A CALL RECOGNITION APPROACH FOR ENDANGERED OR THREATENED CHORUSING AMPHIBIAN SPECIES USING DEEP LEARNING ARCHITECTURES

by

Shafinaz Islam, 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
December 2020

Committee Members:

Damian Valles, Chair

Michael Forstner

Harold Stern

FAIR USE AND AUTHOR'S PERMISSION STATEMENT

Fair Use

Duplication Permission

DEDICATION

To my mother Nilufa Islam

ACKNOWLEDGEMENTS

I would like to express my deep gratitude to Dr. Michael Forstner, the Department of Biology, for giving the opportunity to work under Toadphone development research project. I am also thankful to him for his valuable time, guidance, and feedback.

I feel blessed to have Dr. Damian Valles as my supervisor and my deep gratitude to him for his valuable efforts, time, suggestions, appreciations.

I am grateful to Dr. Harold Stern for accepting to be a member of the thesis committee.

I am thankful to Dr. Vishu Viswanathan for providing guidance throughout the graduate studies and also sharing with me his expert knowledge on audio signal processing.

Last but not the list, I would like to thank my parents; my supportive husband; my relatives; my classmates from Texas State University; who were always there to inspire me and to give me mental support to move on.

TABLE OF CONTENTS

Page

ACKNOWLEDGEMETS v

LIST OF TABLES viii

LIST OF FIGURES ix

ABSTRACT..... xii

CHAPTER

1. INTRODUCTION 1

    1.1. The Present Status of Houston toad and Crawfish frog.....1
    1.2. Problem Statement.....3
    1.3. Contribution.....7
    1.4. Thesis Outline.....7

2. BACKGROUND 9

3. LITERATURE REVIEW 13

    3.1. Audio Data Pre-processing and Feature Extraction.....13
    3.2. Classification Algorithms 15
    3.3. Conclusion 19

4. DEEP LEARNING ALGORITHMS AND ENSEMBLE LEARNING.....20

    4.1. Recurrent Neural Network (RNN).....21
    4.2. Long Short-Term Memory (LSTM) 22
    4.3. Gated recurrent Unit (GRU) 23
    4.4. Convolutional Neural Network (CNN) 24
    4.5. Ensemble Learning 25

5. METHODOLOGY 27

    5.1. Audio Dataset 28
    5.2. Audio Data Pre-processing.....29
    5.3. Dataset Partitioning 32
    5.4. Audio Feature Extraction.....33

5.5. Classification Model Architecture 35
5.6. The List of Experiments 41

6. EXPERIMENTS AND RESULTS 42

6.1. Programming Environment and Computational Resources 42
6.2. Training and Evaluation Results 43

7. CONCLUSION 68

8. FUTURE WORK 71

APPENDIX SECTION 72

REFERENCES 74

LIST OF TABLES

Table **Page**

1. List of Experiments.....41

2. Summary of Results for the “Houston” or “Crawfish” Classification Experiment 61

3. Comparison of Results for “Houston” or “Non-toad” Classification With and Without Ensemble.....65

4. Comparison of Results for “Crawfish” or “Houston” Classification With and Without Ensemble.....65

5. Comparison of Results for “Crawfish” or “Houston” or “Environment” Classification With and Without Ensemble.....66

LIST OF FIGURES

Figure **Page**

1. (a) Houston toad and (b) Crawfish frog 2

2. Flowchart Demonstrating the Working Principle of the Toadphone 1 4

3. Wildlife Acoustic Song Meter 9

4. A Toadphone 1 Structure 10

5. Simple RNN Architecture 21

6. LSTM Internal Architecture 23

7. GRU Internal Architecture 24

8. An Example of Ensemble Learning 26

9. Architectural Building Blocks of Working Process 27

10. Bandpass Filtered Frequency Spectrum of Houston toad Call 30

11. Bandpass Filtered Frequency Spectrum of Crawfish frog Call 30

12. The LSTM and GRU Model Structure for the "Houston toad" or "Non-toad" Classification Experiment with 39 MFCCs 43

13. (a) Accuracy and (b) Confusion Matrix of the LSTM Model with 39 MFCCs for the "Houston toad" or "Non-toad" Classification Experiment 45

14. (a) Accuracy and (b) Confusion Matrix of the GRU Model with 39 MFCCs for the "Houston toad" or "Non-toad" Classification Experiment 46

15. (a) Mel-Spectrogram of Audio File Having Toad Call, (b) Mel-Spectrogram of Audio File Having Non-toad (Only Environment Sound) 46

16. (a) Cropped Mel-Spectrogram of Audio File Having Toad Call, (b) Cropped Mel-Spectrogram of Audio File Having Non-toad (Only Environment Sound) 47

17. CNN Model Structure for the “Houston toad” or “Non-toad” Classification Experiment with Mel-Spectrogram Images 48

18. (a) Accuracy and (b) Confusion Matrix of CNN Model with Mel-Spectrogram for the “Houston toad” or “Non-toad” Classification Experiment 49

19. LSTM and GRU Model Structure for the “Houston” or “Crawfish” Classification Experiment with 39 MFCCs 50

20. (a) Accuracy and (b) Loss Plots of the LSTM for 39 MFCCs with the SGD Optimizer, 0.001 for the Learning Rate, and 200 Epochs 51

21. (a) Accuracy and (b) Loss Plots of the GRU for 39 MFCCs with the SGD Optimizer, 0.001 for the Learning Rate, 200 Epochs 52

22. (a) Accuracy and (b) Loss Plots of the LSTM and 39 MFCCs with the ADAM Optimizer, 0.001 for the Learning Rate, and 200 Epochs 53

23. Confusion Matrix of the LSTM with 39 MFCCs for the ADAM Optimizer, 0.001 for the Learning Rate, and 200 Epochs 53

24. (a) Accuracy and (b) Loss Plots of the GRU for 39 MFCCs with the ADAM Optimizer, 0.001 for the Learning Rate, And 200 Epochs 54

25. Confusion Matrix of the GRU with 39 MFCCs for the ADAM Optimizer, 0.001 for the Learning Rate, and 200 Epochs 54

26. (a) Accuracy and (b) Loss Plots of the LSTM and 39 MFCCs with the ADAM Optimizer, 0.0001 Learning Rate, 300 Epochs 56

27. Confusion Matrix of the LSTM with 39 MFCCs for the ADAM Optimizer, 0.0001 Learning Rate, 300 Epochs 57

28. (a) Accuracy and (b) Loss Plots of GRU with 39 MFCCs for the ADAM Optimizer, 0.0001 Learning Rate, 300 Epochs 57

29. Confusion Matrix of the GRU with 39 MFCCs for the ADAM Optimizer, 0.0001 Learning Rate, 300 Epochs 58

30. LSTM and GRU Model Structure for the “Houston” or “Crawfish” Classification Experiment with 16 SSCs 59

31. (a) Accuracy And (a) Loss Plots of The LSTM with 16 SSCs for ADAM Optimizer, 0.0001 for the Learning Rate, and 300 Epochs 60

32. (a) Accuracy and (b) Loss Plots of the GRU with 16 SSCs for the ADAM Optimizer, 0.0001 for the Learning Rate, and 300 Epochs 60

33. Accuracy/Loss Plots of LSTM with 39 MFCCs for the “Houston” or “Crawfish” or “Environment” Classification Experiment 63

34. Confusion Matrix of LSTM with 39 MFCCs for “Houston” or “Crawfish” or “Environment” Classification Experiment 63

ABSTRACT

Audio signal analysis has become prominent in biological domains for detecting endangered or threatened species like Houston toad and Crawfish frog. Researchers at Texas State University and Texas A&M University are working on a project to steward these species and understanding the causes of their decline. The researchers are currently using an Automated Recording Device (ARD), the Toadphone 1, which is an embedded solution. The hardware platform can perform detection tasks without human interruption and can provide near real-time notification. However, this device's predictive model for the software solution has limited success to serve the primary purpose for which it was developed, which is to provide proper identification of Houston toad calls. Also, the current predictive model for Toadphone 1 was only designed for the Houston toad calls. There is another near-threatened chorusing amphibian, the Crawfish frog, which has become a concern of the researchers working to protect this species.

This thesis research experimented with a modified predictive model for the existing Toadphone 1 software solution, predicting a Houston toad call with decreased false-positive rates. The model can also perform the call recognition task for Crawfish frog calls. This work used the audio data for Houston toad and Crawfish frog collected by the Department of Biology to train the predictive model. Before training, the audio data spectrum was studied to find the frequency range of Houston toad and Crawfish frog call. Next, the audio data have been iteratively preprocessed using digital filters and then

applying framing, the Hamming window function to each frame. Mel-frequency Cepstral Coefficients (MFCCs) with their first and second derivatives or Spectral Sub-band Centroids (SSCs) or Mel-spectrograms audio features have been extracted for each frame. These features were used to train the predictive or classification model for Houston toad or Crawfish frog call prediction. Advanced Recurrent Neural Network (RNN) algorithms such as Long Short-Term Memory unit (LSTM) or Gated Recurrent Unit (GRU) and Convolutional Neural Network (CNN) were utilized, which are sub-fields of deep learning network architectures. Several model architectures were experimented with using different combinations of classifiers and audio features with tuned hyperparameters to build the best predictive model. The voting mechanism of ensemble learning was developed to make the final prediction from the three-best models. Lastly, the predictive model was evaluated on a near real-time prediction system.

1. INTRODUCTION

It is only possible to monitor and then conserve what can be documented, especially for rare taxa that are difficult to locate or detect. For vocalizing species, audio recorder technology enables detection and increases the efficiency of documenting the species' presence at a given pond or habitat patch. In this way, audio signal analysis has become prominent in biological domains for applications in animal call detection. This technology provides crucial data needed for stewardship actions in vocalizing endangered or threatened species by detecting or locating their calls. The Houston toad is already listed as an endangered species, and the Crawfish frog is a near-threatened amphibian species. Biology Department researchers at Texas State University and Texas A&M University are working on a project to steward these species. This thesis research is a part of this project work and used audio data analysis technology for vocalizing species Houston toad and Crawfish frog's call detection.

The following sections discuss the present status of Houston toad and Crawfish frog, provide a problem statement, describe contributions of this thesis, and give an outline of this thesis research.

1.1 The Present Status of Houston toad and Crawfish frog

John Wottring, an amateur herpetologist, first identified the Houston toad (*Bufo [Anaxyrus] houstonensis*) in the 1950s in south Houston, Texas. The Houston toad requires very specific environmental conditions to live in an area. Fourteen eastern-central Texas counties have supported Houston toads including Austin, Bastrop, Burleson, Colorado, Freestone, Fort Bend, Harris, Lavaca, Lee, Leon, Liberty, Milam,

and Robertson counties. Unfortunately, only 20 years after its initial discovery, the population collapsed due to habitat loss and alteration. In 1973, the Houston toad became the first amphibian to be added to the endangered species list. It has undergone several significant reductions in its overall population numbers since its description 70 years ago. Besides the Houston toad, there is another chorusing amphibian species known as the Crawfish frog (*Rana areolate*), which is identified as a *Near Threatened* species by the World Conservation Union [1]. The range of the Crawfish frog is from Texas to Mississippi in the south and from Indiana west to Nebraska in the north, though the species is believed to be extirpated from much of its northern range [1]. The Crawfish frog species is near threatened because of similar issues to the Houston toad, including habitat loss due to drainage of breeding habitat, urban and agricultural development, and fish-stocking. Figure 1(a) from [2] and Figure 1(b) from [3] show the images of Houston toad and Crawfish frog, respectively.

(a)

(b)

Figure 1: (a) Houston toad (*Bufo* [*Anaxyrus*] *houstonensis*) [2] and
(b) Crawfish frog (*Rana areolate*) [3]

1.2 Problem Statement

Biodiversity refers to the variety of living species on earth, including plants, animals, bacteria, and fungi. Losses of biodiversity are both indicators of underlying ecosystem issues and often coincident to the loss of other species within its ecosystem. Healthy ecosystems are required for human health and well-being as it is these systems which provide us with our environment, most critically clean air, water, or land. It is necessary to take stewardship actions to protect and seek to recover those species that are threatened with extinction. However, those species are by default rare, making it increasingly difficult to locate or detect them as they decline. If a species can chorus or call, it is possible to locate them by detecting their call or chorusing through audio solutions and mechanisms to help the conservation efforts. It is possible to locate the Houston toad and Crawfish frog identifying the localization of their mating calls, enabling field researchers to protect their eggs from being eaten by predators. These species can be found mostly in remote areas, such as forest or near a pond. It is an exhaustive work for human beings to perform the detection task by physically staying on those location sites prevalent of the toad and frog. Detecting or identifying species using automatic audio recording and analysis technology has become very popular for rare species monitoring of different animal species. The advancement of Machine Learning (ML) and deep learning techniques enables audio signals to be used in the broader biological domain for detection. Researchers from different organizations, such as the Houston Zoo, Texas Parks and Wildlife Department (TPWD), Texas State University, and the United States Fish and Wildlife Service (USFWS) are currently working to protect the Houston toad from extinction. The goal is to steward wild populations and

also to head start early life-stages, eggs, tadpoles, etc., in the field or by raising them to a larger size in captivity. Currently, researchers are using an Automated Recording Device (ARD), Toadphone 1 [4] in a number of remote sites for the detection of Houston toad calls. The Toadphone 1 is an embedded solution and capable to recording environment audio, performing toad call detection operation from recorded audio automatically without any human interruption, and transmitting near real-time notification of toad call detection. The workflow diagram of Toadphone 1 is shown in Figure 2.

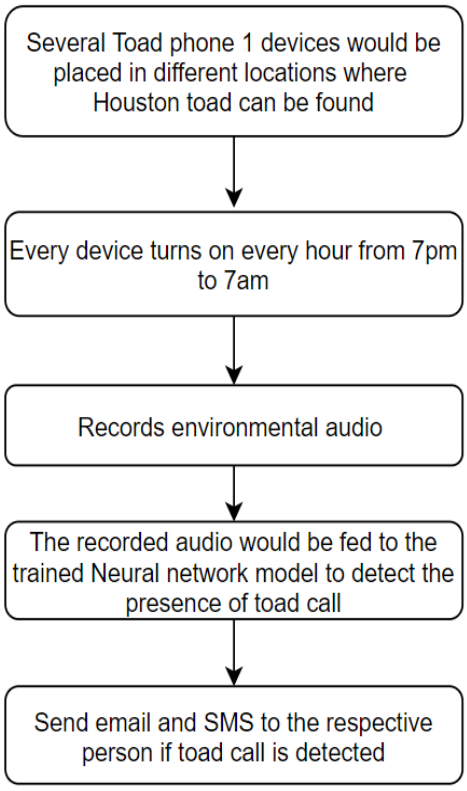

Figure 2: Flowchart Demonstrating the Working Principle of the Toadphone 1

The detection of the toad call from recorded environmental audio is done by the computer programming. Currently this is the software solution of Toadphone 1. The software solution of the Toadphone 1 is mainly the trained model which includes audio data analysis. This trained model would take the recorded audio as a test data and would predict if it is a toad call or not. Two of the major parts in audio data analysis are audio feature extraction and classification algorithms. The Toadphone 1 implemented the Mel-Filterbank and thirteen MFCC for feature extraction techniques, and the Support Vector Machine (SVM) and the Multi-layer Perceptron (MLP) as classifiers [4].

This device has limitations to its software solution in detecting Houston toad calls correctly due to false-positive notifications of Houston toad's presence. The device cannot differentiate among the calls of Houston toad and several other species, such as other chorusing amphibian species, mole cricket, or multispecies chorus composites, as these species or groups can have similarities in frequency, pitch, and intensity of their calls with Houston toad call. The accuracy for Houston toad call detection of the current Toadphone 1 is 66.67% [4].

This is a hinderance to the accurate conservational effort of this endangered species and requires further improvement in design and implementation. As the Crawfish frog is also an impending endangered species, researchers have also planned to add its detection to their conservation efforts. The detection process of the Crawfish frog is similar to the Houston toad, and it also can be performed by identifying the localization of their mating calls. Hence, the software solution in the Toadphone 1 device improvements are to reduce the false-positive notifications using proper detection

mechanisms for the Houston toad call and adding the capability to detect calls of an additional chorusing amphibian Crawfish frog.

As a result, this thesis research work implemented progressive Deep Neural Network classifiers such as Long Short-term Memory (LSTM), Gated Recurrent Units (GRU), and Convolutional Neural Network (CNN) including audio feature extraction algorithms, such as Mel-Frequency Cepstral Coefficients (MFCCs) with delta and delta-delta coefficients, Spectral Sub-band Centroids (SSCs), and Mel-spectrograms. Voting mechanism of ensemble learning is applied. This improved software solution approach is capable of detecting Houston toad with reduced false-positives and is also able to detect Crawfish frog calls.

1.3 Contribution

The contributions of this thesis research have been summarized as follows:

- Development of a modified software solution or a trained and tested model for Toadphone 1 using Recurrent Neural Network algorithms to recognize the endangered chorusing amphibian species “Houston toad” call by reducing false-positives classifications and higher prediction accuracy compared to the existing Toadphone 1 software solution.
- Adding another threatened chorusing amphibian species, the “Crawfish frog,” to the call recognition task.
- Application of Ensemble learning

1.4 Thesis Outline

The thesis begins with a background chapter that discusses the initial and current approaches for toad call detection and the motivation towards developing a modified

software solution for Toadphone 1. The following chapter discusses the relevant literature and methods related to the animal audio or any kind of acoustic scene or environment sound classification and recognition. Chapters 4 is a theoretical chapter discussing the Recurrent neural networks internal architectures, the description of different layers for Convolutional Neural networks and an overview of Ensemble learning. Chapter 5 discusses the methodology of building a modified software solution or the trained model for Toadphone 1. Chapter 6 shows and discusses the experiment results in detail to justify the choices and mechanisms adapted to implement the deep learning model for this work. Conclusions are provided in chapter 7. Finally, in chapter 8, few directions for further research have been described.

