling rate, of toad and non-toad have been created. Among the 3,426 labeled data, 2,075 frames are used for training the SVM and MLP classifier, and 1,351 frames for testing purposes.

Input Audio Signal File

In this step, an audio file containing either a toad or environmental sound is read using the Scipy wav library [19] which returns two-dimensional arrays as stereo channels. The channels data stored in columns are 16-bit integers ranges from −32,768 to 32,767.

The recorded sound has a similar frequency spectrum in both channels but shifted in the time-domain [20]. Thus, in considering any one channel, it will serve the purpose of detecting a HT call in an audio file.

Bandpass Filter

The HT call spectrum as seen in Figure 3.2. shows high-frequency variations from 1,600 Hz to 2,500 Hz. The audio signal is passed into a sharp-edge bandpass filter with a cut-off frequency at this range to narrow down the HT detection computation task.

An elliptic filter design as the sharp-edge bandpass filter [21] is considered in order to obtain a low ripple in the passband and stopband. The filter coefficients, property and frequency response are described, tabulated and plotted in (3.1), Table 3.1 shows a list of conventional classifiers for the VAD and ASR process.

Table 3.1, and

Figure 3.4 (a) respectively. In Figure 3.4 (a), the blue color plot is for amplitude response, and the green color is for an angular response. The filter is applied to the input wav-audio file, and the output of the bandpass filtered signal spectrum is shown in

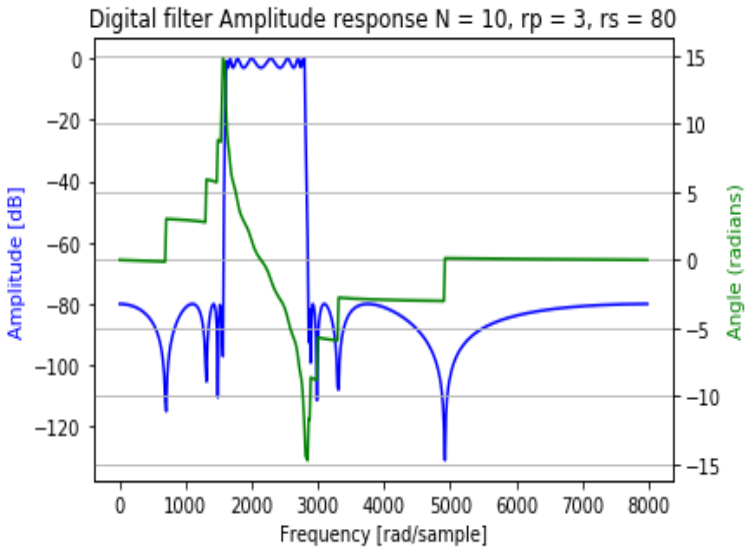
Figure 3.4 (b).

$$H(z)= \frac{B(z)}{A(z)} = \frac{b(1)+b(2)z^{-1}+\ldots+b(2n+1)z^{-2n}}{a(1)+a(2)z^{-1}+\ldots+a(2n+1)z^{-2n}} \quad (3.1)$$

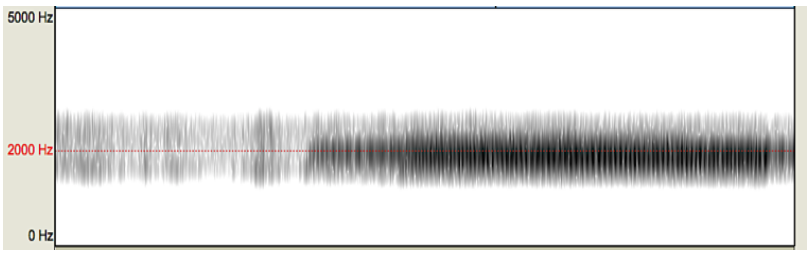
, in where *n* is the filter order. For the filter design consists of a 10^{th}-order filter.

Table 3.2 Elliptic Filter Property

Filter Property	**Value**
filter order, n	10
maximum ripple of rp	3 dB
minimum attenuation of rs	80 dB
lower passband frequency, Wpass1	1600 Hz
upper passband frequency, Wpass2	2500 Hz



(a)



(b)

Figure 3.4 (a) Elliptic Filter Frequency Response in Python, (b) Bandpass Filtered Spectrogram Centered at 2 kHz in Praat

Preprocessing

In the preprocessing step, the one-dimension bandpass filtered signal is separated into small frames of samples that are called frame segmentation or framing [22]. Consequently, to minimize spectral distortion and keep the continuity of the signal, a window is required to multiply on each frame. A Hamming window utilizing (3.2) is applied on each frame to taper the voice sample to zero at the beginning and end of each frame as seen in Figure 3.5 [23]. Note that, if the last frame does not equal to sampling rate, the magnitude of zero is padded at the end of the file to make the same frame length. The frame size is similar to the sampling rate of 16,000 Hz, one-second long, with 0% overlap between consecutive frames.

$$w[n]=0.54 - 0.46 * \cos(2\pi nN-1) \tag{3.2}$$

, where n is the samples, $0 \leq n \leq N-1$, and N is the frame size.

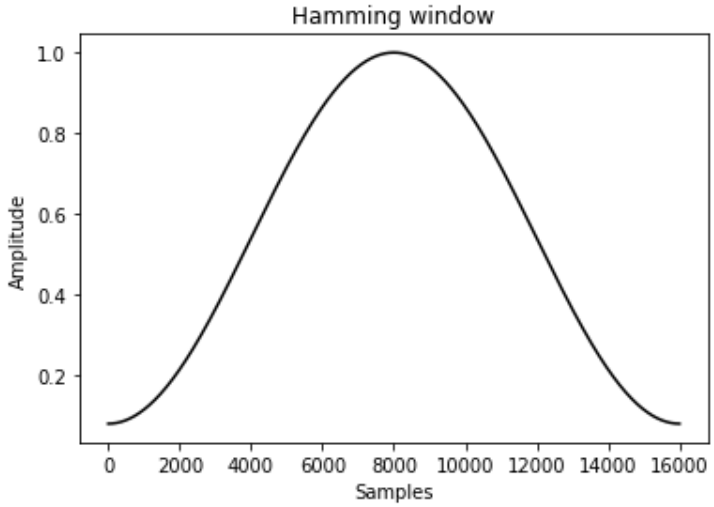


Figure 3.5 Hamming Window

Thresholding

The preprocessed signal generated after windowing, applies thresholds to find potential frames for toad calls in new audio file. Thresholding is not necessary for training the machine learning algorithm shown in Figure 3.3. The thresholds in Table 3.3 are estimated by observing several ideal HT call audio files. The experimented characteristics of the Welch Power Spectral Density (PSD) [24] are compared with the imported frames and identified as potential HT frames. The outcome of this phase is the identification of lesser frames that contain HT calls and some misclassified background noise at around 2 kHz. These frames are fed into the feature extraction algorithm as potential HT frames. The comparisons are necessary to shorten down the number of potential HT call frames in an audio file with these frequency characteristics.

Table 3.3 Thresholds applied in each frame

Welch PSD Peak Frequency (Hz)	**Welch PSD width (Hz)**
1900 ≤ Peak Frequency ≤ 2100	PSD width < 850
2100 < Peak Frequency < 2200	PSD width < 400

Figure 3.6 is a plot of preprocessed frame Welch PSD against frequency from 1,562.5 Hz to 3,125 Hz. A potential HT call has a PSD peak frequency about 2,000 Hz, and width around 600 Hz, actual threshold parameters are mentioned in Table 3.3. The blue vertical line in Figure 3.6 is the peak frequency, and the red horizontal line is the 50 % of the PSD width at the peak frequency.

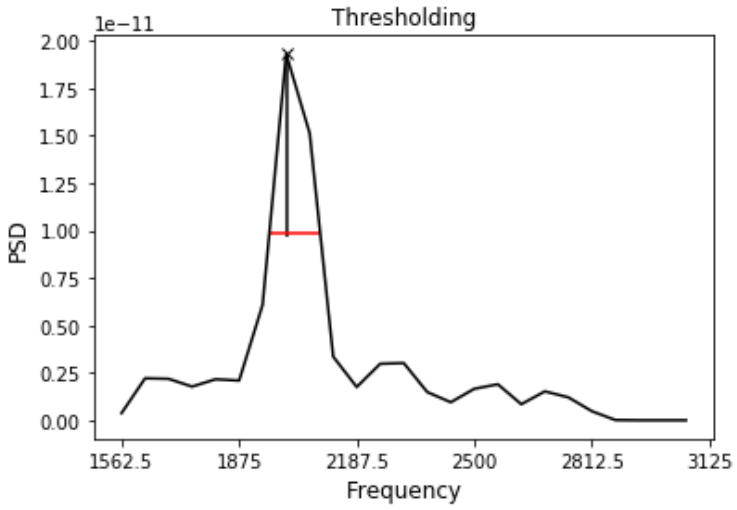


Figure 3.6 Thresholding applied after preprocessed signal

Feature Extraction

After preprocessing and thresholding, the techniques used for feature extraction are the Mel Filterbank, and the Mel-Frequency Cepstral Coefficients (MFCC). These algorithms of feature extraction are applied to generate unique feature coefficients for HT calls. The development for the feature extraction used the Python Librosa [25] library to implement these algorithms.

As in [26], computing filter banks and MFCCs involve the same procedure in where both cases of the filter banks are computed. Later, the MFCCs can be obtained by applying a Discrete Cosine Transform (DCT). The audio signal goes through a bandpass filter, it then gets sliced into non-overlapping frames, and a Hamming window function is applied to each frame as previously stated. Further when implementing threshold, a N-point at 2,048, the Short-Time Fourier Transformation (SFFT) on each frame is computed using (3.3) to obtain the power spectrum or periodogram, subsequently, the Mel filter banks,

$$p=\frac{|FFT(x_i)|^2}{N} \tag{3.3}$$

, where x_i is the ith-frame of signal x.

Mel Filtering Bank

As in [26], the steps to computing filter banks is to apply triangular filters, typically 40 filters, *nfilt* = 40, on a Mel filterbank to the power spectrum to extract frequency bands. The Mel filtebank aims to mimic the non-linear human ear perception of sound by being more discriminative at lower frequencies and less discriminative at higher frequencies. Conversion between Hertz *h(i)* and Mel *m(i)* are performed using (3.4) and (3.5). Each filter in the filterbank is triangular having a response of one in magnitude at the center frequency and decrease linearly towards zero until it reaches the center frequencies of the two adjacent filters where the response is zero. The Mel Filterbank is shown in Figure 3.7.

$$m(i)=2595\log_{10}(1+{}^{f}\!/_{700}) \tag{3.4}$$

$$h(i)=700(10^{{}^{m}\!/_{2595}}-1) \tag{3.5}$$

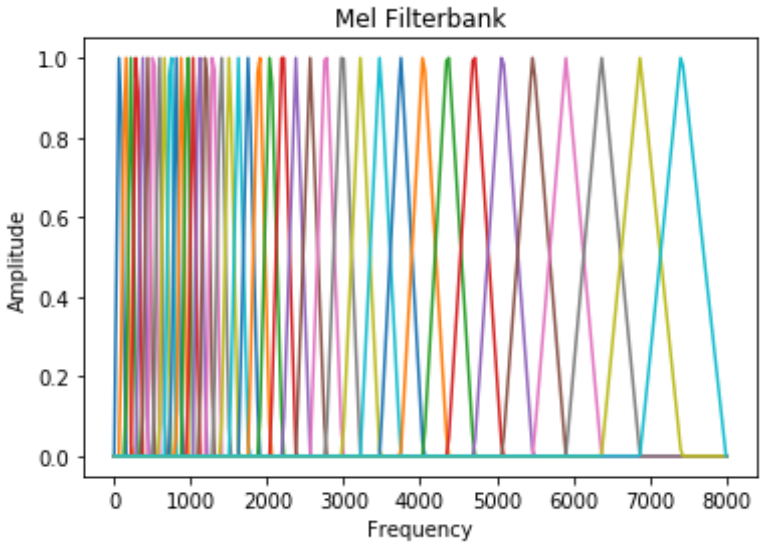


Figure 3.7 Mel Filterbank of 40 Mel filters on 0-8000 Hz

Typically for ASR, the resulting cepstral coefficients 2-13 are retained, and the rest are discarded [26]. However, they represent fast changes in the filter bank coefficients, and these fine details of 40 cepstral coefficients are chosen to contribute to the ASR HT call detection system. For the ARD to have triangular filters with 40 filter banks, 42 points are required. Using (3.4), the upper and lower frequencies are converted to Mels. In this case, lower frequency at 1,600 Hz is 1,340.67 Mels, and the upper frequency 8,000 Hz is 2,840.06 Mels. Alternatively, (3.5) is used to convert Mels back to Hertz. To convert the frequencies to FFT bin numbers *f(i)*, the FFT frame size, of *nfft* = 2,048, and the sample rate *fs* = 16,000, is calculated by (3.6).

