BuzzNet: A Siamese CNN-SVM model for Mosquito Species Identification using Audio Signals

CMSC 199.2 - Research in Computer Science II

Jarred Antonii Afable Acedillo
2019-02304
B.S.C.S.

Presented to the Faculty of the
Division of Natural Sciences and Mathematics

In Partial Fulfillment of the Requirements
For the Degree of
Bachelor of Science in Computer Science

University of the Philippines
TACLOBAN COLLEGE
Tacloban City

June 2023

This research problem, entitled "**BUZZNET: A SIAMESE CNN-SVM MODEL FOR MOSQUITO SPECIES IDENTIFICATION USING AUDIO SIGNALS**", prepared and submitted by **JARRED ANTONII AFABLE ACEDILLO**, in partial fulfillment of the requirements for the degree of **BACHELOR OF SCIENCE IN COMPUTER SCIENCE** is hereby accepted.

Research Adviser:

DR. JOHN PAUL T. YUSIONG

Panel Members:

DR. TECHN. JASMINE A. MALINAO

MS. BEA D. SANTIAGO

Accepted as partial fulfillment of the requirements for the degree of **BACHELOR OF SCIENCE IN COMPUTER SCIENCE**.

DR. JOHN PAUL T. YUSIONG
Chair, DNSM

Acknowledgements

I would first like to acknowledge the assistance and guidance received from my academic advisor, Dr. John Paul Yusiong, whose expertise, patience, and feedback have been invaluable throughout my research process.

Furthermore, I extend my heartfelt appreciation to my colleagues, Ben Julian M. Merales and Carl Joseph P. Mate, my friends, and especially my family, who have provided valuable insights and never-ending support during the development of this research.

Lastly, I am grateful to the school, University of the Philippines Tacloban College, for providing the necessary resources and tools to carry out this research until its completion. Their support has enabled the successful completion of this work.

I am deeply grateful to all those mentioned above and anyone else who has contributed to this manuscript in any way. Your dedication and assistance have played a significant role in shaping the outcome of this research.

Abstract

Mosquitoes are flying insects that are present in almost every part of the world. They are infamous for carrying with them deadly diseases including malaria and dengue. Mosquito species classification can prove to be a crucial method for aiding vector control programs. To contribute to this field of study, this study proposes BuzzNet, a web-based Siamese system that utilizes a hybrid model comprised of a Convolutional Neural Network (CNN) and a Support Vector Machine (SVM) capable of classifying mosquito species through audio signals. The aim of this study is to deploy the Siamese CNN-SVM as a web application that generates two spectrogram images from an audio file and uses those spectrogram images as input in order to classify the mosquito species. This was achieved by introducing a Siamese CNN-SVM model, where the CNN model was based on pre-trained Siamese CNN models, and a Support Vector Machine was used to classify the mosquito species. This model was used as an alternative method for achieving competitive results in the mosquito classification task. In order to determine the effectiveness of the different components and strategies employed on the model to improve performance, an ablation study was conducted. This is followed by conducting state-of-the-art analysis by comparing the performance of the proposed model with other existing methods. In this study, the mosquito wingbeat sounds were converted into three different spectrogram types namely Log-mel, MFCC, and PCEN spectrograms. To create the hybrid model, a pre-trained MosquitoNet model was utilized for feature extraction, while the SVM was used for the classification task. In order to improve model performance, a hyperparameter tuning method, RandomizedSearchCV, was implemented. The performance

of Siamese CNN-SVM models was compared using different spectrogram pairs. The results obtained suggest that along with the hyperparameter tuning method, RandomizedSearchCV, the Siamese CNN-SVM model was able to achieve the highest performance by utilizing the Log-mel and MFCC spectrogram pair as input, yielding an accuracy of 90.94% and a macro-f1 score of 77.56%. The proposed method was able to outperform the single-input CNN model, as well as a state-of-the-art model, MozzBNNv2, in terms of ROC AUC and PR AUC. The proposed model offered a 5.8% improvement on the ROC AUC and a 21.7% improvement on the PR AUC over the MozzBNNv2. The best performing model was deployed as a web application, which offers a more convenient method for classifying mosquito species.

Keywords - Siamese CNN-SVM, VGG-19, SVM, Hybrid model, Transfer Learning, Hyperparameter tuning, RandomizedSearchCV, Web-based Mosquito Species Classification, HumBugDB, Audio Classification, Spectrogram, Log-mel, MFCC, PCEN, Accuracy, Macro-f1, ROC AUC, PR AUC

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

Mosquito-borne diseases have historically been a major cause of worry for public health, with malaria and dengue being the most serious and prevalent. The World Health Organization [16] estimates that malaria causes hundreds of thousands of fatalities each year. However, intervention measures against these vectors remain relatively poor, thus emphasizing the need for better and more efficient intervention measures.

The bite of an Anopheles mosquito carrying the malaria parasite causes transmission of the disease. With the approximate number of 3,600 mosquito species, there are only about 450 Anopheles species, and among those 450 Anopheles species, only roughly 60 of them transmit malaria, according to Neafsey et al. [15]. It is important to have complete and updated data on mosquito species so as to tackle these vector-carrying species effectively, thus low cost and effective methods with the ability to quickly and accurately identify these mosquitoes are crucial for control programs and intervention strategies.

The idea of using animal sounds to be able to identify their respective species have seen a rise in interest, particularly in the field of identifying mosquito species

through their sounds, specifically their wingbeats, has been around for around half a century and this steady increase in interest is due to the worldwide spread of these mosquito-borne diseases [17]. There have even been multiple researches that have used audio data to classify animal species such as in the works of Xu et al. [23] and Zhang et al. [25], but there are insufficient studies related to mosquitoes.

In this study, the main focus would be on classifying mosquito species through their wingbeat sounds. The audio data to be used will be taken from the publicly available dataset, the HumBugDB dataset. This study aims to develop a hybrid model using Convolutional Neural Networks (CNN) and Support Vector Machines (SVM), along with a Siamese Neural Network approach. The model will utilize a pre-trained Siamese CNN model from the MosquitoNet research for feature extraction as well as a Support Vector Machine for classification, along with hyperparameter tuning to improve model performance. The model will then be deployed as a web application as a way to benefit the general public.

1.1 Background of the Study

Mosquitoes and Wingbeats

Mosquitoes are one of the most common, flying insects that we encounter in our lives. They live in most parts of the world and have around 3,600 species. Thousands of these mosquito species bite and feast on the blood of their hosts, which include but are not limited to mammals, reptiles, and birds. These mosquito bites are the primary way of transmitting disease-carrying pathogens such as malaria, although only specific species can transmit them. Of the 3,600 mosquito species, only the Anopheles species is known for transmitting malaria and among the 450 Anopheles

species, only about 60 of them transmit malaria.

According to Raman et al. [19], mosquito flight tones are characteristics of a species and different species have different flight tones. This means that mosquito wingbeat sounds are distinctive and can be used to identify their species through the use of wave analysis methods.

Acoustic Sound Classification

Through the transmission of diseases such as malaria, mosquitoes have infected billions of people and even killed millions every year [24]. This situation led to the rise in interest to improve vector control measures which paved the way for mosquito species identification, in particular, those that utilize acoustic sound classification. In its essence, acoustic sound classification is the process of listening to and evaluating audio recordings. This allows for information retrieval which then leads to multiple applications such as content analysis and monitoring systems.

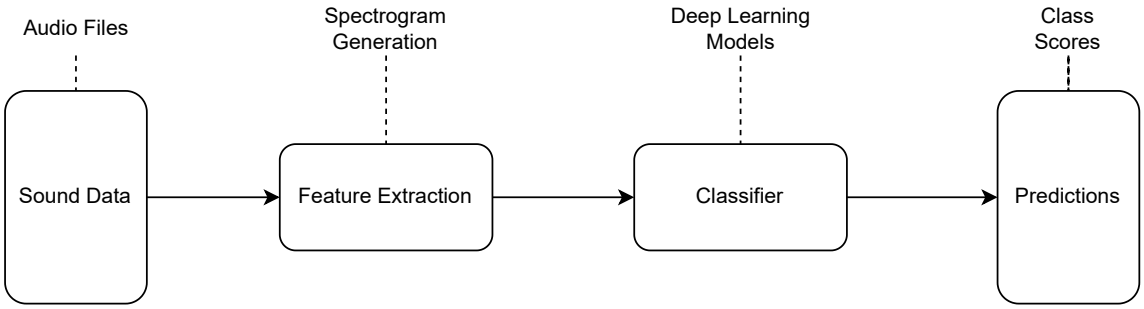

Figure 1.1: Acoustic Sound Classification

Figure 1.1 above shows an example of a simplified diagram of acoustic sound classification for mosquitoes. The works of Offenhauser et al. [17] began the study of utilizing the sounds of mosquitoes, specifically their wingbeat sounds, in order to identify disease-carrying species, and since then, the field of study has only grown

wider with recent works such as Fanioudakis et al. [6], and Khandelwal et al. [8], among others. This study will follow suit with this method of classification with phases such as audio preparation, converting audio files into spectrograms, and then feeding converted spectrograms into a deep learning model for classification.

Audio Files and Spectrograms

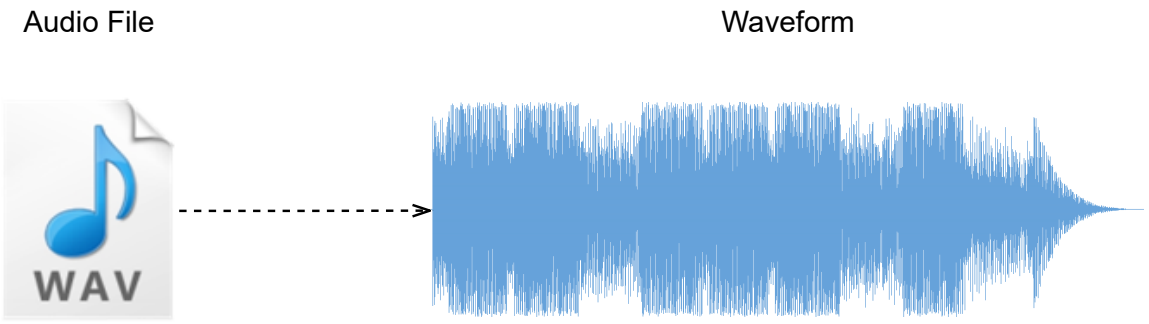

Figure 1.2: Sample Audio File and Waveform

As shown on Figure 1.2, audio files contain audio data and are necessary for audio classification. They are often compiled into datasets with matching types which can then be utilized for studies. One such dataset that this study will utilize is the HumBugDB dataset from the works of Kiskin et al. [9], a publicly available dataset with an extensive multi-species collection acoustic recordings of mosquitoes tracked continuously in free flight.

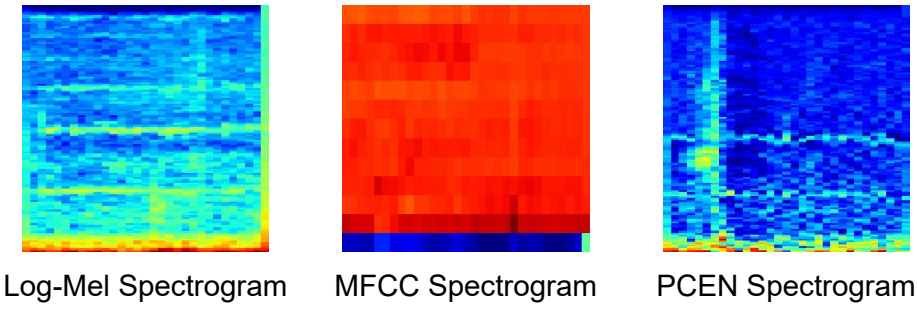

Figure 1.3: Sample Spectrograms of Different Types

Raw audio files alone, however, are not sufficient to be able to classify mosquito species. The raw audio files are to be converted first into their time-frequency representation, spectrograms, as shown in Figure 1.3, to extract useful features. They are often obtained by applying the discrete-time Fourier transform. Spectrograms are capable of capturing the essential features of the audio and are thus most suited to be the input audio data for a deep learning model. As shown in the work of Wang et al. [22], different types of time-frequency representations offer different impact to an algorithm's performance. Therefore, this study will utilize three different spectrograms and assess their respective performances and these are the Log-Mel, Mel-Frequency Cepstral Coefficients (MFCC), and the Per-Channel Energy Normalization (PCEN) spectrograms.

Deep Learning and Convolutional Neural Networks

Deep learning is a branch of machine learning that deals with algorithms and are modeled after the structure and operation of the brain. A hierarchy of deep learning algorithms with varying levels of complexity and abstraction is used where each algorithm performs a nonlinear transformation on its input and outputs a statistical model using what it has learned. Iterations then proceed until the output accuracy is acceptable. Deep learning has grown in favor over the years due to its breakthroughs in image, speech, and audio processing [11], as well as its scalability, where the more data you feed, the better the performance gets.

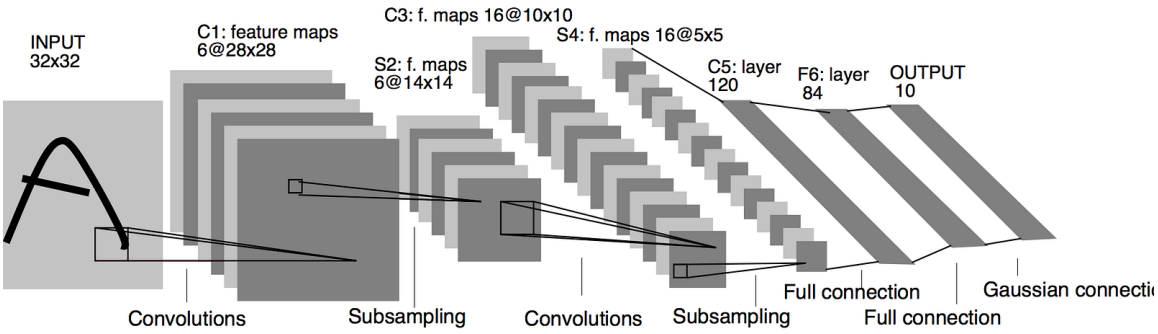

Figure 1.4: The LeNet Architecture

Meanwhile, Convolutional Neural Networks (CNN) are one of the most widely used deep neural networks. Convolution is a mathematical linear action between matrices that gave rise to the name. There are numerous layers in CNNs, including convolutional layer, fully connected layer, nonlinear layer, and pooling layer. CNNs has shown impressive results over the past decade in a plethora of fields as well as serving as a powerful tool for deep learning for a lot of applications such as image processing, and face and voice recognition [2]. Some famous examples of CNN architectures are LeNet and AlexNet, both of which have had significant contributions to the body of knowledge; LeNet contributed to digit recognition, and AlexNet being the first to display how effective deep learning could be in computer vision tasks. Figure 1.4 shows the LeNet architecture. This study will also utilize a CNN, specifically a pre-trained Siamese CNN model from the MosquitoNet research for feature extraction.

Siamese Networks

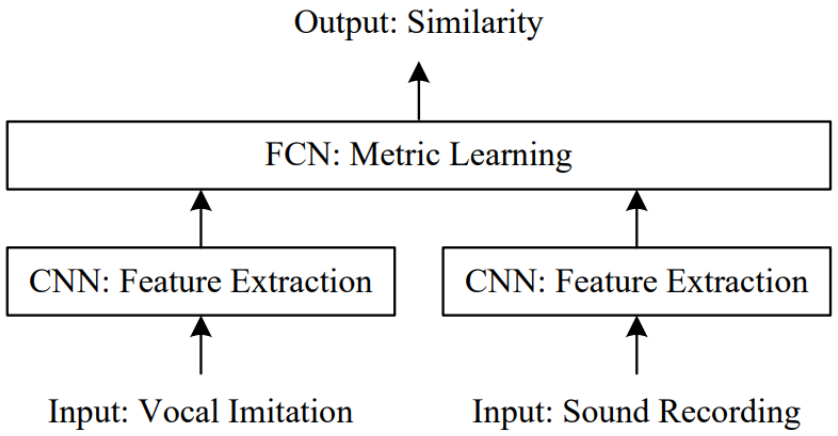

Figure 1.5: Siamese Network Architecture introduced by Zhang et al.

A Siamese Neural Network is a type of artificial neural network that employs the same weights to compute equivalent output vectors from two distinct input vectors simultaneously. A precomputed version of one of the output vectors frequently serves as a benchmark for comparison with the other output vector. There have been some works incorporating a Siamese Neural Network in face verification such as in the work of Taigman et al. [21] as well as for mosquito larva classification such as in the work of Sanchez et al. [20]. Another study that utilizes a Siamese neural network would be in the works of Zhang et al. [26], as shown in Figure 1.5 where they addressed sound search through vocal imitation. There are also works that utilize a Siamese neural network in order to address the detection, classification, and counting of blue whale calls, as done in the study of Zhong et al. [26]

Support Vector Machines

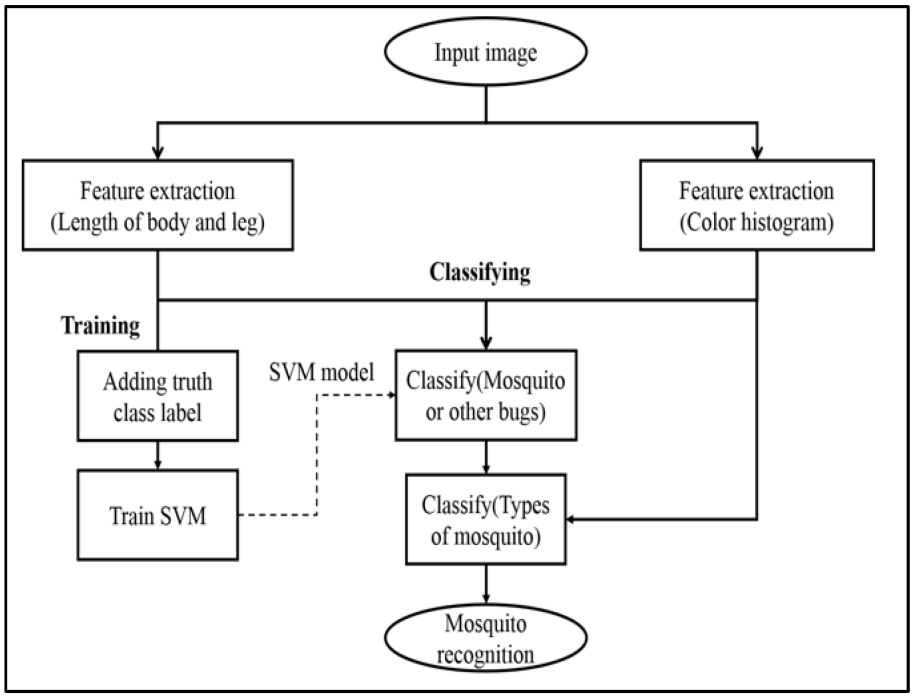

Figure 1.6: Mosquito Classification Diagram using SVM

Support Vector Machines are a widely known type of deep learning algorithm that carries out supervised learning for data group classification or regression. They are often used as alternatives to softmax for classification [3]. Compared to neural networks, SVMs boasts higher speed and better performance given a limited number of samples, around the thousands, which makes this algorithm very suitable for text classification problems. Figure 1.6 shows the architecture diagram adapted in the study of Lukman et al. [14], which also utilizes SVMs for the classification problem, particularly in classifying mosquito species. This study will utilize a Support Vector Machine for its classification using the extracted features from the pre-trained Siamese CNN model.