2. BACKGROUND

As a chorusing amphibian species, Houston toad can be detected using their calls. Several techniques have been performed for its detection using audio signal processing and audio recognition methods. The initial set up approach of the stewardship process involves and depends on human labor, travel, and manual analysis of captured data. Eighty Wildlife Acoustics song meters have been placed at locations where Houston toads can be found throughout the state of Texas. The research team travels to specific sites to collect the data after a few weeks and processes the collected data manually to identify Houston toad call's presence in the audio files. The time gap between the collection of data and identification of the toad calls has made this process a potential opportunity to develop a near real-time detection system solution. Figure 3 shows the setup of a Wildlife Acoustics song meter with its power management system attached to a tree.

Figure 3: Wildlife Acoustic Song Meter

The Toadphone 1 is the current system device being tested by the Biology Department at Texas State University. The Toadphone 1 device is an embedded solution that performs detection operation automatically without any human interruption, with near real-time notification transmission capabilities. The system consists a software solution deployed on a Raspberry Pi-3 board for on-field Houston toad detection, a solar-powered battery for power management, a microphone for audio data recording, an environmental sensor for environmental data collection, a flash drive for collecting audio data and environmental sensor data, and a cellular modem for the internet connection for providing near real-time notifications. All these peripheral components are connected to a Raspberry Pi-3 board using its USB and other serial ports. Figure 4 shows a Toadphone 1 structure placed near a place where toad calls can be recorded.

Figure 4: A Toadphone 1 Structure

The software solution or the trained model of the Toadphone 1 includes audio signal processing algorithms, such as filtering, framing, and windowing audio signals. The Toadphone 1 has advantages over the song meter approach by performing onboard automated detection operations. During the Houston toad's breeding season, about 46 Toadphone 1s have been placed at locations where Houston toad may be found. All the tasks for the Houston toad call detection using audio recording, processing the recorded audio, and recognizing the toad and non-toad call using pattern recognition algorithm are performed on a single board device. An internet module is included with the Toadphone 1 device to transmit near real-time Houston toad call detection notifications and recorded captured audio files. However, the Toadphone 1 has limitations with a large false-positive number of a toad call detections in a file where there is no actual toad call signature present. Hence, the current Toadphone 1 software solution produces an accuracy of only 66.67% of true-positive detections.

This thesis research experimented and developed a model that is able to detect Houston toad along with another near threatened chorusing amphibian specie known as the Crawfish frog. The main objective is significantly improving detection of Houston toad call and adding Crawfish frog call recognition. The goal is thus to build a solution with reduced false-positive rate using ML classifiers and advanced audio features. The development of the ML model will aid the conservation effort in identifying the presence of these species with high accuracy performance rates. According to previous research work on species detection or environmental sound detection, Neural Network architectures have efficient ML techniques for pattern recognition or identification. This thesis work experimented with several neural network architectures such as CNNs,

LSTMs, and GRUs along with feature extraction algorithms such as MFCC, Mel-spectrograms and SSC. Also, ensemble learning has been applied in an attempt to make the predictions more robust. These models were evaluated based on standard ML performance metrics such as accuracy, loss, and confusion matrix.

3. LITERATURE REVIEW

This chapter discusses the previous research works in the field of audio signal processing, recognition, and classification. Previously, most of the research work on voice or audio signal detection was done by utilizing human speech. However, due to the increased attention in ecology centric analytic research, audio signals are also being used in biological domains, such as for animal detection. The main challenge with animal sound is the surrounding environmental background noises. Advancement of ML and neural network techniques have enabled researchers to overcome some of the related problems. The techniques followed by the researchers in the field of animal voice detection, or any kind of audio detection, is useful for the Houston toad and Crawfish frog recovery research work. This chapter is divided in three subsections:

1.5. Audio data pre-processing and feature extraction

1.6. Classification algorithms

1.7. Conclusion

3.1 Audio Data Pre-Processing and Feature Extraction

This subsection discusses previous research works in the field of audio signal, data preprocessing, and feature extraction.

Authors in [5] used digital filters for a pre-processing stage of animal audio signals for segmentation, as strong interference noise from the environment can shadow the vocalizations within the data. After choosing this preprocessing step, the success rate of identifying segmented bio signal improved dramatically, and false positive rate reduced. The success rate for this work in [5] is 98% which is 8% higher compare to the

commercial song scope for segmentation of bio signals. The method of using digital filters is a good choice for this work as the Houston toad call or the Crawfish frog call is surrounded by environment noise. The false positive issue can be reduced for Houston toad and Crawfish frog call detection by applying digital filters to the signals.

Authors in [6] compared MFCC, LFPC, Spectral, and PLPC audio features for singing voice detection or characterizing vocal and non-vocal portions of a song. Authors in this paper used music genres database such as rock, pop, folk, funk, and jazz. Independent datasets of popular music recordings were used for training, validation, and testing. The validation database consists of 63 fragments of 10 seconds and an independent evaluation was conducted on a testing database of 46 manually annotated songs, for a total duration of 3 hours. The results showed that MFCCs and their derivatives are the most appropriate features. The paper [6] gained 84% accuracy with MFCC features and 60% to 70% accuracy with LFPC, Spectral, and PLPC features for classifying the vocal and non-vocal parts of a song.

Authors in [7] used the TIMIT database which is a database with English connected speech prepared by Texas Instruments and MIT. From 6300 utterances in the TIMIT database, 5670 of them are used for the training system, and 630 utterances were used for the test. Authors in [7] used three different features such as 12 Linear Predictive Cepstral Coefficients (LPCC), 26 Spectral Sub-band Centroids (SSC) and 38 Linear Predictive Spectral Sub-bands (LPSS) which is a combination of LPCC and SSC features. The LPCC features gained 96.2%, the SSC features gained 97.1% accuracy for speaker recognition and combining LPCC and SSC features produced the highest accuracy which is 99%.

Authors in [8] used Mel-frequency cepstral coefficients (MFCCs) or spectrograms as audio features for acoustic scene classification from real life audio recordings. The MFCC features were extracted using a Hamming window with window-size 40ms and 50% overlap. 39 MFCCs were extracted including Δ- and Δ2 coefficients. For the experiments in [8], the authors used either MFCCs + Δ + Δ2 features, or raw 1025-bin log magnitude spectrograms. The spectrogram features produced 79.1% accuracy where MFCC features gained 78% accuracy for real world acoustic scene classification.

3.2 Classification Algorithms

A) Recurrent neural network for audio analysis

This subsection discusses previous research works related to the implementation of recurrent neural networks for audio detection or classification.

Authors in [8] proposed an efficient framework for environmental acoustic scene and event classification. The authors of [8] evaluated their framework on environmental sound scenes of the IEEE Detection and Classification of Acoustic Scenes and Events challenge (DCASE2016). The challenge comprised four tasks: acoustic scene classification, sound event detection in synthetic audio, sound event detection in real-life audio, and domestic audio tagging. Authors indicated the great success of deep recurrent networks for sequence modeling like audio data and used gated recurrent neural networks (GRNNs) as a classification algorithm for acoustic scene classification and acoustic event detection. This system in [8] reaches an overall accuracy of 79.1% that has improved and outperformed the baseline Gaussian Markov Model (GMM) system by 8.34%.

In [9], authors proposed a method to perform sound event classification and detection. In this work, they used Gated Recurrent Units (GRUs), which is one type of GRNNS. The work in [9] classified fifteen classes of sound. Authors in this paper claimed that Recurrent neural networks (RNN) showed their effectiveness and flexibility in working with various problems in audio analysis. Among 15 classes, each class achieved classification accuracy of more than 80% with an overall accuracy of 82.09%.

In [10], the authors proposed an approach to detect polyphonic sound events in real-life recordings. Due to the presence of multiple and overlapping sounds, this problem is known as polyphonic detection. Authors used a multilabel bi-directional long short-term memory (BLSTM)-RNNs model with multiple hidden layers to map the acoustic features to class activity indicator vectors. This method was experimented on a large database of real-life recordings, with 61 classes, such as music, car sounds, and speech sounds, from ten different everyday contexts. Their average F1-score was 65.5% on one-second blocks and 64.7% on single frames and improved previous state-of-the-art approaches such as Hidden Markov models (HMMs), and Feedforward neural network (FNN) by 6.8% and 15.1% respectively.

In [11], authors proposed a new method for singing voice detection based on a BLSTM-RNN. This classifier is able to take a past and future temporal context into account to decide on the presence/absence of a singing voice. Authors used the Jamendo Corpus, a publicly available dataset including singing voice activity annotations. It contains 93 copyright-free songs. The corpus is divided into three sets: the training set contains 61 files; the validation and test sets contain 16 songs each. Authors found the best architecture for BLSTM-RNN with three hidden layers whose sizes are 30, 20 and

40. Their method significantly outperformed state-of-the-art methods such as Support Vector Machines (SVMs), Hidden Markov Models (HMMs), Random Forests or Artificial Neural Networks (ANNs) and gained an accuracy of 91.5%.

B) Convolutional neural network (CNN) for audio data analysis

This subsection discusses previous research works related to the implementation of convolutional neural network for audio detection or classification.

Authors in [12] present a novel approach for pattern classification of animal voice that is composed of multiple CNNs and SVM. The dataset contains three Classes such as anurans, birds, and insects. Anuran sounds are recorded with 44.1 kHz sampling rate from their natural habitats, bird sounds were collected from http://www.ebird.org, insect sounds were collected from Korea Wild Animal Sound Dictionary released by National Institute of Biological Resources. Among them, orthopteran including crickets and grasshoppers is selected. The database collection contains a total of 52,765 segments for 102 species. In detail, the database is composed of 4,878 segments for 8 anuran species, 21,749 segments for 43 bird species, and 26,138 segments for 51 insect species. To classify the species into three Classes, three CNNs that are pretrained for classifying each of the three species are applied for feature extraction. CNN took the spectrograms as input and produced mid-level features for an SVM classification model. The proposed model obtained an accuracy of 98.91% for classifying an anuran, a 77.235% for classifying a bird, and an 81.69% for classifying an insect.

In [13], a bird song classification method was introduced using a popular encoder-decoder CNN structure. The input to the network is a spectrogram, and the output is an image of the same size with pixels labeled as either background or one of the bird

species. Authors in [13] have acquired true positive rate (TPR) of 0.988 and false positive rate (FPR) of 0.02. This work suggests developing an RNN model that considers the temporal relation between bird syllables.

The work in [14] evaluates a set of popular ML approaches on audio data from the cat and dog families. Data has been used from three different data sources: the Macaulay Library, Freesound.org, and the Google Audio set. Authors employed two libraries to perform the audio feature extraction task with Librosa and pyAudioAnalysis in Python. The Librosa library produces 40 MFCCs, twelve chroma features for pitch information, 128 Mel-spectrograms, seven spectral contrasts, and six tonnetz quantities for tonal distance characterizations. It also extracted spectrogram images as 2D arrays. Different ML techniques has been evaluated: k-Nearest Neighbors (kNN), Logistic Regression, SVM, DNN, and CNN. The CNN trained with spectrograms extracted from exclusively high-quality audio files achieved over 91% high accuracy on the high-quality test data.

Authors in [15] used deep-learning methods for feature extraction and classification of frog calls. They did not use any predesigned features but allowed a deep-learning algorithm to find the features that are most important for classification. The audio recordings came from a mixture of web-based collections and field recordings made in Florida. Different CNN implementations were tested on a dataset of about 200 calls from fifteen frog species. Average classification accuracy reached is 77% due to the low number of samples in its datasets.

Authors in [16] performed binary classification of Anuran species using CNN. They used MFCCs audio features in the form of image as input. The dataset used in [16]

experiments contains ten anuran species from four different families. The sound recordings were collected in the anuran habitat, under real noise conditions. The audio recordings are stored in wav format with a sample rate of 44.1 kHz and 32 bits per sample. Results are compared with other classifiers such as k-Nearest Neighbors (kNN), Decision Tree (DT), Quadratic Discriminant Analysis (QDA) and Support Vector Machine (SVM) and the proposed approach outperformed others and gained accuracy of more than 90%.

3.3 Conclusion

The goal of this research is the call recognition of two chorusing amphibian species Houston toad and Crawfish frog. The task of chorusing amphibian call recognition is also close to the task of speech recognition, singing voice detection, or music audio classification. Previous works experimented and compared different feature extraction algorithms and found MFCC with delta and delta-delta coefficients, SSCs, Mel-spectrograms to be the most robust features that provide high accuracy performance. According to previous work, utilizing neural network algorithms such as LSTM, GRU, and CNN as classification algorithms showed promising results for animal call detection or any audio detection. Based on these related research efforts, this thesis work experimented and evaluated MFCCs with delta and delta-delta coefficients, Mel-spectrograms as audio features and included the LSTM, GRU, CNN as classifiers for two chorusing amphibian species Houston toad and Crawfish frog call detection

4. DEEP LEARNING ALGORITHMS AND ENSEMBLE LEARNING

Deep Learning is a subfield of machine learning concerned with algorithms inspired by the brain's structure and its function called artificial neural networks. At the same time, much of the literature and concepts on deep learning are concerned with computer vision and natural language processing (NLP) applications, audio data analysis that includes automatic speech/audio recognition, digital signal processing, audio classification, tagging, and generation. Deep learning models can achieve state-of-the-art accuracy, sometimes exceeding human-level performance. This thesis research aims to recognize the call of two chorusing amphibian species as audio data analysis by implementing deep learning architectures to perform the call recognition task.

Among all the deep learning architectures, the Recurrent Neural Network (RNN) has become a popular technique for sequential data analysis like audio data [17]. This thesis research experimented with two modified RNN architectures, Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU) for Houston toad and Crawfish frog call detection.

Audio data can also be visualized as an image, which is a spectrogram. A spectrogram is a detailed view of audio, able to represent time, frequency, and amplitude all on one graph. Analysis of an audio signal based on its spectrogram can be considered as a computer vision task. Convolutional Neural Network (CNN, or ConvNet) is a part of deep learning which is most applied to analyzing visual imagery. This thesis work also experimented with CNN for audio analysis based on spectrograms.

The following sections give an overview of RNN with its modified versions and CNN in detail.

4.1 Recurrent Neural Network (RNN)

An RNN is an extension of a conventional feedforward neural network, which can handle a variable-length sequence input. The RNN handles the variable-length sequence by having a recurrent hidden state whose activation at each time is dependent on that of the previous time. The problem mentioned in this research is the accurate detection of Houston toad and Crawfish frog using their calls, for which the audio signal classification and processing requires sequence modeling that can be performed by RNNs. Figure 5 from [18] shows the basic architecture of RNN.

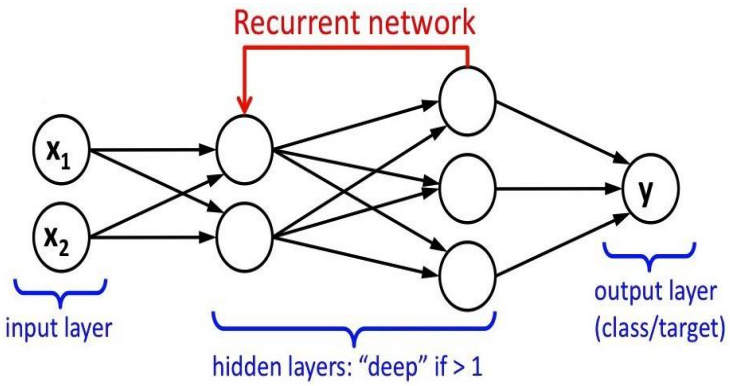

Figure 5. Simple RNN Architecture [18]

Some researchers in the field of audio signal recognition have observed that it is challenging to train RNNs to capture long-term dependencies because the gradients tend to either vanish or explode [8]. To overcome this problem, they have suggested using other types of RNNs, Gated Recurrent Neural Network (GRNNs), whose hidden units are gate based [8]. There are two types of GRNNs: LSTMs and GRUs. According to recent research, both LSTM and GRU have achieved tremendous success for accurate classification or detection from sequential or time-series data like audio data. This

research experimented and evaluated the performances of LSTM and GRU for the proper detection of two chorusing amphibian species call or audio data with a reduced false-positive rate.

4.2 Long Short-Term Memory (LSTM)

LSTM is an artificial recurrent neural network (RNN) architecture used in the field of deep learning. LSTM has feedback connections which makes it different from standard feedforward neural networks. It is capable of processing single data points such as images, and entire sequences of data such as speech, audio, or video. That is why LSTM is applicable to audio recognition, speech recognition or handwriting recognition. A LSTM unit has an input-gate, an output-gate, and a forget-gate. The cell remembers values over time intervals, and the three gates of LSTM are used to regulate the flow of information into and out of the cell's networks. LSTM is capable of classifying, processing, and forecasting based on time-series data. LSTMs were developed to deal with the exploding and vanishing gradient problems encountered when training traditional RNNs. This thesis research work deals with the detection and classification of two chorusing amphibian species, the Houston toad and Crawfish frog using audio data in a sequential data format. As LSTM has higher performance for time-series data classification, this research used LSTM to predict or detect two chorusing amphibian species to reduce the false-positive rate. Figures 6 from [19] shows the internal architecture of the LSTM.