$$f(i)=\text{floor}[(nfft+1)\frac{h(i)}{f_s}\ ] \tag{3.6}$$

The total number of the Filterbank is 42 which ranges from 1,600 Hz to 2,500 Hz with a 2,048-point FFT size. The equation (3.7) is computed for calculating Filterbank.

$$H_m(k)=\begin{cases} 0, & f(m-1)<k<f(m-1) \\ \dfrac{k-f(m-1)}{f(m)-f(m-1)}, & f(m-1)\leq k<f(m) \\ 1, & k=f(m) \\ \dfrac{f(m+1)-k}{f(m+1)-f(m)}, & f(m)<k\leq f(m+1) \end{cases} \quad (3.7)$$

, where m = 40 is the number of filters. After applying the filter bank to the power spectrum (periodogram) of the signal, the Mel Filterbank matrix is generated.

Mel-frequency Cepstral Coefficients (MFCCs)

As in [26], the filter bank coefficients computed in the previous step are highly correlated, which is problematic in some machine learning algorithms. Therefore, the DCT technique is applied to decorrelate the filter bank coefficients and yield a compressed representation of the filter banks. Thus, the output of the DCT generates 40 MFCC coefficients for each frame that are fed into a classifier algorithm.

Classifiers

A voice predictor model needs to be generated to recognize these feature vectors as a toad or a non-toad by using the classifier algorithms. A template of features will be created from a toad call and used as a reference for the recognition process. Meaning, each frame of the feature vector will be compared to a reference model to discriminate a toad or non-toad call. The output of the classifier should provide a final result that shows the recognition of the HT call. SVM and MLP are types of classifiers that are applied for the toad call detection process. The classifier algorithms can also be supervised or unsupervised data

formats. For this case, supervised learning is implemented in where labels are assigned to data with known outcomes that helped to model the classifier to recognize desired outcomes. The goal is to develop a classifying model that can provide a high level of confidence in recognizing the HT call.

According to the Scikit-learn preprocessing MinMaxScaler documentation [27], before feeding data into the classifier, the feature matrix has to be preprocessed to scale all the features ranging from 0 to 1, where transformation is given by (3.8) and (3.9).

$$X_{std} = \frac{X - X.min(axis=0)}{X.max(axis=0) - X.min(axis=0)} \tag{3.8}$$

$$X_{scaled} = X_{std}(max-min) + min \tag{3.9}$$

, where *min = 0*, *max = 1, X.min (axis=0) and X.max (axis=0)* represents the minimum and maximum value of each column in matrix *X*. This estimator scales and translates each feature vector individually such that it is in the given range on the training and testing set between 0 and 1. For the classifier design of the HT call detection, it was scaled to 2,075 frames for training and 1,351 frames for testing data have feature values ranges from 0 to 1.

Support Vector Machine

As in the Scikit-learn documentation [28], the SVM is a classifier that separates data points using a hyperplane with the most considerable amount of margin. For this reason, the SVM classifier is also known as a discriminative classifier. The SVM finds and constructs an optimal hyperplane in multidimensional space at an iterative manner to separate different classes minimizing error, which is visualized in Figure 3.8, in where the

x-axis and y-axis are features. The SVM uses a kernel method that takes a low-dimensional input space and transforms it into a higher dimensional space. Meaning, it converts the non-separable problem to separable problems by adding more dimensions to it to build a more accurate classifier. For the classifier development, the Linear and the Sigmoid kernel were implemented to justify the accuracy of the binary classifier for the audio files that contain HT call or none.

The main objective is to segregate the given dataset in the best possible way. The distance between either nearest points in the binary classification is known as the margin, a hyperplane with the maximum reasonable margin between support vectors in the given dataset [28]. The Linear and Sigmoid SVM kernels are measured separately for finding accuracy in the Mel Filterbank, MFCC, and both features at the same time.

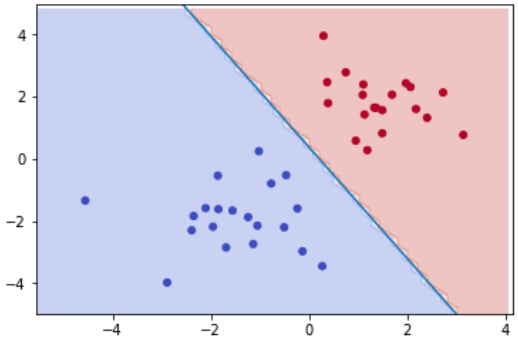


Figure 3.8 Typical Linear SVM

Multi-layer Perceptron (MLP)

As in [29], the Multi-layer Perceptron (MLP) is a supervised learning neural network algorithm that learns a function $f(X)$ by training on a dataset, where m is the number of dimensions for its input. Given a set of features $X = \{x_1, x_2, ..., x_m\}$ and a target y, it can learn a non-linear function approximator for either classification or regression. It is different from a logistic regression, in that between the input and the output layers, there

can be one or more non-linear layers called the hidden layers. For the MLP design, 2,075 training examples with 80 feature vector and 1,351 testing samples were utilized to check the accuracy by using the Scikit-learn MLP algorithm. The layer number, as well as hidden nodes, have been varied to get the maximum efficiency of the neural network. In the following section single neuron neural network and MLP are discussed.

Single Neuron Neural Network

A single neuron standard neural network model is shown in

Figure 3.9. This is called an artificial neuron that has one or several inputs and only one output neuron. When stacked, it is called fully-connected neural network. From Figure 3.9, the X_i are the inputs of the neuron, the weight W_i, and the bias b_i are the learning parameters of the neurons. The weight quantifies how much entry is essential in the final output, whereas the bias is an offset. The weights multiply the inputs, and it helps the network to adjust faster to the known targets. The output is a function of the weighted sum of the input and the bias. The goal of each neuron is to evaluate the weight and bias value for a model iterating over the training samples.

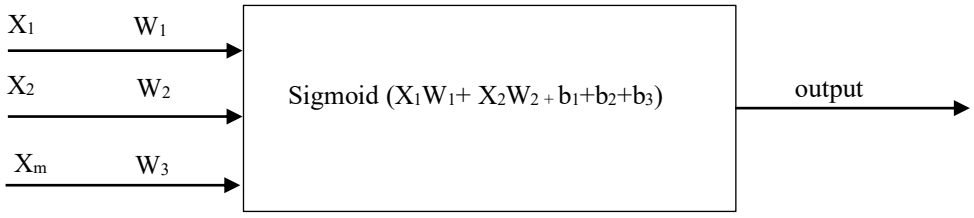


Figure 3.9 A single neuron neural network

A neural network develops a function that fits a given set of points. If the output is linear, it means that a simple linear equation, such as $y = wx+b$, can fit the data. However, the approach does not work well when dealing with nonlinear applications that cannot be represented by a linear function. This property can be achieved by a nonlinear function called the Activation function. Without any Activation function, the connected neurons would be equivalent to only one neuron with specific weights and therefore, a simple weighted sum as an output [30]. Figure 3.10 is the plot of sigmoid, Rectified Linear Units (ReLU) and Hyperbolic Tangent *tanh* Activation functions that bring nonlinearity to the neural network.

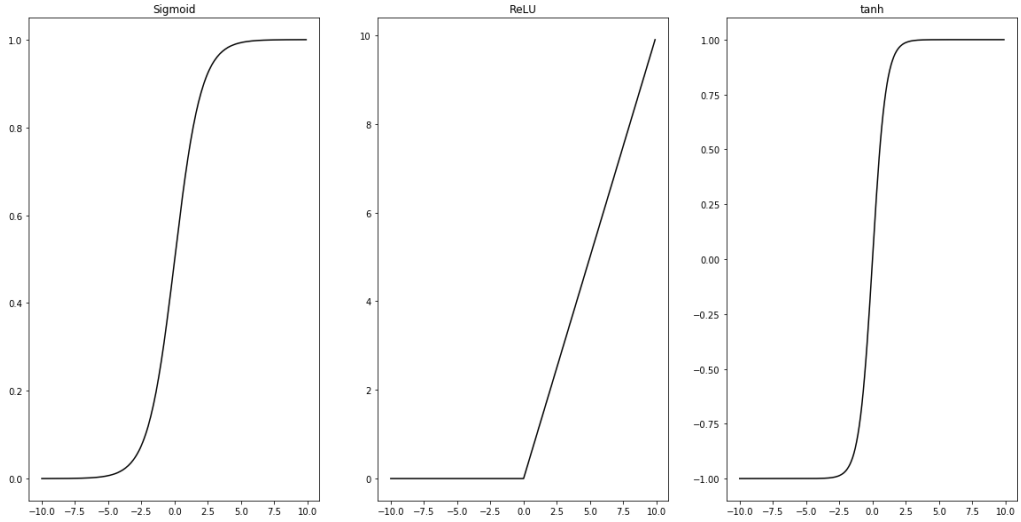


Figure 3.10 Sigmoid function, ReLU and tanh Activation functions

The objective of neural network is to find the right values for weights and bias so that the output of the neuron matches with the desired output. To do this, randomly changing the values of *Wi* and *b* would take too much time; also, there are endless possibilities when network is dense [30]. To predict an output of a linear relationship for a specific input, a line of best fit to the data needs to be determined. It is obtained through

optimization that involves minimizing the error between the data points and the line of best fit. One possibility would be least squares that minimizes the square of a distance from the line to a point is the cost function of the neural network model. The error function aiming to minimize error is defined in (3.10) [30]. The error is minimized when the gradient is equal to zero. The values of W_i and b are found using gradient descent, where N in the number of samples. Therefore, the Mean Square Error (MSE) is evaluated for loss or error function.

$$\text{Error}(W,b)=\frac{1}{N}\sum_{i=1}^{N}(y_i-(W_i x_i+b) \tag{3.10}$$

The step size is kept constant. However, if it is too large, the solution can diverge. A method called backtracking line search is sometimes used to avoid divergence, where a certain amount decreases the size at each iteration. In practice, an approximate of the step size from a formula is calculated, or the exact optimum step size is used [30]. Ideally, the step size decreases monotonically as the algorithm converges. For the classifier design a constant step size of 0.001 is used. The MSE loss function determines the deviation from the desired output to actual output, and the gradient descent calculates the derivative of the loss function with respect to the parameters W_i and b. Through gradient descent, the result of this calculation tells how the error is changed if the parameters are changed.

MLP Neural Network

A large number of connected neurons create MLP NN that can eliminate one neuron generalization in the single NN model and introduce non-linearity with multiple neurons to solve HT detection problem. That way a more accurate predictor model is created which

can be shown in Figure 3.11. It represents "neurons" perspective in a multi-layer neural net [31].