Hybrid Machine Learning Models

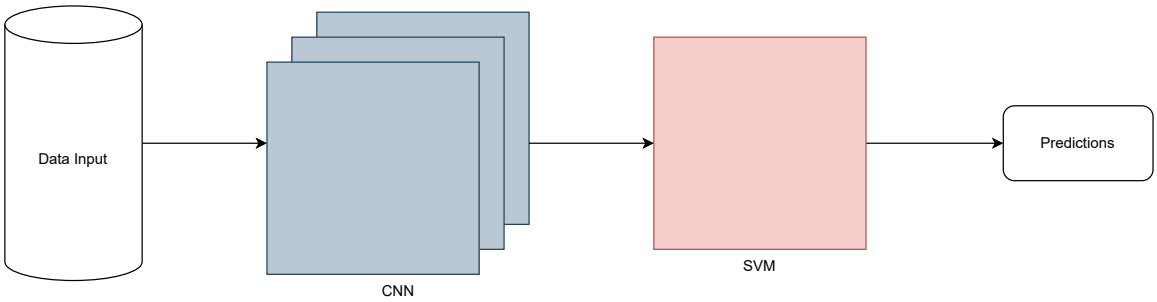

Figure 1.7: Simple Hybrid Machine Learning Architecture

Hybrid machine learning models are models that are created by combining two or more machine learning algorithms to solve a specific problem. This hybrid architecture aims to make the most out of the advantages of each model to output results that beat the performance of the individual models. Figure 1.7 depicts a diagram of a simple hybrid machine learning architecture. Examples of these are in the works of Duan et al. [5] where they proposed a hybrid deep learning CNN-ELM model for classifying age and gender. They utilized CNN for extracting features from the input images while using ELM for classifying the results. Another work would be in the works of Ahlawat et al. [1] where they aimed to create a hybrid model capable of recognizing handwritten digits from the MNIST dataset. Their model used a CNN as for extracting features automatically, and an SVM as a binary classifier. Both works presented good results and exhibits the viability of using a hybrid model for the classification problem.

RandomizedSearchCV

RandomizedSearchCV is a technique where the performance of a machine learning model is assessed using a range of hyperparameter values. It is similar to another

commonly used hyperparameter tuning approach called GridSearchCV [10]. RandomizedSearchCV is a hyperparameter tuning technique used to improve model performance, where it explores various combinations of values within a defined range given a specified number of iterations to be done. It is effective in examining a broad spectrum of values and often quickly discovers highly favorable combinations. When comparing GridSearchCV and RandomizedSearchCV, due to the exhaustive exploration of all intermediate hyperparameter combinations, GridSearchCV incurs significant computational costs. RandomizedSearchCV addresses the limitations of GridSearchCV by examining a predetermined number of hyperparameter settings instead of exploring all possible combinations. It navigates the parameter grid randomly to identify the optimal hyperparameter set, which reduces unnecessary computations. Additionally, RandomizedSearchCV considers the distribution of values when selecting hyperparameters. The study of Pravallika et al. [18] explores the effects of using RandomizedSearchCV on a Random Forest classifier and found that their multi-class classification model had gained an improvement in performance of up to 0.57%.

1.2 Problem Statement

Mosquitoes are one of the most dangerous animals in the planet, causing millions of deaths every year. Due to the recent climatic changes to our planet such as global warming, the climatic suitability of mosquitoes has increased leading to numerous people being at risk of malaria and dengue. According to the World Health Organization, mosquito bites causes millions of deaths each year, with an estimated number of 627,000 deaths due to malaria in 2020. There have been numerous frequently advised methods of intervention such as mosquito repellents as well as clearing stagnant

water, however there are better alternatives. Having complete and up-to-date data on mosquitoes is essential to understanding how to better prevent mosquito-borne diseases. Therefore, detecting and identifying pathogen-carrying mosquito species for gaining necessary information can prove to be useful.

As such, the use of innovative approaches to mosquito identification at scale such as a low-cost, accurate, and effective deep learning approach is a viable way in aiding vector control and intervention. While there already exists some models that deal with mosquito species identification, these models usually come in the form of accepting only a single spectrogram image as its input to deep learning models. In contrast to the single spectrogram image input approach, this study proposes an alternative method for addressing the mosquito species classification problem which is to use two spectrogram images as input, rather than just one. Through this approach, it allows the proposed model to provide better features for discriminating between different classes by investigating two different spectrograms for each audio source.

In addition, the current body of knowledge is still lacking in terms of having researches that utilizes hybrid models specifically for mosquito species identification through audio signals. While there are studies that utilize both CNNs and SVMs for mosquito species identification using audio signals, they mostly come in the form of being separate models. There are very limited works when it comes to using hybrid models and are mostly used for facial emotion and speech recognition, rather than mosquito species identification.

Therefore, the goal of this study is to address these research gaps of having a deep neural network that accepts multi-spectrogram input in contrast to the usual single spectrogram input from other existing studies, as well as utilizing a hybrid model for

species identification. This study aims to create a hybrid deep learning model that will utilize a Siamese network approach for identifying mosquito species using audio signals that is capable of accepting two spectrogram images as input and implement it as a web application. This multi-spectrogram input approach can serve as a viable method for mosquito species classification since different types of spectrograms offer different impacts to an algorithm's performance [22], and therefore could potentially aid the model in discriminating among the different classes better and produce more accurate predictions.

1.3 Aim of the Work

General Objective

1. Develop and deploy a Siamese CNN-SVM model for the identification of mosquito species using two spectrogram images from audio signals.

Specific Objectives

1. Introduce a Siamese CNN-SVM network as an alternative model to generate comparable prediction accuracy for mosquito species identification using audio signals and analyze the combination of different spectrogram inputs and determine how it affects the model's performance.

2. Perform an ablation study to determine the effectiveness of the different strategies for improving model performance.

3. Perform state-of-the-art analysis on the Siamese CNN-SVM model to compare and evaluate the accuracy of the proposed model with the best known methods identified to date.

4. Deploy the Siamese CNN-SVM model as a web application for mosquito species identification.

Scope and Limitations

This study will be utilizing a hybrid approach using a pre-trained MosquitoNet model, trained on the HumBugDB dataset, for feature extraction as well as a Support Vector Machine (SVM) for classification, along with a Siamese neural network approach using various strategies such as audio data augmentation, and fine-tuning, among others. This research will utilize the publicly available dataset, the HumBugDB dataset, but not the entire dataset. This study will only address the classification of specific mosquito species with the most populated species by recording, following the approach of Kiskin et al. [9]. These species are *Ae. aegypti*, *An. arabiensis*, *An. coustani*, *An. funestus ss*, *An. squamosus*, *Culex pipiens complex*, *Ma. africanus*, and *Ma. uniformis*, as well as some other non-mosquito noises labeled as "audio" and "background". The deep learning model will also be deployed as a web application.

Significance of the Study

Studies utilizing deep learning models to identify and detect mosquito species using the HumBugDB dataset are currently very limited, let alone those that employ a hybrid model. There is also yet to be a web-based mosquito species identification system, specifically based on the Siamese CNN-SVM model, that takes advantage of deep learning. Utilizing a Siamese approach and CNNs with an SVM algorithm as a classifier can improve accuracy over other existing approaches, an enhanced deep learning model can prove to be an effective and efficient method of preventing mosquito-borne

diseases through the use of a web-based mosquito species identification system.

1.4 Theoretical and Conceptual Framework

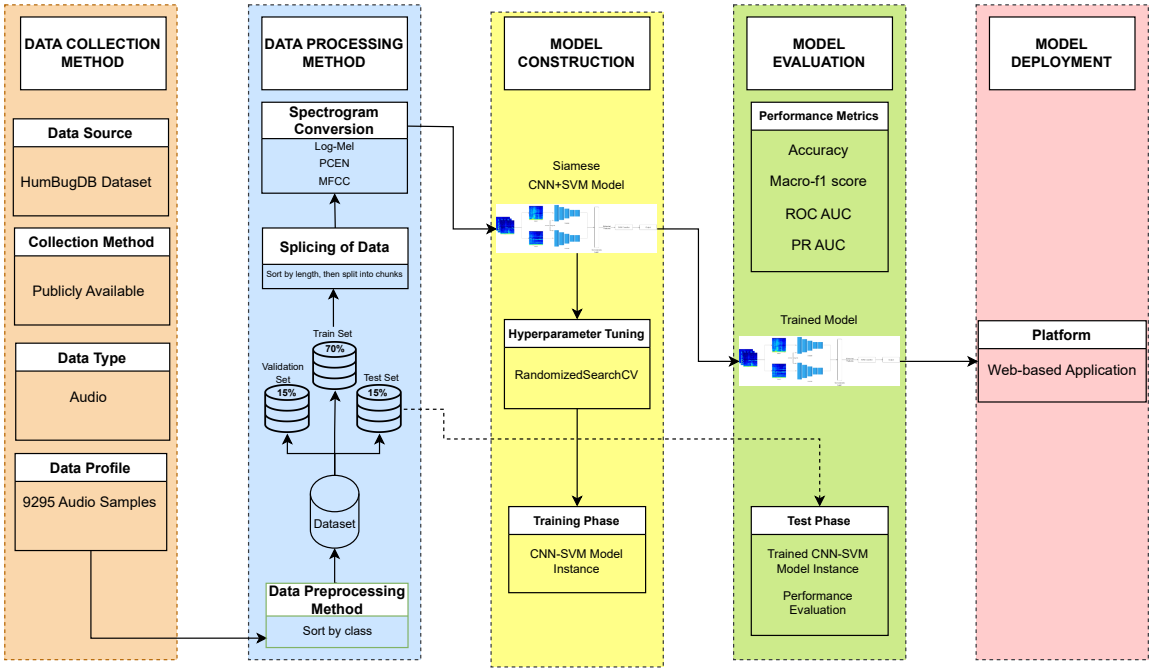

Figure 1.8: Theoretical and Conceptual Framework

Figure 1.8 shows the theoretical and conceptual framework that this study will use. For the data collection method, this research will be utilizing the publicly available audio dataset, the HumBugDB dataset on the Zenodo repository. In total, the dataset contains 9,295 audio samples, which include mosquito wingbeat sounds from different species, background noise, and audio, which refers to other non-mosquito related noise. It includes 71,286 seconds (20 hours) of labelled mosquito data with 53,227 seconds (15 hours) of corresponding background noise, all recorded globally at different sites from 8 experiments. It includes a broad range of both indoor and outdoor background environments taken from different countries such as USA, UK,

Kenya, Thailand, and Tanzania. Of these samples, 64,843 seconds contain metadata of their species, which consists of a total of 36 different species or species complexes.

As for the data processing method, the first step is to sort the raw audio files according to their classes. The audio files are then to be sorted by length from longest to shortest, with length referring to the duration of each audio file. Afterwards, the audio files will then be split into 3 different sets namely the train set, the validation set, and the test set, with a ratio of 70:15:15, respectively. From here, the audio files will then be spliced into chunks of 1.92 seconds, following the HumBugDB author's approach [9]. Any chunks that are less than the specified length are to be discarded. Then these spliced chunks will then be converted into 3 different types of spectrograms namely Log-based Mel (Log-Mel) spectrogram, the Mel-Frequency Cepstral Coefficient (MFCC) spectrogram, and the Per-Channel Energy Normalization (PCEN) spectrogram. These spectrograms will also be normalized.

Proceeding to the next step, we have the model construction, and in here the model will utilize a Siamese network approach and adapt a hybrid approach utilizing a Convolutional Neural Network (CNN) along with a Support Vector Machine (SVM), where it will take two spectrogram images as input and output the resulting class. The deep learning model to be utilized will be a hybrid model, where a pre-trained MosquitoNet model will be used for feature extraction, and the features extracted from the MosquitoNet model will be fed into the SVM classifier for the classification of the mosquito species, instead of using the softmax layer of the MosquitoNet model.

In order to improve the model's performance, a hyperparameter tuning method will be implemented. The RandomizedSearchCV will be used as the hypertuning method which will determine the optimal set of parameters for the model to yield the

best results.

For the model evaluation, the implemented model will be trained, tested, and evaluated using the three aforementioned sets. The training and validation sets will be used during the training phase and the test set will be used during the evaluation phase. Evaluation metrics such as accuracy, macro-f1 score, ROC AUC, and PR AUC, will be used to evaluate the model performance.

Finally we have the deployment phase where the best performing model will be deployed as a web application that will utilize HTML and CSS as its front-end while also utilizing the latest version of Flask as its back-end. The creation of the models will utilize libraries such as TensorFlow, Keras, NumPy, scikit-learn, matplotlib, among others, in Python, while Flask will be utilized to deploy the best performing model. The creation of the web application will utilize HTML and CSS, which can then be used in a number of platforms such as Google Chrome, Microsoft Edge, Firefox, Opera, Safari, and others.

Chapter 2

Review of Related Literature

There have been multiple works regarding mosquito species identification from audio signals. However, they vary in terms of their type. There are image-based classification, audio-based classification, and even video-based classification. With this study, we are more concerned with audio-based classification, where the methods are mainly divided into different phases which include splitting a dataset into different sets usually the train set, the validation set, and the test set, the conversion of audio signals into spectrograms, and the feeding of the data into a CNN.

One such work concerned with audio-based mosquito species classification would be in the study of Khandelwal et al. [8]. Their study aimed to present a novel method for improving the efficiency of audio machine learning approaches by incorporating pre-processing methods into a deep learning model. They utilized the publicly available dataset, the HumBugDB dataset, consisting of over 20 hours of mosquito flight recordings. They used two different models namely a Shifted Windows (Swin) architecture as well as a deep learning CNN model (ConvNeXt) and combined it with data pre-processing techniques including Per-Channel Energy Normalization (PCEN) as a trainable front-end, external data augmentation, trainable kernels, and random

masking. As for their method of verifying the significance of the pre-processing methods, they conducted an ablation study where they removed each component in their pipeline one at a time and retained the model. This aided them in quantifying their results. They found that among the different components, the "random masking" on spectrograms and "PCEN" are the most significant in their pipeline, whereas "random masking" prevented overfitting. Overall, their method was able to outperform the published baseline by around 212% on the unpublished test set from the ACM'22 challenge.

In a similar fashion, the work of Fanioudakis et al. [6] is also concerned with mosquito species classification. However, unlike the previous work where they utilized a publicly available dataset, in this work, they utilized their own recorded dataset, which took place in Biogents in Regensburg, Germany. Their dataset consisted of 279,566 flight recordings with 6 species of mosquitoes and was split into two different sets, the train and test sets with a ratio of 80:20, respectively. They utilized deep learning approaches such as DenseNet121, MobileNet, and Xception, as well as shallow learning approaches such as XGBoost and LightGBM. In their experiments, they utilized different means of acquiring features such as "Raw Samples" which indicates that the time domain recording was used without feature extraction, "PSD" that stands for a 129-vector dimension vector which corresponded to the log-power of the frequencies of the different recordings, and finally "Spectrogram" which is a time-frequency domain representation. Notice that the first two are 1D signals while the Spectrogram is a 2D signal. Overall they were able to achieve satisfactory accuracies in both their deep learning approaches and shallow approaches, with accuracies ranging from 91% to 96% for the deep learning approaches, and accuracies of 81%

and 82% for the two shallow learning approaches.

Aside from Convolutional Neural Networks (CNNs), there are also other alternatives models that can be used for mosquito species classification. One such alternative is the use of Support Vector Machines (SVMs). SVMs are known for their relatively high speeds and performance on a limited number of samples, making them suitable for text classification problems. That being said, they are also proven to be a viable method for the mosquito species classification problem.

A study that presents this is in the works of Lukman et al. [14], where they aimed to create a method of classifying mosquito species using SVM. They sought to improve the SVM performance by using different kernel functions instead of using back-propagation from previous research. Their SVM model takes in features using MFCC from mosquito noise and outputs the classification. These features are extracted through a series of steps which includes applying pre-emphasis, frame-blocking, windowing, then changing each frame into a frequency domain through Fast Fourier Transform, obtaining the Mel-filter bank, and lastly getting the logarithmic scale frequency using the bank. In their classification experiments, they utilized different kernels including linear, polynomial, RBF, and sigmoid. From their results, their model was able to outperform back-propagation by using SVM with RBF kernel by a significant margin. The SVM with an RBF kernel obtained an accuracy of 75.55% as compared to the 72.56% of back-propagation. The study of Hagiwara et al. [7] also used SVMs for classifying mosquito species, where they utilized the HumBugDB dataset as their species classification dataset. In their experiments, among the non-DL based models that they used, SVM was the model that outputted good performance, beating out other neural networks in the classification tasks, resulting

in an accuracy of 77.9%.

Aside from having different models for audio data classification, there are also studies that utilize a different architecture entirely. One such example would be the use of a Siamese network. Essentially, Siamese networks are a type of neural network architecture comprising of two or more equivalent sub-networks that are often employed for tasks requiring the comparison of various input samples.

The study of Zhong et al. [26] showcases the use of a Siamese neural network. Their study aimed to address the detection, the classification, as well as the counting of blue whale calls. The use of a Siamese neural network was suggested as an alternative to the popular CNN to carry out classifications, particularly when the number of training data is constrained. In the deeper layer, the Siamese neural network focuses on discovering embeddings that group similar classes together,resulting in a more efficient way of learning semantic similarities. In their experiments, they utilized a CNN, specifically a DenseNet-201 architecture which serves as their baseline for classifying blue whale calls as well as counting the number of calls. An image triplet is fed into their model as a single sample, where their Siamese architecture enables their networks to learn from very little data. In their results, their Siamese neural network achieved better performance compared to CNN across the board, reaching higher scores in accuracy, sensitivity, and specificity, in all four populations of blue whales. Their model was able to achieve an average of 92.2% accuracy, 92.1% sensitivity, and 92.2% specificity, across all four populations. While this paper does not explicitly tackle mosquito species classification, this does still fall in line with the audio classification problem, proving that the Siamese neural network approach can be viable alternative for classifying mosquito species.

While the aforementioned studies were able to successfully create a model capable of classifying mosquito species, they were unable to deploy their models into anything tangible. However, in the works of Li et al. [12], they were able to produce an android app called "MozzWear" capable of identifying mosquito species using audio signals. In this study, they also utilized the publicly available dataset, the HumBugDB dataset. For their feature extraction methods, they utilized one of the more well-known and widely used spectrogram for extracting audio features, which is the Mel-Frequency Cepstral Coefficients (MFCC), since it led to the best detection accuracy. As for their detection and classification method, the researches utilized a two-stage classification paradigm for detecting mosquitoes where a binary class support vector machine (SVM) was utilized in the inital step to determine whether mosquitoes were present. In the second stage, a multi-species classification employing a one-versus-one multi-class method along with the SVM was performed in order to determine the precise species of mosquito. Throughout their training, the researchers explored numerous training strategies in order to achieve the most effective classification result. These included dividing the recordings into 0.1 second audio clips along with assigning labels to the clips. Random sampling was also performed and was later determined to generate the best performance. They were able to produce satisfactory accuracies ranging from 68% to 92% after 100 trials. This study was able to show that deep learning models can be deployed on mobile devices, and potentially even as a web application.