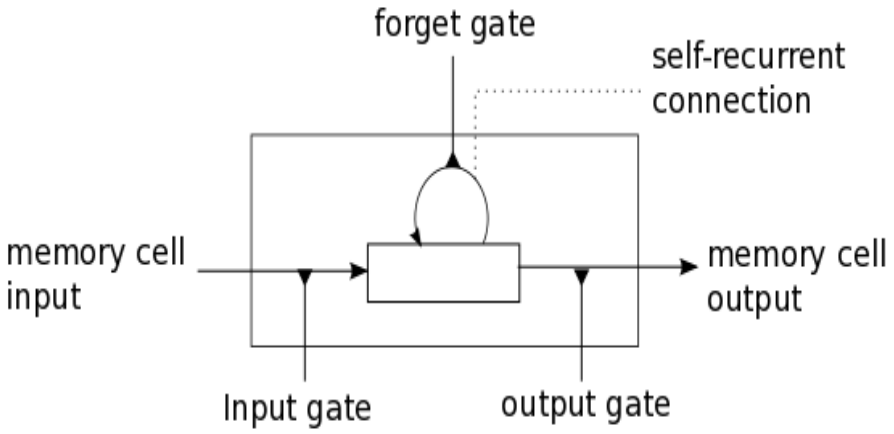

Fig. 6: LSTM Internal Architecture [19]

4.3 Gated Recurrent Unit (GRU)

A GRU is a gating mechanism in RNN deep learning techniques. The GRU performs same as LSTM with a forget-gate but it has fewer parameters compare to the LSTM. The GRU lacks an output-gate. GRU's performance on specific tasks of polyphonic music modeling, audio signal modeling, and speech signal modeling was found to out-perform the LSTM configuration and implementation [20]. The GRUs have exhibited even better performance on specific smaller datasets due to its design's lower complexity. According to recent research, GRUs have comparable performance to LSTM in sequence modeling with lower computational costs [21]. This work also experimented with the GRU for classifying two toad/frog species because it can perform the detection task with a higher performance like LSTM but less complex computations. It can also help to reduce the detection time for the Houston toad and Crawfish frog. Figure 7 from [22] shows the internal architecture of the GRU.

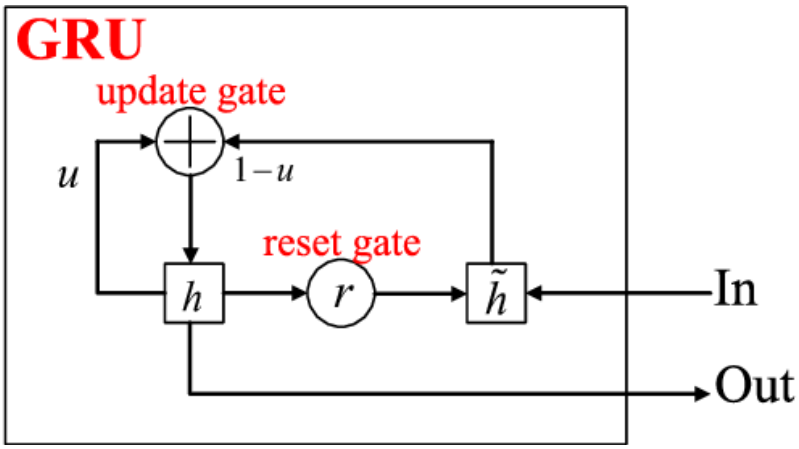

Fig. 7: GRU Internal Architecture [22]

4.4 Convolutional Neural Network (CNN)

The convolutional neural network (CNN) has shown excellent performance in computer vision, machine learning, and pattern recognition problems. CNNs are useful in many applications that deal with image-related tasks for the deconstruction of features in an image to identify feature patterns. Applications of CNNs include image classification, image semantic segmentation, object detection in images, and other image and video implementations. This thesis research focuses on image classification.

Convolutional Layer

For this layer, filters are applied to the original image or other feature maps in a deep CNN. This layer contains most of the user-specified parameters are in the network. The number of kernels and the size of the kernels are the most critical parameters of this layer. Audio Mel-spectrogram images have been used as the input in this work and filters were applied to the Mel-spectrogram images.

Pooling Layer

This layer is similar to convolutional layers, but it has a specific function to perform such as max pooling or average pooling. Max pooling takes the maximum value in a specific filter region. However, average pooling takes the average value in a filter region. This layer is usually used to reduce the dimensionality of the network.

The Fully Connected Layer

The configuration of the fully connected layer just reflects its name. it is fully connected with the output of the previous layer. Fully connected layers are usually used in the last stages of the CNN model and it connects to the output layer. At last it provides the desired output.

4.5 Ensemble Learning

Ensemble learning is a machine learning paradigm where multiple learners are trained to solve the same problem. An ensemble contains several learning models, also known as learners. Base learners can be made of different classifiers or the same type of classifiers with different hyperparameters, and the same set of input data will be used for all the models included in the ensemble technique. The generalization ability of an ensemble is usually more robust than that of base learners. Ensemble learning can assign confidence to the decision made by the model. There are various techniques to ensemble several architectures. This thesis research experimented with the "*majority voting ensemble*," as this technique can make the prediction more accurate or trustworthy. Figure 8 shows an example where three models are being ensembled to give the prediction. In Figure 8 the same input data are fed to Model A, Model B and Model C

and these models make predictions individually. The class which is predicted by at least any two of Model A, Model B, Model C is selected as the final prediction.

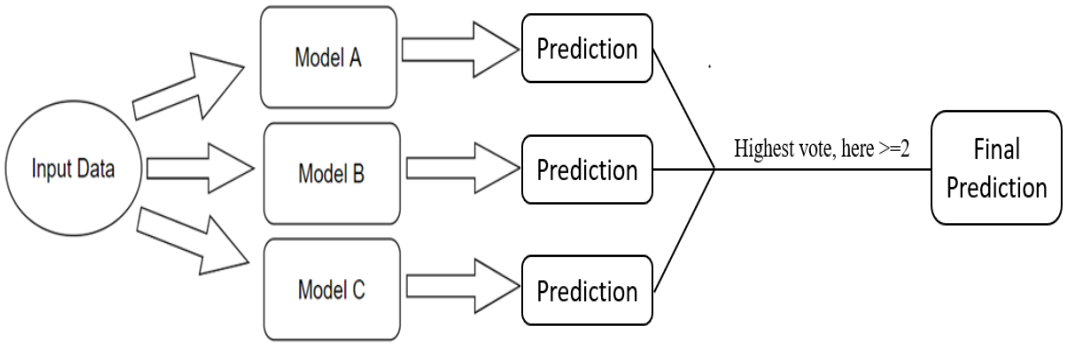

Figure 8: An Example of Ensemble Learning

From [23], a voting ensemble works by combining the predictions from multiple models. It can be used for classification or regression. In the classification case, the predictions for each label are summed and the label with the majority vote is predicted. There are two approaches to the majority vote prediction for classification: hard voting and soft voting. Hard voting involves summing each class label's predictions and predicting the class label with the most votes. Soft voting involves summing the predicted probabilities for each class label and predicting the class label with the largest probability. This work used hard voting to apply the voting mechanism of ensemble learning.

5. METHODOLOGY

This chapter discusses a modified software solution's overall architectural framework, which includes the recorded audio signal processing task to develop a trained and tested classification model for the existing Toadphone 1 system. Figure 9 shows the architectural building blocks of the framework of the design methodology. At first, the recorded audio data were labeled and preprocessed. The next tasks were splitting the training and test samples, extracting audio features, and training and testing the classifier models to predict the target output. Several classification methods have been tested and evaluated in the prediction of the classification of the toad and frog calls.

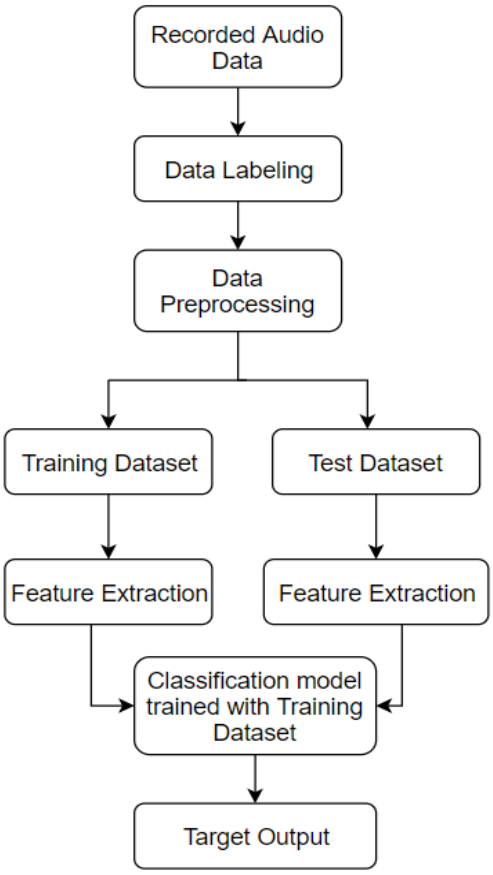

Figure 9: Architectural Building Blocks of Working Process

5.1 Audio Dataset

Real-world audio recordings were collected by the Biology Department at Texas State University. All the recorded audio files for Houston toad were in the WAC (Wildlife Acoustic) format as those were collected by Wildlife Acoustic song meters and converted to wav format. The “wac2wav” software was used for the WAC to WAV conversion. All recorded files have been converted to the WAV format as the programming languages such as Python and MATLAB are more adaptable with WAV audio format compared to WAC. The audio files for Crawfish frog were already in WAV file format. The call detection experiments have been performed in three ways: “Houston toad” or “no-toad” classification experiment, “Crawfish” or “Houston” classification experiment, and “Houston” or “Crawfish” or “Environment” classification experiment.

- The “Houston toad” or “no-toad” classification experiment consisted in using a dataset that included a total of 860 audio files. Among the 860 audio files, 430 audio samples contained the Houston toad call, and 430 audio samples contained only environmental sounds. Environmental sounds were from other species or animals, wind, rain, thunder, and other weather-related events. The audio samples containing Houston toad calls were labeled as 1, and the audio samples having environmental sounds were labeled as 0.
- The “Crawfish” or “Houston” classification experiment utilized a total of 1070 audio samples. Among 1070 audio files, 490 audio samples contained the Crawfish call, and 580 audio samples contained the Houston toad call. The audio samples with the Crawfish frog call were labeled as 1, and the audio files having Houston toad call were labeled as 0.

- The “Houston” or “Crawfish” or “Environment” classification experiment utilized a total of 1,000 audio files. Among the 1,000 audio files, 370 samples contained the Houston toad call, 370 samples contained the Crawfish frog call, and 260 samples contained only environmental sounds. The audio files with environmental sounds were labeled as 0, the audio files having the Houston toad call were labeled as 1, and audio files with the Crawfish frog call were labeled as 2.

5.2 Audio Data Pre-processing

Preprocessing steps, such as framing, window function, and noise reduction were performed for optimal recognition of audio signals from the samples.

- **Audio Data Filtering:** Recordings which are acquired from natural habitats are mostly overlapped with the sounds emitted by other species and different kinds of background noise. The conventional techniques are not adaptable with background noise in a voiced audio. Digital filters are commonly used in the pre-processing stage of animal audio signals, as strong interference noise from the environment can shadow the vocalizations within the data [5].
  - ❖ The “Houston toad” or “no-toad” classification experiment: The Houston toad call frequency varies between 1,500 Hz to 2,600 Hz. All audio files narrowed down to 1,500Hz to 2,600 Hz frequency range using a bandpass digital filter. Figure 10 shows the bandpass filtered frequency spectrum of Houston toad call.
  - ❖ The “Crawfish” or “Houston” classification experiment: The Crawfish frog call frequency varies between 300Hz to 1,600Hz. All audio files were filtered down

to 300 Hz to 1,600 Hz frequency range using a bandpass digital filter. Figure 11 shows the bandpass filtered frequency spectrum of Crawfish frog call.

- The “Houston” or “Crawfish” or “Environment” classification experiment: The audio files that contained the Houston toad call were filtered between the 1,500Hz to 2,600Hz frequency range. The audio files that contained the Crawfish frog call were filtered to the 300 Hz to 1,600Hz frequency range. Among the 240 audio samples containing only environmental sounds, 120 audio samples were filtered to the 1,500Hz to 2,600 Hz frequency range, and 120 audio samples were filtered between the 300Hz to 1,600Hz frequency range. The bandpass filtering process was used as the digital filter for the desired frequency ranges.

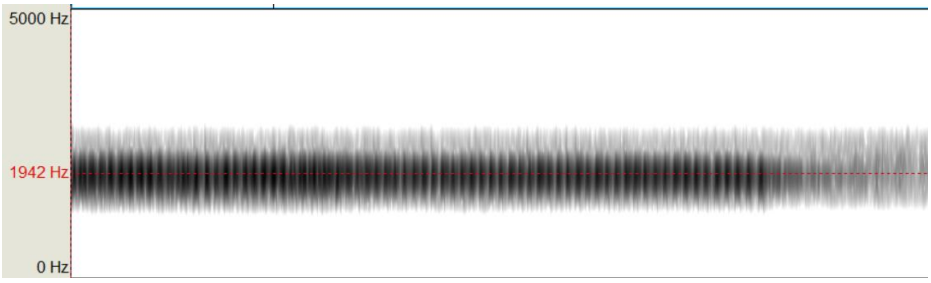

Figure 10: Bandpass Filtered Frequency Spectrogram of the Houston toad Call

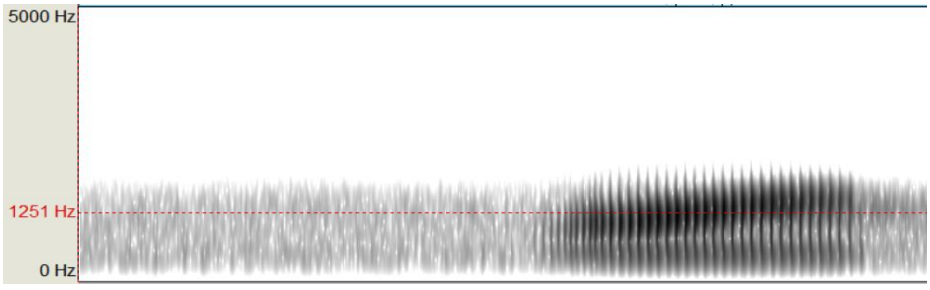

Figure 11: Bandpass Filtered Frequency Spectrogram of the Crawfish frog Call

- **Framing and the Window Function:** After applying the digital filter to the audio signals, the next preprocessing step was the framing process and applying the window function to the audio signal. Framing converts the audio signals into a set of frames. If the frame duration is too long, it is not possible to determine the time-varying characteristics of the audio signals. On the other hand, if the frame duration is too short, then it is not possible to get valid acoustic features. For this research work, the time duration of each frame was set to 80 milliseconds with a 50% overlap. The sample rate for all audio data was 16KHz, frame size was 16,000*(80/1,000) = 1,280 sample points, and the overlap for each frame was 16,000*(40/1,000) = 640 sample points. There can be discontinuity between the last frame and the first frame after the converting each audio signal into a number of frames. This discontinuity is called the spectral leakage, and it is problematic for time-domain to frequency-domain conversion. To reduce the effect of the spectral leakage, a window function is applied that smooths out any discontinuity [24]. For this work, each frame was multiplied by a "Hamming" window function.

  From [24] the Hamming window is defined as

  $$w(n)=0.54-0.46\cos\left(\frac{2\pi n}{M-1}\right) \qquad 0 \leq n \leq M-1 \quad \text{.....(1)}$$

  It was recommended for smoothing the truncated autocovariance function in the time domain.

5.3 Dataset Partitioning

- In the “Houston toad” or “no-toad” classification experiment, the dataset was split among 860 audio files. The training split contained 800 audio samples, and 60 audio files were used for testing. It is a good practice to use 20-30% data for validation from the training set. Therefore, among 800 training audio files, 80% audio samples were used for training, and 20% audio samples were used for validation. Among the 60 test audio samples, 30 samples contained the Houston toad call, and 30 audio samples contained no Houston toad call signatures.
- In the “Crawfish” or “Houston” classification experiment, among the 1,070 audio files, 1,000 audio samples were used for training and 70 audio samples were used for testing. Among the 1,000 training audio files, 80% audio samples were used as training data, and 20% audio samples were used for validation. Among the 70 test audio files, 35 samples contained the Crawfish frog calls, and 35 audio samples contained the Houston toad call.
- In the “Houston” or “Crawfish” or “Environment” classification experiment, among the 1,000 audio files, 940 audio samples were used for training, and 60 audio samples were used for testing. Among the 940 training audio files, 80% audio samples were used for training, and 20% audio samples were used for validation. Among the 60 test audio files, 20 samples contained the Crawfish frog call, 20 audio samples contained the Houston toad call, and 20 samples contained only environmental sounds.

5.4 Audio Feature Extraction

Feature-extraction methods aim at obtaining the most distinctive features from audio signals, and the choice of features can have a large impact on the quality of classification results. Frequently used audio features for voice or animal call detection are Mel-Frequency Cepstral Coefficients (MFCC) with their derivatives, Linear Predictive Codes (LPC), Perceptual Linear Predictive Coefficients (PLPs), Log Frequency Power Coefficients (LFPCs), Spectral Sub-band Centroids (SSCs), and Mel-spectrograms. MFCC and their derivatives have proven to be the most accurate and mostly used features for audio [6]. The features can be extracted by using Python's library '*LibROSA*' [25], '*Python_speech_features'* [26], Hidden Markov Model Toolkit (HTK) [27], and MATLAB libraries.

This thesis research experimented and evaluated the MFCCs with their first and second derivatives, SSCs, and Mel-spectrograms audio features.