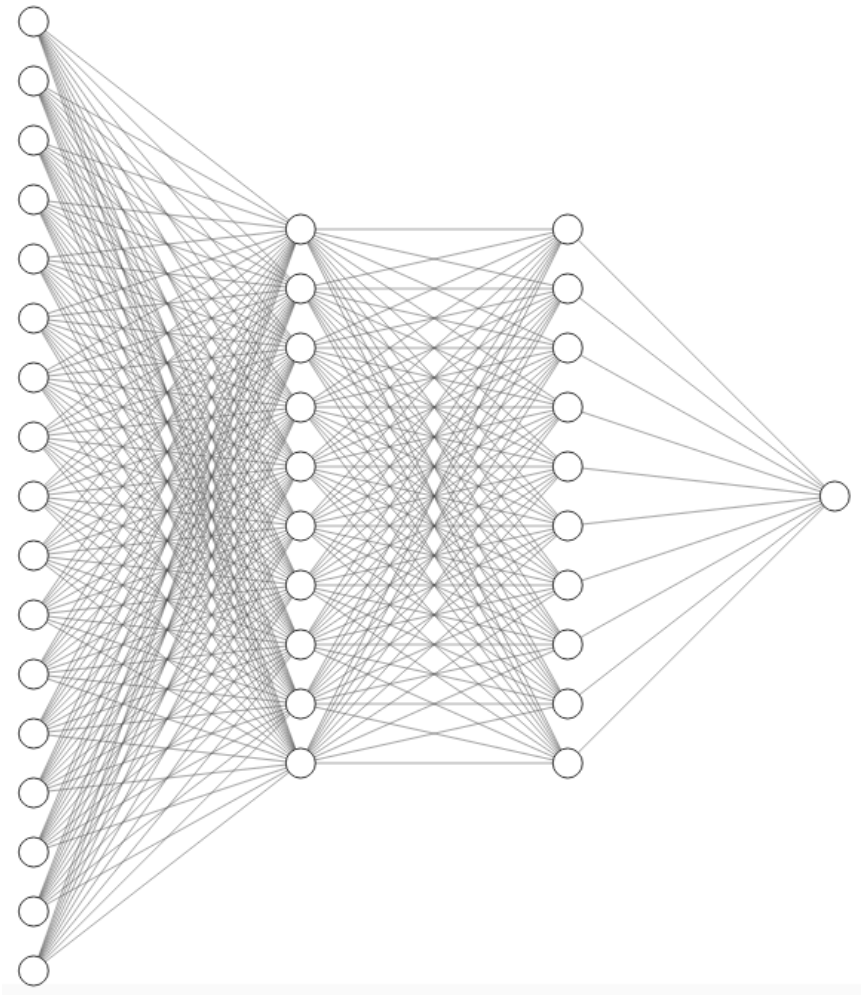


Figure 3.11 An example MLP Neural Network (17 input layer, 10 neuron hidden layer-1, 10 neuron hidden layer-2, 1 output layer (Toad or Non-Toad))

The network is split into three parts called the input, hidden, and output layer. The input layer takes the input data, Mel Filterbank, or MFCC, or both features, that contains as many neurons as there are inputs. For the network design, Mel Filterbank and MFCC each contains 40 features of 2,075 training data. The output layer gives the result of the network, which contains as many neurons as there are output categories. In this paper, there

is only two categories, either HT is present (=1) or no toad (=0) in the output layer. That is why one neuron is required at the output layer and called binary classification. The number of neurons in the hidden layer is everything in between the input and output that depend on the problem, and it is experimental. The neuron layers in Figure 3.11 are called fully-connected layers or MLP because every neuron of a layer is connected to every neuron of the previous and the next layer. The computation of a complete MLP neural network, mathematical expressions of forward and backward propagations, and parameter update formulas are discussed in the subsequent sections.

Forward Propagation

As in [32], the input X_i provides the initial information that propagates to the hidden units at each layer and finally produces the output Y. The architecture of the network entails determining its depth, width, and Activation functions used on each layer. Depth is the number of hidden layers, and width is the number of units (nodes) on each hidden layer since neither input layer nor output layer dimensions can't be controlled. There are quite a few sets of Activation functions typically used such as ReLU, sigmoid, tanh, etc. From case to cases, deeper networks outperform to shallower networks with more hidden units. Therefore, it is always better and won't hurt to train a deeper network with diminishing returns.

In this stage, the input, hidden and output neurons respectively X, H and Y. The input neurons to next hidden layer weights and this hidden layer to output layer weights are respectively W_1 and W_2. Hidden and output bias respectively B_1 and B_2. The equations in Equation (3.11) and (3.12) represents the forward bias.

$$H = \text{sigmoid}\,(XW1 + B1) \tag{3.11}$$

$$Y = \text{sigmoid}\,(HW2 + B2) \tag{3.12}$$

At this point, input, weight, and bias matrix calculations forward from input to output layer. In every iteration of the forward propagation, every sample in the training set is multiplied with the same weight and bias matrix. After all training set is passed through the network once, the new weight and bias matrix are evaluated by backward propagation.

Backward Propagation

Backpropagation is the reverse path propagating from output to the input neurons when parameters are updated. Otherwise, the network would output the same result every time. First, the derivative of the loss function needs to be calculated with respect to W and B. Equations (3.13), (3.14), (3.15) and (3.16) are the backward propagation of the gradient loss functions with respect to W and B.

$$\frac{\partial J}{\partial B_2} = \delta_2 = \left(Y - Y^*\right) * f'(HW_2 + B_2) \tag{3.13}$$

$$\frac{\partial J}{\partial B_1} = \delta_1 = \delta_2 . W_2^t * f'(XW_1 + B_1) \tag{3.14}$$

$$\frac{\partial J}{\partial W_2} = H^t . \delta_2 \tag{3.15}$$

$$\frac{\partial J}{\partial W_1} = X^t . \delta_1 \tag{3.16}$$

These gradients follow the steep of the function to find the minimum loss (which means actual output = desired output). The MSE loss function is a power of two, which is

a convex function. In convex function, there is only one minimum, so the learning process will not be stuck in a local minimum but will approach to the global minimum.

Update Parameter

The last piece of the puzzle on building MLP model is updating weights and bias parameters. In the forward propagation, the network output is calculated, and then the derivative of the loss function with respect to the parameters is determined. The weight and bias parameters are updated to reduce the loss. It is done by deducting the old values with the gradient multiplied by alpha, α. In Equations (3.17), (3.18), (3.19) and (3.20), alpha is an arbitrary constant called learning rate, which is multiplied with the gradient.

$$B_2^{new} = B_2^{old} - \alpha * \frac{\partial J}{\partial B_2} \tag{3.17}$$

$$B_1^{new} = B_1^{old} - \alpha * \frac{\partial J}{\partial B_1} \tag{3.18}$$

$$W_2^{new} = W_2^{old} - \alpha * \frac{\partial J}{\partial W_2} \tag{3.19}$$

$$W_1^{new} = W_1^{old} - \alpha * \frac{\partial J}{\partial W_1} \tag{3.20}$$

Since loss function is convex like x^2, a derivative of f(x) with respect to x goes toward zero. Now, iterating the training dataset through this process ultimately decreases error. However, constant step size is simple but is often not the optimum value. Although in some cases, it can lead to the divergence of the solution. In MLP Neural network, multiple variable functions (weights and bias) gradient is a vector, and it requires to point

toward the direction that will increase the function as quickly as possible. That is why In this paper for faster convergence, Limited-memory Broyden-Fletcher-Goldfarb-Shanno (LBFGS) optimizer is used [33].

Tuning SVM and MLP classifier parameters, a predictor model is selected for the identification of HT calls. The model parameters are exported and used for new audio file HT signature detection.

Limiting HT Call Duration

Each frame is 125ms in length in the HT call detection for each new audio file. A classifier makes a decision for each frame in order to measure the duration of the HT calls. All the potential frames in the audio file that are less than 1 second away are considered to be in the same HT call. The duration of a HT call is an essential factor as it is required to receive a notification for the HT calls that are significantly larger than 5 seconds long. If less than 5 seconds, the HT call duration is then rejected as noise.

HT trailing detection

It is observed that the HT call contains a trailing section before the steady signal as the Welch PSD peak frequency decreases gradually as in shown in Figure 3.12. The classifier model identifies the HT call either identifies the trailing section or not. The requirement of notification should be doen via Email or SMS only if the trailing is present in the HT call. Thus, the classifier detected HT call starting frame is identified first. One second before the identified frame is taken into consideration for trailing detection. If there

exist three consecutives Welch PSD peak frequency decreases, the call is considered to be a trailing HT call.

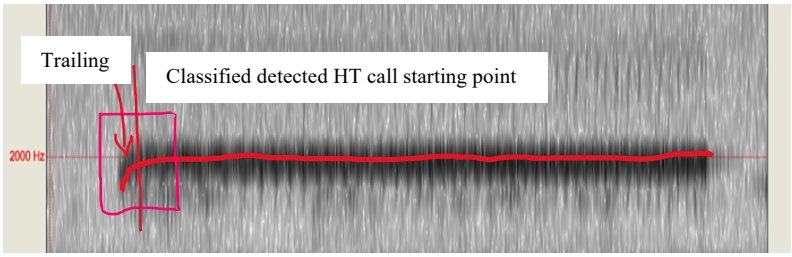


Figure 3.12 HT Trailing frequency spectrum

Audio file HT Detection

In summary, a new audio file is analyzed by a bandpass filtering, framing and windowing, thresholding, feature extracting, and a predictor model that outputs toad call frames. After the classification of each frame, limiting the HT call duration for more than 5-seconds long and three consecutive frames decreasing the PSD peak frequency signifies trailing HT call, process described in Figure 3.13. The ARD sends notification at the prescribed schedule to the Department of Biology for the presence and detection of a HT.

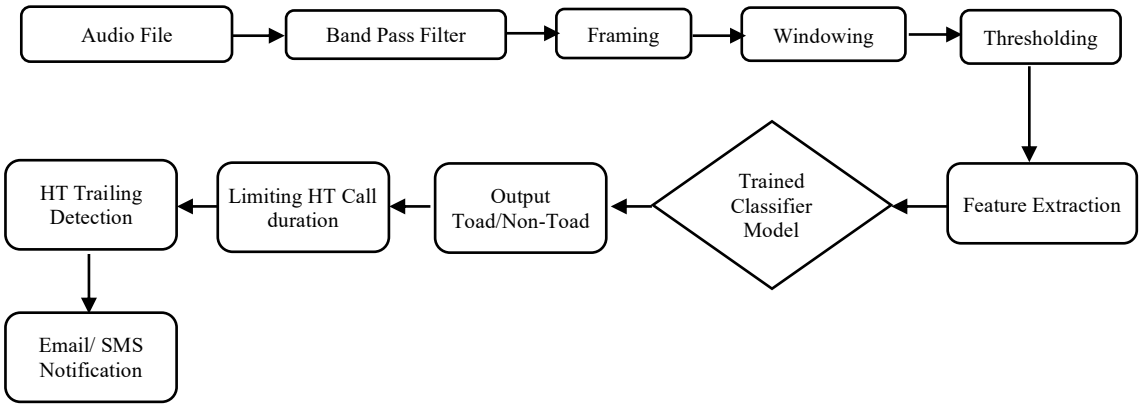


Figure 3.13 Houston Toad Audio Detection block diagram

4. HARDWARE DESIGN

The ARD concept design supports the software that classifies the HT call with signal processing and a NN model explained in Chapter 0, and the hardware implementation for the ARD concept. The software design handles speech signal processing to the recorded audio file for the presence of the HT call. An NN model [17] has already been developed by using a several audio HT call files for training and testing purposes. The software is written in Python and run under the Raspbian Operating System (OS). This chapter focuses on the hardware design and software integration with the OS and NN model. The hardware consists of a power-managed microcomputer that is solar-powered, battery sourced, General Packet Radio Service (GPRS) enabled autonomous, weatherproof microphone recording, and a mircoSD storing device. The software integration enforces the Raspberry Pi microcomputer to develop a bash script and Python program to configure the Witty Pi 2 power-management board, sound recording, signal processing, and modem settings to send Email and SMS notification via the GPRS module each time it boots up.