The key difference that this study will have with the aforementioned studies as well as other existing studies and systems will be the use of a Siamese network approach along with a combination of CNN with SVM, particularly VGG-19 + SVM

for classifying mosquito species. As there are not that many studies with similar approaches that is created for detecting and classifying mosquito species through audio signals, this study will be able to contribute greatly to the body of knowledge, where this study will utilize multi-spectrogram inputs as opposed to existing models utilizing only single spectrogram input. This study will also utilize a portion of the HumBugDB dataset as its source of mosquito audio signals and the best performing model obtained from this study will be deployed as a web application.

Chapter 3

Methodology

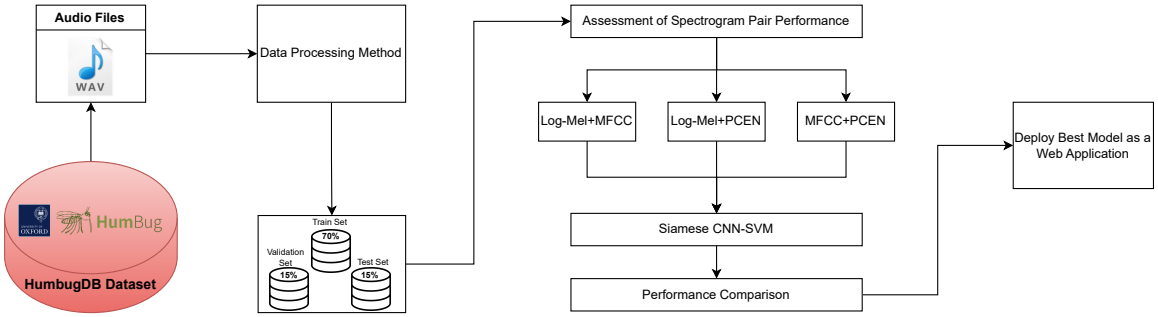

Figure 3.1: The BuzzNet Framework

Figure 3.1 represents a high level overview of the methodology. Initially, select audio files are to be taken from the publicly available dataset, the HumBugDB dataset. The audio files will then go through a data processing method, following the author's approach [9], where the output of this method would be three different spectrogram types for each audio file, namely Log-mel, MFCC, and PCEN. These three spectrogram types will be used to assess the model's performance. As this model follows a Siamese neural network approach, spectrogram pairs, which are Log-mel+MFCC, Log-mel+PCEN, and MFCC+PCEN, will be investigated to determine which spectrogram pairing would produce the highest performance. These spectrogram pairs will be used as inputs for the hybrid CNN-SVM model, where the best performing

model will then be deployed as a web application.

Network Design and Architecture

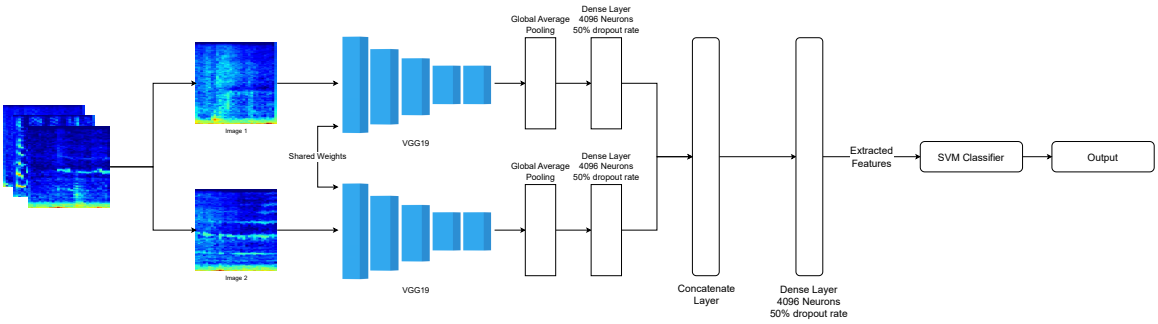

Figure 3.2: Siamese CNN-SVM Architecture

As shown in Figure 3.2, this study will utilize a Siamese network approach and adapt a hybrid implementation utilizing a convolutional neural network (CNN) along with a Support Vector Machine (SVM), where it will take two spectrogram images as its input and output the mosquito species class. These are made by combining two identical sub-networks which their outputs are then fed into a concatenate layer, followed by a dense layer with a 50% dropout rate. The CNN model to be used would be a pre-trained MosquitoNet model, trained on the HumBugDB dataset, which will be used as a feature extractor. The features extracted here will then be passed on to the SVM classifier to carry out the classification task. The inputs to be fed in this Siamese neural network are the different spectrogram pairs obtained from the pre-processing method, namely Log-Mel+MFCC, Log-Mel+PCEN, and MFCC+PCEN.

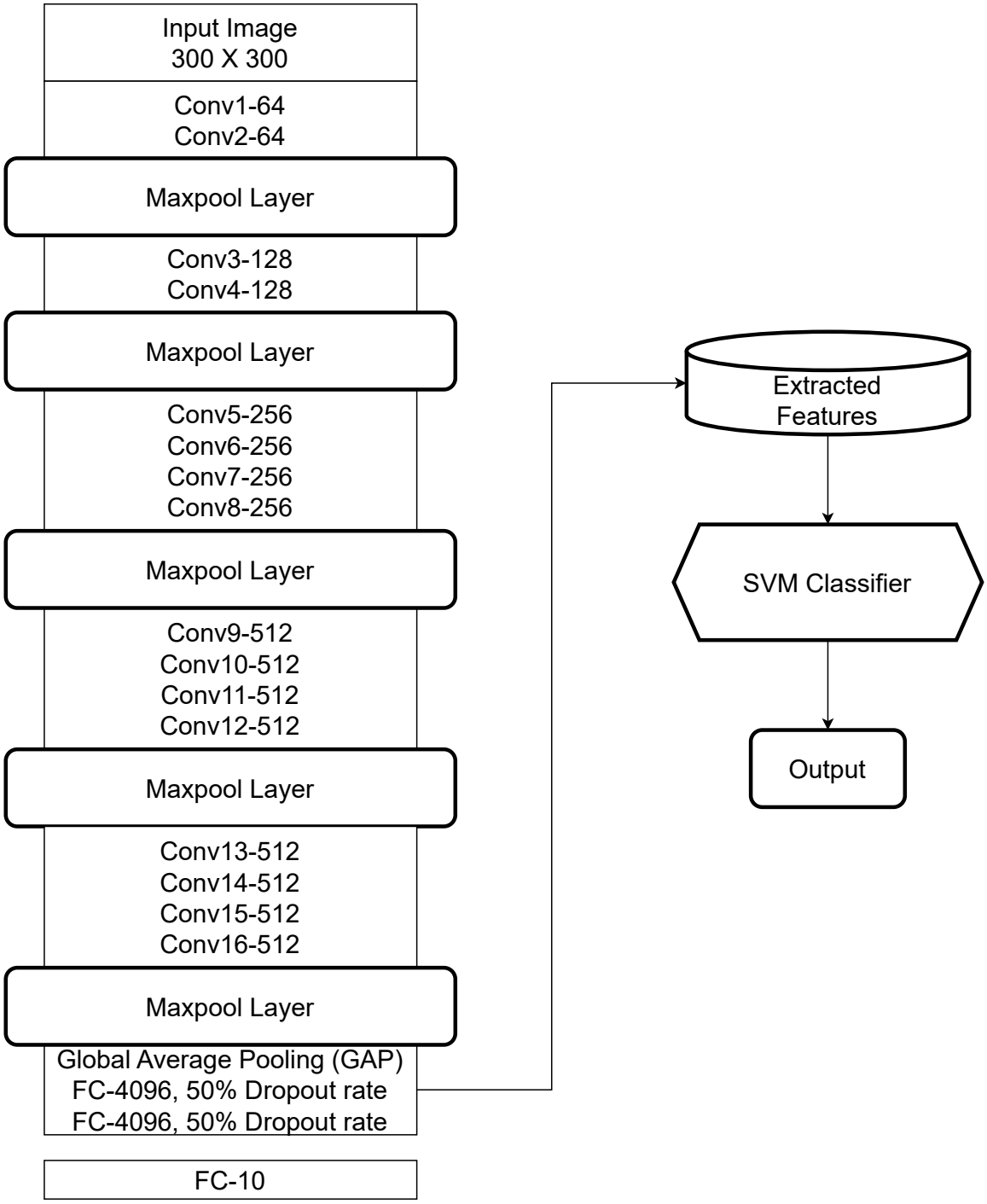

Figure 3.3: VGG-19+SVM Model Architecture

Figure 3.3 shows that the deep learning model to be utilized will be a hybrid CNN+SVM

model, where a pre-trained MosquitoNet model will be used for feature extraction and the features extracted from the model will be fed into a Support Vector Machine (SVM) for the classification of the mosquito species. It will take a spectrogram image of size 300x300 pixels as input, and then feed them into the pre-trained model consisting of 16 convolution layers, with 5 max-pooling layers. The tail of the architecture contains a Global Average Pooling (GAP) layer, along with two dropout layers with a 50% dropout rate. Finally, instead of using the softmax layer of the MosquitoNet model, extracted features will instead be passed to a Support Vector Machine where the classification of the mosquito species will take place. The output of this model would be the predicted mosquito class.

Datasets

For this research, a publicly available audio dataset will be used. Said dataset would be the HumBugDB Dataset which will be acquired from the Zenodo Repository based on the works of Kiskin et al. [9]. The dataset contains a total of 9,295 audio files, containing mosquito wingbeat sounds from different species, background noise, and other non-mosquito audio files. It includes 71,286 seconds (20 hours) of labelled mosquito data with 53,227 seconds (15 hours) of corresponding background noise, all recorded globally at different sites from 8 experiments. It includes a broad range of both indoor and outdoor background environments taken from different countries such as USA, UK, Kenya, Thailand, and Tanzania. Of these samples, 64,843 seconds contain metadata of their species, which consists of a total of 36 different species or species complexes. Table 3.1 below shows the profiling of the original dataset of the HumBugDB dataset.

Table 3.1: Profiling of Original HumBugDB Dataset

Class	Number of Instances
ae aegpyti	123
ae albopictus	79
an albimanus	55
an albimanus	55
an arabiensis	1985
an atroparvus	13
an barbirostris	10
an coluzzii	14
an coustani	92
an dirus	129
an farauti	13
an freeborni	52
an funestus	18
an funestus sl	104
an funestus ss	381
an gambiae	39
an gambiae sl	42
an gambiae ss	737
an harrisoni	124
an leesoni	3
an maculatus	117
an maculipalis	79
an merus	4
an minimus	20
an pharoensis	3
an quadriannulatus	70
an rivulorum	1
an sinesis	11
an squamosus	141
an stephensi	45
an ziemanni	9
audio	600
background	1900
coquilettidia sp	2
culex pipiens complex	545
culex quinquefasciatus	678
culex tarsalis	39
culex tigripes	18
ma africanus	78
ma uniformis	131
toxorhynchites brevipalpis	4

Table 3.2 below shows the modified table of instances for the mosquito species with its respective number of samples in the three different sets, which is the output

tabulation after the data pre-processing method.

Table 3.2: Profiling of Modified HumBugDB Dataset

Class	Train	Validation	Test
ae aegpyti	617	20	14
an arabiensis	8481	298	298
an barbirostris	35	3	3
an coluzzii	54	12	5
an coustani	526	16	14
an dirus	360	34	26
an funestus sl	709	34	16
an funestus ss	3495	128	61
an gambiae	32	7	4
an gambiae sl	183	7	7
an gambiae ss	784	70	70
an harrisoni	403	26	19
an maculatus	320	32	22
an maculipalis	942	56	15
an minimus	22	3	3
an squamosus	976	28	22
an stephensi	26	3	1
an ziemanni	46	2	4
audio	565	80	80
background	26037	279	280
culex pipiens complex	3800	131	82
culex quinquefasciatus	3146	264	143
culex tigripes	70	3	3
ma africanus	352	12	12
ma uniformis	772	20	20
toxorhynchites brevipalpis	26	8	8

Following the authors' approach [9], this study will only utilize specific mosquito species with the most populated species by recording, which are *Aedes aegypti*, *Anopheles arabiensis*, *Anopheles coustani*, *Anopheles funestus ss*, *Anopheles squamosus*, *Culex pipiens complex*, *Mansonia africanus*, and *Mansonia uniformis*, as well as some other non-mosquito noises labeled as "audio" and "background". Table 3.3 below shows the portion of the dataset that will be used in this study.

Table 3.3: Profiling of Modified HumBugDB Dataset to be used

Class	Train	Validation	Test
ae aegpyti	617	20	14
an arabiensis	8481	298	298
an coustani	526	16	14
an funestus ss	3495	128	61
an squamosus	976	28	22
audio	565	80	80
background	26037	279	280
culex pipiens complex	3800	131	82
ma africanus	352	12	12
ma uniformis	772	20	20
TOTAL	45621	1012	883

From the profiling of the dataset as shown in in Table 3.3, it is apparent that there is a severe class imbalance among the classes which can affect the performance of the model in such a way that it might produce biased results. Figure 3.4 shows us the total class distribution of the dataset to be used.

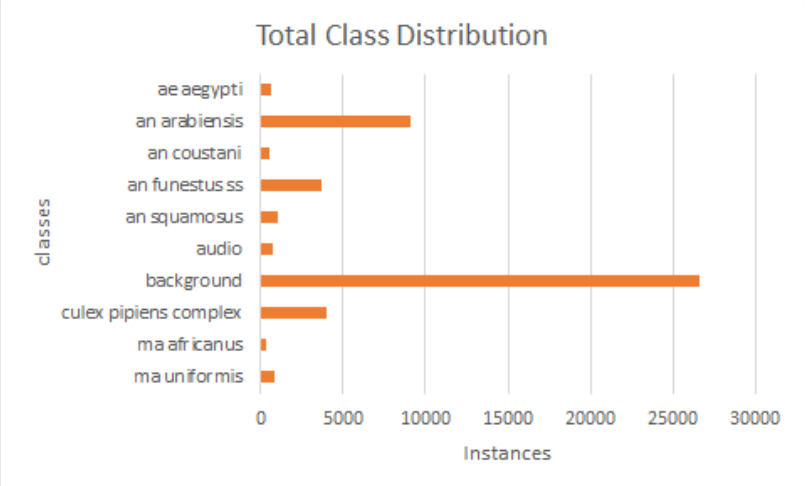

Figure 3.4: Data Processing Method

Data Processing

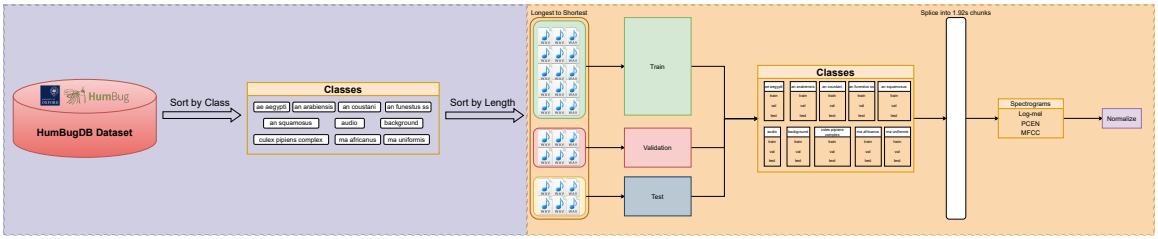

Figure 3.5: Data Processing Method

Figure 3.5 shows the entire data processing method, from obtaining the audio files from the dataset, to splitting them into the three different sets, all the way to converting the audio files into spectrograms and normalizing them. In this data processing method, we have two components, highlighted accordingly. To explain them in more detail, the Figures 3.6 and 3.7 presents as closer look into each component.

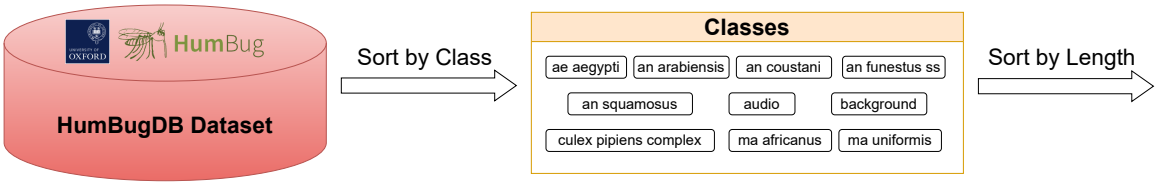

Figure 3.6: Data Processing Method (Component 1)

Figure 3.6 shows the first component of the data processing method, where the audio files taken from the HumBugDB dataset will initially be sorted and grouped into their respective classes, which are mosquito sounds (which are also split into their respective species), background, and audio. From here, the audio files will then be sorted according to length, starting from the longest down to the shortest.

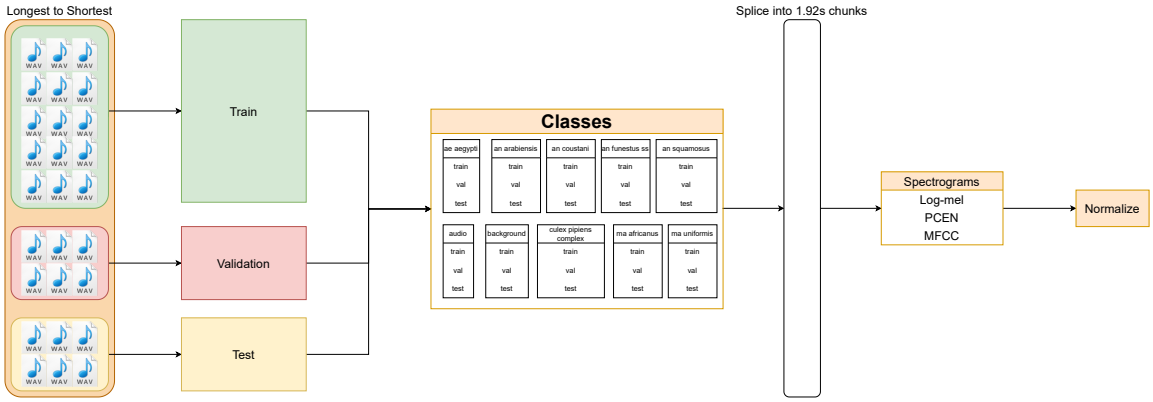

Figure 3.7: Data Processing Method (Component 2)

Figure 3.7 shows the second component of the data processing method, where after the files are sorted, they are then to be divided into three different sets with a 70:15:15 ratio, representing the train set, test set, and the validation set, respectively.

Once the data have been divided the respective sets, each audio file will then be spliced into 1920 millisecond (1.92 seconds) chunks. Any chunks that are less than the specified length are to be discarded.

Finally, spliced audio files will then be converted into normalized spectrograms with 3 different types namely the Log-based Mel (Log-Mel) Spectrogram, the Mel-Frequency Cepstrum Coefficient (MFCC) Spectrogram, and the Per-Channel Energy Normalization (PCEN) Spectrogram. Image normalization was also implemented to scale the image so that the minimum and maximum pixel values are 0 and 1, respectively. This was done using Keras' Rescaling layer which standardizes the pixel values to be in the range [0,1].