- Mel Frequency Cepstral coefficients (MFCCs): MFCC is recognized as one of the best and most widely used acoustic signal feature used for audio or speech recognition [30]. This thesis work used 39 MFCCs with delta and delta-delta coefficients. These features were extracted using Python's libraries "Librosa" and "Python_speech_features". For generating MFCCs, the Mel-Filterbanks were needed to be calculated. For computing filter banks, triangular filters are typically applied on a Mel-scale to the power spectrum and it extracts frequency bands. The goal of Mel-scale is to mimic the human ear perception of sound as this scale is less discriminative at higher frequencies and more discriminative at lower frequencies. It

is possible to convert between Hertz (f) and Mel (m) using the following (2) and (3) [28].

$$m = 2595\log_{10}(1+\frac{f}{700}) \quad \text{.....(2)}$$

$$f = 700\,(10^{m/2595}-1) \quad \text{.....(3)}$$

For generating MFCCs, Discrete Cosine Transform (DCT) is applied to the filter bank coefficients. Using thirteen MFCCs, thirteen delta coefficients, which are the difference of each MFCCs and thirteen delta-delta coefficients, which are the difference between thirteen delta MFCCs coefficients, were determined.

- Spectral Sub-band Centroids (SSCs): SSCs audio features are also proven to be a good feature for audio or speech recognition. These features have better performance for noisy environments [7][29]. To compute SSCs, the frequency band is divided into a fixed number of sub-bands. According to [30] Spectral centroid is the weighted average frequency for a given sub-band, where the weights are the normalized energy of each frequency component in that sub-band. Since this measure captures the center of gravity of each sub-band it can locate large peaks in sub-bands. These peaks correspond to the approximate location of formants or pitch frequencies.
- Mel-spectrogram: A spectrogram is a representation of a signal like audio signal which shows the evolution of the frequency spectrum in time. A Mel-spectrogram is one kind of spectrogram and here the frequencies are transformed to the Mel-scale. A spectrogram is computed by calculating the Fast Fourier Transform (FFT) over a series of overlapping frames those are extracted from the original signal. This research used 2,048 sample points as the frame length, and 1,024 sample points as the

overlap length. The process of dividing the signal in short-term sequences of fixed size and applying FFT on those independently is known as Short-Time Fourier Transform (STFT). The spectrogram is computed as the squared and complex magnitude of the STFT. Extracting short term windows of the original image affects the enumerated spectrum by producing spectral leakage. This work used “Hamming” window function to reduce the spectral leakage effect during extracting Mel-spectrograms. This thesis used MATLAB to extract Mel-spectrogram.

5.5 Classification Model Architecture

This thesis work experimented and evaluated three classifiers: LSTM, GRU, and CNN. The architecture of the classification model is about defining the numbers, types and arrangement of the different types of layers in the model along with the values of any variables associated with these layers, such as the number of neurons in LSTM or GRU layer, the size of filters in the convolutional and max pooling layers, or the number of nodes in the fully-connected layer, and the selection of the hyperparameters such as optimizers, learning rate, activation function, loss function, batch size, epochs, validation split.

The process of developing the proper architecture to accomplish a certain job requires conducting many experiments since the architecture is dependent on the characteristics and size of the dataset used to train the model. Therefore, there are no formulas or certain rules to follow while designing a classification model.

LSTM or GRU Model Architecture

Designing the LSTM and GRU model architectures was very similar. For the architecture with GRU, the first layer is GRU cell layer, the second layer is a dense layer, the third layer is a dropout layer, fourth layer is a flatten layer, and the fifth or the last layer is the output dense layer. For the architecture with LSTM, the first layer is LSTM cell layer, the second layer is a dense layer, the third layer is a dropout layer, fourth layer is a flatten layer, and the fifth or the last layer is the output dense layer. The input for LSTM or GRU architecture was 39 MFCCs or 16 SSCs audio features. Following paragraphs discusses the hyperparameters of each layer for LSTM or GRU architecture.

LSTM or GRU Layer

- ***Return Sequence:*** The parameter "*return_sequences*" was kept "True" for both LSTM and GRU layer. Thus, the LSTM or GRU layer returned the output for each timestep to the next layer. This way the next dense layer got information from every time step for that sample and performed proper sequential analysis.
- ***Recurrent Activation:*** LSTM or GRU layer was associated with "*recurrent_activation*" function, several choices for that activation function were tested; however, the "*Rectified Linear Unit*" function was found to produce the best performance.

Dropout Layer

- ***Rate:*** The rate for the dropout layer was given 0.3 for this work. Which means 30% of the output of the layer were dropped out and it helped preventing overfitting.

Flatten Layer

After a dropout layer, the fatten layer is used to convert two-dimensional feature vectors into a one-dimensional array. This layer has made the output from dropout layer compatible for the last output dense layer as the shape of the train data is one dimensional array.

Output Dense Layer

The output layer for this model architecture was a dense or fully connected layer. After performing all the tasks in previous layers, this layer produced the outcome of the model.

- ***Activation Function:*** As the model architecture for this thesis work is designed for binary classification, "Sigmoid" was used as the activation function for this layer.

CNN Model Architecture

The input for this model was a Mel-spectrogram of audio files which is images. The Mel-spectrogram images used for this work were RGB images. This architecture was implemented for the "Houston" or "Crawfish" or "Environment" experiment. The model was designed using one convolutional layer, one max pooling layer, the flatten layer, dropout layer, and the last fully connected or dense layer. Following paragraphs discuss the hyperparameters of these layers used to design the CNN model for this thesis research.

Convolutional Layer

- ***Filters:*** It refers to the dimensionality of the output space or the number of output filters in the convolution. This is an integer value and the most variable parameter of the convolutional layer. The filter count of this research was 32.

- ***Kernel Size:*** It is a tuple or list of two integers, specifying the height and width of the 2D convolution window. 3x3 or 5x5 is a common choice for the kernel size. This thesis work used 3x3 as the kernel size.
- ***Stride:*** Stride is a component of convolutional neural networks, or neural networks tuned for the compression of images. For example, if the stride is set to 1, the filter will move one pixel, or unit, at a time. Naturally, as the stride, or movement, is increased, the resulting output will be smaller. For this layer, it was kept at the default value 1.
- ***Padding:*** The kernel does not perfectly fit the input length of the convolutional layer. There are two options when the misalignment occurs. Either pad the input tensor with zeros so that it fits, known as zero-padding, or by dropping the part of the image where the filter does not fit, known as valid-padding. In this work, zero-padding was adapted.
- ***Activation Function:*** Each convolutional layer is associated with an activation function. Several choices such as *"Exponential Linear Unit,"* "*Rectified Linear Unit*," "*LeakyRelu*" for the activation function were tested; however, the "*Rectified Linear Unit*" function was found to produce the best performance.

Max-Pooling Layer

- ***Kernel Size:*** The size of the pooling operation or filter is smaller than the size of the feature map; specifically, it is almost always 2×2 pixels. This configuration was adapted for this work
- ***Stride:*** Stride is set to 2 for the max-pooling layer for this work. It means the filter will move two pixels at a time.

- ***Dropout Layer Rate:*** The rate for the dropout layer was given 0.25 for this work. Which means 25% of the output of the layer were dropped out and it helped prevent overfitting.

Other Hyperparameters for LSTM, GRU or CNN Model

- ***Number of Epochs:*** The number of epochs is a hyperparameter that defines the number of times that the classification algorithms will work through all the samples in the training dataset. It is a common practice to observe both the training and validation learning curves to define the number of epochs. These plots help to decide if the classification model overfitted, underfitted, or perfectly learned. These models should be trained for the adequate number of epochs to allow the learning algorithm to run until the error is sufficiently minimized. Number of epochs were varied for each experiment and selected by trial and error method.
- ***Batch Size:*** The batch size defines the number of samples that propagate through the network during training the model. It also refers to the number of samples that are used for one iteration of a single epoch. Popular values for batch sizes include 16, 32, 64, and 128 samples. The batch size for this work was selected using trial and error method.
- ***Optimizer:*** Optimizers are algorithms or methods used to change the attributes of the neural network, such as weights and learning rates to reduce the losses to achieve the most accurate results. Usually, neural network algorithms use the Stochastic Gradient Descent (SGD) algorithm to initialize and update the weights used in all the layers [31]. The SGD uses a single learning rate throughout the training process. The Adam optimization algorithm is an extension of the SGD, which optimizes the learning rate

based on the first and second-order moments, such as the gradient mean and element-wise squared gradient, respectively. The Adam optimizer has been shown to produce significantly better results for image and natural language problems [32]. This research experimented with the SGD and Adam optimizers during the training and testing of the models.

- ***Learning Rate:*** The learning rate controls the change in a model in response to the estimated error each time the model weights are updated. It is a challenging task to choose the learning rate. If the value is too small, it may result in a long training process that could get stuck. If the value is too large, it may result in learning a sub-optimal set of weights too fast or an unstable training process. However, the learning rates determined by the trial and error method are the 0.01, 0.001 and 0.0001 values experimented, and the learning rate value of 0.0001 performed the best.
- ***Loss Function:*** The error for the current state of the model must be estimated repeatedly as part of the optimization algorithm. This requires the choice of an error function, conventionally called a "*loss function*", that can be used to estimate the loss of the model so that the weights can be updated as it converges to smaller gradient losses. The choice of the loss function must match to the specific predictive modeling problem, such as classification or regression. This thesis work is focused only on the classification predictive modeling approach. Selecting a loss function is also dependent on the targeted output of the model, such as binary or multiclass classification predicted outcomes. Cross Entropy is used as a good loss function for neural network classification problems, because it minimizes the distance between two probability distributions such as predicted and actual. The "Houston toad" or

"no-toad," and "Crawfish" or "Houston" experiments are binary classification problems and used the 'binary_crossentropy' loss function when compiling the model. The "Houston" or "Crawfish" or "Environment" experiment is multiclass classification problem and used the "*sparse_categorical_crossentropy*" loss function as the class labels are integers 0, 1, and 2.

5.6 The List of Experiments

Table 1 shows a list of the experiments based on dataset sample calls, audio features, and types of classifiers used for evaluation. The next chapter discusses the results of these experiments in detail.

Table 1: List of Experiments

Classification Experiment	Audio Feature	Classifier
"Houston toad" or "no-toad"	39 MFCCs	LSTM or GRU
	Mel-spectrogram	CNN
"Crawfish" or "Houston"	39 MFCCs or 16 SSCs	LSTM or GRU
"Houston" or "Crawfish" or "Environment"	39 MFCCs	LSTM

6. EXPERIMENS AND RESULTS

5.1 Programming Environment and Computational Resources

The Python 3.5.7 programming language was used for the development of the models, preprocessing, training, and testing. Preprocessing stages such as filtering, framing, windowing, and extracting audio features have been implemented using Python, verified through MATLAB, the Hidden Markov Model Toolkit (HTK), and MATLAB are widely accepted as audio signal processing and feature extraction toolkits. The "*Scipy"* Python library has been used to read and write the wav audio files. Butterworth has been imported using "*Scipy"* library for applying Bandpass digital filter to the audio files. The "*Python_speech_features"* and "*Librosa"* Python libraries has been used to extract audio features. For the classification design, deep learning algorithms, such as LSTM, GRU, and CNN were examined for their accuracy performance.

Python contains special libraries for deep learning, such as *Keras*, which is one of the most widely used libraries that support the development of LSTMs, GRUs, and CNNs using back-end libraries from "*TensorFlow"*. The LEAP (Learning, Exploration, Analysis, and Processing) cluster, which is a next-generation High-Performance Computing (HPC) cluster of Texas State University [33] was used to train and test the developed models. From [33], the cluster has 120 compute nodes, each with 28 CPU cores via two (14-core) 2.4 GHz E5-2680v4 Intel Xeon (Broadwell) processors, 128 GBs of memory, and 400 GBs of SSD storage per node. The compute nodes provide an aggregate of 15 TBs of memory and 48 TBs of local storage. The LEAP cluster supports Python installations through *"conda",* and version install control for its libraries.

5.2 Training and Evaluation Results

Many experiments have been carried out while developing a reliable classification model to detect the Houston toad or the Crawfish frog call.

The "Houston toad" or "Non-toad" Classification Experiment

The dataset included a total of 860 audio files, where 430 audio samples contained the Houston toad call, and 430 audio samples contained only environmental sounds. Among the 860 samples, 60 data samples were used for testing. For this experiment, the 39 MFCCs with their first and second derivatives were used as audio features. Each audio sample contained 149 frames with 80 millisecond frame size, and 40 millisecond (50%) overlap.

- ***LSTM and GRU as Classifier and MFCCs as Audio Feature***

The 39 MFCCs were extracted for each frame, and the input tensor for the classifiers is defined as (149, 39) for the LSTM and GRU input layer. Figure 12 shows the model structure for LSTM and GRU with 39 MFCCs features.

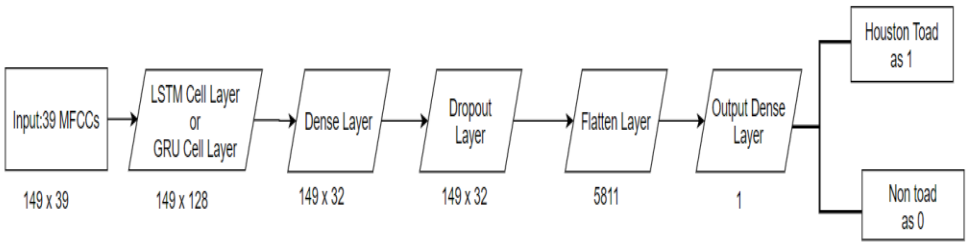

Figure 12: The LSTM and GRU Model Structure for the "Houston toad" or "Non-toad" Classification Experiment with 39 MFCCs

The architecture for both models is similar except for the first LSTM and GRU cell layer. The output shape of the LSTM and GRU cell layer is a tensor of (149, 128), where 149 is the time steps as the number of frames for each audio sample, and 128 is the

number of neurons for the LSTM and GRU cell layer which was selected through trial and error. The "*return_sequences"* which is a parameter for LSTM or GRU layer was kept "*True*". Then for sequential computation a dense layer was used with 32 neurons which was also selected by trial and error method. Then, a 20% dropout was used which reduced the overfitting issue of the model. A flatten layer was used to make the output shape from the dropout layer compatible for the output dense layer. The output of the flatten layer was a 1D array of 5811. The number of neurons for output dense layer was one providing a binary output, such as 1 for the "Houston toad" class, and 0 for the "no-toad" class. Time steps or number of frames is 149 was carried till the dropout layer and each layer till the dropout layer produced the output for each timestep. The *sigmoid* activation function is used for the output dense layer, and the *binary_crossentropy* was used as the loss function for classifying the "Houston toad" or "non-toad" classes.

From Figure 13(a), the LSTM model has gained a 87% training and a 82% validation accuracy for the classification of the "Houston toad" or "no-toad" class labels. The LSTM model has a 78% test accuracy, and this is able to predict 80% of the 30 test Houston toad call data correctly and this model misclassified 23% non-toad samples as Houston toad call as seen in Figure 13(b). So the false positive rate for this LSTM model architecture is 23%. The GRU model achieved a 85% training and a 78% validation accuracy as shown in Figure 14(a). From Figure 14(b), the GRU model achieved a 72% test accuracy and classifies with a 73% accuracy of the 30 test Houston toad call samples, and 30% of the non-toad samples were misclassified as Houston toad. So, the false positive rate for this GRU architecture is 30%. The validation and test accuracy for LSTM and GRU models are very close to training accuracy so both LSTM and GRU

models were not overfitted. Also, the LSTM model shows higher testing accuracy compare to the GRU model. The LSTM model predicted 80% of the 30 Houston toad call test data correctly with 23% false positives and the GRU model predicted 73% of the 30 Houston toad call test data correctly with 30% false positives. Here, the LSTM model performed better compare to GRU model to predict Houston toad call correctly. It is also noticeable that the LSTM model architecture has a lower false positive rate compared to GRU model architecture.