The solar-powered and battery connected ARD system uses a microphone and an environmental sensor module to capture sounds, temperature, humidity, and pressure data of the atmosphere, respectively. The Witty Pi 2 is integrated for the power-management and a cellular module for transmission of notifications. The captured sound and environmental readings are recorded with exact date and time and stored in a 200 GB microSD card from where the Raspbian OS runs.

Figure 4.1 shows the detail components of the hardware design that encapsulates the ARD design.

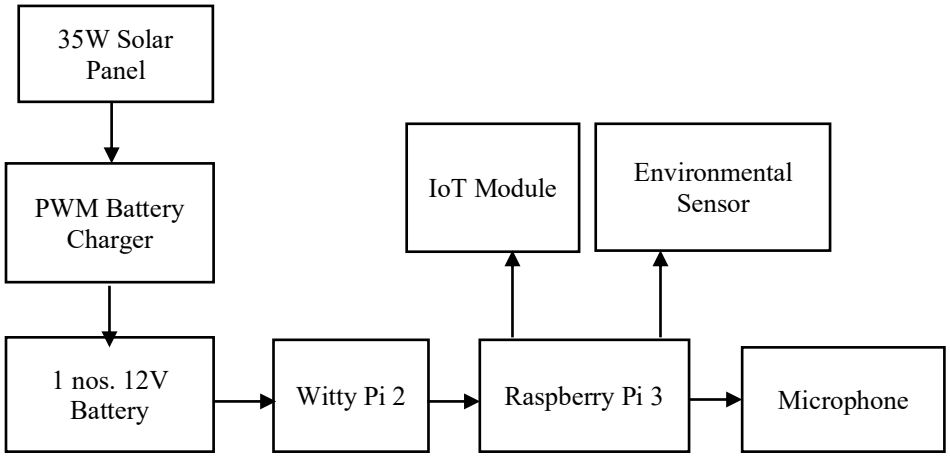


Figure 4.1 ARD Hardware Design Block Diagram

Raspberry Pi

A Raspberry Pi 3 is chosen, on the contrary of Arduino, as the microcontroller for the device because it is cost-effective, powerful, capable of performing signal processing tasks, and supports onboard USB ports as seen in Figure 4.2. The peripherals such as the Witty Pi 2, environmental sensor module, and the GPRS module offer support for the Raspberry Pi. It is also capable of supporting the memory and processing intensive functions, such as Python packages, that are integrated and compatible with the Raspbian OS.

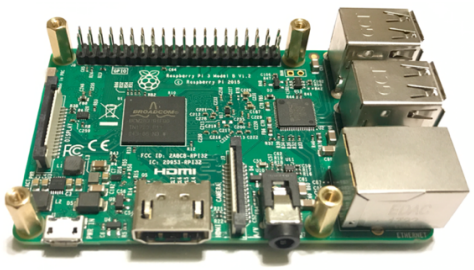


Figure 4.2 Raspberry Pi

Solar Panel and Solar Charger

The solar panel is implemented to harness solar energy to power the Raspberry Pi over the lead-acid battery and to make the ARD a standalone embedded solution. The designed system utilizes a 35W solar module and 12V / 20A Pulse Width Modulation (PWM) solar charger to charge the batteries and provide power to the Raspberry Pi over the USB port. While fully charged the controller has nominal voltage of more than 13 V. Specifically, if it is more than 13.5 V, that means battery is fully charged.

Assembling a solar panel is crucial. At first, the user must secure the PWM solar charger to the enclosure box using the Velcro/Gorilla tape. The solar charger should be placed as close to the battery as possible. This will allow the user to use the shortest length of wires, which will reduce power loss through the cables. Then, the user must connect the positive and negative connections of the battery to the battery terminals of the solar charger. After that, the positive and negative connections of the solar panel must be connected to the solar panel terminals of the solar charger. It is mandatory that for assembling solar panel, battery should be connected first and for disassembling, battery should be the last thing to disconnect. The solar charger has two USB connectors to supply load, and Witty Pi, that is placed on top of Raspberry Pi, can use any of the ports.


Figure 4.3 Solar Panel (Left), PWM Solar Charger (Right)

Lead-acid Battery

One lead-acid battery cell shown in Figure 4.4 has a rating of 12V/20A, is selected to charge and provide sufficient power required for the Raspberry Pi, the connected peripherals, and sensors during operation from dusk to dawn. In the adverse weather during cloudy days, it can deliver sufficient power to maintain operation, and continue recording and detecting HT calls by the ARD.

Figure 4.4 Lead-acid Battery cell

Microphone

The omnidirectional microphone seen in Figure 4.5 captures sound from all directions in the environment. The selected microphone is a condenser type transducer, and the frequency-response ranges from 50 Hz to 16 kHz. This microphone is configured within the Python script that starts recording while the Raspberry Pi boots up to capture audio.

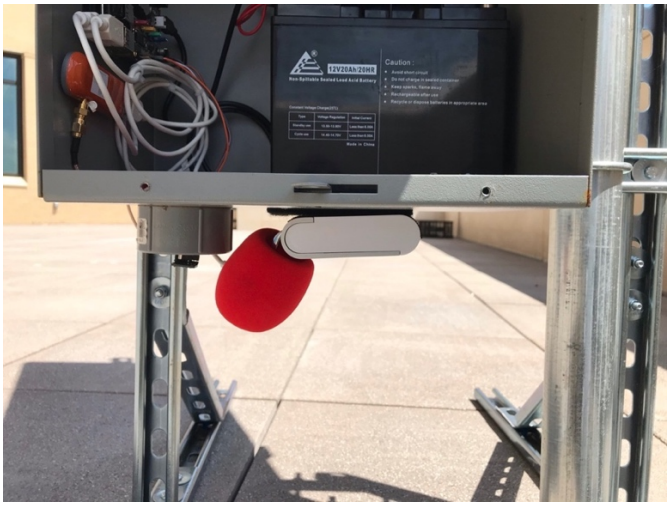


Figure 4.5 Microphone coved by red windscreens foam

Environmental Sensor Integration

The Adafruit BME280 shown in Figure 4.6 is a combined module with humidity, pressure, and a temperature sensor interfaced with the Raspberry Pi over the Inter-Integrated Circuit (I2C) communication port. The module's I2C connectivity with the Raspberry Pi is programmed in the Python script, which starts capturing data right after audio recording is completed. According to Adafruit [18], the environmental sensor is connected with the Raspberry Pi by the Serial Peripheral Interface (SPI) communication protocol. For the ARD design, the sensor Chip Select (CS) pin is connected with the

Raspberry Pi Digital 5 pin displayed in Figure 4.6 shown wired to the SPI port and the environmental sensor, Adafruit BME 280, connected to the Raspberry Pi.

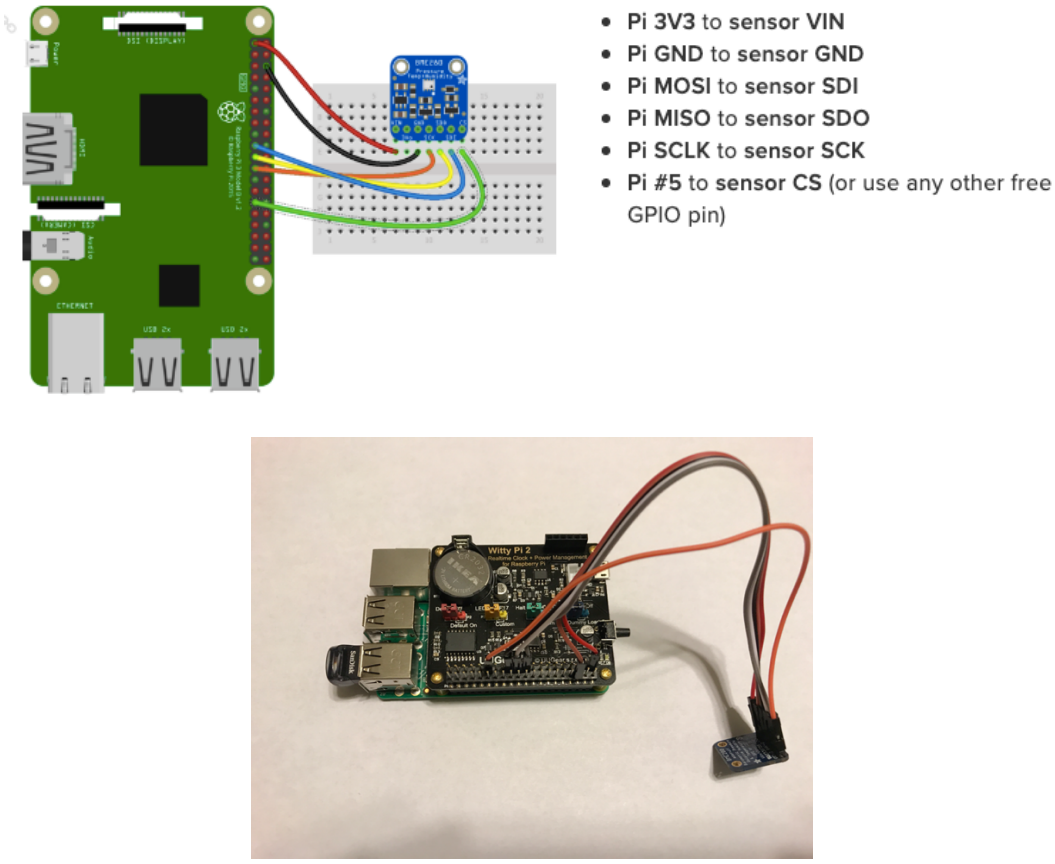


Figure 4.6 The Raspberry Pi 3 Model B wired with SPI (top) Environmental Sensor, Adafruit BME 280, connected with Raspberry Pi (bottom)

General Packet Radio Service (GPRS) Module

The GPRS modem features multi-communication functionalities: Global System for Mobile Communications (GSM), GPRS, and other features of communication. that supports SMS, phone calls, MMS, Email, and other forms of transmission formats. The modem is configured in the Raspberry Pi OS and integrated into the Python script so to run only when an HT call is detected in a file. The program first tries to send an Email, and if

internet connectivity fails, then it sends an SMS notification with a time stamp and duration of the HT call via a cellular network.

Figure 4.7 Huawei GPRS Modem (Orange color) connected with Raspberry Pi in ARD

Witty Pi 2

The Raspberry Pi is a power expensive device to run continuously without eventually draining significant battery current. The objective of Raspberry Pi is to boot up every day in the evening to record sound and environmental data for ten minutes. Then, it does all the signal processing for HT detection and shuts down before next hour. This operation at hourly intervals is executed by the Witty Pi 2 power management board. From 7 PM to 7 AM, the Witty Pi module wakes the Raspberry Pi up to take readings and turns it off when not in use.

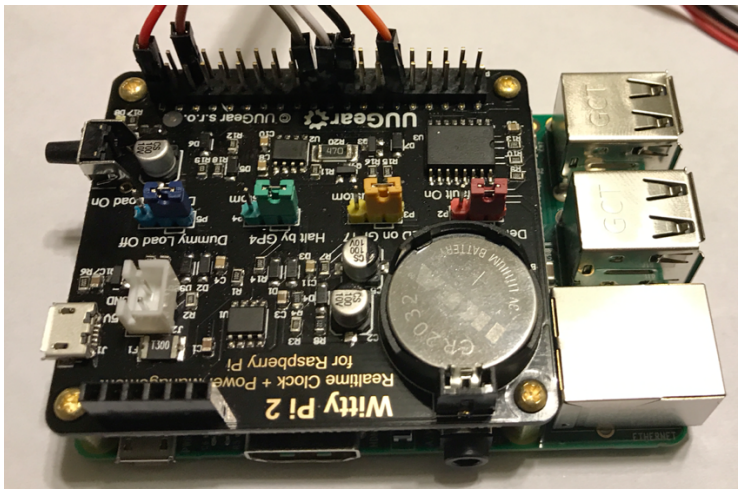


Figure 4.8 Witty Pi placed on top or Raspberry Pi

5. RESULTS

The ARD consists of two parts: the HT detection software, and hardware with sensor integration. Chapter 0 explained the HT signal processing techniques and machine learning algorithms applied to the design. The HT detection stages in an audio file are shown in

Figure 3.1. These steps involve preprocessing involving bandpass filtering and framing, thresholding, feature extraction methods, with the Mel Filterbank and Mel-Frequency Spectral Coefficient (MFCC), and classify with the Support Vector Machine (SVM) and Multi-layer Perceptron (MLP) NN. The output of the classifier provides a final result that shows the recognition of a HT call, one for HT and zero for no HT. If the classifier detects a HT call in an audio file, the GPRS module sends Email, or due to internet unavailability an SMS, at the scheduled time set in the ARD. In this section, the best neural network trained model will be evaluated and deployed in the ARD for the further evaluation of HT in a new or unseen file. Chapter 0 described the hardware and sensor components used in this project. This chapter will bring both chapters together to build an ARD.