Table 3.4: Parameters Used for Generating Spectrograms

	Log-mel	MFCC	PCEN
Sampling Rate	8,000	8,000	8,000
NFFT	2,048	2,048	2,048
Hop Length	512	512	512
n_mels	128	128	128
window	30	30	30
n_mfcc		13	
S			melspec * (2 ** 31)
time_constant			0.06
epsilon			1e-6
gain			0.8
power			0.25
bias			10

Table 3.4 above presents the parameters employed for generating the spectrogram images in this study. These parameters were derived from the Feat B. from the author's approach [9]. The sampling rate assumes significance as it determines the number of samples acquired per second during the analysis. The length of the FFT window, denoted by NFFT, becomes an essential factor, as it corresponds to the number of points involved in the Fast Fourier Transform (FFT) algorithm. Moreover, the hop length, representing the interval between successive frames in the spectrogram or Short-Time Fourier Transform (STFT), plays a crucial role in the analysis process. Additionally, the parameter n_mels assumes significance by indicating the number of Mel filterbanks employed for computing the mel spectrogram. These filterbanks, designed as triangular filters, effectively approximate the frequency response observed

in the human auditory system. Furthermore, the window parameter assumes importance as it specifies the type and length of the window function employed during the computation of the STFT or spectrogram. Another critical parameter, n_mfcc, denotes the number of Mel-frequency cepstral coefficients (MFCCs) derived from the audio signal, which are widely recognized and utilized in audio signal processing. The spectrogram, denoted by the symbol S, represents the input to which the Per-Channel Energy Normalization (PCEN) transformation is applied. Typically, this transformation is performed on the mel spectrogram derived from the audio signal. The time_constant of the first-order infinite impulse response (IIR) filter utilized in PCEN plays a significant role and requires careful consideration. Additionally, the term eps or epsilon, representing a small positive constant, is introduced to ensure numerical stability during the computation. Moreover, the gain factor applied to the PCEN transformation contributes to the overall processing. The power, serving as the exponent in the PCEN computation, becomes a determining factor. Lastly, the bias term, representing a scalar value, is incorporated into the PCEN transformation to further enhance its effectiveness.

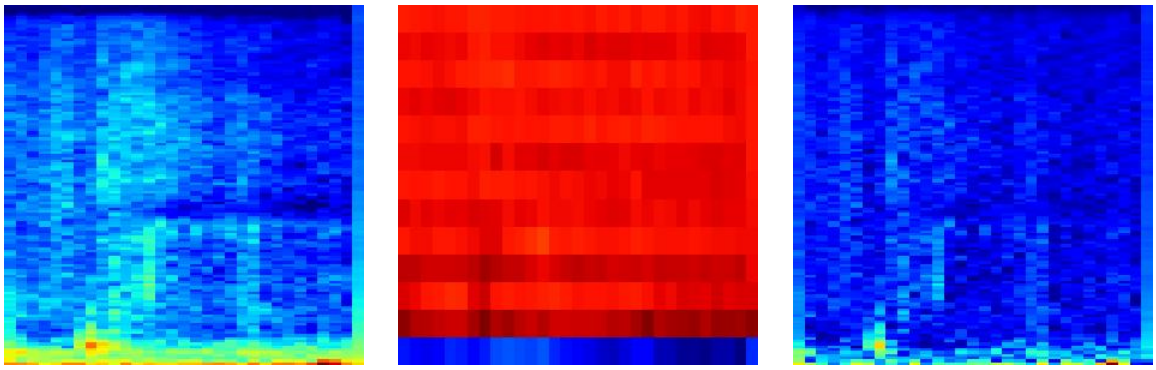

Figure 3.8: Spectrogram Images of Sample *Aedes aegypti* class audio file from test set

Figure 3.8 presents the three generated spectrogram images from a sample *Aedes aegypti* class audio file that has been spliced, taken from the test dataset.

The parameters specified in Table 3.4 were utilized in the process of generating Log-mel spectrograms using the librosa.feature.melspectrogram function from the *Librosa* library. This particular function facilitated the computation of the mel spectrogram from the given audio data. To accomplish this, the function took into account the audio data, denoted as x, along with the sampling rate, sr, and additional parameters such as n_mels, nfft, w, and hop_length. Subsequently, the power spectrogram was transformed into the logarithmic scale using the librosa.power_to_db function. This transformation was implemented to enhance the suitability of the spectrogram for visualization and analysis purposes. Finally, the function produced the computed Log-mel spectrogram as a 2D NumPy array, which was subsequently converted into a 300x300 Log-mel spectrogram image.

In a similar fashion, the parameters specified in Table 3.4 were also employed during the generation of MFCC spectrograms. To achieve this transformation, the audio data was processed using the mfcc_transform function provided by the *Librosa* library, which facilitated the conversion into a Mel-frequency Cepstral Coefficients (MFCC) spectrogram representation. Notably, an additional parameter, n_mfcc, was specified to determine the precise number of Mel-frequency Cepstral Coefficients computed. These coefficients serve as a representation of the short-term power spectrum of sound, designed to model the intricacies of the human auditory system. Specifically, 13 coefficients were calculated in this case. The librosa.feature.mfcc function, an integral component of the Librosa library, played a vital role in computing the

MFCC spectrogram from the given audio data. In executing this function, the necessary inputs comprised the audio data denoted as x, the corresponding sampling rate represented by sr, and several additional parameters such as n_mfcc, n_mels, nfft, w, and hop_length, each contributing to the spectrogram calculation. Consequently, the function returned the computed MFCC spectrogram as a 2D NumPy array, which was subsequently converted into a visually meaningful 300x300 MFCC spectrogram image.

And for the generation of the PCEN spectrograms, using the parameters specified in Table 3.4, the convertion process entailed utilizing the pcen_transform function from the *Librosa* library, which facilitated the conversion of audio data into a Per-Channel Energy Normalization (PCEN) spectrogram representation. Preceding this transformation, the librosa.feature.melspectrogram function from the *Librosa* library was employed to compute the mel spectrogram of the given audio data. For the purpose of this computation, the function necessitated inputs including the audio data (x), the sampling rate (sr), and various additional parameters (n_mels, nfft, w, and hop_length) essential for the spectrogram calculation. Subsequently, the PCEN transformation was applied to the mel spectrogram using the librosa.pcen function. PCEN serves as a normalization technique that scales the spectrogram based on local energy and applies compression to enhance relevant features. The librosa.pcen function required inputs such as the mel spectrogram (melspec) multiplied by a scaling factor, along with various parameters encompassing the time constant, epsilon value, gain, power, and bias. Additionally, the sampling rate (sr) and hop length were included as pertinent inputs. Finally, the function yielded the computed PCEN spectrogram as a 2D NumPy array, which was further transformed into a visually meaningful 300x300

PCEN spectrogram image.

Model Training and Evaluation

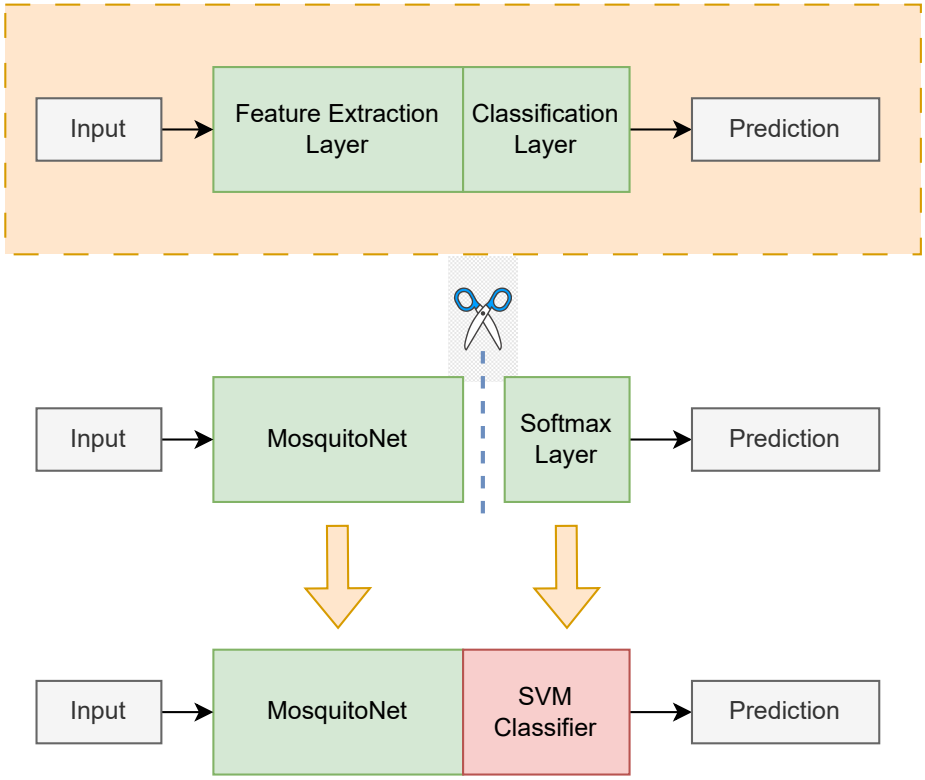

Figure 3.9: Transfer Learning Diagram

To train the model, the train and validation sets were utilized. Here, the model used the different types and pairs of spectrograms from each mosquito species. These served as the input for the pre-trained MosquitoNet model for feature extraction. The MosquitoNet model is a VGG-19 based CNN model that was pre-trained on the HumBugDB dataset. The pre-trained model was loaded along with its custom loss function, the categorical loss function, and its layers are frozen so that the layers will not be trainable. This was done so that the weights of the mode will not be updated. This was achieved by setting *model.trainable = False*. Then, the softmax

layer of the MosquitoNet model was removed so that the model will not output any predictions, and instead, extract the features. This entire process follows the principle of transfer learning where a pre-existing model is used and its learned features will then be transferred onto another neural network, instead of training from scratch, similar to the approach in the works of Choi et al. [4]. Figure 3.9 exhibits how this study performed transfer learning.

Once the features have been extracted, these features were then passed onto the SVM classifier for classification. The SVM used in this study utilized the Scikit-learn implementation which is capable of multi-class classification. From here, the SVM classifier was trained using the extracted features as input. In order to improve model performance, hyperparameter tuning was conducted. To find the optimal values for the hyperparameters to improve model performance, RandomizedSearchCV was implemented, similar to the author's approach [18].

The test set derived from the pre-processed data was then used to evaluate the model in the evaluation phase. In this phase, the performance of the model was assessed using a number of evaluation criteria, including accuracy and macro-f1 score. The receiver operating characteristic area-under-curve (ROC AUC) and precision-recall area-under-curve (PR AUC) metrics were utilized to compare the best model to the baseline MozzBNNv2 model from Kiskin et al. [9].

Additionally, since this study will deal with the classification of multiple different classes of mosquito species, this means that there will be at least two or more output labels. Therefore, the loss function to be used will be the categorical focal loss function [13], where it is an alternative method to the commonly used cross-entropy loss. This loss function is used so as to address the severe class imbalance present in the

dataset, which could potentially lead to biases towards the majority class and relatively low performance on the minority classes. In this function, each class is given a weight based on its importance. For classes that are underrepresented have higher weights, while classes that are overrepresented have lower weights. Additionally, it adds a modulator known as the "focusing parameter", which changes the weights accordingly for samples that are simpler to categorize and those that are difficult to categorize. As a result, the model can concentrate more on the minority classes and improve their performances. The categorical focal loss function can be defined using the formula,

$$FL_(p_t) = -(1 - p_t)^{\gamma} \cdot log(p_t) \tag{3.0.1}$$

where p_t is the predicted probability of the correct class and γ is the focusing parameter. When instances are correctly identified, the function minimizes loss, but when they are incorrectly classified, it increases loss. The amount of reduction or increase is determined by the value of γ, which can be adjusted to reach the desired degree of class balance.

Deployment

The best performing Siamese CNN-SVM model, as shown in Table 4.1, was deployed as a web application utilizing HTML and CSS for its front-end, as well as Flask for its back-end.

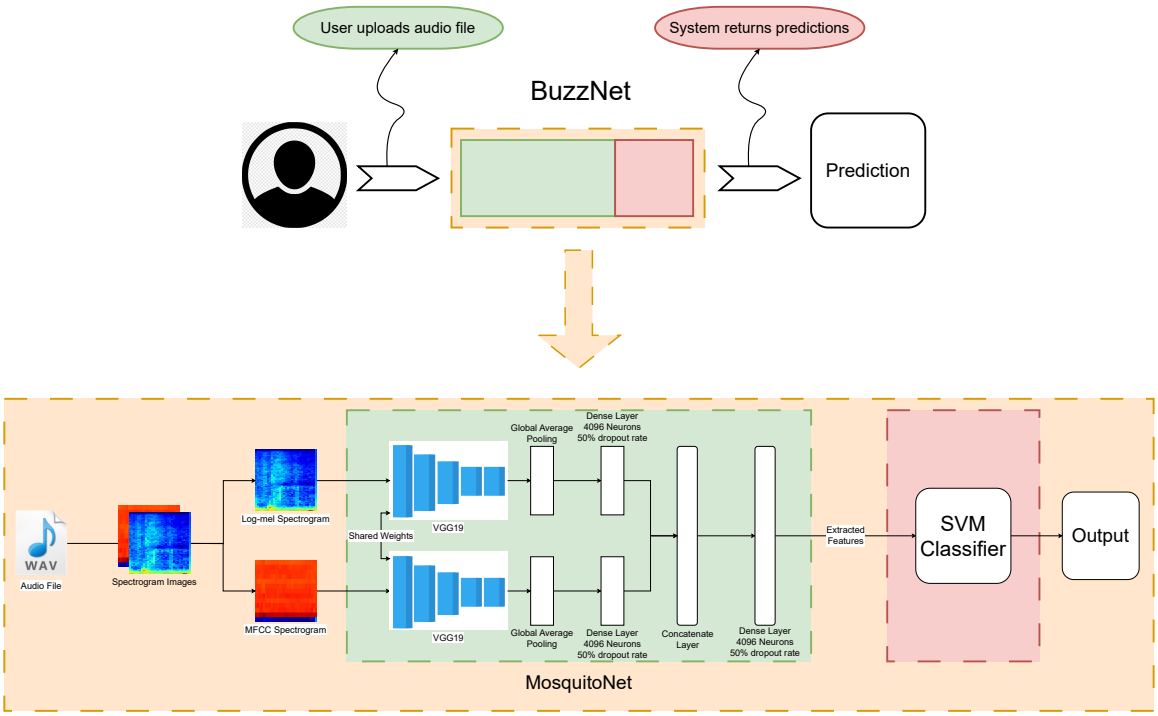

Figure 3.10: Overview of BuzzNet, the Web-based Mosquito Species Classification System

Figure 3.10 presents a general overview of the BuzzNet web application, a mosquito classification system. The user can upload audio files with wav format. BuzzNet then processes the uploaded audio file by converting it into Log-mel and MFCC spectrograms. The pixel values are also normalized to [0,1]. The spectrogram images are then fed into model to generate the predictions. The predicted mosquito class is then displayed to the user through the web application's user interface.

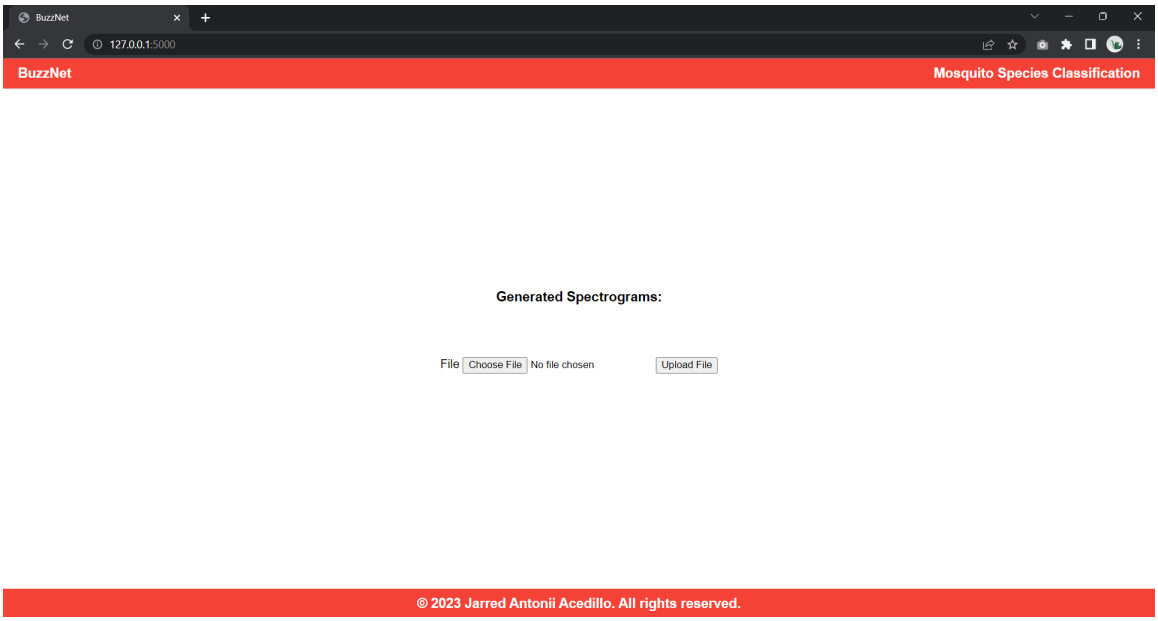

Figure 3.11: The BuzzNet Application's Home Page

Figure 3.11 show's the web application's home page. The Flask web server hosts the web application, which can be accessed locally through the address 127.0.0.1:5000 using any up-to-date web browser, such as the latest version of Google Chrome. The user can input an audio file by clicking the "Choose File" button, although the web application's capability is limited to only being able to process wav files. The user can then navigate through their system files and select their desired audio file. Once the user uploads the file by clicking the "Upload File", the web application then processes the audio file and predicts the mosquito species. The predicted species is then displayed in the user interface for the user to see, along with the Log-mel and MFCC spectrograms generated from the uploaded audio file.

Chapter 4

Results and Discussion

Performance Evaluation Metrics

The performance evaluation metrics used to evaluate model performance in this study included accuracy and macro average F1-score. These metrics were chosen for the classification task in order to take into consideration the class imbalance present in the dataset, similar to the approach of the author [7].

Accuracy is defined as the total correctly classified samples divided by the total number of classified samples. The equation for accuracy is

$$accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{4.0.1}$$

where TP stands for True Positive, TN stands for True Negative, FP stands for False Positive, andFN stands for False Negative.

Meanwhile, the macro average F1-score serves as a comprehensive evaluation metric for a model's performance across multiple classes. It is computed by calculating the harmonic mean of precision and recall for each label, and then averaging the results across all labels. This metric provides an overall assessment of the model's

performance in a manner that accounts for variations across different classes. It can be defined as follows:

$$Macro - F1 = \frac{1}{N} \sum_{i=1}^{N} \frac{2XP_iXR_i}{P_i + R_i} \tag{4.0.2}$$

where N represents the number of classes, $\sum_{i=1}^{N}$ represents the summation over the range 1 to N, P_i is the precision for class i, and R_i is the recall for the class i. The calculation of the macro average F1-score involves summing the individual F1-scores for each class and then dividing the resulting sum by the total number of classes. This process yields a single score that represents the average F1-score across all classes.

Conversely, the existing state-of-the-art approach, which utilized the HumBugDB dataset, specifically the MozzBNNv2 [9], established the primary baseline for evaluating the proposed model. In this evaluation, the performance of the models were assessed using two metrics: receiver operating characteristic area-under-curve (ROC AUC) score and precision-recall area-under-curve (PR AUC) score. These metrics were employed to analyze and compare the performance of the proposed model against MozzBNNv2 [9] as the state-of-the-art method.