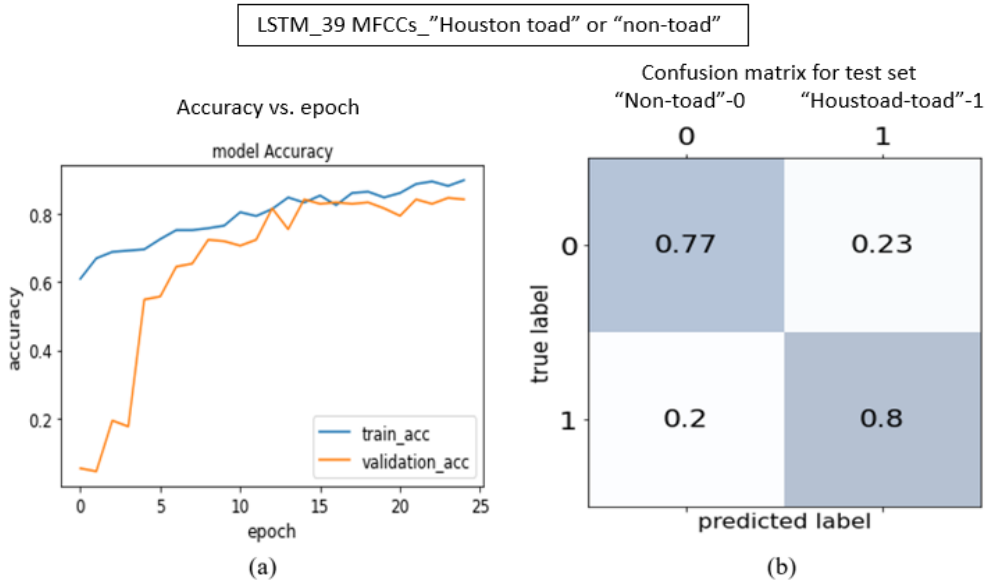

Figure 13: (a) Accuracy and (b) Confusion Matrix of the LSTM Model with 39 MFCCs for the "Houston toad" or "Non-toad" Classification Experiment

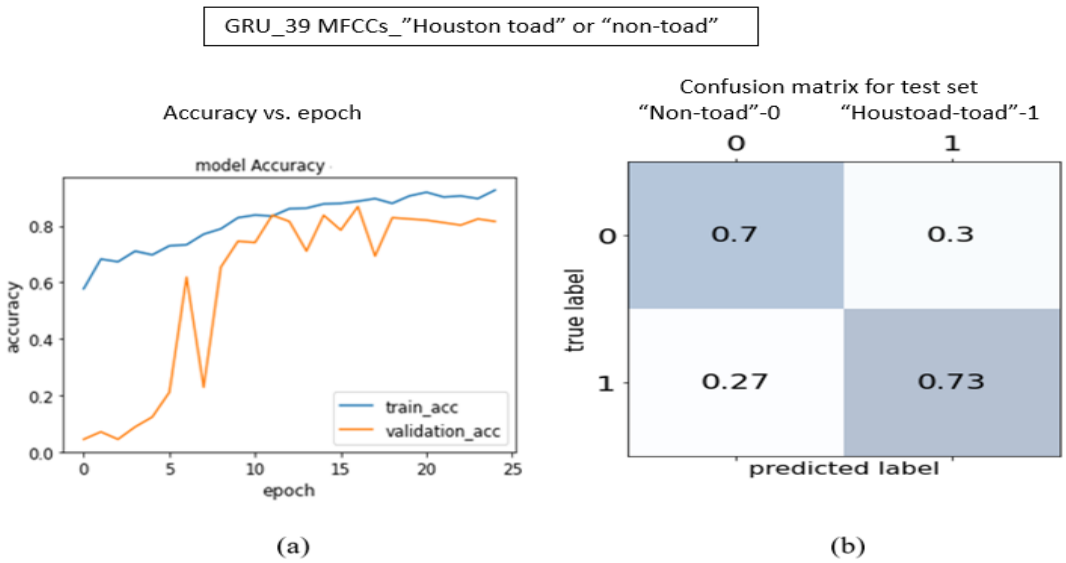

Figure 14: (a) Accuracy and (b) Confusion Matrix of the GRU Model with 39 MFCCs for the "Houston toad" or "Non-toad" Classification Experiment

- ***CNN as Classifier and Mel-spectrogram Images as Audio Feature***

For this experiment Mel-spectrogram of audio signals has been extracted using MATLAB. Figure 15(a) shows the Mel-spectrogram of an audio file of a Houston toad call. Figure 15(b) shows the Mel-spectrogram of an audio file without a Houston toad call, or the background noise or environment sound.

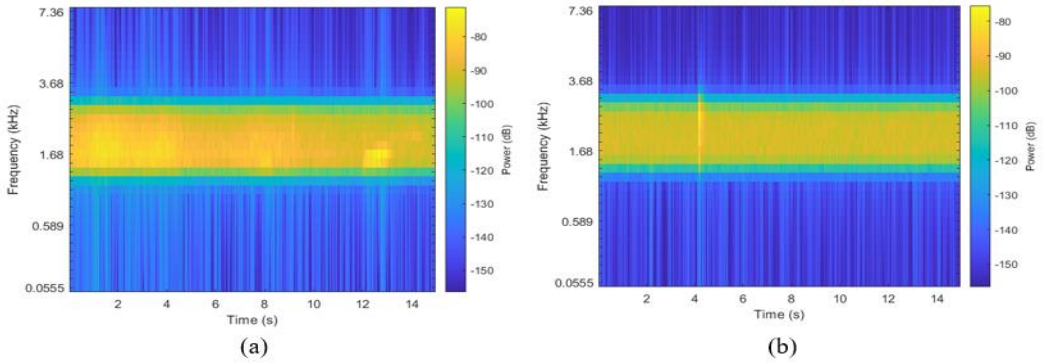

Figure 15: (a) Mel-Spectrogram of Audio File Having Toad Call, (b) Mel-Spectrogram of Audio File Having Non-toad (Only Environment Sound)

Each Mel-spectrogram is an image of the sound in the frequency-domain, which the CNN learns to classify in the same way that traditional image recognition paradigms work. These Mel-spectrogram's images were used as the input for CNN architecture. 840 Mel-spectrograms have been generated from the 840 audio files for the "Houston toad" or "non-toad" classification experiment. Among the 840 images, 420 Mel-spectrogram images represented the Houston toad audio call and are labeled as "toad," and 420 Mel-spectrogram images were for the environmental sound and are labeled as "non-toad". Among the two classes, 60 Mel-spectrogram images were used for test, where 30 Mel-spectrogram images were for the Houston toad and 30 Mel-spectrogram images were for the environmental sound. 20% data from the training set were used for validation. The images were cropped to center the spectrogram features in the image.

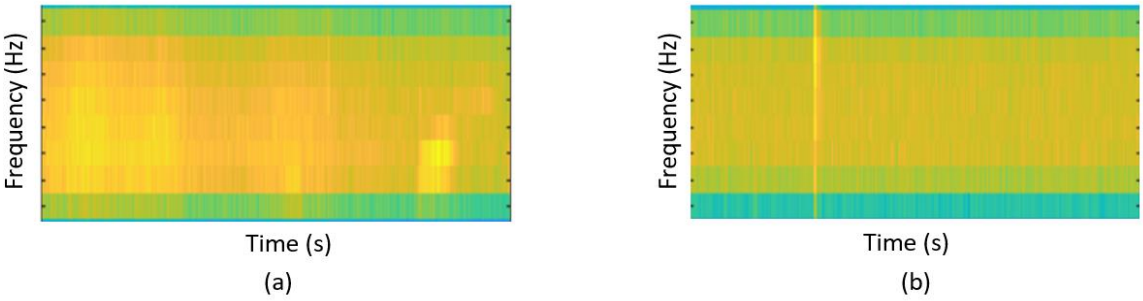

Figure 16: (a) Cropped Mel-Spectrogram of Audio File Having Toad Call, (b) Cropped Mel-Spectrogram of Audio File Having Non-toad (Only Environment Sound)

Figure 16(a) shows the cropped Mel-spectrogram images of audio files having a toad call. Figure 16(b) shows the cropped Mel-spectrogram of audio files not having a toad call or having only environment sound . The original size of the cropped image has a 598x140 original resolution. The image size was converted to a 128x128 resolution for avoiding dimension error and for less computational time. The input shape of the CNN model was

(128, 128, 3). As the Mel-spectrogram images used here are RGB images, the third values, 3, is defined for three channels of RGB images. Figure 17 shows the model structure of the CNN architecture used as the spectrogram image classifier of the audio samples. The ADAM optimizer with 0.0001 learning rate has been used, a batch size of 128, and an epoch number of 50 performed at its maximum capacity.

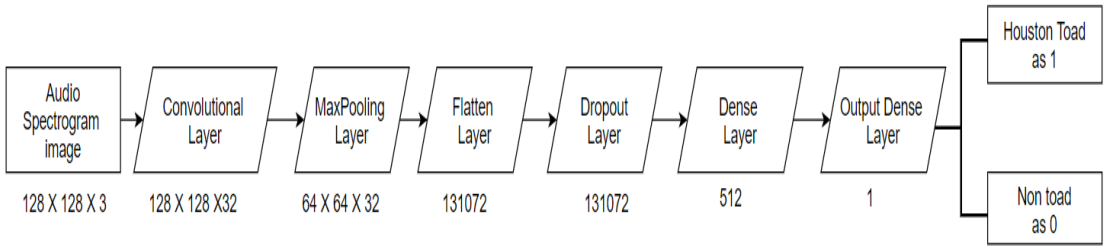

Figure 17: CNN Model Structure for the “Houston toad” or “Non-toad” Classification Experiment with Mel-Spectrogram Images

From Figure 18(a), a 91% training accuracy was achieved with 60% validation accuracy, which clearly shows the overfitting issue as the training accuracy is much higher compared to the validation accuracy. Also, the overall test accuracy of the model was 63% with 60% test accuracy on Houston toad achieved and it misclassified 33% non-toad sample as Houston toad call which can be seen from the confusion matrix in Figure 18(b). Hyperparameters like dropout layer were tuned manually in an attempt to solve the overfitting issue but it did not solve. The higher training accuracy and lower validation and test accuracy indicates that the model has learned features of the training data too well, but it did not generalize, or it did not work well on new data and it produced a high false positive rate. The overfitting issue may be solved by increasing number of data samples.

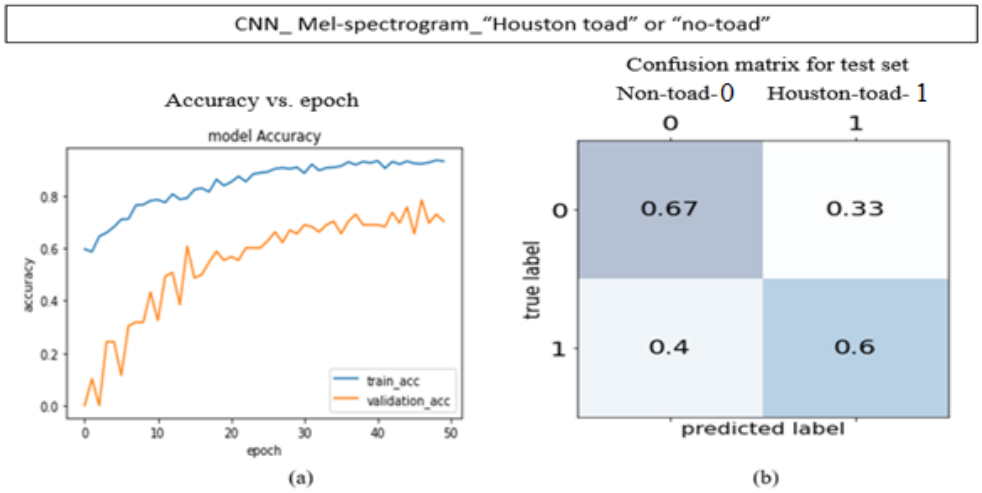

Figure 18: (a) Accuracy and (b) Confusion Matrix of CNN Model with Mel-Spectrogram for the "Houston toad" or "Non-toad" Classification Experiment

The "Crawfish" or "Houston" Classification Experiment

As mentioned in the "Methodology" chapter, among the total 1,070 audio data samples, 1,000 samples were used for training, with 455 audio samples containing the Crawfish calls and 545 audio samples containing the Houston toad call. Among the 70 test data samples, 35 audio samples have the Houston toad calls, and 35 audio samples have the Crawfish frog calls. Of the 1,070 training audio files, 80% of the audio samples were used as training samples, and the remaining 20% of the audio samples were used for validation. The duration of each audio file containing the Crawfish frog call is 60 seconds, and each audio file containing the Houston toad call is 1-15 seconds.

Each training audio file was fragmented into one-second clips to avoid the dimension error during training the model. These one-second audio clips were labeled the same as the parent audio files and were not overlapped with consequent fragments. The Hamming window function was applied to the fragmented clips, and these one-second

fragments are processed through the framing and windowing process for feature extraction. Each one-second clip has 24 frames, with 80 milliseconds frame size, and a 50% overlap (40 milliseconds). Thirty-nine MFCCs or sixteen SSCs were extracted for each frame. The classification models were trained using the extracted features from each frame.

- ***LSTM and GRU as Classifier and MFCCs as Audio Feature***

For this experiment, 39 MFCCs with their first and second derivatives were used as audio features. As each one audio clip has 24 frames and features were extracted for each frame, the input shape is (24, 39) for the LSTM and GRU layer. Figure 19 shows the model structure for LSTM and GRU with 39 MFCCs features. The architecture for both models is similar except for the first LSTM and GRU cell layer. The output of the LSTM and GRU cell layer is (24, 128), where 24 is the time steps for each one-second audio clip, and 128 neurons cells for the LSTM and GRU layer selected through trial and error.

Here the output dense layer provided a binary output, such as 1 for the "Crawfish" class, and 0 for the "Houston" class. Time steps or number of frames is 24 was carried till the dropout layer and each layer till the dropout layer produced the output for each timestep.

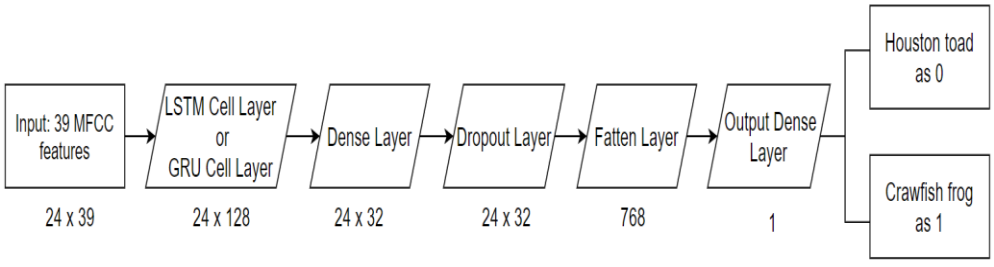

Figure 19: LSTM and GRU Model Structure for the "Houston" or "Crawfish" Classification Experiment with 39 MFCCs

Several combinations of hyperparameters had been evaluated for this experiment. Initially, the SGD optimizer was used with 200 epochs. The learning rate was tuned manually and 0.001 was selected. Figure 20 and Figure 21 shows the accuracy and loss plots for the experiment with the LSTM and GRU as classifier with 39 MFCCs as audio features for the SGD optimizer, 0.001 for the learning rate, and 200 epochs. From Figure 20, the LSTM model gained a 65% training and 63% validation accuracy with 58% losses which is very high. From Figure 21 the GRU model gained 68% training and 50% validation accuracy with 57% losses. Both LSTM and GRU models for this architecture gained low training and validation accuracies to learn the features for Houston toad and Crawfish frog, and from Figure 21, the GRU model shows an overfitting issue as there is a big gap between its training and validation accuracy and validation loss is not decreasing.

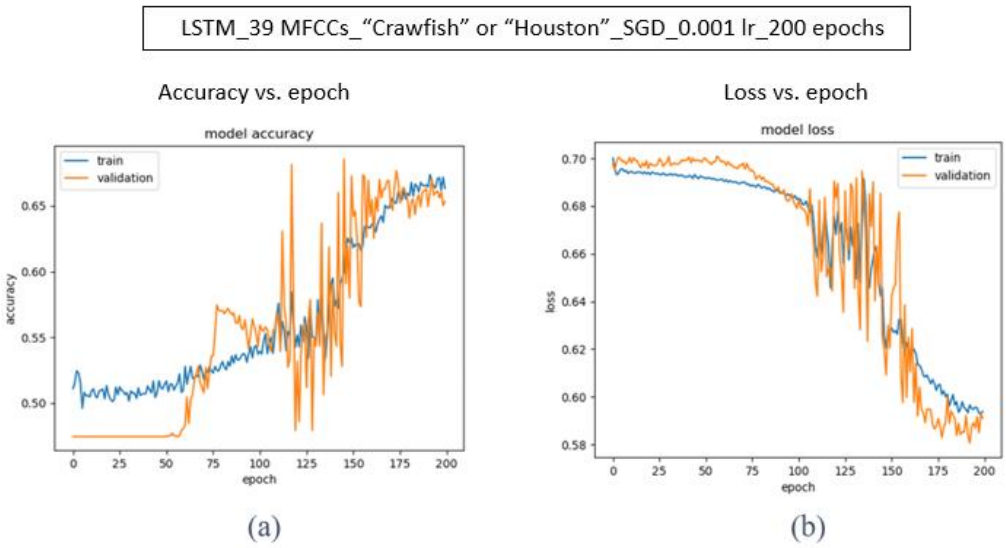

Figure 20: (a) Accuracy and (b) Loss Plots of the LSTM for 39 MFCCs with the SGD Optimizer, 0.001 for the Learning Rate, and 200 Epochs

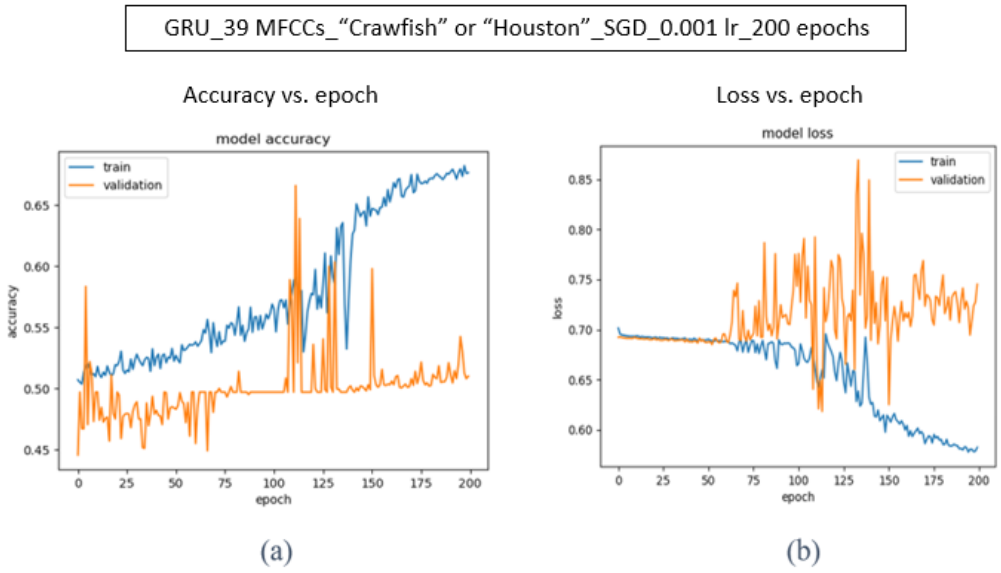

Figure 21: (a) Accuracy and (b) Loss Plots of the GRU for 39 MFCCs with the SGD Optimizer, 0.001 for the Learning Rate, 200 Epochs

The optimizer was changed to ADAM, which is an extension version of SGD. Using ADAM as optimizer with 0.001 learning rate and 200 epochs, from Figure 22, the LSTM model with 39 MFCCs gained 75% training and 75% validation accuracy with 50% training and 52% validation losses. From Figure 24, the GRU model gained 73% training and 74% validation accuracy with 53% training and validation losses. It is clearly visible that both the LSTM and GRU model showed better performance with ADAM optimizer. Though the accuracy of both models increased, loss is still more than 50%, which is much higher. From the confusion matrices, the test accuracy for the LSTM model was 61%, with a 69% accuracy for the "Houston" class and46% false positives. So, the accuracy based on true and false positives is 60% for Houston toad call. and a 54% accuracy for the "Crawfish" class as shown in Figure 23. The test accuracy of the GRU model was 58%, with a 63% accuracy for the "Houston" class, and a 54% accuracy for the "Crawfish" class which is shown in Figure 25. Both models have produced low test

accuracies. However, the LSTM model showed comparatively better performance than the GRU model with this model architecture. Also, for LSTM and GRU models, the accuracy for Houston toad call classification is higher compare to the accuracy for Crawfish frog. False positive rate for this architecture is also very high.