HT Detection Software

The detection system starts with applying an elliptic bandpass filter where frequency ranges from 1,600 Hz to 2,500 Hz to the potential HT call audio file. The filtered signal is factored into frames of one-second duration and each frame is multiplied with the Hamming window for smoothing discontinuities at the beginning and end of the sampled signal. The selected frames are feature extracted using the Mel Filterbank and MFCC

method. In total, 80 features are generated for each frame and each extraction method generates 40 features.

These features are the input matrix to the binary classifier to train and test the system. Initially, the Mel Filterbank and MFCC feature extraction methods separately used for the SVM and MLP classifier. For the classifier design, 2,075 training examples with 80 feature vector and 1,351 testing samples were used to check the accuracy applied using the Scikit-learn MLP algorithm.

SVM Classifier Model Selection

The highest accuracy of 90.52% is observed when the Mel Filterbank and MFCC both features are incorporated in the SVM binary classifier. The summary of the accuracy in different kernels is tabulated in Table 5.1. It is noted that only the Mel Filterbank or MFCC features do not provide higher accuracies as opposed to using both the feature vectors together for binary classification.

Table 5.1 SVM Kernel vs Accuracy

Sl. No	**SVM Kernel**	**Accuracy for (MEL Feature vector=40)**	**Accuracy (MFCC Feature vector=40)**	**Accuracy (MEL + MFCC Feature vector=40+40=80)**
1.	Linear	63.36 %	81.64 %	90.52 %
2.	Sigmoid	62.99 %	62.99 %	62.99 %

MLP Classifier Model Selection

The MLP classifier model was trained with the previously recorded audio files to detect HT call. The best trained NN predictor model is the model that has the highest accuracy, exported as a file, and utilized to identify the signature of the HT call in an audio file. The newly recorded audio data is processed in Python through bandpass filtering, framing, thresholding, the feature extracting, and MLP classifying predictor model. The Python program is executed by a bash script deployed at the Raspberry Pi's boot up. The layer number, as well as hidden nodes, have been varied to get the maximum efficiency of the neural network as listed in Table 5.2. The estimated highest accuracy is obtained in a two-layer Neural network having 200 and 100 nodes respectively.

Table 5.2 Neural Network Layer vs Accuracy

Sl. No	Layer-1 Nodes	Layer 2 Nodes	Layer 3 Nodes	Layer 4 Nodes	Layer 5 Nodes	Accuracy
1.	200	-	-	-	-	95.78%
2.	200	100	-	-	-	98.07%
3.	100	300	200	-	-	98.07%
4.	300	200	100	-	-	98.00%
5.	100	300	400	200	-	86.67%
6.	100	200	300	400	-	96.40%
7.	100	400	200	300	500	85.49%
8.	100	200	300	400	500	92.70%

Model Selection

Although a linear kernel SVM generated maximum accuracy of 90.52%, the MLP outperformed the SVM with an accuracy of 98.07% with just two layers iterated over 2,075 training data and 1,351 test data. It reflects that the MLP model can do better estimate on identifying HT calls and be used as a predictor model. Thus, the best candidate model is the two-layer MLP-NN having 200 and 100 nodes, respectively. This model weight and bias matrix is exported and deployed in the ARD.

Raspberry Pi HT detection Software Integration

The MLP is the classifying model implemented and integrated with the signal processing techniques to identify HT calls. The signal processing programs execute every time Raspberry Pi boots up, store the recorded sound file in the flash drive connected with the Raspberry Pi via USB port, and do all the signal processing tasks with the MLP predictor model to identify HT calls. The OS is loaded in Raspberry Pi MicroSD card, and it was created from an iso-image file using the Apply-Pi Baker software and restored to reproduce the ARD HT detection task. A snapshot of backing up the Raspbian OS that has the HT detection software is presented in Figure 5.1.

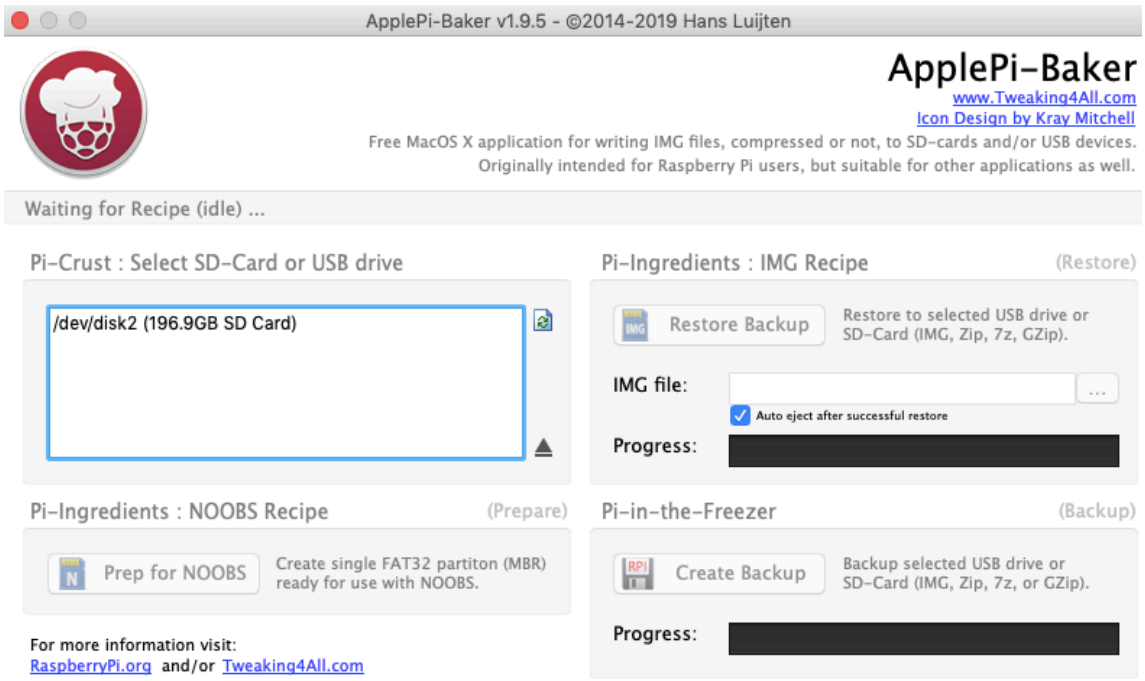


Figure 5.1 Creating Image of Raspbian OS

ARD Component Selection and Integration

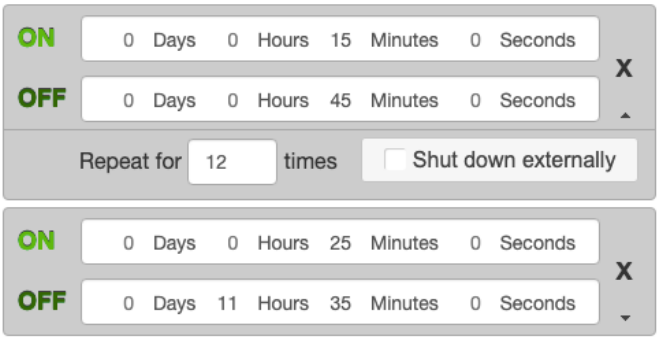
The ARD hardware consists of a solar-powered, battery sourced, GPRS enabled autonomous and weatherproof recording, and storing device. The ARD component lists are tabulated in

Table 5.3. The Raspberry Pi is a cost-effective and relatively powerful microcomputer with onboard USB ports and capacity to implement multiple IoT devices for communication purposes. The microphone and environmental sensor are considered based on performance, required outputs, and cost. To record the captured sound, a single USB 3.0 Flash drive is used with a capacity of 200 GB, which can store up to 180 days of audio data. The Raspbian OS and HT detection program is loaded on a 16 GB microSD card.

Table 5.3 ARD Component List

Component	**Product Name**
Microcomputer	Raspberry Pi 3 Model B
Microphone	Blue Snowflake Digital Microphone
Temperature, Humidity and Pressure Sensor	BME 280 environmental sensor
USB Stick	200GB GB USB 3.0 flash drive
MicroSD Card	SanDisk 16GB
Power Management Board	Witty Pi 2
Modem	Huawei USB Modem
Solar panel	ACOPOWER 35 W Polycrystalline Photovoltaic Panel
PWM battery charger	ALLPOWERS 20A Solar Charger Controller
Lead-acid battery	Expert Power 12 Volt 20 Ah Rechargeable Battery

The Witty Pi 2 is configured to schedule the power state of the Raspberry Pi as shown in Figure 5.2. The Huawei USB Modem is selected considering the cost, power consumption, and reliability of communicating information that sends a notification via Email or SMS to the research lab at Texas State University.




```
Run the script at 2019-05-18 07:00:00
Raspberry Pi turns OFF at: 2019-05-18 07:25:00
Raspberry Pi turns ON at:  2019-05-18 19:00:00
Raspberry Pi turns OFF at: 2019-05-18 19:15:00
Raspberry Pi turns ON at:  2019-05-18 20:00:00
Raspberry Pi turns OFF at: 2019-05-18 20:15:00
Raspberry Pi turns ON at:  2019-05-18 21:00:00
Raspberry Pi turns OFF at: 2019-05-18 21:15:00
Raspberry Pi turns ON at:  2019-05-18 22:00:00
Raspberry Pi turns OFF at: 2019-05-18 22:15:00
Raspberry Pi turns ON at:  2019-05-18 23:00:00

Raspberry Pi turns OFF at: 2019-05-18 23:15:00
Raspberry Pi turns ON at:  2019-05-19 00:00:00
Raspberry Pi turns OFF at: 2019-05-19 00:15:00
Raspberry Pi turns ON at:  2019-05-19 01:00:00
Raspberry Pi turns OFF at: 2019-05-19 01:15:00
Raspberry Pi turns ON at:  2019-05-19 02:00:00
Raspberry Pi turns OFF at: 2019-05-19 02:15:00
Raspberry Pi turns ON at:  2019-05-19 03:00:00
Raspberry Pi turns OFF at: 2019-05-19 03:15:00
Raspberry Pi turns ON at:  2019-05-19 04:00:00

Raspberry Pi turns OFF at: 2019-05-19 04:15:00
Raspberry Pi turns ON at:  2019-05-19 05:00:00
Raspberry Pi turns OFF at: 2019-05-19 05:15:00
Raspberry Pi turns ON at:  2019-05-19 06:00:00
Raspberry Pi turns OFF at: 2019-05-19 06:15:00
Raspberry Pi turns ON at:  2019-05-19 07:00:00
Raspberry Pi turns OFF at: 2019-05-19 07:25:00
Raspberry Pi turns ON at:  2019-05-19 19:00:00
```

Figure 5.2 Witty Pi Scheduling script (Top), Raspberry Pi turn on/off check (bottom)

After Raspberry Pi completes recording and HT detection task, it turns off automatically. The Witty Pi 2 scheduler triggers the Raspberry Pi to complete the whole operation in 205 minutes as listed in Table 5.4. The polycrystalline 35W solar panel, number of batteries, and solar panel rating shown in Table 5.5 are sufficient for the per-hour requirement of the load generated by the ARD.