The ROC AUC score is determined by calculating the area under the receiver operating characteristic (ROC) curve. The ROC curve illustrates the relationship between the true positive rate (TPR) and the false positive rate (FPR) across different threshold settings. The AUC score represents the probability that a randomly selected positive instance will be ranked higher than a randomly selected negative instance by the classifier. The AUC score ranges from 0 to 1, with a score of 1 indicating perfect classification and a score of 0.5 representing random guessing.

Implementation Details and Training Protocols

The BuzzNet model in this study was implemented using Keras. Training of the model was done using a single NVIDIA GeForce GTX 1080 Ti with 11 GB of GDDR5X memory and an Intel(R) Core(TM) i7-8700 CPU running at 3.20GHz. The hybrid model comprised of a pre-trained MosquitoNet model, a model based on a pre-trained VGG-19 model which was fine-tuned on the HumBugDB dataset, for feature extraction, and a Support Vector Machine for the classification. This was implemented by utilizing the MosquitoNet for extracting the features and passing those features into the SVM, instead of the softmax classification layer of the MosquitoNet model. The SVM used in this study utilized the implementation of Scikit-learn capable of multi-class classification, particularly the SVC implementation. The SVC is a C-support vector classification based on libsvm which handles multi-class support according to a one-versus-one scheme. The Siamese architecture was implemented by utilizing different pairs of spectrogram images taken from the pre-processed data as input, and feeding them into the BuzzNet model where it would output the classification results.

In order to improve the performance of the standard SVM, a technique called RandomizedSearchCV was implemented, following the approach of the author [18], where it trains the SVM model by randomly passing a specified set of tuning parameters and calculate the score to determine which set of parameters gives the best scores. The tuning parameters for the classifier included the regularization parameter C, the kernel coefficient *gamma*, and the tolerance for stopping criterion *tol*, where the values of C, *gamma*, and *tol*, were float values ranging from a specified range. These aforementioned parameters were used in the RandomizedSearchCV. Other available

parameters of the classifier included the kernel type *kernel*, where its values were chosen between either "rbf", "sigmoid", "linear", or "poly", *break_ties*, which is responsible for breaking ties depending on the confidence values of the decision function, which was set to "True" in order to produce better classification results, as opposed to "False" which just sets the prediction to whichever the first class is among the tied classes, and lastly *class_weight*, where it was set to "balanced", which automatically adjusts the weights of each class in proportion to class frequencies. This can be obtained by calculating $n\_samples/(n\_classes * np.bincount(y))$. This parameter is important so that the model sets their weights inversely proportional to the class frequencies in the input data. This parameter ensures that the model does not gear towards the majority class and produce low performance on the minority classes.

Experimental Setup

Experiment I

Experiment 1 was conducted to address this study's first specific objective by introducing a Siamese CNN-SVM model that accepts a spectrogram pair (Log-mel+MFCC, Log-mel+PCEN, MFCC+PCEN) as input. The goal is to present a state-of-the-art Siamese CNN-SVM architecture in mosquito species classification that can output competitive performance against other existing architectures, such as a single-input CNN model. Initially, a comparative evaluation was conducted on three network variants, each utilizing different pairs of spectrograms, specifically Log-mel + MFCC, Log-mel + PCEN, and PCEN + MFCC, as inputs. The objective was to identify the spectrogram pair that yielded the most optimal performance. Subsequently, the Siamese CNN-SVM model that demonstrated the best performance, in conjunction with the selected spectrogram pair, underwent an ablation study.

Experiment II

Experiment 2 was conducted to address this study's second specific objective by conducting an ablation study in order to determine the effect of the individual components to the overall model's performance. The best performing model obtained from Experiment 1 was subjected to this ablation study. Here, the strategies explored to achieve the model's performance was the utilization of a hyperparameter tuning method, the RandomizedSearchCV, as well as the addition of the SVM classifier on top of the pre-trained MosquitoNet.

Experiment III

Experiment 3 was conducted to address this study's third specific objective by conducting a state-of-the-art analysis to create comparison between the best performing BuzzNet model with other current best known methods, specifically those that also utilize the HumBugDB dataset so as to provide a direct comparison. To date, only the study conducted by Kiskin et al. [9] has addressed mosquito species classification using the HumBugDB dataset in the existing literature. Consequently, the model proposed by Kiskin et al. [9] served as the primary benchmark for comparing the proposed model in this current study. In line with their approach, the evaluation metrics employed were specifically the receiver operating characteristic area-under-curve (ROC AUC) and precision-recall area-under-curve (PR AUC).

However, while this study followed Kiskin et al.'s [9] methodology of segmenting audio files into 1.92 second chunks and using the eight mosquito classes, several additional steps were incorporated. These included merging the mosquito species identification and classification tasks into a single classification task by including the

Background and Audio classes with the remaining eight mosquito classes, arranging audio files based on their duration from longest to shortest in terms of time domain length, and introducing a validation set in addition to the training and testing sets.

Therefore, it is important to note that the comparison between Kiskin et al.'s [9] model and the proposed BuzzNet model in this study is not a direct one-to-one comparison in terms of performance due to slight differences in the datasets and methodology. Instead, it can be considered as a close approximation. However, this slight variance is expected to have negligible impact when the model is tested on new and unseen data.

Experiment IV

Experiment 4 was conducted to address this study's fourth specific objective by deploying the best performing Siamese CNN-SVM model as a web application for mosquito species classification using HTML and CSS as its front-end while utilizing Flask for its back-end.

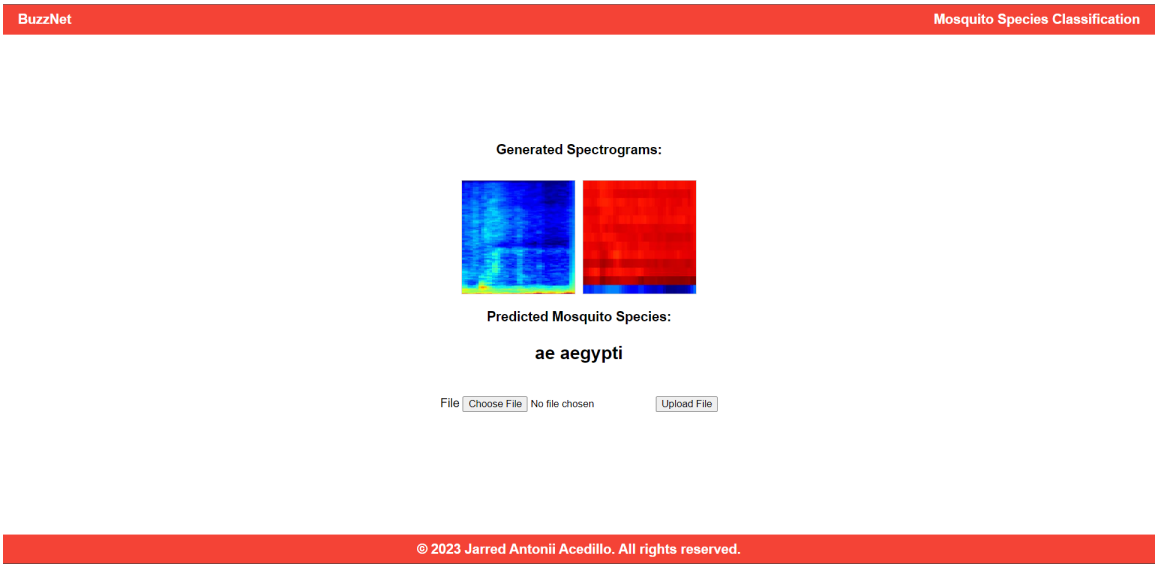

Figure 4.1: The BuzzNet Application With An Uploaded Audio File

Figure 4.1 shows the web application's user interface after the user has uploaded an audio file. In this example, the 220291.wav was uploaded. This audio file is classified as *Aedes aegpyti*. As shown in the figure, the web application displays the two generated spectrogram images along with predicted mosquito species. The decision to convert the audio file into Log-mel and MFCC spectrograms was driven by the fact that the Siamese CNN-SVM model achieved the best performance when Log-mel + MFCC spectrograms were used as inputs. Subsequently, the audio file underwent processing and was inputted into the BuzzNet model to generate predictions. In this particular instance, the model accurately predicted the true class of the audio file, which is *Aedes aeggypti*.

Results and Discussion

Table 4.1: Performances of Single-Input CNN-SVM Model and Siamese CNN-SVM Model with Different Spectrogram Pairs as Input

Spectrogram	Training Method	Accuracy	Macro-F1
Log-mel+MFCC	Siamese CNN-SVM + RandomizedSearchCV	**0.9094**	**0.7756**
Log-mel+PCEN		0.8890	0.7294
MFCC+PCEN		0.8913	0.7073
Log-mel	CNN-SVM + RandomizedSearchCV	0.9049	0.7381
PCEN		0.8947	0.6897
MFCC		0.8143	0.5254

Table 4.1 shows the performance of the proposed Siamese CNN-SVM model with the different spectrogram pairs, as well as the Single-Input CNN-SVM model. The results in the table above corresponds to Experiment 1 of this study which is to address the first specific objective. For a fair comparison, both were trained by using

the RandomizedSearchCV with their optimal parameters for each model to produce their best results. From Table 4.1, we can infer that Siamese CNN-SVM model that used Log-mel+MFCC spectrograms as input yielded the best performance with an accuracy of 90.94% and a macro-f1 score of 77.56%. This model outperforms the best Single-Input CNN-SVM model, which only achieved an accuracy of 90.49% and a macro-f1 score of 73.81%. The Siamese CNN-SVM model with Log-mel+MFCC was able to boost the accuracy by 0.45% and macro-f1 score by 3.75% over the Single-Input CNN-SVM model. The findings of this study provide evidence that supports the initial hypothesis that a Siamese Convolutional Neural Network (CNN) architecture holds promise for mosquito species classification. This is due to the inherent advantage of the Siamese CNN-SVM in leveraging discriminative features from multiple spectrogram representations, as opposed to a Single-Input CNN-SVM that can only utilize a single spectrogram representation. Furthermore, the outcomes of the study demonstrate that the Siamese CNN-SVM model attained better performance when Log-mel+MFCC spectrograms were utilized as inputs. This observation serves as additional evidence, indicating that the model was capable of learning more robust features when Log-mel + MFCC spectrograms were employed, in comparison to Log-mel+PCEN and MFCC+PCEN combinations, as well as pointing to the fact that using different spectrogram images as inputs can yield varying results from one another.

Table 4.2: Ablation Study of the Best Performing Siamese CNN-SVM Model with Log-mel+MFCC

Spectrogram	Training Method	Accuracy	Macro-F1
Log-mel+MFCC	Siamese CNN-SVM + RandomizedSearchCV	**0.9094**	**0.7756**
	Siamese CNN-SVM	0.8845	0.6585
	Siamese CNN (MosquitoNet)	0.9094	0.7543
Log-mel+PCEN	Siamese CNN-SVM + RandomizedSearchCV	0.8890	0.7294
	Siamese CNN-SVM	0.8584	0.6041
	Siamese CNN (MosquitoNet)	0.8856	0.7254
MFCC+PCEN	Siamese CNN-SVM + RandomizedSearchCV	0.8913	0.7073
	Siamese CNN-SVM	0.8618	0.5953
	Siamese CNN (MosquitoNet)	0.8811	0.6451

Table 4.2 presents the ablation study conducted to the best performing Siamese CNN-SVM model with Log-mel+MFCC as spectrogram inputs. The components in this experiment were removed individually in order to determine their contribution towards the model's performance.

The first component is the hyperparameter tuning method, the RandomizedSearchCV. This method was used to improve the model's performance, following the author's approach [18]. Fine-tuning involved finding finding the optimal values for the hyperparameters to improve model performance. The tuning parameters that were included were C, *gamma*, and *tolerance*, where the value of C were float values ranging from 0 to 2, *gamma* with float values ranging from 0.01 to 2, and *tolerance* with float values ranging from 0.4 to 0.6. RandomizedSearchCV is responsible for randomly selecting the float values from the specified ranges of each values, and fitting the model using these combinations of parameters. These specific range of values

were found to be the most optimal range from the experiments conducted. Other parameters that was used for the classifier include the *kernel* which included "rbf", "sigmoid", "linear", and "poly" for testing, *break_ties* which was set to "True", and the *class_weights* which was set to "balanced". The rest of the available parameters were left at their default values. From the results shown on Table 4.2, the RandomizedSearchCV improves model accuracy by 2.49% and macro-f1 score by 11.71%.

The next component is the addition of the SVM classifier. The SVM classifier's parameters here are set to default, other than *break_ties* which was set to "True", and the *class_weights* which was set to "balanced". From the results, we can see that the addition of the SVM classifier to the MosquitoNet model showed a decrease in both accuracy and macro-f1 score. This can be attributed to the lack of optimal parameters in the SVM classifier, leading to lower model performance.

From Table 4.2, despite the relatively lower performance of the Siamese CNN-SVM model compared to the MosquitoNet model, the addition of the hyperparameter tuning method, the RandomizedSearchCV, was able to produce results that outperformed the rest of the models, achieving equal accuracy, but a substantial increase in macro-f1 score of 2.13% over the Siamese CNN model, showing that the absence of the hyperparameter tuning method resulted in a marginal performance decrease. We can also observe that the other spectrogram pairs, Log-mel+PCEN and MFCC+PCEN, produced lower performance compared the spectrogram pair that yielded the best performing model, the Log-mel+MFCC spectrogram pair.

Table 4.3: ROC AUC and PR AUC Comparison with MozzBNNv2 [9], MosquitoNet, and BuzzNet tested on the Test Set

Class	AUC	MozzBNNv2	MosquitoNet	BuzzNet
An. arabiensis	ROC	86.6	**98.6**	98.2
	PR	80.9	**99.2**	96.9
Culex pipiens complex	ROC	86.7	93.7	**96.2**
	PR	66.9	**73.1**	72.9
Ae aegypti	ROC	96.4	**99.3**	98.6
	PR	74.4	**83.9**	78.9
An. funestus ss	ROC	92.3	93.6	**97.5**
	PR	80.9	**82.9**	82.5
An. squamosus	ROC	85.2	**97.7**	95.4
	PR	35.6	**65.0**	55.3
An. coustani	ROC	88.4	**95.5**	94.2
	PR	26.6	**64.3**	56.8
Ma. uniformis	ROC	82.0	**94.2**	87.3
	PR	29.6	**42.8**	30.6
Ma. africanus	ROC	91.3	91.7	**98.8**
	PR	22.3	**47.8**	47.2
Total	ROC	92.7	97.6	**97.7**
	PR	71.6	88.0	**93.2**

Table 4.3 shows the ROC AUC and PR AUC comparisons between the MozzBNNv2 [9], the MosquitoNet model, and the proposed BuzzNet model. These scores are given by their micro-average. It can be inferred from the table that the proposed BuzzNet model was able to outperform the MozzBNNv2 across the board, while also outputting comparable results with the MosquitoNet model. Despite the MosquitoNet model having higher scores on some classes, the difference between the proposed BuzzNet model results are not far off, enough to where the proposed BuzzNet model was able to achieve higher total scores in both ROC and PR in comparison. The

BuzzNet model was able to achieve 98.5% Total ROC AUC and 93.3% Total PR AUC, resulting in a 5.8% increase over the MozzBNNv2 and a 0.8% increase over the MosquitoNet Total ROC AUC scores, and a 21.6% increase over the MozzBNNv2 and a 5.3% increase over the MosquitoNet Total PR AUC scores.

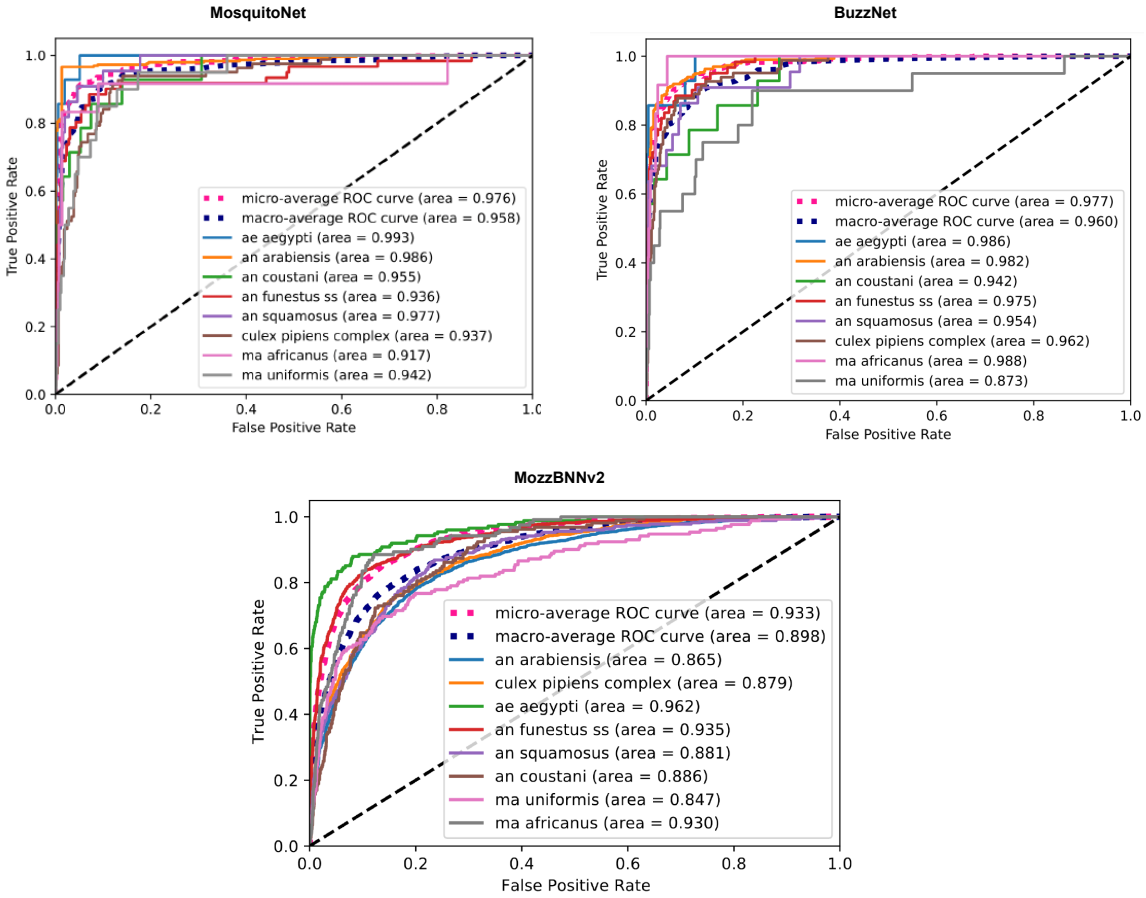

Figure 4.2: ROC Curves and Areas Comparison between MozzBNNv2 [9], MosquitoNet, and BuzzNet

Figure 4.2 presents the comparison of the ROC curves and areas of the proposed BuzzNet model against the MozzBNNv2 [9] and MosquitoNet models. From Figure 4.2, the micro-average ROC curves of the proposed BuzzNet model was overall higher and the BuzzNet model was able to cover more area compared to both the

MozzBNNv2 [9] and the MosquitoNet. A higher micro-average ROC curve reflects improved overall performance of a classification model in terms of its capacity to differentiate between different classes. It indicates that the model has achieved higher true positive rates while maintaining lower false positive rates, which is typically desirable in classification tasks. This signifies enhanced performance in correctly identifying positive instances and minimizing false positive errors. Therefore, the BuzzNet model was able to achieve improved overall performance compared to the other models.