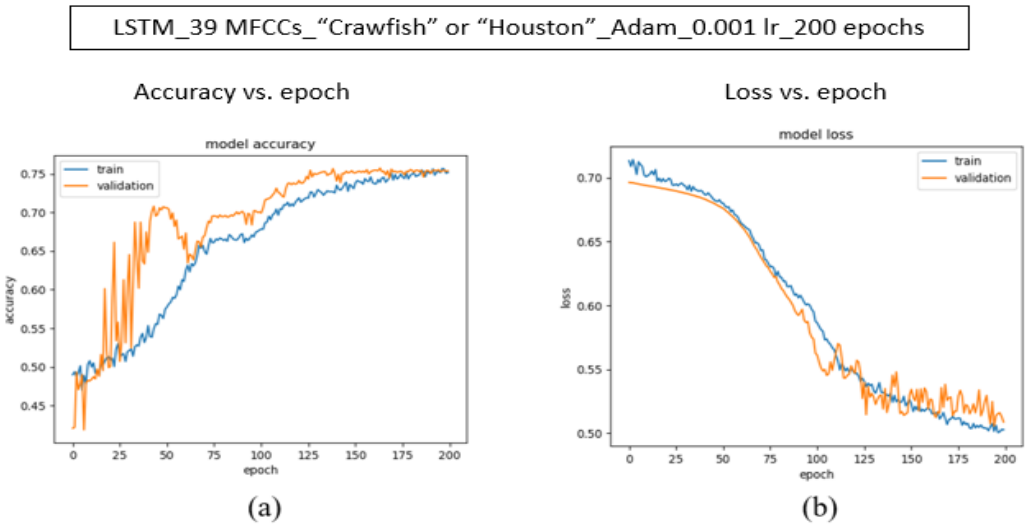

Figure 22: (a) Accuracy and (b) Loss Plots of the LSTM and 39 MFCCs with the ADAM Optimizer, 0.001 for the Learning Rate, and 200 Epochs

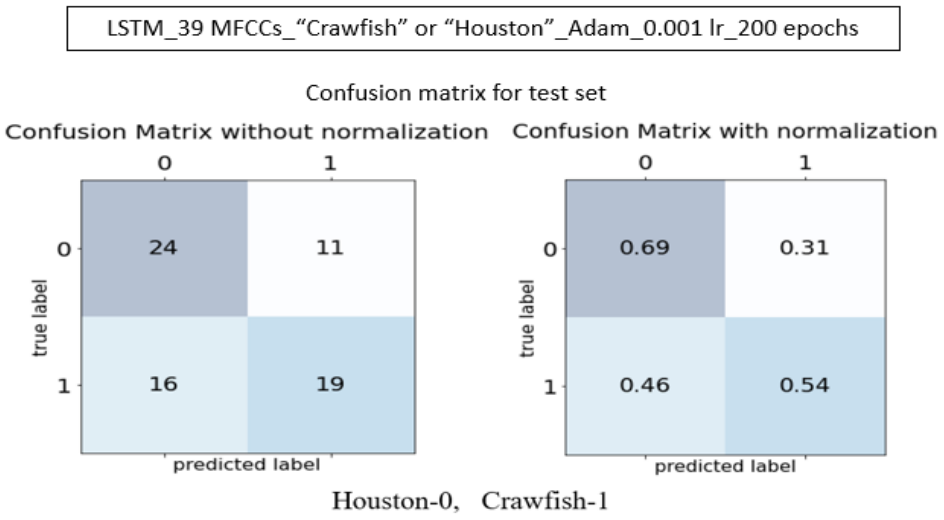

Figure 23: Confusion Matrix of the LSTM with 39 MFCCs for the ADAM Optimizer, 0.001 for the Learning Rate, and 200 Epochs

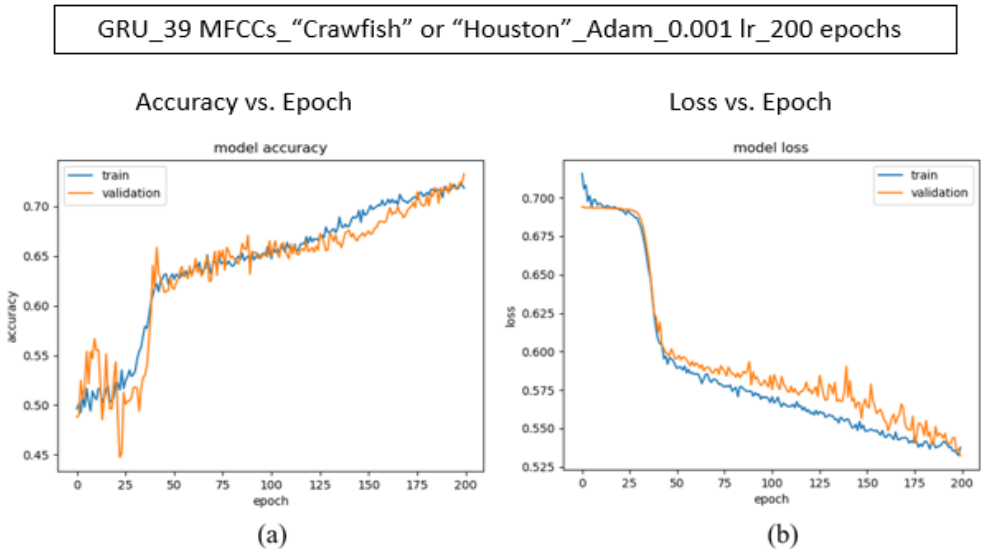

Figure 24: (a)Accuracy and (b) Loss Plots of the GRU for 39 MFCCs with the ADAM Optimizer, 0.001 for the Learning Rate, And 200 Epochs

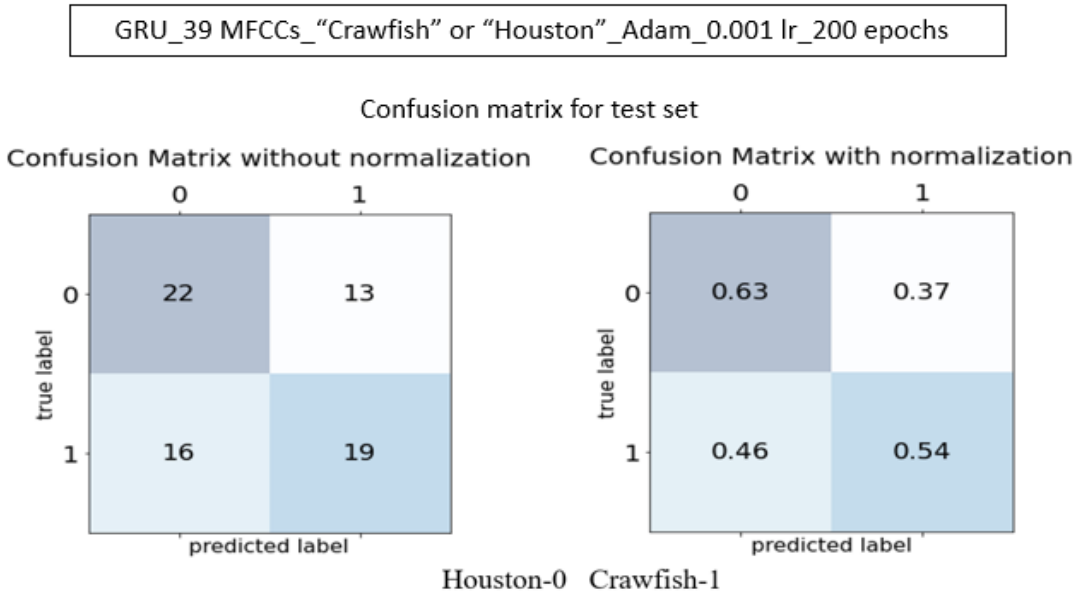

Figure 25: Confusion Matrix of the GRU with 39 MFCCs for the ADAM Optimizer, 0.001 for the Learning Rate, and 200 Epochs

From Figure 22 and Figure 24, it is observed that the accuracy curves for the LSTM and GRU models move upward indicating the accuracy may improve if the

number of epochs increases. After increasing the number of epochs to 300, and decreasing the learning rate to 0.0001, significant improvement was observed from Figure 26 and Figure 28 on the learning curves for the LSTM and GRU. From Figure 26, the LSTM model achieved an 84% training accuracy, an 82% validation accuracy, a 32% training loss, and a 38% validation loss. From Figure 28, the GRU model achieved an 80% training accuracy, a 79% validation accuracy, a 35% training loss, and a 40% validation loss. The very small gap between the training and validation accuracy and the decreased validation loss with decreased training loss for both LSTM and GRU model showed both the LSTM and GRU model were not overfitted. From the confusion matrices in Figure 29, the test accuracy for the LSTM model was 76%, with an 80% accuracy for the "Houston" class, and a 71% accuracy for the "Crawfish" class. The false positive rate for Houston toad was 29% and the false positive rate for crawfish call was 20%. The test accuracy for the GRU model was 73% with 77% accuracy on "Houston", 31% crawfish frog audio samples were misclassified as Houston toad calls and 69% accuracy on "Crawfish", 23% Houston toad audio samples were misclassified as Crawfish call which is shown in Figure 29. From the test accuracies for "Houston" and "Crawfish" it can be noticed that both LSTM and GRU architectures have higher individual test accuracy for Houston toad call detection compared to Crawfish frog call. The audio files that have the Crawfish frog calls are 60 seconds long and do contain more environment background noise than the Houston toad call audio files. For the crawfish frog call, the model learned extracted environment background noise as the Crawfish call when training the model. It is possible that more Crawfish frog test samples were misclassified as the environment background noise of the Houston toad call. Here, again

the training, validation and test accuracies gained by the LSTM model were higher compared to the GRU model to classify Houston toad or Crawfish frog. False positive rate for Houston toad calls and Crawfish frog calls was also less with the LSTM model compared to GRU model architecture.

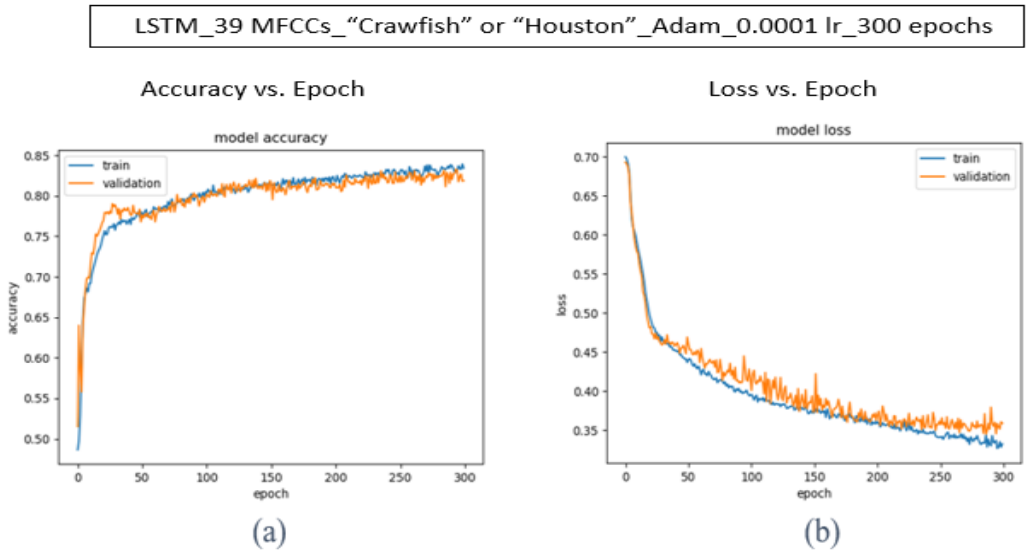

Figure 26: (a) Accuracy and (b) Loss Plots of the LSTM and 39 MFCCs with the ADAM Optimizer, 0.0001 Learning Rate, 300 Epochs

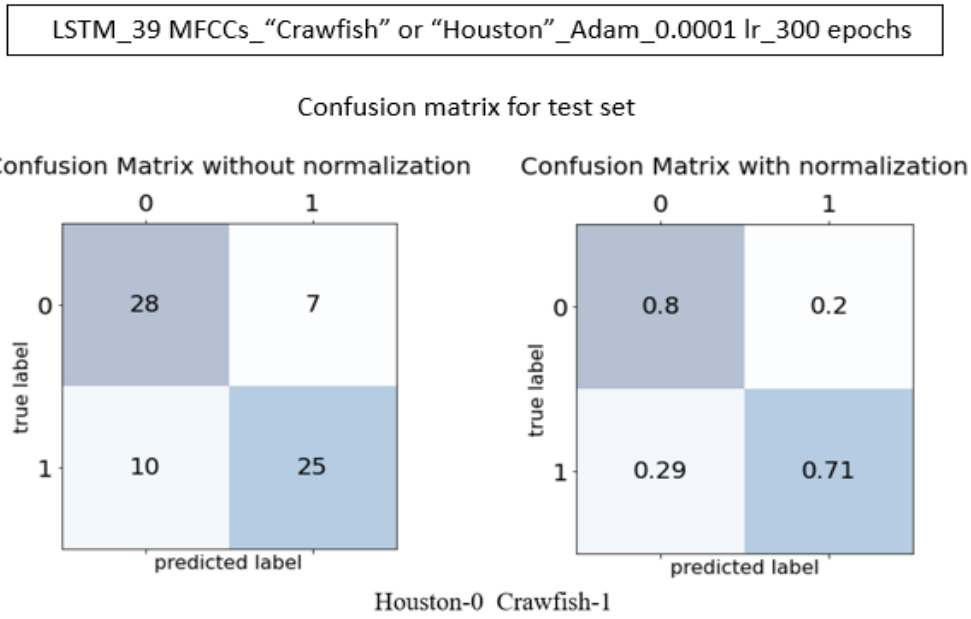

Figure 27: Confusion Matrix of the LSTM with 39 MFCCs for the ADAM Optimizer, 0.0001 Learning Rate, 300 Epochs

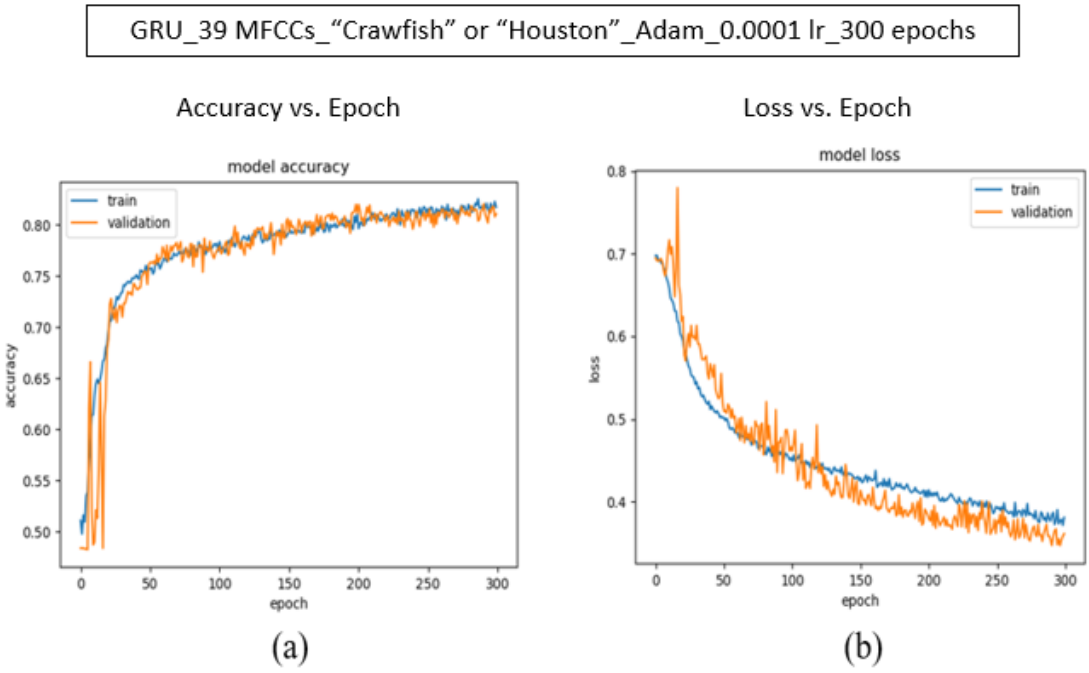

Figure 28: (a) Accuracy and (b) Loss Plots of GRU with 39 MFCCs for the ADAM Optimizer, 0.0001 Learning Rate, 300 Epochs