Table 5.4 Raspberry Pi maximum possible operation time

Operation Time	**Calculation**	**Result**
7 PM to 6AM	15 mins * 12 hours	180 mins
7 AM	25 mins	25 mins

Total Time = 205 minutes

Table 5.5 Solar Panel Specification

Description	**Rating**
Peak Power, Pmax	35W
Peak Circuit Voltage, Voc	22.0V
Max Power Voltage, Vmp	17.5V
Short Circuit Current, Isc	2.0A
Weight	4.1kg
Dimension	730*360*25mm

The Raspberry Pi can draw a maximum of 10W of power from the 5V source. The number of the battery requirements is calculated in Table 5.6 while considering this

maximum power consumption. The result shows that a single battery can continue operation without solar power for two days during cloudy days.

Table 5.6 Worksheet for battery sizing

Description	**Calculation**	**Result**
Raspberry Pi maximum power consumption	5V * 2 A	10 W
Required Watt-Hour per day (Table 5.4)	10 W *205 mins	34.2 W-hr
Days of storage required	-	2 days
Total Watt-Hour (W-hr)	34.2 W-hr * 2 days	68.4 W-hr
Depth of discharge for longevity of battery	-	50 %
Required total W-hr	68.4/0.5	136.8 W-hr
De-rate battery for low temperatures (-1 ºC) by multiplying by	-	1.6
Total Watt-hour required	136.8 * 1.6	218.88 W-hr
W-hr capacity of Each selected battery	12V * 20Ah	240 W-hr
No. of batteries required	218.88/240	1

In the San Marcos area, the average number of sun-hours per day during the least sunny month of the year is four hours, and the solar panel angle is roughly at 30 degrees. Table 5.7 demonstrates that the Raspberry Pi requires one solar panel of minimum 8.55W to charge by full in a day. However, the solar panel used in the ARD is a 35W, which is

higher in capacity than the required operational power and provide reliability in potential source.

Table 5.7 Worksheet for sizing solar panel

Description	**Calculation**	**Result**
Raspberry Pi maximum power consumption	5V * 2 A	10 W
Raspberry Pi Operating hours (dusk to dawn, 10 mins every 13 hours)	10 * 205 mins = 2050 minutes	3.42 hours
Required Watt-Hour per day	10 W * 3.42 hrs	34.2 W-hr
The average number of sun-hours per day during the least sunny month of the year	-	4 hours
The number of watt hours need to generate per hour of full sun	34.2 W-hr / 4 hrs	8.55 W
The actual power produced by selected 35W PV panel per hour (rated Amperage x battery voltage during charging)	2 A * 13 V	26 W
Number of panels required for the system	8.55 W / 26 W	1 panel

The ARD Raspberry Pi has four USB ports that are interfaced with a microphone, USB Modem, and a Pen Drive. Figure 4.1 shows the block diagram representation of the ARD components. The environmental sensor is joined with the Witty Pi 2 module which is placed on top of the Raspberry Pi. The solar panel and battery terminal are connected in the solar charger. The charger USB port supplies power to the Raspberry Pi as shown in Figure 5.3.

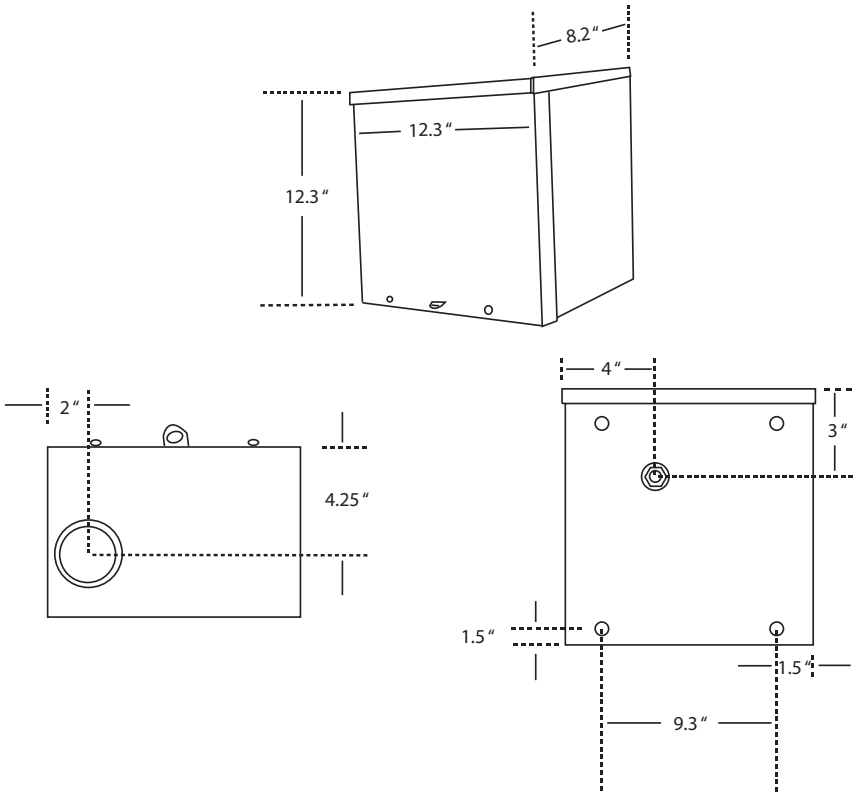
Figure 5.3 ARD Circuit Arrangements

ARD Hardware Design

The hardware items used in the ARD is listed in Table 5.3. The significant parts involve an Unistrut channel, enclosure box, and a galvanized pipe. The Unistrut is the building block for the framework of the ARD in which the pipe holds the solar panel with the structure, and enclosure box is the housing for the electrical circuitry.

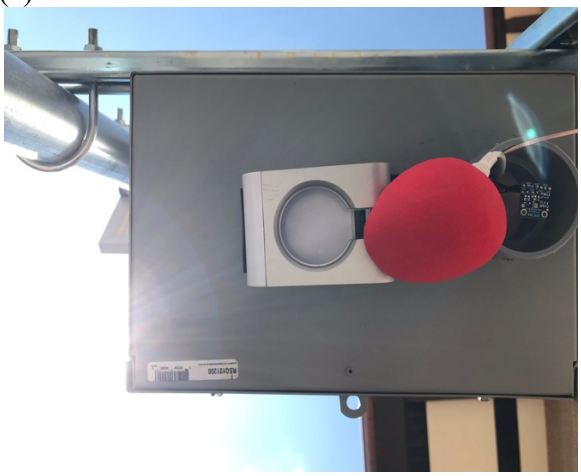
Enclosure Box

The enclosure box is waterproof and contains all electrical components, Raspberry Pi, microphone, modem, and all peripherals that are connected with the box by Velcro tape. Service entrance connector for the solar panel connections ensures water impermeability in the enclosure box. The 2-inch PVC connector at the bottom of the box is erected to place the microphone, antenna, and environmental sensor outside to capture data. The enclosure box dimension drawing and the original view is shown in Figure 5.4.



(a)

(b)



(c)

Figure 5.4 (a) Enclose Box Design (b) front view (c) bottom view

Base Framework

The Unistrut is cut into pieces and coupled together to build the basement framework for the ARD. The framework holds the solar panel and enclosure box and carries the ARD's weight ensuring the stability of the device. The structure has base support to distribute the weight of the ARD load. The spring nuts, bolts, and washers are screwed at different joints to embrace it tight and steady.

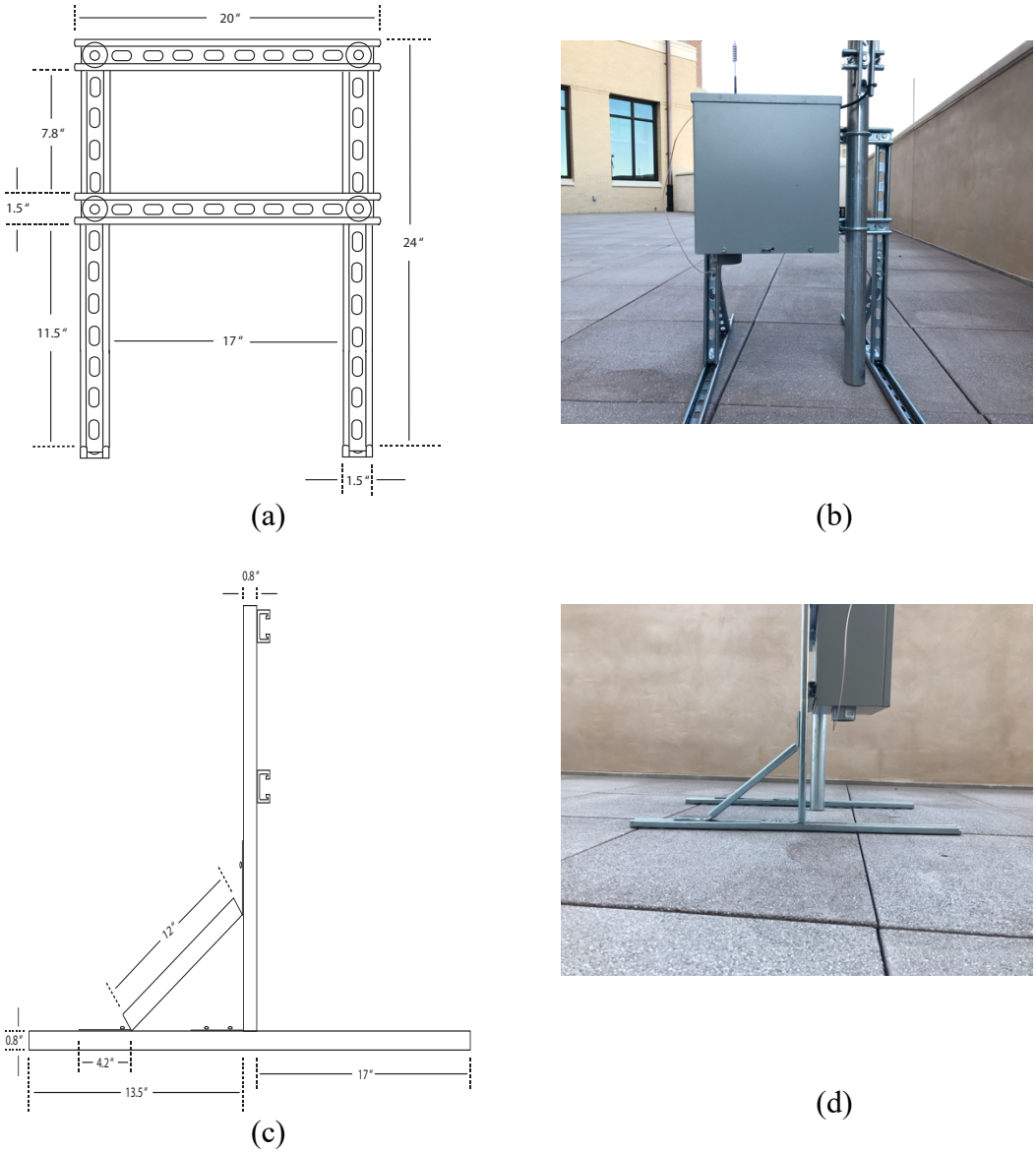
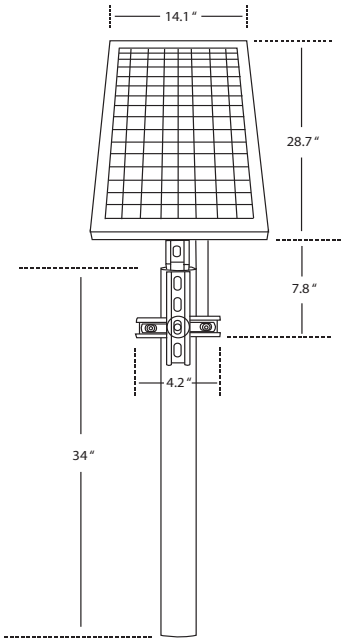