Finally, the best performing state-of-the-art BuzzNet model was deployed as a web application capable of classifying mosquito species.

Table 4.4: Sample Audio Files Used to Test the Web Application

Class	File Name
Ae. aegypti	220291.wav
An. arabiensis	207459.wav
An. coustani	222237.wav
An. funestus ss	222584.wav
An. squamosus	221699.wav
Audio	777.wav
Background	201347.wav
Culex pipiens complex	220089.wav
Ma. africanus	220811.wav
Ma. uniformis	220586.wav

Table 4.4 shows the audio files that were used to test the web application. 10 audio files were taken from the test set, one audio file for each mosquito class. These audio files were then uploaded into the web application for processing and outputting the class predictions. The results for each audio file are as follows:

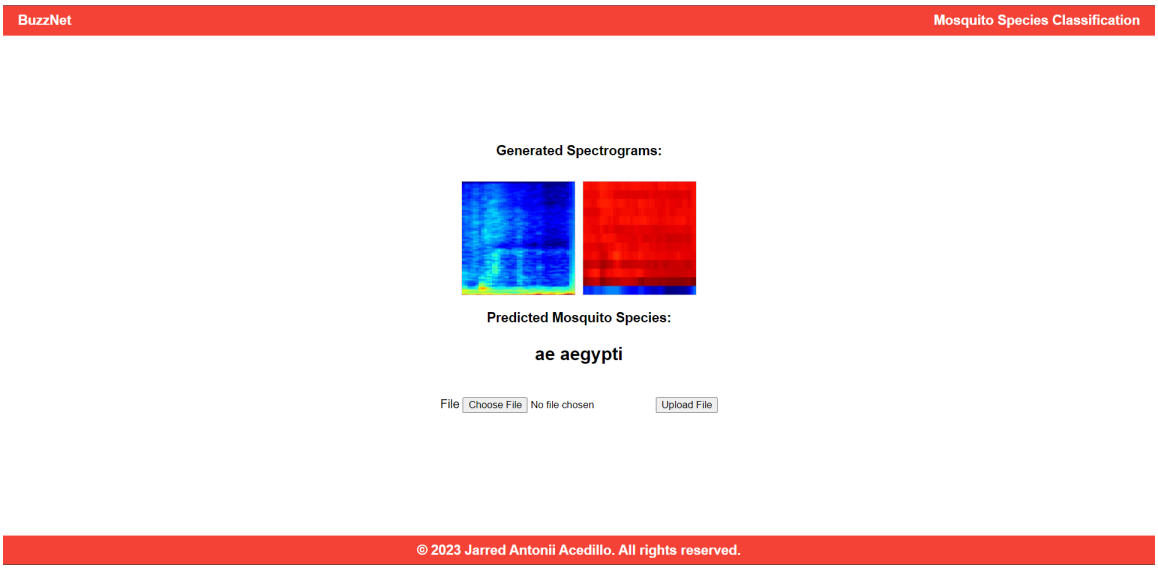

Figure 4.3: Web Application's Predictions for An. aegypti Class Audio File

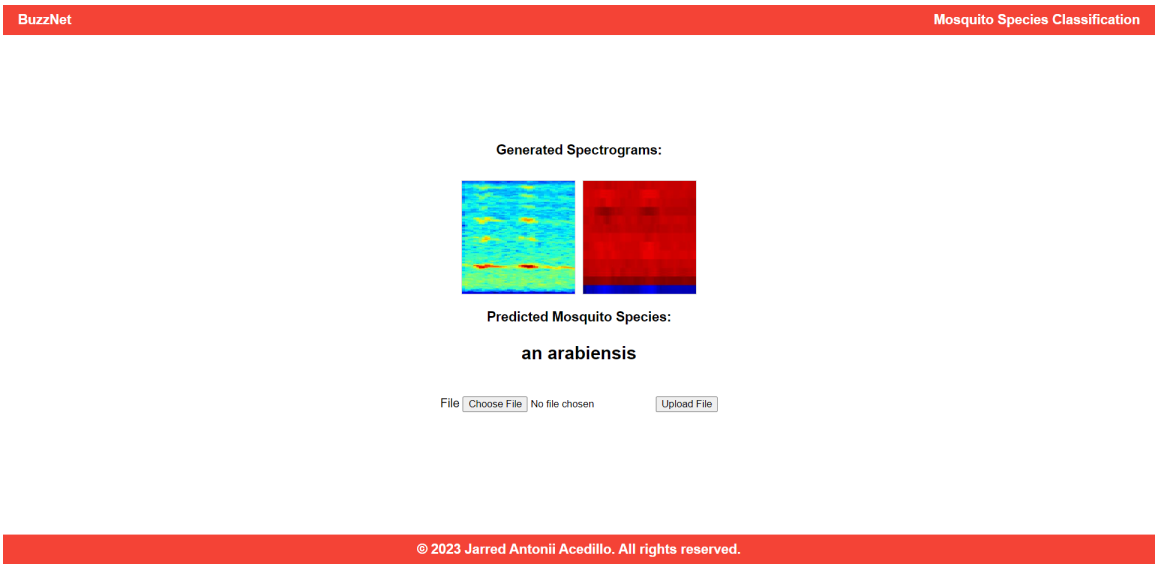

Figure 4.4: Web Application's Predictions for An. arabiensis Class Audio File

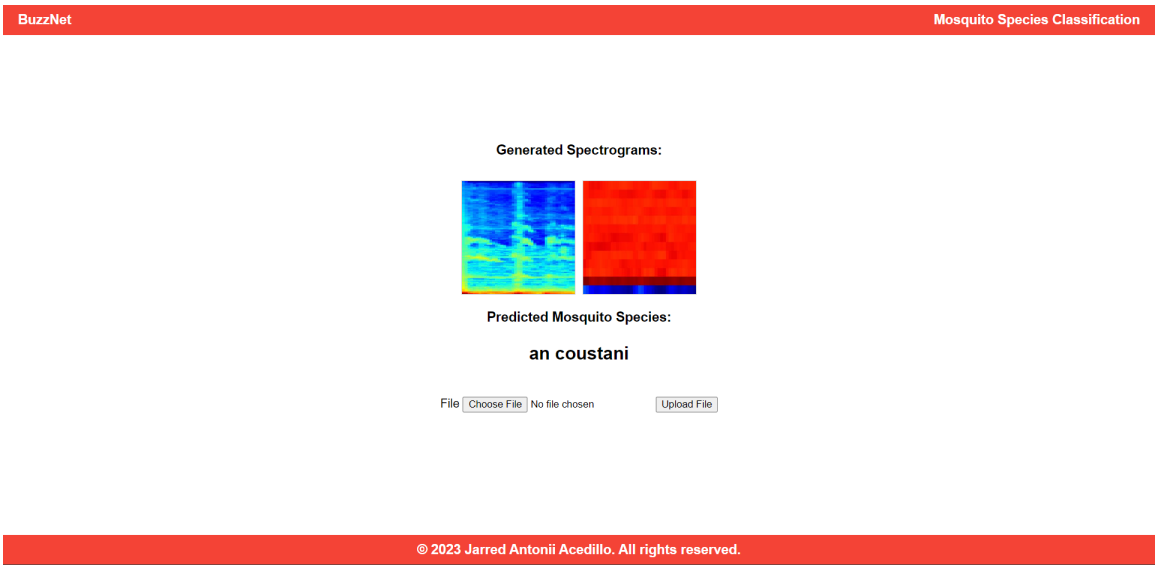

Figure 4.5: Web Application's Predictions for An. coustani Class Audio File

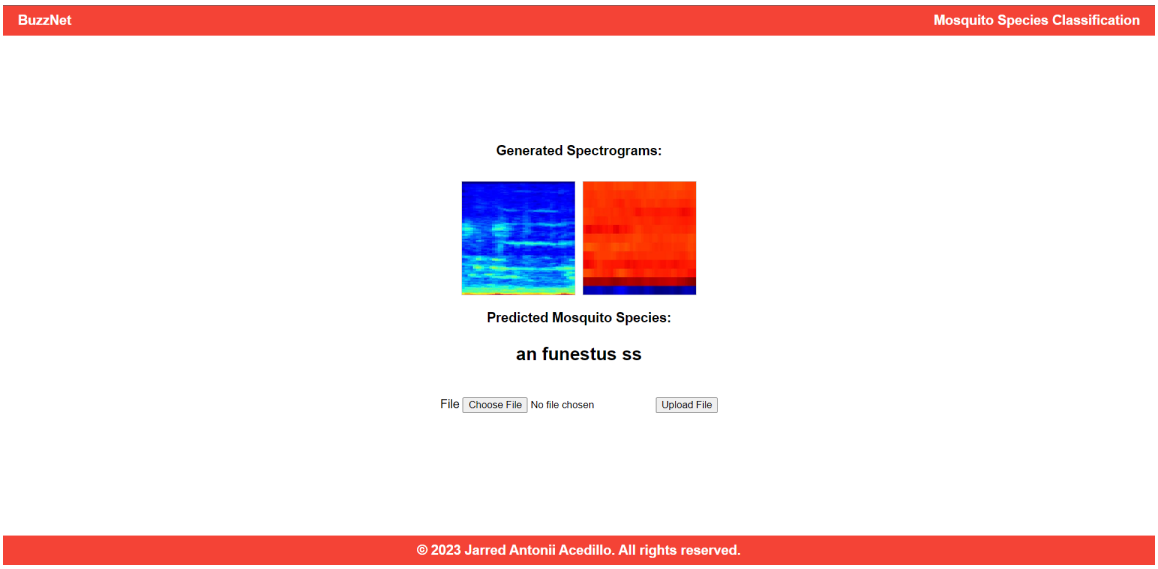

Figure 4.6: Web Application's Predictions for An. funestus ss Class Audio File

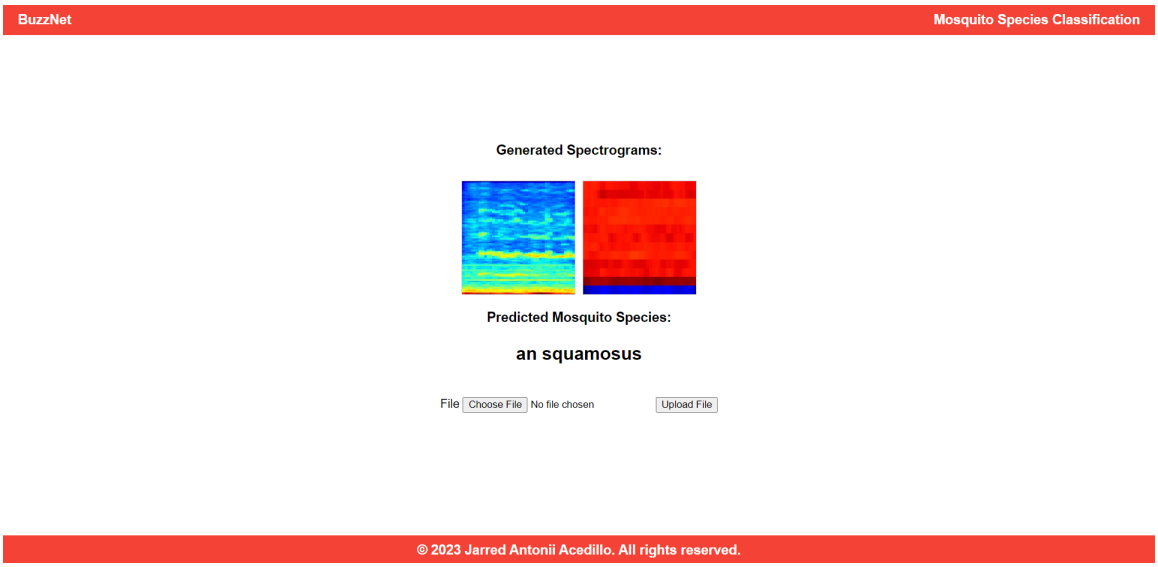

Figure 4.7: Web Application's Predictions for Ae. aegypti Class Audio File

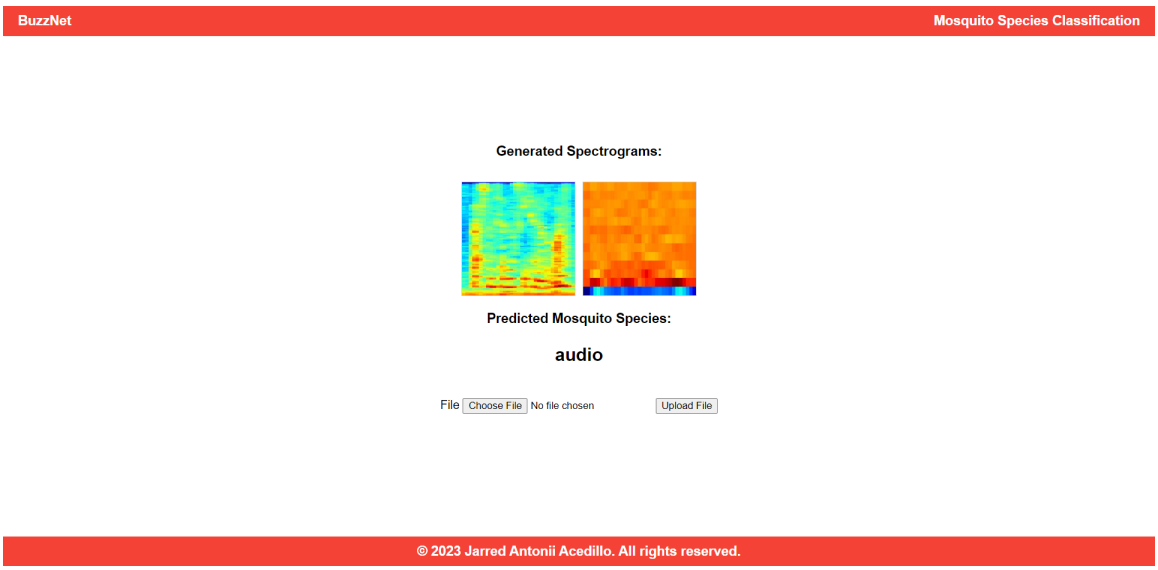

Figure 4.8: Web Application's Predictions for Audio Class Audio File

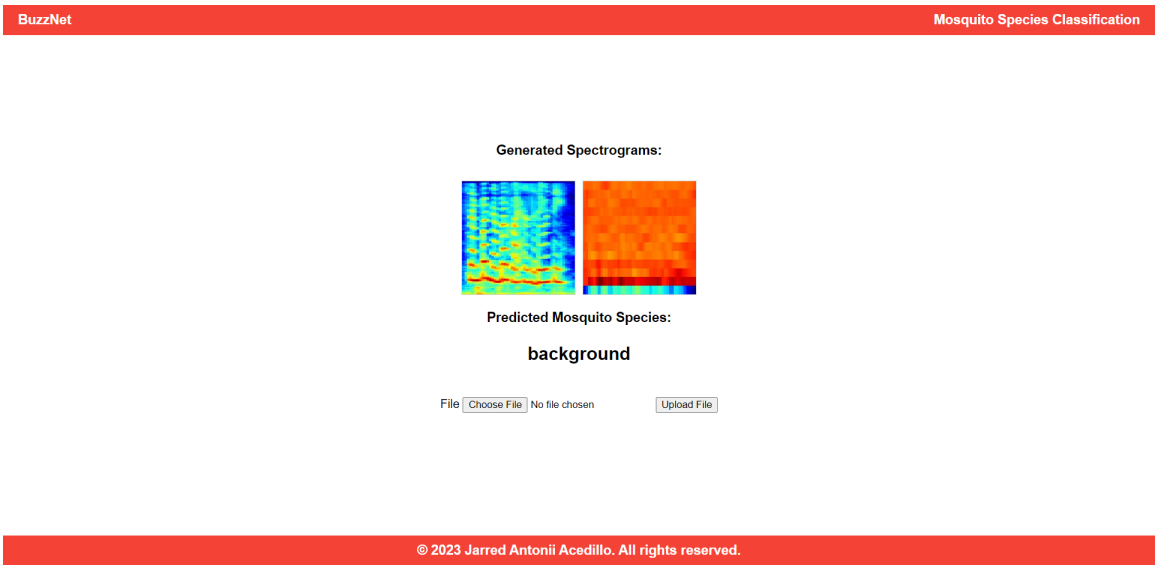

Figure 4.9: Web Application's Predictions for Background Class Audio File

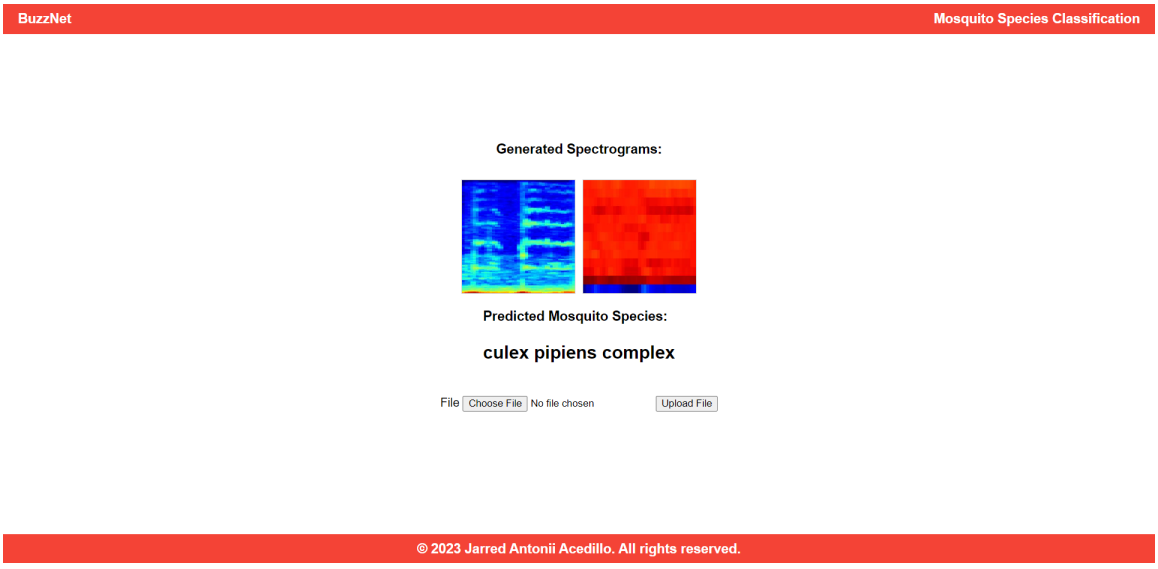

Figure 4.10: Web Application's Predictions for Culex pipiens complex Class Audio File