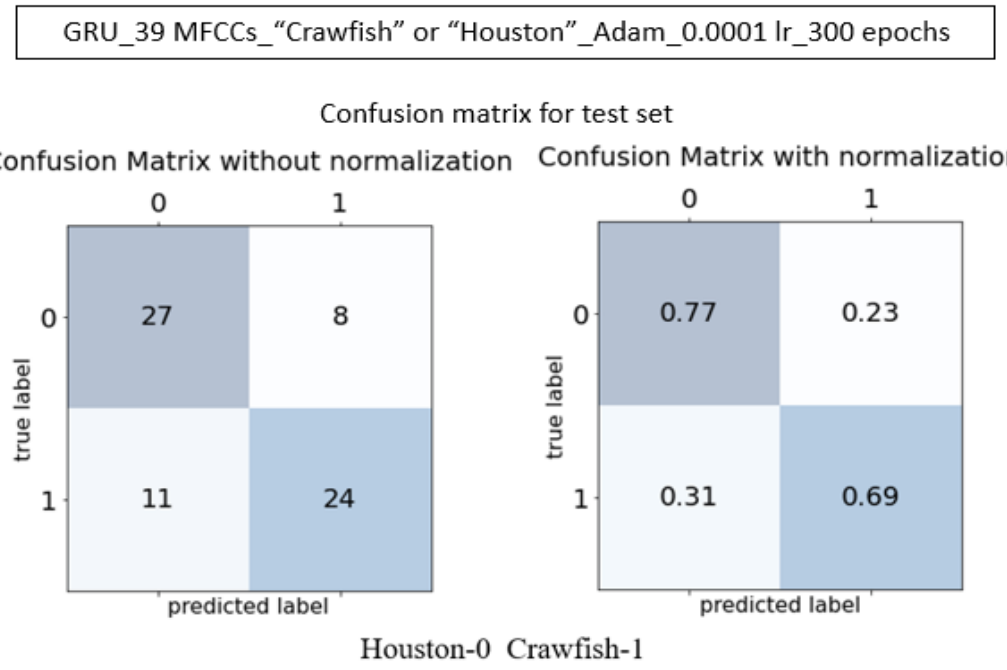

Figure 29: Confusion Matrix of the GRU with 39 MFCCs for the ADAM Optimizer, 0.0001 Learning Rate, 300 Epochs

- ***LSTM and GRU as Classifier and 16 SSCs as Audio Feature***

For this experiment, spectral centroids were extracted for 16 sub-bands for each frame. As each audio clip has 24 frames and 16 SCCs features were extracted from each frame, the input shape is (24, 16) for the LSTM and GRU layer. Figure 30 shows the model structure for the LSTM and GRU with 16 SSC features. The structure is the same as the model structure already described for the *"LSTM and GRU as classifier and SSCs as audio feature"* experiment except the input shape. The input shape is (24, 16), where 24 is the number of frames or timesteps, and 16 is the number of SSC features.

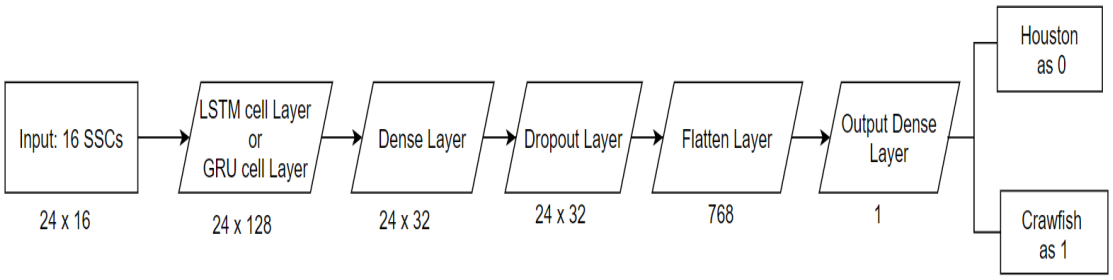

Figure 30: LSTM and GRU Model Structure for the "Houston" or "Crawfish" Classification Experiment with 16 SSCs

Figure 31 shows the accuracy and loss plots for the LSTM, with sixteen SSCs as audio features with the ADAM optimizer, 0.0001 for the learning rate, and 300 epochs. The LSTM model performance is shown in Figure 31 with a 69.9% training accuracy, a 67.7% validation accuracy, and a 68.5% test accuracy with a 71.3% accuracy for the Houston toad and a 65.7% accuracy for the Crawfish frog call using SSC. From Figure 32, the model with GRU has a 66.5% training, a 65% validation, and a 64.2% test accuracy with a 68% accuracy for the Houston toad and a 60% accuracy for Crawfish frog call. From the accuracy plots, both LSTM and GRU models training and validation accuracy decreased after changing the audio features to SSCs.

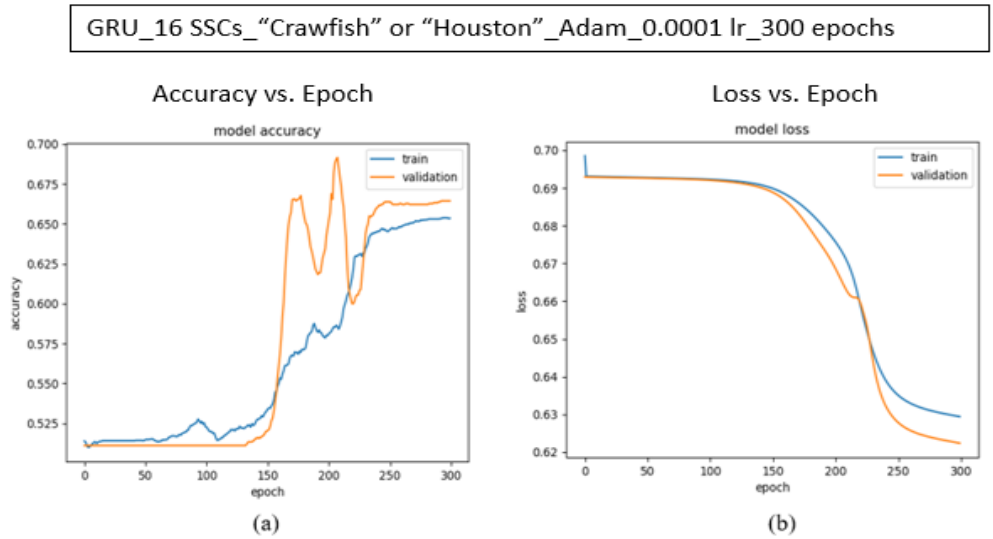

Figure 31: (a) Accuracy And (a) Loss Plots of The LSTM with 16 SSCs for ADAM Optimizer, 0.0001 for the Learning Rate, and 300 Epochs

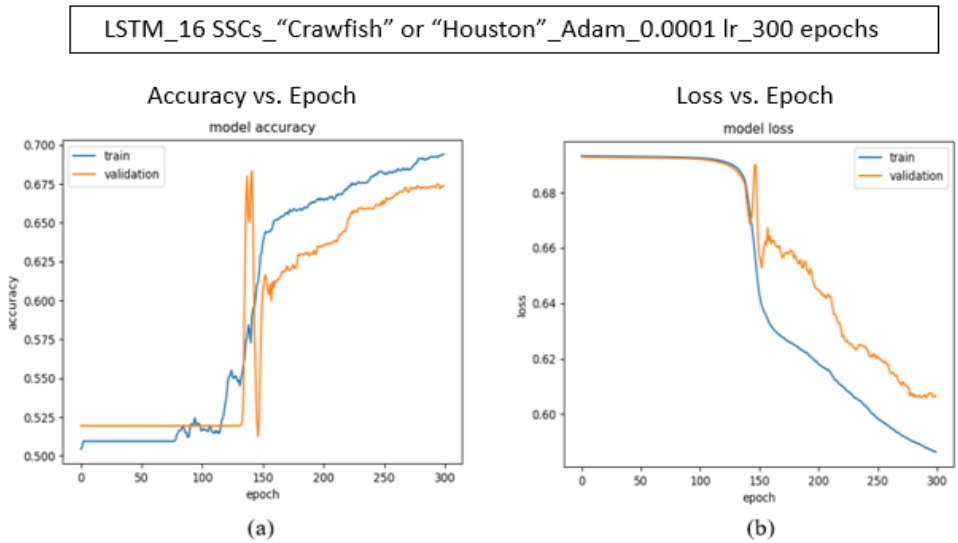

Figure 32: (a) Accuracy and (b) Loss Plots of the GRU with 16 SSCs for the ADAM Optimizer, 0.0001 for the Learning Rate, and 300 Epochs

Table 2 summarizes the results achieved using LSTM and GRU classifiers with MFCC and SSC features using the ADAM optimizer, 0.0001 learning rates, and 300 epochs for "Houston" or "Crawfish" classification. LSTM with 39 MFCCs and LSTM with 16 SSCs have shown higher training validation and testing accuracy compare to

GRU with 39 MFCCs and GRU with 16 SSCs. The SSC features were experimented here based on the literature review that it has good performance on noisy data [29]. But this feature did not show good performance with either Houston toad or Crawfish frog call.

Table 2: Summary of Results for the "Houston" or "Crawfish" Classification Experiment

Classifier	Audio Feature	Training Accuracy	Validation accuracy	Overall Test Accuracy	Test accuracy of Houston	Test accuracy of Crawfish
LSTM	39 MFCCs	84%	82%	76%	80%	71%
GRU	39 MFCCs	80%	79%	73%	77%	69%
LSTM	16 SSCs	69.9%	67.7%	68.5%	71.3%	65.7%
GRU	16 SSCs	66.5%	65%	64.2%	68%	60%

The "Houston" or "Crawfish" or "Environment" Classification Experiment

As previously denoted, a total of 1,000 audio files were used for this experiment. Among the 1,000 audio files, 370 samples contained the Houston toad call, 370 samples contained the Crawfish frog calls, and 260 samples contained only environmental sounds. The duration of each audio file containing the Crawfish frog calls is 60 seconds, each audio file containing the Houston toad call is 1-15 seconds, and each audio file containing environmental sounds is 15 seconds. Like the "Houston" or "Crawfish" classification experiment, each training audio file was fragmented into one-second clips to avoid the dimensional error during training the model and labeled the same as the parent audio files.

Several experiments were performed for the classification of the “Houston” or “Crawfish” or “Environment” using the LSTM and GRU classifiers with 39 MFCCs or 16 SSCs as audio features. The hyperparameters, such as the batch size, epochs, optimizer, and the learning rate were tuned manually to get the best fit. Among these experiments, the LSTM model with 39 MFCCs audio features, 0.0001 for the learning rate, a 32-batch size, 100 epochs, and the ADAM optimizer provided the best training, validation, and test accuracies. Figure 33 shows the training accuracy of this model at 78% and a validation accuracy at 77%. From the confusion matrices in Figure 34, this model achieved a 73.33% test accuracy with a 75% for the Houston toad, a 65% on Crawfish frog call, and an 70% for environmental or background noise. It is noticeable that after adding another class to the model which is the background noise, the false positive rate for “Houston” and “Crawfish” dropped down compare to the binary class classification for “Houston” and “Crawfish” as mostly Crawfish frog sample were misclassified with background noise. From Figure 34, 15% background noise samples were misclassified as Houston toad, 15% background noise samples were misclassified as Crawfish frog.

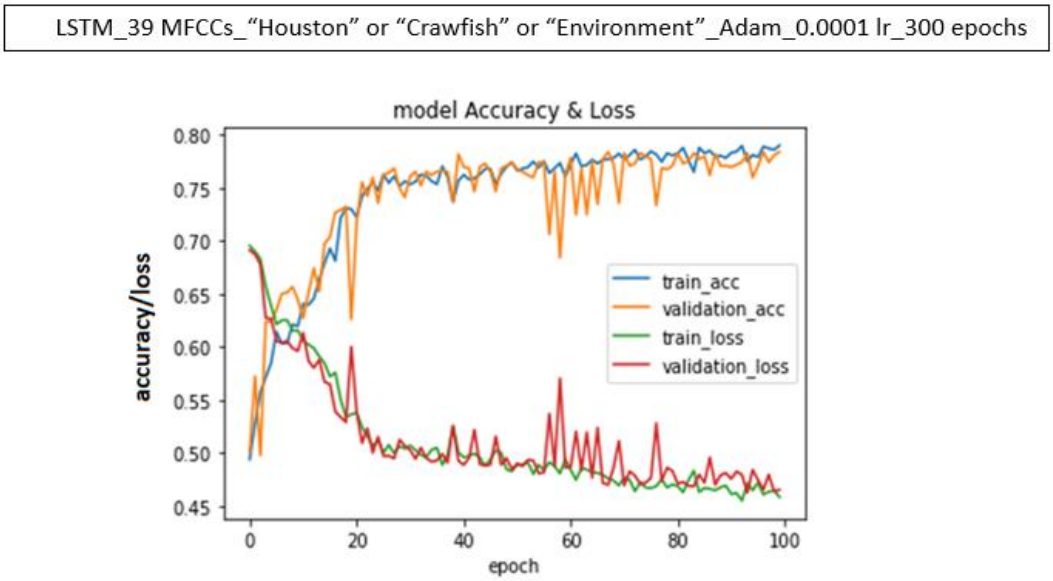

Figure 33: Accuracy/Loss Plots of LSTM with 39 MFCCs for the "Houston" or "Crawfish" or "Environment" Classification Experiment

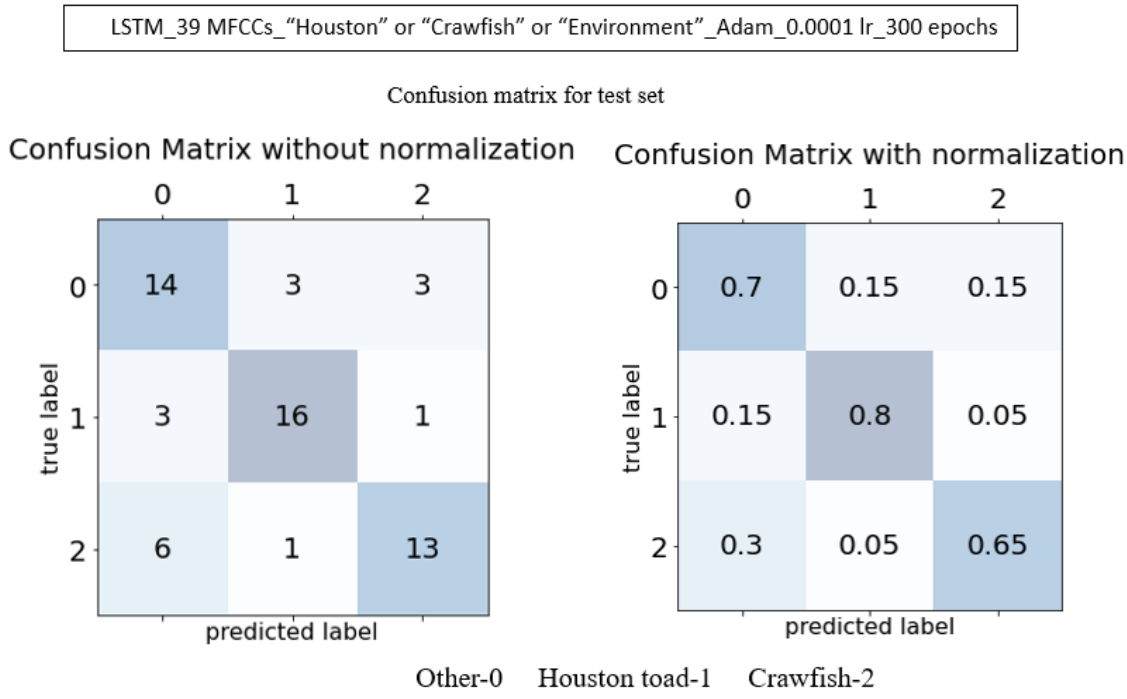

Figure 34: Confusion Matrix of LSTM with 39 MFCCs for "Houston" or "Crawfish" or "Environment" Classification Experiment

Ensemble Learning Implementation

The trained models for the “Houston toad” or “no-toad”, the “Crawfish” or “Houston” and the “Houston” or “Crawfish” or “Environment” classification experiments were saved. The prediction was done using a majority voting mechanism as an ensemble learning modeling technique. In the majority voting mechanism, the input samples are fed to the selected trained models and predictions are made by individual models. The class predicted the most frequently is selected as the final prediction of the system. This makes the predictions more robust or trustworthy.

For the “Houston toad” or “no-toad,” the “Crawfish” or “Houston,” and the “Houston” or “Crawfish” or “Environment” classification experiments, the three best model architectures were selected based on number of epochs. For “Houston toad” or “no-toad,” classification, 30 and 20 epochs were selected for GRU and LSTM model with 39 MFCCs, for “Crawfish” or “Houston,” classification, 300 and 200 epochs were selected for GRU and LSTM model with 39 MFCCs, for “Houston” or “Crawfish” or “Environment” classification, 100 and 80 epochs were selected for GRU and LSTM model with 39 MFCCs as these numbers of epochs provided best results based on the previous discussed experiment results. The ADAM optimizer and 0.0001 for the learning rate were used for all the selected model architectures.