Figure 5.5 ARD Base Framework (a) front view design (b) front view original sight (c) side view design (d) side view original sight

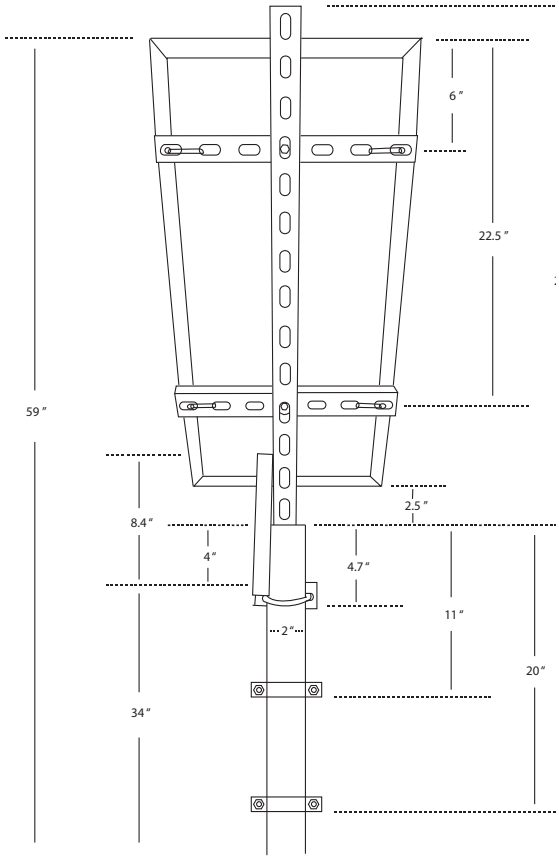
Solar Panel Assembly

The solar panel is affixed with the Unistrut pieces, and at the bottom of the center strut is bended at an angle of 30 degrees. The strut is attached with pipe by the strut pieces, clamps, through bolts, spring nuts, self-driving screws, and flat washers. The two clamps hold the solar panel pipe with the base framework. An Unistrut supports the base solar center Unistrut and is screwed to make an angle adjustment. Solar panel assembly design front and side view are shown in Figure 5.6.



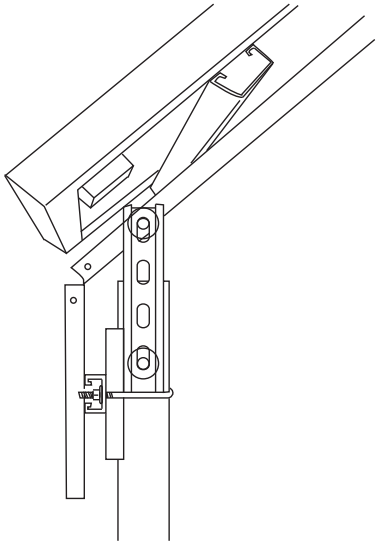
(a)

(b)



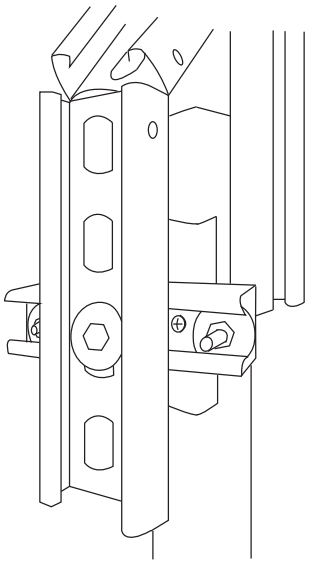
(c)

(d)



(e)

(f)



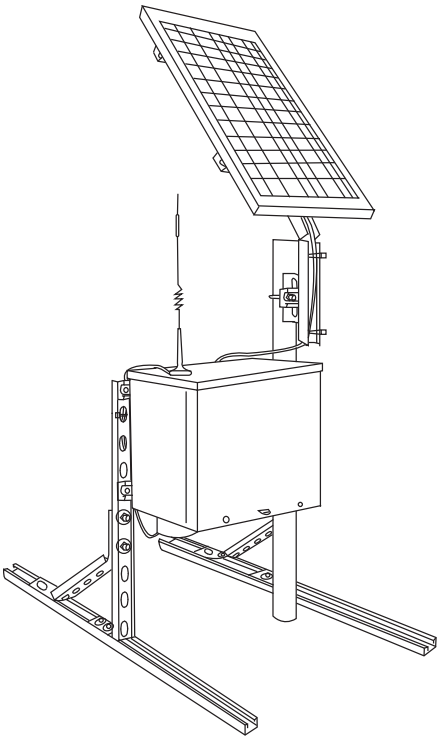
(g)

(h)

Figure 5.6 Solar Panel Connection (a) front design (b) original front (c) back design (d) original back (e) joint from side design (f) joint from side – original (g) joint side design (h) original joint side view

ARD Components Assembly

Once the enclosure box, basement framework, and solar panel assembly are ready, the PV panel cable is drawn to enclosure box through service entrance connector inside the Unistrut and tied with cable tie as seen in black in color in Figure 5.6 (d). The complete ARD prototype design and original view is displayed in Figure 5.7.



(a)

(b)

Figure 5.7 (a) Designed ARD (b) original ARD

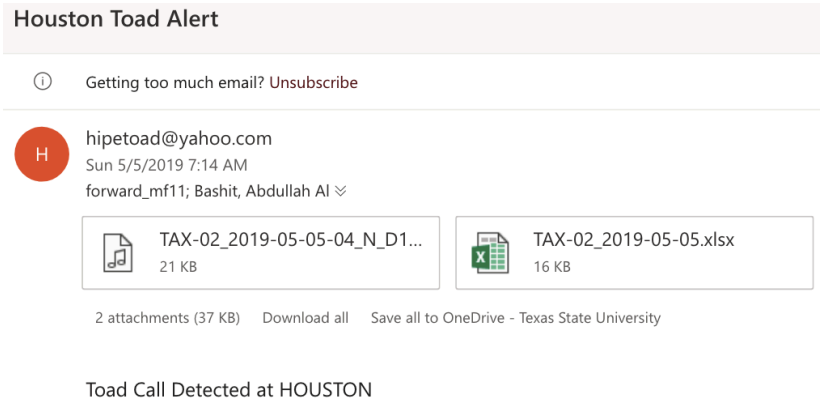
ARD Deployment Results

The ARD was deployed in the spring 2019 and started operation at the Griffith League Scout Ranch in Bastrop, Texas as displayed in Figure 5.8. The Department of Biology used to record sound on that location using the Wildlife Acoustics datalogger software, and the prevalence of the HT calls was confirmed before deployment. Up to May 20, the recorded sound file is observed, and the first HT was notified on April 7 via Email.

Figure 5.8 Bastrop, Texas where ARD Deployed

The deployed ARD boots up every day at 7 PM, records 10 minutes audio file each hour, and processes the recorded file for the HT signature detection. At 7 AM of the next

day, a summary sheet is generated, and if HT call is detected, the ARD begins the Email notification attached with the highest HT duration call audio file with summary sheet. However, if the ARD fails to send an Email, then it tries to send it over an SMS message with the time stamps of the HT call. The notification Email contains the whole overnight hourly HT detection status, temperature, humidity, pressure value, trailing (Spectral width in Excel Sheet by (T)RUE or (F)ALSE status), and the summary sheet as shown in Figure 5.9 (b).

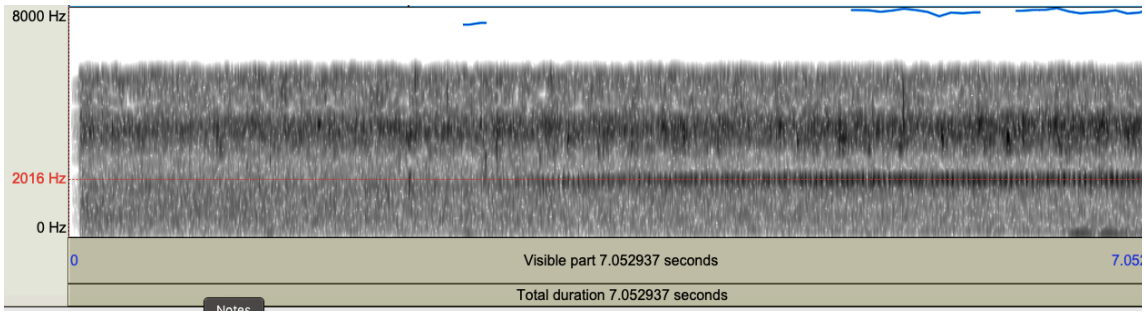


(a)

	A	B	C	D	E	F	G	H	I
1	File	Duration	Toad(sec)	trailing	_Spectral_width	Power	Temp_C	Humid_%	Pressure_hPa
2	TAX-02_2019-05-10-01_N.wav	28.25	(460.38, 488.5)	[1187.5 1375. 1437.5 175	TRUE	[1812.5 1750. 11	16.43	100	995.75
3	TAX-02_2019-05-10-02_N.wav	26.25	(219.88, 246.0)	[1125. 1125. 1250. 1187	TRUE	[1812.5 1750. 17	15.97	100	995.59
4	TAX-02_2019-05-10-00_N.wav	26	(76.0, 101.88)	[1812.5 1812.5 1812.5 17	TRUE	[1812.5 1812.5 1	16.9	100	996.24
5	TAX-02_2019-05-10-01_N.wav	22.62	(396.88, 419.38)	[1187.5 1187.5 1187.5 11	TRUE	[1812.5 1750. 17	16.43	100	995.75
6	TAX-02_2019-05-10-00_N.wav	21.25	(522.62, 543.75)	[2062.5 1812.5 1812.5 17	TRUE	[1812.5 1812.5 1	16.9	100	996.24
7		20.88	(414.0, 434.75)	[1812.5 1312.5 1875. 181	TRUE	[1875. 1875. 187	16.9	100	996.24
8		19.75	(471.75, 491.38)	[1812.5 1750. 1750. 1812	TRUE	[1875. 1812.5 18	16.9	100	996.24
9	TAX-02_2019-05-09-22_N.wav	9.75	(22.12, 31.75)	[1500. 1875. 1562.5 1562	TRUE	[2000. 1750. 168	19.72	94.73	996.3
10	TAX-02_2019-05-10-00_N.wav	7.88	(192.88, 200.62)	[1812.5 1812.5 1812.5 18	TRUE	[1812.5 1812.5 1	16.9	100	996.24
11		6.12	(593.88, 599.88)	[1875. 1812.5 1812.5 168	TRUE	[1812.5 1812.5 1	16.9	100	996.24
12	TAX-02_2019-05-09-22_N.wav	5	(46.62, 51.5)	[2375. 1812.5 2375. 1812	TRUE	[2062.5 2000. 20	19.72	94.73	996.3

TAX-02_2019-05-10-05_N.wav | TAX-02_2019-05-10-06_N.wav | TAX-02_2019-05-10-07_N.wav | summary

(b)



(c)

Figure 5.9 (a) May 05 Email notification with attachment (b) Excel file (c) HT audio 7s

Table 5.8 HT detection Email Notification date, duration and True value

Sl. No.	**Email Notification Date, hour (24 hours)**	**HT call duration, sec**	**Detected HT**	**True HT**
1	April 07, 02	42	T	T
2	April 07, 20	30	T	T
3	April 08, 22	80	T	T
4	April 10, 03	91	T	T
5	April 11, 01	23	T	T
6	April 12, 21	05	T	T
7	April 18, 22	12	T	F
8	April 19, 21	20	T	F
9	April 21, 00	14	T	F
10	April 22, 03	17	T	T

Sl. No.	Email Notification Date, hour (24 hours)	HT call duration, sec	Detected HT	True HT
11	April 23, 02	13	T	T
12	April 24, 01	26	T	T
13	April 26, 01	06	T	F
14	April 27, 07	09	T	F
15	April 28, 21	18	T	T
16	April 30, 00	07	T	T
17	May 01, 22	14	T	T
18	May 05, 04	17	T	F
19	May 06, 02	14	T	T
20	May 07, 01	16	T	T
21	May 07, 23	08	T	F
22	May 09, 02	14	T	F
23	May 10, 01	28	T	T
24	May 12, 23	13	T	F
25	May 14, 01	16	T	T
26	May 15, 01	19	T	T

Sl. No.	Email Notification Date, hour (24 hours)	HT call duration, sec	Detected HT	True HT
27	May 16, 01	07	T	T
28	May 17, 01	08	T	F
29	May 18, 00	08	T	T
30	May 18, 21	12	T	T

From spring to early summer of 2019, the ARD Email notification date, detected HT call duration, true value is presented in Table 5.8. Highlighted red cells are the false detections. Out of the 30 notifications, 20 true-positives and 10 false-positives trailing HT Email were generated marking an efficiency of the system at 20/30 = 66.67 %.