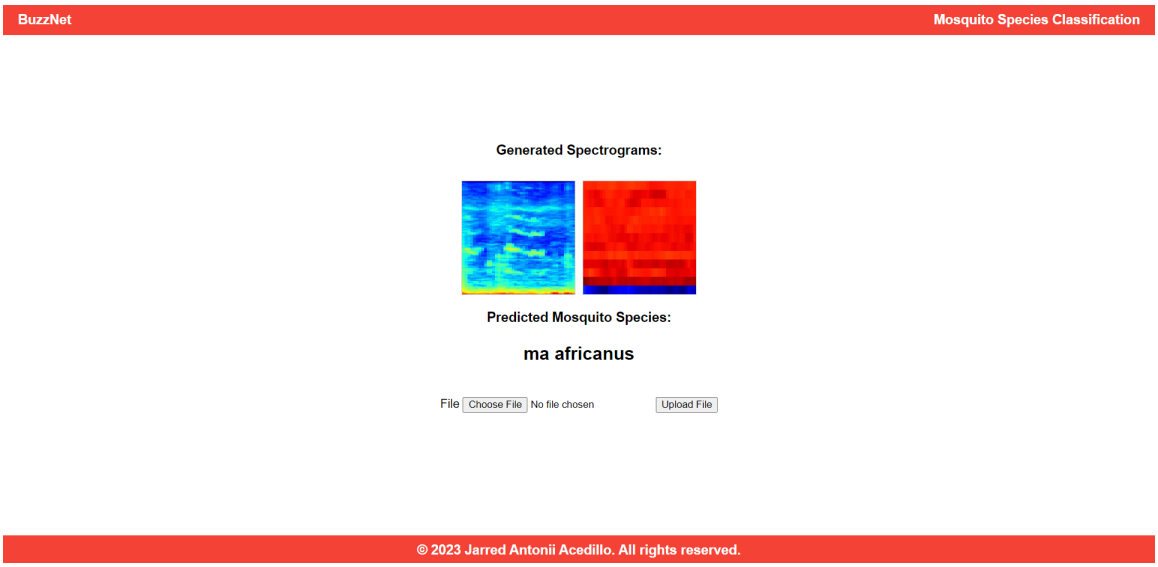

Figure 4.11: Web Application's Predictions for Ma. africanus Class Audio File

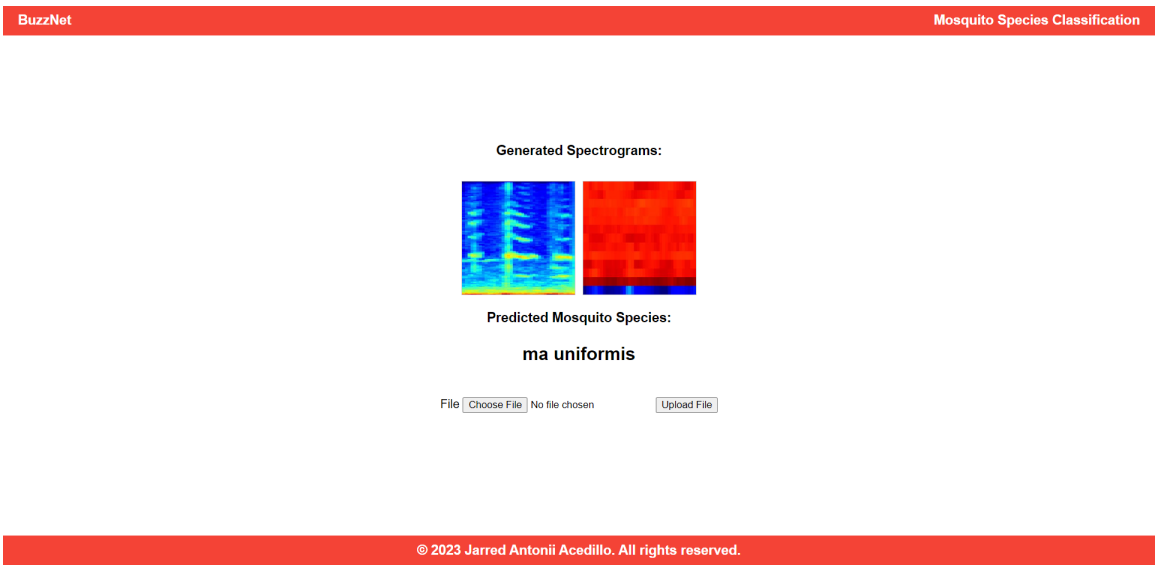

Figure 4.12: Web Application's Predictions for Ma. uniformis Class Audio File

As shown in Figures 4.3 through 4.12, the web application was able to successfully predict all the classes correctly. This entails that the model was successfully deployed as a web application and that it can accurately classify mosquito species using audio files with wav format.

Chapter 5

Conclusions and Recommendations

This research presents BuzzNet, a state-of-the-art web-based Siamese CNN-SVM model, which demonstrates the capability to classify mosquito species based on their audio wingbeat sounds. The findings of this study indicate that the proposed Siamese CNN-SVM model, coupled with RandomizedSearchCV and Log-mel+MFCC spectrogram inputs, achieves competitive or even improved performance compared to existing methods. Notably, the proposed model outperforms the baseline model MozzBNNv2 [9], which utilizes the HumBugDB dataset, as well as the feature extractor MosquitoNet used in the BuzzNet model. The proposed model was able to achieve an accuracy of 90.94% and a macro-f1 score of 77.56%. This study also determined that employing a Siamese CNN-SVM architecture capable of harnessing the discriminative features from two-spectrogram inputs leads to enhanced performance compared to a Single-Input CNN model, which can only accommodate a single-spectrogram input. Specifically, using Log-mel+MFCC as spectrogram inputs yielded the most robust outcomes.

The ablation study investigated the impact of removing specific components on the performance of the best-performing training method. Alongside employing a

pre-trained MosquitoNet as a feature extractor and incorporating an SVM classifier, the introduction of a hyperparameter tuning method, namely RandomizedSearchCV, resulted in substantial improvements in model performance.

The creation of a cost-effective and easily accessible mosquito species identification system, such as a web-based platform like BuzzNet, has the potential to greatly enhance public health outcomes, particularly in regions susceptible to dangerous mosquito vectors, and aid in vector control programs. This study presents an alternative system that employs a state-of-the-art Siamese CNN-SVM model for accurate mosquito species identification. Further research and collaboration with relevant experts from various fields, including healthcare and community engagement, would play a vital role in refining and validating the system, leading to its optimal effectiveness in global endeavors focused on combating mosquito-borne diseases.

Furthermore, this research acknowledges certain limitations that can be addressed in future works, particularly the severe class imbalance within the dataset and the impact of time and hardware constraints on certain design choices. It is widely recognized that augmenting the dataset with a larger and more evenly distributed data can generally yield improved model performance. This can be addressed by utilizing different techniques such as the use of undersampling and oversampling the data, changing the loss function to better accomodate the minority classes, as well as utilizing data augmentation techniques such as SpecMix to create more instances and populate the dataset. Additionally, allocating more computational resources or exploring alternative hardware configurations could facilitate the exploration of more sophisticated models, advanced techniques, or larger datasets, potentially leading to enhanced results. These future directions hold substantial potential to fortify the

robustness and generalizability of forthcoming models within the field of mosquito species classification.

While the proposed method has demonstrated favorable performance, there are still opportunities for further research to enhance the model's effectiveness. One avenue for improvement is the exploration of ensemble models, which combine multiple models or techniques to leverage their individual strengths and improve predictive accuracy and robustness. Additionally, the use of a better pre-trained model can also serve as an avenue for improving overall model performance. Hyperparameter optimization can also be extended to optimize the generation of spectrogram representations, further refining the input features. By addressing these aspects, it is possible to achieve higher model accuracy and improved generalization capabilities.

Overall, this study proposed a web-based Siamese CNN-SVM model capable of classifying mosquito species, which showed significant performance improvements over existing methods [9]. The integration of the Siamese CNN-SVM model with RandomizedSearchCV yielded the most favorable outcomes. The ablation study highlighted the importance of each component, with RandomizedSearchCV demonstrating the most pronounced influence on model performance. The development of a cost-efficient and easily accessible system holds promise for significantly enhancing public health outcomes. Collaborative efforts with experts from various fields can further optimize the system's effectiveness in addressing global challenges associated with mosquito-borne diseases.

Appendix A

Appendix

Performances of Models

Table A.1: Performances of Single-Input CNN-SVM Model

Spectrogram	Training Parameters	Accuracy	Macro-F1
Log-Mel	class_weight="balanced", break_ties=True	**0.8879**	**0.6685**
PCEN		0.8800	0.6244
MFCC		0.7837	0.4080

Table A.2: Performances of Single-Input CNN-SVM Model with RandomizedSearchCV

Spectrogram	Training Parameters	Accuracy	Macro-F1
Log-Mel	C = 1.3762464135817707 gamma = 0.021622588342567 tol = 0.4976246725146712 class_weight="balanced" break_ties=True	**0.9049**	**0.7381**
PCEN	C = 1.1924268714500825 gamma = 0.0102102570196358 tol = 0.4985809671426218 class_weight="balanced" break_ties=True	0.8947	0.6897
MFCC	C = 1.0779362572819235 gamma = 0.0126148910287565 tol = 0.5086537265810241 class_weight="balanced" break_ties=True	0.8143	0.5254

Table A.3: Comparison of Single-Input CNN-SVM Model and Single-Input Fine-tuned CNN-SVM Model Performances

		CNN-SVM Model	
Spectrogram	Metric	without RandomizedSearchCV	with RandomizedSearchCV
Log-Mel	Accuracy	0.8879	**0.9049**
	Macro-F1	0.6685	**0.7381**
PCEN	Accuracy	0.8800	**0.8947**
	Macro-F1	0.6244	**0.6897**
MFCC	Accuracy	0.7837	**0.8143**
	Macro-F1	0.4080	**0.5254**

Table A.4: Performances of Siamese CNN-SVM Model

Spectrogram Pair	Training Parameters	Accuracy	Macro-F1
Log-Mel+MFCC	class_weight="balanced", break_ties=True	**0.8854**	**0.6585**
Log-Mel+PCEN		0.8584	0.6041
MFCC+PCEN		0.8618	0.5953

Table A.5: Performances of Siamese CNN-SVM Model with RandomizedSearchCV

Spectrogram Pair	Training Parameters	Accuracy	Macro-F1
Log-Mel+MFCC	C = 1.9884643826792696 gamma = 0.01449109913934881 tol = 0.56851248360355033 class_weight="balanced" break_ties=True	**0.9094**	**0.7756**
Log-Mel+PCEN	C = 1.1931801455619087 gamma = 0.01081140326751328 tol = 0.55859152986547102 class_weight="balanced" break_ties=True	0.8890	0.7294
MFCC+PCEN	C = 1.3887395160488329 gamma = 0.01961095726457633 tol = 0.50271274109647551 class_weight="balanced" break_ties=True	0.8913	0.7073

Table A.6: Comparison of Siamese CNN-SVM Model and Siamese Fine-tuned CNN-SVM Model Performances

		Siamese CNN-SVM Model	
Spectrogram Pair	Metric	without RandomizedSearchCV	with RandomizedSearchCV
Log-Mel+MFCC	Accuracy	0.8845	**0.9094**
	Macro-F1	0.6585	**0.7756**
Log-Mel+PCEN	Accuracy	0.8584	**0.8890**
	Macro-F1	0.6041	**0.7294**
MFCC+PCEN	Accuracy	0.8618	**0.8913**
	Macro-F1	0.5953	**0.7073**

Table A.7: Comparison of Siamese CNN Model and Siamese Fine-tuned CNN-SVM Model Performances

		Model	
Spectrogram Pair	Metric	Siamese CNN	Siamese CNN-SVM
Log-Mel+MFCC	Accuracy	**0.9094**	**0.9094**
	Macro-F1	0.7543	**0.7756**
Log-Mel+PCEN	Accuracy	0.8856	**0.8890**
	Macro-F1	0.7254	**0.7294**
MFCC+PCEN	Accuracy	0.8811	**0.8913**
	Macro-F1	0.6451	**0.7073**

Table A.8: Comparison between Classification Reports of Siamese CNN Log-Mel+MFCC and Siamese CNN-SVM Log-Mel+MFCC

	Siamese CNN Model			Siamese CNN-SVM Model			
	Metric						
	precision	recall	f1-score	precision	recall	f1-score	instances
ae aegypti	0.7857	**0.7857**	0.7857	**0.9167**	**0.7857**	**0.8462**	14
an arabiensis	**0.9730**	0.9664	**0.9697**	0.9238	**0.9765**	0.9494	298
an coustani	0.8000	**0.5714**	0.6667	**1.0000**	**0.5714**	**0.7273**	14
an funestus ss	**0.8367**	0.6721	0.7455	0.7963	**0.7049**	**0.7478**	61
an squamosus	0.6071	**0.7727**	0.6800	**0.7143**	0.6818	**0.6977**	22
audio	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	80
background	**0.9929**	**0.9964**	**0.9947**	**0.9929**	**0.9964**	**0.9947**	280
culex pipiens complex	0.6500	**0.7927**	0.7143	**0.7011**	0.7439	**0.7219**	82
ma africanus	0.6250	0.4167	0.5000	**0.6667**	**0.5000**	**0.5714**	12
ma uniformis	0.5294	**0.4500**	0.4865	**0.5625**	**0.4500**	**0.5000**	20
accuracy			**0.9094**			**0.9094**	883
macro avg	0.7800	**0.7424**	0.7543	**0.8274**	0.7411	**0.7756**	883
weighted avg	**0.9127**	**0.9094**	**0.9089**	0.9073	**0.9094**	0.9065	883

Classification Reports

Table A.9: Classification Report of Single-Input CNN-SVM using Log-mel without RandomizedSearchCV

	precision	recall	f1-score	instances
ae aegypti	0.7143	0.7143	0.7143	14
an arabiensis	0.8750	0.9866	0.9274	298
an coustani	0.8000	0.2857	0.4211	14
an funestus ss	0.8000	0.6557	0.7207	61
an squamosus	0.7500	0.4091	0.5294	22
audio	1.0000	1.0000	1.0000	80
background	0.9825	1.0000	0.9912	280
culex pipiens complex	0.6556	0.7195	0.6860	82
ma africanus	0.6667	0.1667	0.2667	12
ma uniformis	0.7500	0.3000	0.4286	20
accuracy			0.8879	883
macro avg	0.7994	0.6238	0.6685	883
weighted avg	0.8823	0.8879	0.8759	883

Table A.10: Classification Report of Single-Input CNN-SVM using Log-mel with RandomizedSearchCV

	precision	recall	f1-score	instances
ae aegypti	0.7059	0.8571	0.7742	14
an arabiensis	0.9331	0.9832	0.9575	298
an coustani	0.6154	0.5714	0.5926	14
an funestus ss	0.8400	0.6885	0.7568	61
an squamosus	0.7647	0.5909	0.6667	22
audio	1.0000	1.0000	1.0000	80
background	0.9823	0.9929	0.9876	280
culex pipiens complex	0.6897	0.7317	0.7101	82
ma africanus	0.5714	0.3333	0.4211	12
ma uniformis	0.6000	0.4500	0.5143	20
accuracy			0.9049	883
macro avg	0.7703	0.7199	0.7381	883
weighted avg	0.9004	0.9049	0.9008	883

Table A.11: Classification Report of Single-Input CNN-SVM using PCEN without RandomizedSearchCV

	precision	recall	f1-score	instances
ae aegypti	1.0000	0.7857	0.8800	14
an arabiensis	0.8353	0.9698	0.8975	298
an coustani	1.0000	0.1429	0.2500	14
an funestus ss	0.8776	0.7049	0.7818	61
an squamosus	0.6364	0.3182	0.4242	22
audio	1.0000	1.0000	1.0000	80
background	0.9790	1.0000	0.9894	280
culex pipiens complex	0.6630	0.7439	0.7011	82
ma africanus	0.0000	0.0000	0.0000	12
ma uniformis	0.8000	0.2000	0.3200	20
accuracy			0.8800	883
macro avg	0.7791	0.5865	0.6244	883
weighted avg	0.8708	0.8800	0.8621	883

Table A.12: Classification Report of Single-Input CNN-SVM using PCEN with RandomizedSearchCV

	precision	recall	f1-score	instances
ae aegypti	1.0000	0.7857	0.8800	14
an arabiensis	0.9085	0.9664	0.9366	298
an coustani	0.4286	0.2143	0.2857	14
an funestus ss	0.8824	0.7377	0.8036	61
an squamosus	0.6250	0.4545	0.5263	22
audio	1.0000	1.0000	1.0000	80
background	0.9859	1.0000	0.9929	280
culex pipiens complex	0.6562	0.7683	0.7079	82
ma africanus	0.5000	0.3333	0.4000	12
ma uniformis	0.4615	0.3000	0.3636	20
accuracy			0.8947	883
macro avg	0.7448	0.6560	0.6897	883
weighted avg	0.8872	0.8947	0.8881	883

Table A.13: Classification Report of Single-Input CNN-SVM using MFCC without RandomizedSearchCV

	precision	recall	f1-score	instances
ae aegypti	0.6667	0.1429	0.2353	14
an arabiensis	0.6787	0.9497	0.7916	298
an coustani	0.0000	0.0000	0.0000	14
an funestus ss	0.6333	0.3115	0.4176	61
an squamosus	0.5000	0.0455	0.0833	22
audio	0.9753	0.9875	0.9814	80
background	0.9298	0.9929	0.9603	280
culex pipiens complex	0.6444	0.3537	0.4567	82
ma africanus	1.0000	0.0833	0.1538	12
ma uniformis	0.0000	0.0000	0.0000	20
accuracy			0.7837	883
macro avg	0.6028	0.3867	0.4080	883
weighted avg	0.7524	0.7837	0.7397	883

Table A.14: Classification Report of Single-Input CNN-SVM using MFCC with RandomizedSearchCV

	precision	recall	f1-score	instances
ae aegypti	0.6000	0.2143	0.3158	14
an arabiensis	0.7690	0.9497	0.8498	298
an coustani	0.5000	0.2143	0.3000	14
an funestus ss	0.6226	0.5410	0.5789	61
an squamosus	0.5000	0.0909	0.1538	22
audio	0.9524	1.0000	0.9756	80
background	0.9452	0.9857	0.9650	280
culex pipiens complex	0.5714	0.3902	0.4638	82
ma africanus	0.6667	0.3333	0.4444	12
ma uniformis	0.3333	0.1500	0.2069	20
accuracy			0.8143	883
macro avg	0.6461	0.4869	0.5254	883
weighted avg	0.7881	0.8143	0.7886	883

Table A.15: Classification Report of Siamese CNN-SVM using Log-mel+MFCC without RandomizedSearchCV

	precision	recall	f1-score	instances
ae aegypti	0.9091	0.7143	0.8000	14
an arabiensis	0.8559	0.9765	0.9122	298
an coustani	1.0000	0.2143	0.3529	14
an funestus ss	0.8113	0.7049	0.7544	61
an squamosus	0.7692	0.4545	0.5714	22
audio	1.0000	1.0000	1.0000	80
background	0.9859	0.9964	0.9911	280
culex pipiens complex	0.6824	0.7073	0.6946	82
ma africanus	0.3333	0.0833	0.1333	12
ma uniformis	0.5000	0.3000	0.3750	20
accuracy			0.8845	883
macro avg	0.7847	0.6152	0.6585	883
weighted avg	0.8768	0.8845	0.8722	883

Table A.16: Classification Report of Siamese CNN-SVM using Log-mel+MFCC with RandomizedSearchCV

	precision	recall	f1-score	instances
ae aegypti	0.9167	0.7857	0.8462	14
an arabiensis	0.9238	0.9765	0.9494	298
an coustani	1.0000	0.5714	0.7273	14
an funestus ss	0.7963	0.7049	0.7478	61
an squamosus	0.7143	0.6818	0.6977	22
audio	1.0000	1.0000	1.0000	80
background	0.9929	0.9964	0.9947	280
culex pipiens complex	0.7011	0.7439	0.7219	82
ma africanus	0.6667	0.5000	0.5714	12
ma uniformis	0.5625	0.4500	0.5000	20
accuracy			0.9094	883
macro avg	0.8274	0.7411	0.7756	883
weighted avg	0.9073	0.9094	0.9065	883