For the “Houston” or “no-toad” classification experiment, the three model architectures were selected for a majority voting mechanism. Once the models have made the predictions, the class that is most common among the three predictions is classified as the final label for the input. Table 3 shows the comparison of results for this classification with and without ensemble method. The test accuracy decreased by 5-6% with the

ensemble model compared to the individual accuracy for each model architecture for "Houston" or "no-toad" classification.

Table 3: Comparison of Results for "Houston" or "No-toad" Classification With and Without Ensemble

			Without Ensemble		With Ensemble	
Classifier	Audio Feature	Epochs	Test Accuracy for "Houston"	Test Accuracy for "no toad"	Test Accuracy for Houston	Test Accuracy for Crawfish
LSTM	39 MFCCs	30	80%	77%	76%	69%
GRU	39 MFCCS	30	73%	70%		
LSTM	39 MFCCs	20	76%	62%		

For the "Crawfish" or "Houston" classification experiment, three model architectures were selected for a majority voting mechanism. Table 4 shows the comparison of results for this classification experiment with and without ensemble method. The results in table 4 show similar scenario as table 3, with the ensemble model having 5-6% less accuracy compared to the accuracy without ensemble.

Table 4: Comparison of Results for "Crawfish" or "Houston" Classification With and Without Ensemble

			Without Ensemble		With Ensemble	
Classifier	Audio Feature	Epochs	Test Accuracy for Houston	Test Accuracy for Crawfish	Test Accuracy for Houston	Test Accuracy for Crawfish
LSTM	39 MFCCs	300	80%	71%	75%	65%
GRU	39 MFCCS	300	77%	69%		
LSTM	39 MFCCs	200	69%	54%		

For the “Crawfish” or “Houston” or “Environment” classification experiment, three model-architectures were again selected for a majority voting mechanism. Table 5 shows the comparison of the results for this classification experiment with or without ensemble method.

Table 5: Comparison of Results for “Crawfish” or “Houston” or “Environment” Classification With and Without Ensemble

			Without Ensemble			With Ensemble		
Classifier	Audio Feature	Epochs	Test Accuracy for Houston	Test Accuracy for Crawfish	Test Accuracy for Environment	Test Accuracy for Houston	Test Accuracy for Crawfish	Test Accuracy for Environment
LSTM	39 MFCCs	100	80%	65%	75%	77%	63%	73%
GRU	39 MFCCS	100	76%	65%	73%			
LSTM	39 MFCCs	80	78%	63%	70%			

From tables 3, 4, and 5, it is observed that most of the architecture’s individual accuracies performed better compared to the ensembled model. The ensembled model has 5-6% less accuracy compare to each architecture accuracy without ensemble.

The ensemble learning method did not provide an improved result due to the improper model selection. For the three classification experiments, among the selected three models, one has good performance, one has average performance and the third one has bad performance and overfitted. So, the three model’s contribution was not good enough to the ensemble model.

Near Real-time Prediction System

A system was designed to test the saved trained model for near real-time prediction of the Houston toad or Crawfish frog call. The Python code was executed in a

laptop computer to perform the test with the different audio files. The sounds containing the Houston toad and Crawfish frog call were played on a mobile device. The Python code recorded the sounds using the "*sounddevice"* Python library. The recorded audio sound was saved in a directory in the "wav" format. The "wav" formatted sound file was imported and pre-processed for extracting audio features. Finally, the extracted audio features were used as inputs for the loaded saved trained model. The model generated predicted output and saved the predication as a text format.

For the "Houston toad" or "no-toad" classification model, three audio files having the Houston toad call and two audio files having background noises were played in front of the trained model. The model predicted all five calls as Houston toad calls with a 60% accuracy classification.

For the "Crawfish" or "Houston" classification model, three Crawfish frog calls and three Houston calls were played. Among the three Crawfish calls, one was predicted correctly, and the model misclassified two as a Houston toad call. Among the three Houston toad calls, one was misclassified as a Crawfish call and two Houston toad calls were predicted correctly.

The class with the highest prediction accuracy, which was "Houston toad" during training, was predicted the most frequently in this test. Also, the audio files were played in a controlled room environment. It has a level of influences on the prediction as the room environment background noises such as air conditioning and noise from vehicles from outside were fully unknown to the trained model.

7. CONCLUSION

The Houston toad is an endangered species, and the Crawfish frog is a near threatened or impending endangered amphibian species, and both require conservation stewardship. As both species vocalize, they can be detected using their calls. An automated detection eliminates the human observer from detection; hence it minimizes failures to detect the species when present due to human error and increases detection robustness. An automated recording device, Toadphone 1, is currently being used for the Houston toad call detection. The existing trained or predictive model of Toadphone 1 is designed only for the Houston toad call detection and has limited success for that taxon. It cannot detect Houston toad with high efficiency and provides false identification confirmations at an unacceptable rate.

This work experimented with several methods or architectures to design a modified and more accurate classification or predictive model for the Toadphone 1, which can classify both the Houston toad and Crawfish frog audio calls. Experiments have been performed in three ways. The first experiment was the classification of "Houston toad" or "Non-toad," the second experiment was the classification of "Houston" or "Crawfish" and the third experiment was the classification of "Houston," "Crawfish," or "Environmental sound". For these experiments, the first task was to preprocess the data using several signal processing techniques, such as filtering to narrow the frequency range, framing to break down the signal and make it more uniform, and applying the Hamming window function to avoid spectral leakage of the framed signal. The audio features were extracted from each frame of the signals. This thesis work utilized MFCCs with their first and second derivative, SSCs and Mel-spectrogram images as the feature extractors.

Deep learning architectures, such as LSTM, GRU, and CNN, were used as the classification algorithms. The LSTM and GRU are advanced or modified RNN architecture, which are suitable for sequential or time-series data analysis like audio samples. Several model architectures have been experimented using combinations of classifier and audio features. For all the classification experiments, the LSTM as classifier with 39 MFCCs audio performed the best in average accuracy testing values. Results from the "Houston" or "Crawfish" and "Houston" or "Crawfish" or "Environmental sound" classification experiments have shown that all architectures evaluated performed for these experiments have a higher success rate to recognize Houston toad call compared to crawfish call. For "Houston" or "Crawfish" classification experiment, the model architecture for LSTM with 39 MFCCs was able to predict 80% of the true Houston call among 35 Houston toad call test samples correctly but this model also misclassified 29% Crawfish call as Houston toad call. So, the accuracy based on true positives and false positives for Houston toad call was 73%. For Crawfish frog call prediction, this model architecture was able to predict 71% of the true Crawfish call among 35 Crawfish call test samples correctly but this model also misclassified 20% Houston toad call as Crawfish call. So, the accuracy based on true positives and false positives for Crawfish call was 78%. This highest accuracy was achieved using LSTM as classifier and 39 MFCCs as audio features. For the "Houston" or "Crawfish" or "Environmental sound" classification experiment, the accuracy for Houston toad based on true positives and false positives was 80% and the accuracy for crawfish frog was 76%. The model architecture with LSTM and 39 MFCCs has shown promising result for Houston toad call predictions with less false positives.

The ensemble learning technique with hard majority voting was experimented with the same input data fed to the selected trained models and predictions were made with different samples. The class that predicted the most was selected as the final prediction. Though the accuracy was decreased by 5-8%, it was an attempt to make the prediction more robust or trustworthy as the final prediction was made from three different architectures.

8. FUTURE WORK

The future development of this work is to implement this experimented predictive model in the field. Here, the results indicate the production of 75%-80% classification accuracy for Houston toad, but for Crawfish frog it varied between 65%-71%, which is comparatively low. This needs to be addressed to enable higher accuracy, particularly for the Crawfish frog. Increasing the number of data samples to train the model could be the right solution. The same length for all audio samples could be chosen for all classes so that the machine learning model should not be biased to any of the classes. The less accuracy of the ensemble model needs to be addressed and the appropriate best models which are not overfitted should be selected for the ensemble method. Another future advancement or improvement of this work is experimenting with other deep learning algorithms such as a combination of LSTM or GRU with CNN, where CNN will be used to generate more robust features from the conventional audio features. The model can be trained by other chorusing amphibian species calls such as Gulf coast toad, and Woodhouse toad to make this model a more generic one.

APPENDIX SECTION

This section includes the result of the experiment which was not included in the thesis.

LSTM and GRU as classifier with 39 MFCCs + 16 SSCs as audio features for the "Crawfish" or "Houston" classification

In this experiment, 39 MFCCs and 16 SSCs audio features were combined, and a total of 55 features were used as input to the LSTM or GRU model. The model architecture of this experiment is shown in Figure 35. This architecture is the same as the model architecture used for the "Crawfish" or "Houston" classification experiment with only 39 MFCCs or only 16 SSCs with LSTM and GRU model except for the input shape.

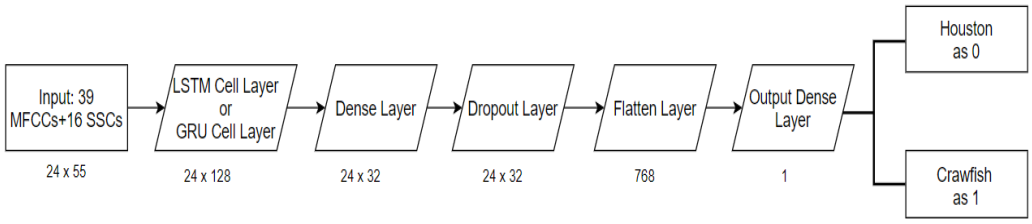

Figure 35: LSTM and GRU Model Structure for the "Houston" or "Crawfish" Classification Experiment with Combined 39 MFCCs and 16 SSCs

Table 6 has listed the training and validation accuracies of combined 39 MFCCs and 16 SSCs (55 features), only 39 MFCCs and only 16 SSCs audio features as input for LSTM or GRU model. From Table 6, combined MFCCs and SSCs features gave a better performance than the experiment with only 16 SSCs features, but it did not show good performance than the experiment with only 39 MFCCs.

Table 6: The Training and Validation Accuracies with Combined 39 MFCCs and 16 SSCs (55 Features), Only 39 MFCCs, Only 16 SSCs Audio Features

ML algorithm	Audio Feature	Training Accuracy	Validation Accuracy
LSTM	39 MFCCs+16 SSC	0.7682	0.757
GRU	39 MFCCs+16 SSC	0.771	0.741
LSTM	39 MFCCs	84%	82%
GRU	39 MFCCs	80%	79%
LSTM	16 SSCs	69.9%	67.7%
GRU	16 SSCs	66.5%	65%

As the combined MFCCs and SSCs audio features have shown poor performance compared to only 39 MFCCs, this method was not furthered experimented with and was not used for ensemble modeling or near real-time prediction system.

REFERENCES

[1] "Crawfish frog," [Online]. Available: https://rareearthtones.org/ringtones/species/Crawfish_frog.html. [Accessed: 15-Sep-2019]

[2] "The Endangered Houstonian: Houston Toad Populations on the Road to Recovery," [Online]. Available: https://www.houstonzoo.org/blog/endangered-houstonian-houston-toad-populations-road-recovery/. [Accessed: 10-Oct-2020]

[3] "*Rana areolata*; Crawfish Frog," Copyright © 2011 Michael Graziano, Missouri, US, Mar 2011, [Online]. Available: https://calphotos.berkeley.edu/cgi/img_query?seq_num=370844&one=T. [Accessed: 10-Oct-2020]

[4] A. A. Bashit, "A Comprehensive Solar Powered Remote Monitoring and Identification of Houston toad Call Automatic Recognizing Device System Design," M.S. thesis, Ingram school of Engineering, Texas State University, San Marcos, Texas, 2019. [Online]. Available: https://digital.library.txstate.edu/ [Accessed: 8-Sep-2019]

[5] 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*, Liberia, 2014, pp. 86-92

[6] M. Rocamora and P. Herrera, "Comparing audio descriptors for singing voice detection in music audio files," in *Brazilian Symposium on Computer Music, 11th.* San Pablo, Brazil, 2007, vol. 26, p. 27

[7] M. Qarachorloo and G. Farahani, "New Features to Improve Speaker Recognition Efficiency with Using LPCC and SSC Features," in *International Journal of Signal Processing Systems,* Vol. 4, No. 4, pp. 295-299, August 2016

[8] M. Zohrer and F. Pernkopf, "Gated Recurrent Networks Applied to Acoustic Scene Classification and Acoustic Event Detection," in *Proc. Detection and Classification of Acoustic Scenes and Events 2016*, Budapest, Hungary, 2016, pp. 115–119

[9] T. H. Vu and J. C. Wang, "Acoustic scene and event recognition using recurrent neural networks," *in Proc. Detection and Classification of Acoustic Scenes and Events 2016,* Budapest, Hungary, 2016

[10]G. Parascandolo, H. Huttunen, and T. Virtanen, "Recurrent neural networks for polyphonic sound event detection in real life recordings," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP),* Shanghai, China, Mar. 2016, pp. 6440–6444

[11]S. Leglaive, R. Hennequin and R. Badeau, “Singing voice detection with deep recurrent neural networks,” in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Brisbane, Australia, 2015, pp. 121-125

[12]K. Ko, S. Park, and H. Ko, Senior Member, “Convolutional Feature Vectors and Support Vector Machine for Animal Sound Classification,” in *2018 40th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, Honolulu, HI, 2018, pp. 376–379

[13]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)*, March 2017, pp. 146–150

[14]L. Zhang; I. Saleh; S. Thapaliya; J. Louie; J. F. Hernandez; H. Ji, “An Empirical Evaluation of ML Approaches for Species Identification through Bioacoustics,” in *2017 International Conference on Computational Science and Computational Intelligence (CSCI)*, pp. 489–494

[15]J. Strout, B. Rogan, S.M. M. Seyednezhad, K. Smart, M. Bush, E. Ribeiro, “Anuran call classification with deep learning”, in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, New Orleans, LA, 2017, pp. 2662-2665

[16]J. Colonna, T. Peet, C. A. Ferreira, A. M. Jorge, E. F. Gomes, and J. Gama, “Automatic classification of anuran sounds using convolutional neural networks,” in *Proc. International C* Conference on Computer Science & Software Engineering*, Porto, Portugal, pp. 73-78, July 2016

[17]“Sequence Models & Recurrent Neural Networks (RNNs) | by Santhoopa Jayawardhana,” [Online]. Available: https://towardsdatascience.com/sequence-models-and-recurrent-neural-networks-rnns-62cadeb4f1e1. [Accessed: 10-Oct-2019]

[18]“Recurrent Neural Network-Artificial Intelligence,” [Online]. Available: https://leonardoaraujosantos.gitbooks.io/artificial-inteligence/content/recurrent_neural_networks.html.[Accessed: 10-Oct-2019]

[19]“LSTM Networks for Sentiment Analysis,” [online]. Available: http://deeplearning.net/tutorial/lstm.html. [Accessed: 10-Oct-2018]

[20]G. Parascandolo, H. Huttunen, and T. Virtanen, “Recurrent neural networks for polyphonic sound event detection in real life recordings,” in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Shanghai, China, Mar. 2016, pp. 6440–6444

[21]Y. Tang, Y. Huang, Z. Wu, H. Meng, M. Xu and L. Cai, "Question detection from acoustic features using recurrent neural network with gated recurrent unit," *in 2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Shanghai, 2016, pp. 6125-6129

[22]L. Zhen & Y. Yizhou, "Protein Secondary Structure Prediction Using Cascaded Convolutional and Recurrent Neural Networks," in *2016 International Joint Conference on Artificial Intelligence (IJCAI)*, July 2016, New York City, pp. 2560–2567

[23]"How to Develop Voting Ensembles With Python", [Online]. Available: https://machinelearningmastery.com/voting-ensembles-with-python/ [Accessed: 5-Sep-2020]

[24]"Window function-Wikipedia", [Online]. Available: http://en.wikipedia.org/wiki/Window_function. [Accessed: 25-Sep-2019]

[25]"Librosa", [Online]. Available: https://librosa.github.io/librosa/ [Accessed: 20-Sep-2019]

[26]"Welcome to python_speech_features's documentation!" [Online]. Available: https://Python-speech-features.readthedocs.io/en/latest/ [Accessed: 10-Sep-2019]

[27]"HTK Speech Recognition Toolkit", [Online]. Available: http://htk.eng.cam.ac.uk/ [Accessed: 20-Sep-2019]

[28]"Speech Processing for Machine Learning: Filter banks, Mel Frequency Cepstral Coefficients (MFCCs) and What's In-Between." [Online]. Available: https://haythamfayek.com/2016/04/21/speech-processing-for-machine-learning.html. [Accessed: 10-Sep-2019]

[29]T. Kinnunen, B. Zhang, J. Zhu, and Y. Wang, "Speaker verification with adaptive spectral sub-band centroids," in *International Conference on Biometrics*, August 2007, Springer, Berlin, Heidelberg, vol. 4642 LNCS, pp. 58– 66

[30]D. Hosseinzadeh and S. Krishnan, "Combining Vocal Source and MFCC Features for Enhanced Speaker Recognition Performance Using GMMs," in *2007 IEEE 9th Workshop on Multimedia Signal Processing*, Crete, 2007, pp. 365-368

[31]I. Sutskever, J. Martens, G. Dahl, and G. Hinton, "On the importance of initialization and momentum in deep learning," in *2013 30th International Conference on Machine Learning*, February 2013, pp. 1139–1147

[32]D. P. Kingma and J. Ba, "Adam: A method for stochastic Optimization," in *3rd International Conference for Learning Representations*, San Diego, 2015

[33]“LEAP - High Performance Computing Cluster,” [Online]. Available: https://doit.txstate.edu/rc/leap.html . [Accessed: 10-Aug-2020]