6. CONCLUSIONS

The goal of this thesis is to design an embedded system to record, process, detect, and notify Houston Toad (HT) mating calls. In order to accomplish that, a standalone solar based device called Automatic Recognizing Device (ARD) has been built. It serves the purpose of signal processing for the HT detection on board, and cellular communication to notify their presence.

Summary

The ARD, which can remain in the field without maintenance for a long-extended amount of time, is able to record sounds at prescribed intervals and durations and operate uninterrupted between service periods. The ADR can handle captured data with enough storage and transmit notifications of HT calls. The temperature, barometric pressure, and humidity are also integrated as beneficial readings to record where ARD is located. With these factors in mind, an ARD prototype has been developed which meets the sponsor's expectations at this stage.

The ARD utilizes Witty Pi, Raspberry Pi, Blue Snowflake Digital Microphone, BME 280 environmental sensor chip, cellular module, and USB flash drive. Witty Pi module triggers Raspberry Pi at prescribed intervals, and Raspberry Pi plays key role toward recording, storing, processing HT detection algorithms and sending notification with the help of other peripherals. The rechargeable battery, PWM controller, and solar panel supplies electric power for the continuous operation of Raspberry Pi in the remote

areas. The hardware stuffs include waterproof housing, solar panel structure and base framework that solidifies and ensures robust architecture of the ARD design.

Several voice detection techniques have been implemented to accomplish an efficient and effective way of this identification task. The signal processing algorithm steps include filtering, preprocessing, framing, applying thresholds, feature extraction, predictor classifier model, limiting HT call duration, and trailing thresholds to identify HT call in an audio file. First the recorded audio signal is filtered to narrow down the frequency spectrum. Then, that signal is framed and windowed to break down and uniform the signal which is later applied to thresholding to further shortening the number of potential frames. These identified frames are feature extracted with Mel Filterbank and MFCC algorithms. In the final stage, the matrix is fed into the predictor model for the actual judgement of the potential HT calls. The HT call duration and trailing stages are done to find minimum HT call length and a graduation trailing at the beginning of the HT call. These two steps are imposed on the request of the sponsors to get Email or SMS notifications only when the HT trailing calls are present.

The signal processing algorithm was developed and executed in the Python environment. When the Raspberry Pi boots up, the signal processing algorithm program will record the audio file for a 10-minute duration and analyze for HT call signature. For the HT detection, a prototype of a trained neural network predictor model was implemented to identify HT onboard is deployed, and If the recorded file contains toad call signature, the ARD will send a notification as Email or MMS of the spotted audio file along with environmental data and recording summary to the Department of Biology. The ARD is capable of recording and storing sound and environmental data for months at prescribed

intervals in remote areas. Finally, the design software is working in the solar panel-based hardware at the Houston Toad breeding site. It is tested that the developed system works around 66.67% accurately in the detection of an HT call. The current plan is to reproduce this prototype in the field during the Houston Toad breeding season in 2020.

Future works

For the speech signal processing, the Recurrent Neural Network (RNN) is another classifier that is typically used for periodic audio signals which are yet to be tested for performance and accuracy analysis for the classifier deployment model into the Raspberry Pi. Also, the adaptive signal processing techniques to reduce the number of frames for the feature extractor, as well as the classifier, would significantly reduce the computation as well as the accuracy of the detection system. The testing of thresholds analyzed for the detection system has been carried out over a file that needs to be extended for a large number of files in a high-preforming computational server. Furthermore, different available feature extraction algorithms namely LPCC, PLPC, Cepstrum Analysis, LFCC with RNN or applied MLP might generate a better model that might outperform the existing developed one that can be implemented and tested for better performance and accuracy. Experimenting with different techniques, the output of it from an audio input would record the time stamps of the presence of the HT calls. Extension development for the ARD would be for the Raspberry Pi to continue to record environmental sounds and process a decision of HT calls along with other species such as crickets, birds and other toads.

REFERENCES

[1] C. Y. Yeo, S. A. R. Al-Haddad, and C. K. Ng, "Animal voice recognition for identification (ID) detection system," in *2011 IEEE 7th International Colloquium on Signal Processing and its Applications*, 2011, pp. 198–201.

[2] E. Ramdinmawii and V. K. Mittal, "Gender identification from speech signal by examining the speech production characteristics," in *2016 International Conference on Signal Processing and Communication (ICSC)*, 2016, pp. 244–249.

[3] M. Gupta, S. S. Bharti, and S. Agarwal, "Support vector machine based gender identification using voiced speech frames," in *2016 Fourth International Conference on Parallel, Distributed and Grid Computing (PDGC)*, 2016, pp. 737–741.

[4] A. A. M. Abushariah, T. S. Gunawan, J. Chebil, and M. A. M. Abushariah, "Voice based automatic person identification system using Vector Quantization," in *2012 International Conference on Computer and Communication Engineering (ICCCE)*, 2012, pp. 549–554.

[5] W. Meiniar, F. A. Afrida, A. Irmasari, A. Mukti, and D. Astharini, "Human voice filtering with band-stop filter design in MATLAB," in *2017 International Conference on Broadband Communication, Wireless Sensors and Powering (BCWSP)*, 2017, pp. 1–4.

[6] R. Narasimhan, X. Z. Fern, and R. Raich, "Simultaneous segmentation and classification of bird song using CNN," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2017, pp. 146–150.

[7] N. Garcia, E. Macias-Toro, J. F. Vargas-Bonilla, J. M. Daza, and J. D. López, "Segmentation of bio-signals in field recordings using fundamental frequency detection," in *3rd IEEE International Work-Conference on Bioinspired Intelligence*, 2014, pp. 86–92.

[8] S. Khanum and A. Firos, "Text independent gender identification in noisy environmental conditions," in *2017 International Conference on Computing, Communication and Automation (ICCCA)*, 2017, pp. 63–66.

[9] Y. Jung, Y. Kim, H. Lim, and H. Kim, "Linear-scale filterbank for deep neural network-based voice activity detection," in *2017 20th Conference of the Oriental Chapter of the International Coordinating Committee on Speech Databases and Speech I/O Systems and Assessment (O-COCOSDA)*, 2017, pp. 1–5.

[10] H. Bae, H. Lee, and S. Lee, "Voice recognition based on adaptive MFCC and deep learning," in *2016 IEEE 11th Conference on Industrial Electronics and Applications (ICIEA)*, 2016, pp. 1542–1546.

[11] S. Ozaydin, “Design of a text independent speaker recognition system,” in *2017 International Conference on Electrical and Computing Technologies and Applications (ICECTA)*, 2017, pp. 1–5.

[12] A. Gupta, R. Jain, R. Joshi, and R. Saxena, “Real time remote solar monitoring system,” in *2017 3rd International Conference on Advances in Computing,Communication Automation (ICACCA) (Fall)*, 2017, pp. 1–5.

[13] I. Tripathi, “Wireless environmental parameters monitoring and SMS alert system,” in *2016 International Conference on Information Technology (InCITe) - The Next Generation IT Summit on the Theme - Internet of Things: Connect your Worlds*, 2016, pp. 166–171.

[14] S. R. Parekar and M. M. Dongre, “An intelligent system for monitoring and controlling of street light using GSM technology,” in *2015 International Conference on Information Processing (ICIP)*, 2015, pp. 604–609.

[15] “We make solar simple.” [Online]. Available: https://www.wholesalesolar.com. [Accessed: 20-May-2019].

[16] “WittyPi2_UserManual.pdf.” .

[17] A. A. Bashit and D. Valles, “A Mel-Filterbank and MFCC-based Neural Network Approach to Train the Houston Toad Call Detection System Design,” in *2018 IEEE 9th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON)*, 2018, pp. 438–443.

[18] “Python & CircuitPython Test | Adafruit BME280 Humidity + Barometric Pressure + Temperature Sensor Breakout | Adafruit Learning System.” [Online]. Available: https://learn.adafruit.com/adafruit-bme280-humidity-barometric-pressure-temperature-sensor-breakout/python-circuitpython-test. [Accessed: 10-Oct-2018].

[19] “scipy.io.wavfile.read — SciPy v0.14.0 Reference Guide.” [Online]. Available: https://docs.scipy.org/doc/scipy-0.14.0/reference/generated/scipy.io.wavfile.read.html. [Accessed: 07-Jul-2019].

[20] “Mono vs Stereo - Difference and Comparison | Diffen.” [Online]. Available: https://www.diffen.com/difference/Mono_vs_Stereo. [Accessed: 07-Jul-2019].

[21] “Elliptic filter design - MATLAB ellip - MathWorks India.” [Online]. Available: https://in.mathworks.com/help/signal/ref/ellip.html. [Accessed: 18-May-2019].

[22] S. Gupta, J. Jaafar, W. F. wan Ahmad, and A. Bansal, “Feature Extraction Using Mfcc,” *Signal & Image Processing : An International Journal*, vol. 4, no. 4, pp. 101–108, Aug. 2013.

[23] “Pal Singh - 2014 - An Approach to Extract Feature using MFCC.pdf.” .

[24] P. Welch, “The use of fast Fourier transform for the estimation of power spectra: A method based on time averaging over short, modified periodograms,” *IEEE Transactions on Audio and Electroacoustics*, vol. 15, no. 2, pp. 70–73, Jun. 1967.

[25] “LibROSA — librosa 0.6.2 documentation.” [Online]. Available: https://librosa.github.io/librosa/index.html. [Accessed: 15-Aug-2018].

[26] “Speech Processing for Machine Learning: Filter banks, Mel-Frequency Cepstral Coefficients (MFCCs) and What’s In-Between.” [Online]. Available: http://haythamfayek.com/2016/04/21/speech-processing-for-machine-learning.html. [Accessed: 05-Jun-2018].

[27] “sklearn.preprocessing.MinMaxScaler — scikit-learn 0.19.2 documentation.” [Online]. Available: http://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.MinMaxScaler.html. [Accessed: 23-Sep-2018].

[28] “Support Vector Machines in Scikit-learn,” *DataCamp Community*, 12-Jul-2018. [Online]. Available: https://www.datacamp.com/community/tutorials/svm-classification-scikit-learn-python. [Accessed: 16-Aug-2018].

[29] “1.17. Neural network models (supervised) — scikit-learn 0.19.2 documentation.” [Online]. Available: http://scikit-learn.org/stable/modules/neural_networks_supervised.html. [Accessed: 16-Aug-2018].

[30] “Neural Network from scratch: Part 1 - Theory - ’Cause You’re Stuck.” [Online]. Available: https://causeyourestuck.io/2017/06/12/neural-network-scratch-theory/. [Accessed: 10-May-2019].

[31] “NN SVG.” [Online]. Available: http://alexlenail.me/NN-SVG/index.html. [Accessed: 07-Jul-2019].

[32] “Neural Network from scratch: Part 2 - Practice,” *’Cause You’re Stuck*, 25-Jun-2017..

[33] “sklearn.neural_network.MLPClassifier — scikit-learn 0.21.1 documentation.” [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.neural_network.MLPClassifier.html. [Accessed: 19-May-2019].