Table A.17: Classification Report of Siamese CNN-SVM using Log-mel+PCEN without RandomizedSearchCV

	precision	recall	f1-score	instances
ae aegypti	1.0000	0.5000	0.6667	14
an arabiensis	0.7784	0.9664	0.8623	298
an coustani	1.0000	0.2857	0.4444	14
an funestus ss	0.8919	0.5410	0.6735	61
an squamosus	0.7273	0.3636	0.4848	22
audio	1.0000	1.0000	1.0000	80
background	0.9689	1.0000	0.9842	280
culex pipiens complex	0.6790	0.6707	0.6748	82
ma africanus	0.0000	0.0000	0.0000	12
ma uniformis	0.7500	0.1500	0.2500	20
accuracy			0.8584	883
macro avg	0.7795	0.5478	0.6041	883
weighted avg	0.8520	0.8584	0.8382	883

Table A.18: Classification Report of Siamese CNN-SVM using Log-mel+PCEN with RandomizedSearchCV

	precision	recall	f1-score	instances
ae aegypti	0.7857	0.7857	0.7857	14
an arabiensis	0.9025	0.9631	0.9318	298
an coustani	0.6667	0.5714	0.6154	14
an funestus ss	0.8571	0.5902	0.6990	61
an squamosus	0.6667	0.3636	0.4706	22
audio	1.0000	1.0000	1.0000	80
background	0.9722	1.0000	0.9859	280
culex pipiens complex	0.6452	0.7317	0.6857	82
ma africanus	0.7273	0.6667	0.6957	12
ma uniformis	0.5385	0.3500	0.4242	20
accuracy			0.8890	883
macro avg	0.7762	0.7022	0.7294	883
weighted avg	0.8843	0.8890	0.8827	883

Table A.19: Classification Report of Siamese CNN-SVM using MFCC+PCEN without RandomizedSearchCV

	precision	recall	f1-score	instances
ae aegypti	0.8000	0.5714	0.6667	14
an arabiensis	0.7898	0.9832	0.8759	298
an coustani	0.5714	0.2857	0.3810	14
an funestus ss	0.8372	0.5902	0.6923	61
an squamosus	0.7500	0.4091	0.5294	22
audio	1.0000	0.9750	0.9873	80
background	0.9790	1.0000	0.9894	280
culex pipiens complex	0.7286	0.6220	0.6711	82
ma africanus	0.0000	0.0000	0.0000	12
ma uniformis	0.4000	0.1000	0.1600	20
accuracy			0.8618	883
macro avg	0.6856	0.5537	0.5953	883
weighted avg	0.8426	0.8618	0.8424	883

Table A.20: Classification Report of Siamese CNN-SVM using MFCC+PCEN with RandomizedSearchCV

	precision	recall	f1-score	instances
ae aegypti	0.8125	0.9286	0.8667	14
an arabiensis	0.8875	0.9799	0.9314	298
an coustani	0.5385	0.5000	0.5185	14
an funestus ss	0.8478	0.6393	0.7290	61
an squamosus	0.6923	0.4091	0.5143	22
audio	1.0000	1.0000	1.0000	80
background	0.9859	1.0000	0.9929	280
culex pipiens complex	0.6914	0.6829	0.6871	82
ma africanus	0.5714	0.3333	0.4211	12
ma uniformis	0.5000	0.3500	0.4118	20
accuracy			0.8913	883
macro avg	0.7527	0.6823	0.7073	883
weighted avg	0.8833	0.8913	0.8838	883

ROC Curves

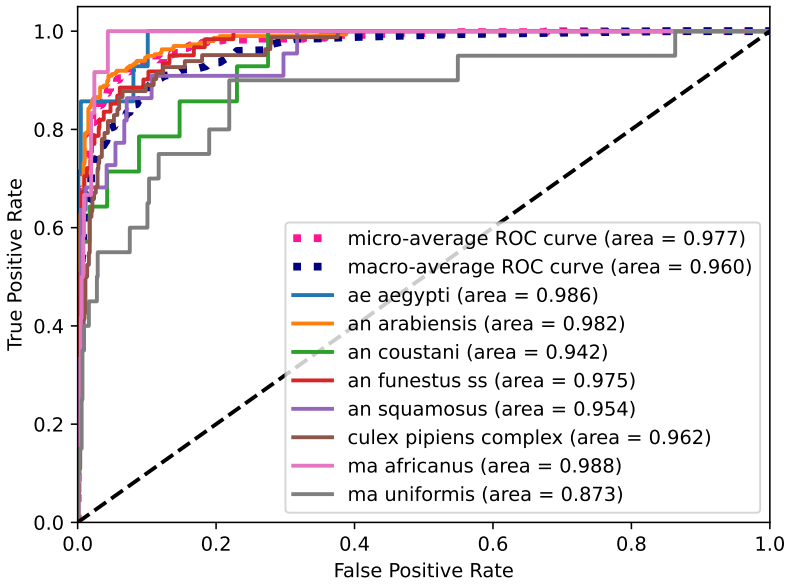

Figure A.1: ROC Curve for Siamese CNN-SVM with RandomizedSearchCV using Log-mel+MFCC

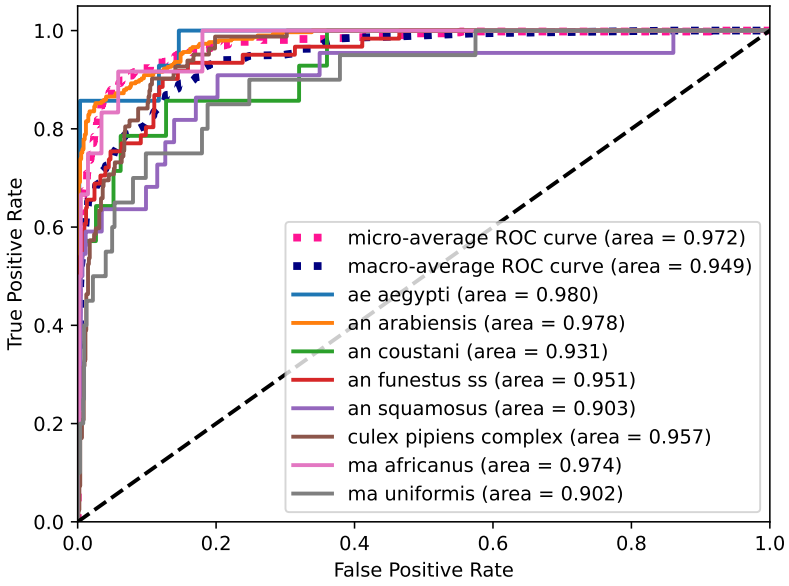

Figure A.2: ROC Curve for Siamese CNN-SVM with RandomizedSearchCV using Log-mel+PCEN

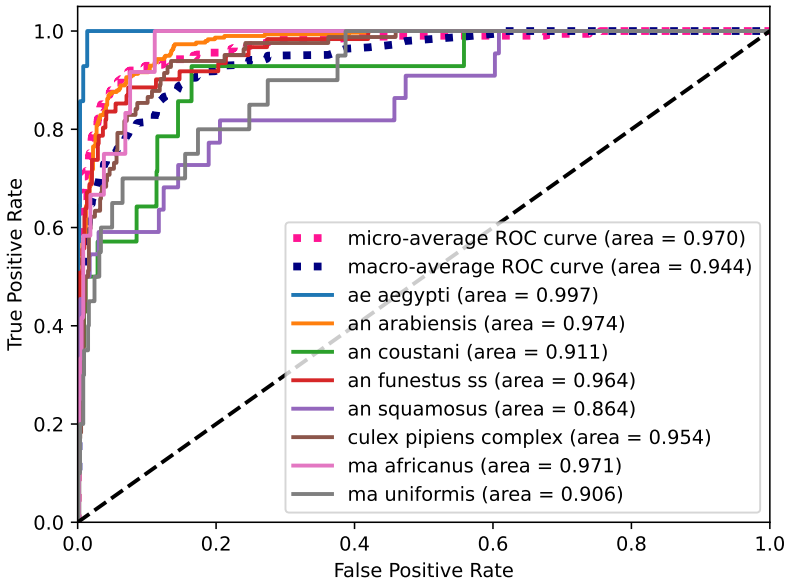

Figure A.3: ROC Curve for Siamese CNN-SVM with RandomizedSearchCV using MFCC+PCEN

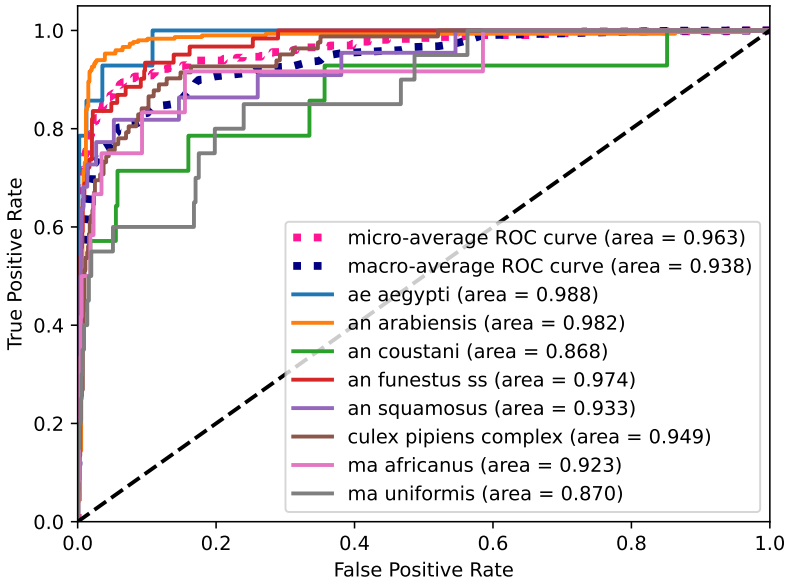

Figure A.4: ROC Curve for Siamese CNN-SVM without RandomizedSearchCV using Log-mel+MFCC

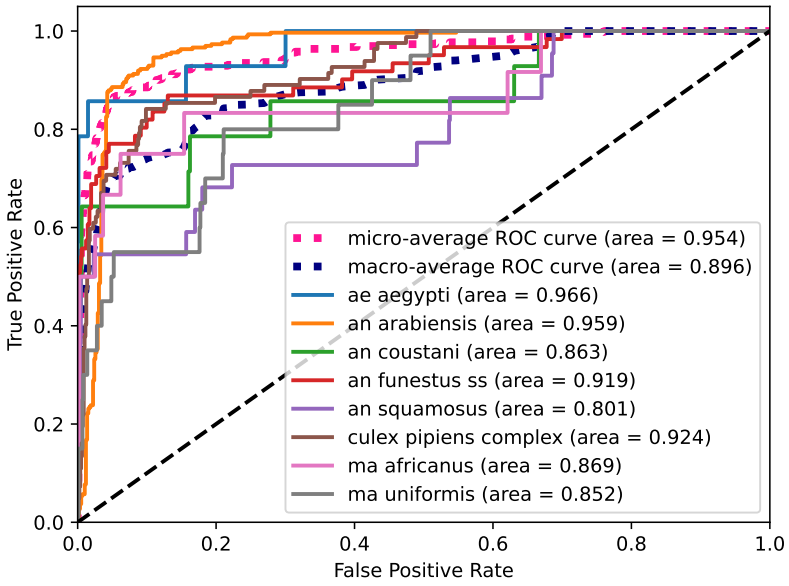

Figure A.5: ROC Curve for Siamese CNN-SVM without RandomizedSearchCV using Log-mel+PCEN

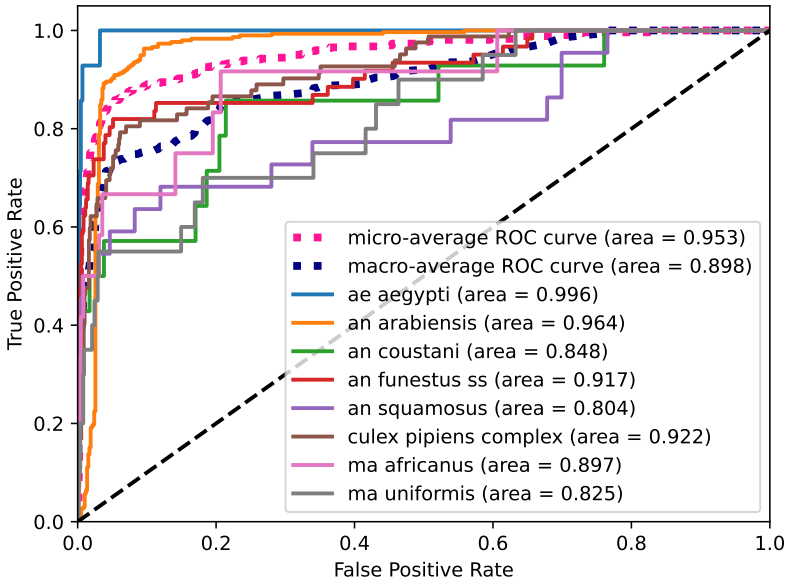

Figure A.6: ROC Curve for Siamese CNN-SVM without RandomizedSearchCV using MFCC+PCEN

Bibliography

[1] Savita Ahlawat and Amit Choudhary. Hybrid cnn-svm classifier for handwritten digit recognition. *Procedia Computer Science*, 167:2554–2560, 2020.

[2] Saad Albawi, Tareq Abed Mohammed, and Saad Al-Zawi. Understanding of a convolutional neural network. In *2017 international conference on engineering and technology (ICET)*, pages 1–6. Ieee, 2017.

[3] Bernhard E Boser, Isabelle M Guyon, and Vladimir N Vapnik. A training algorithm for optimal margin classifiers. In *Proceedings of fifth annual workshop on Computational learning theory*, pages 144–152, 1992.

[4] Keunwoo Choi, György Fazekas, Mark Sandler, and Kyunghyun Cho. Transfer learning for music classification and regression tasks. *arXiv preprint arXiv:1703.09179*, 2017.

[5] Mingxing Duan, Kenli Li, Canqun Yang, and Keqin Li. A hybrid deep learning cnn–elm for age and gender classification. *Neurocomputing*, 275:448–461, 2018.

[6] Eleftherios Fanioudakis, Matthias Geismar, and Ilyas Potamitis. Mosquito wingbeat analysis and classification using deep learning. In *26th European Signal Processing Conference (EUSIPCO)*, pages 2410–2414. IEEE, 2018.

[7] Masato Hagiwara, Benjamin Hoffman, Jen-Yu Liu, Maddie Cusimano, Felix Effenberger, and Katie Zacarian. Beans: The benchmark of animal sounds. In

ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pages 1–5. IEEE, 2023.

[8] Devesh Khandelwal, Sean Campos, Shwetha Nagaraj, Fred Nugen, and Alberto Todeschini. Deep learning-based acoustic mosquito detection in noisy conditions using trainable kernels and augmentations. *arXiv preprint arXiv:2207.13843*, 2022.

[9] Ivan Kiskin, Marianne Sinka, Adam D Cobb, Waqas Rafique, Lawrence Wang, Davide Zilli, Benjamin Gutteridge, Rinita Dam, Theodoros Marinos, Yunpeng Li, et al. Humbugdb: a large-scale acoustic mosquito dataset. *arXiv preprint arXiv:2110.07607*, 2021.

[10] HM Veena Kumari, DS Suresh, and PE Dhananjaya. Clinical data analysis and multilabel classification for prediction of dengue fever by tuning hyperparameter using gridsearchcv. In *2022 14th International Conference on Computational Intelligence and Communication Networks (CICN)*, pages 302–307. IEEE, 2022.

[11] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436–444, 2015.

[12] Yunpeng Li, Davide Zilli, Henry Chan, Ivan Kiskin, Marianne Sinka, Stephen Roberts, and Kathy Willis. Mosquito detection with low-cost smartphones: data acquisition for malaria research. *arXiv preprint arXiv:1711.06346*, 2017.

[13] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988, 2017.

[14] Achmad Lukman, Agus Harjoko, and Chuan-Kay Yang. Classification mfcc feature from culex and aedes aegypti mosquitoes noise using support vector machine. In *2017 International Conference on Soft Computing, Intelligent System and Information Technology (ICSIIT)*, pages 17–20. IEEE, 2017.

[15] Daniel E Neafsey, Robert M Waterhouse, Mohammad R Abai, Sergey S Aganezov, Max A Alekseyev, James E Allen, James Amon, Bruno Arcà, Peter Arensburger, Gleb Artemov, et al. Highly evolvable malaria vectors: the genomes of 16 anopheles mosquitoes. *Science*, 347(6217):1258522, 2015.

[16] World Health Organization. Office of Library and Health Literature Services. Styles for bibliographic citations : guidelines for who-produced bibliographies, 1988.

[17] Wm H Offenhauser Jr and Morton C Kahn. The sounds of disease-carrying mosquitoes. *The Journal of the Acoustical Society of America*, 21(3):259–263, 1949.

[18] ASSM Pravallika and J Rajanikanth. Improving nids performance by using two decision trees and fs technique. 2021.

[19] D Raj Raman, Reid R Gerhardt, and John B Wilkerson. Detecting insect flight sounds in the field: Implications for acoustical counting of mosquitoes. *Transactions of the ASABE*, 50(4):1481–1485, 2007.

[20] A Sanchez-Ortiz, A Fierro-Radilla, Antonio Arista-Jalife, Manuel Cedillo-Hernandez, Mariko Nakano-Miyatake, Daniel Robles-Camarillo, and V Cuatepotzo-Jiménez. Mosquito larva classification method based on convolutional neural networks. In *2017 International Conference on Electronics, Communications and Computers (CONIELECOMP)*, pages 1–6. IEEE, 2017.

[21] Yaniv Taigman, Ming Yang, Marc'Aurelio Ranzato, and Lior Wolf. Deepface: Closing the gap to human-level performance in face verification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1701–1708, 2014.

[22] Helin Wang, Yuexian Zou, and Dading Chong. Acoustic scene classification with spectrogram processing strategies. *arXiv preprint arXiv:2007.03781*, 2020.

[23] Weitao Xu, Xiang Zhang, Lina Yao, Wanli Xue, and Bo Wei. A multi-view cnn-based acoustic classification system for automatic animal species identification. *Ad Hoc Networks*, 102:102115, 2020.

[24] Hoonbok Yi, Bijaya R Devkota, Jae-seung Yu, Ki-cheol Oh, Jinhong Kim, and Hyun-Jung Kim. Effects of global warming on mosquitoes & mosquito-borne diseases and the new strategies for mosquito control. *Entomological Research*, 44(6):215–235, 2014.

[25] Chongsheng Zhang, Pengyou Wang, Hui Guo, Gaojuan Fan, Ke Chen, and Joni-Kristian Kämäräinen. Turning wingbeat sounds into spectrum images for acoustic insect classification. *Electronics Letters*, 53(25):1674–1676, 2017.

[26] Ming Zhong, Maelle Torterotot, Trevor A Branch, Kathleen M Stafford, Jean-Yves Royer, Rahul Dodhia, and Juan Lavista Ferres. Detecting, classifying, and counting blue whale calls with siamese neural networks. *The Journal of the Acoustical Society of America*, 149(5):3086–3094, 2021.