KANTIPUR ENGINEERING COLLEGE

(Affiliated to Tribhuvan University)

Dhapakhel, Lalitpur

[Subject Code: CT755]

A MAJOR PROJECT FINAL REPORT ON

FEATHERFIND : BIRD SPECIES IDENTIFICATION FROM AUDIO

Submitted by:

Gaurav Giri	**[Kan077bct034]**
Iza K.C.	**[Kan077bct039]**
Prajwal Khatiwada	**[Kan077bct056]**
Samrat Kumar Adhikari	**[Kan077bct074]**

A MAJOR PROJECT SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENT FOR THE DEGREE OF BACHELOR IN COMPUTER ENGINEERING

Submitted to:

Department of Computer and Electronics Engineering

March, 2025

FEATHERFIND : BIRD SPECIES IDENTIFICATION FROM AUDIO

Submitted by:

Gaurav Giri	**[Kan077bct034]**
Iza K.C.	**[Kan077bct039]**
Prajwal Khatiwada	**[Kan077bct056]**
Samrat Kumar Adhikari	**[Kan077bct074]**

Supervised by:

Prof. Dr. Subarna Shakya

Department of Electronics & Computer Engineering

Pulchowk Campus, Institute of Engineering

Tribhuvan University

A MAJOR PROJECT SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENT FOR THE DEGREE OF BACHELOR IN COMPUTER ENGINEERING

Submitted to:

Department of Computer and Electronics Engineering

Kantipur Engineering College

Dhapakhel, Lalitpur

March, 2025

ABSTRACT

This project presents FeatherFind, an innovative system for automated bird species identification using audio recordings. The system combines two deep learning models: a bird sound detection model based on modified ResNet-50 architecture, and a species classification model utilizing EfficientNetB3 and CNN-LSTM architectures. The bird sound detection model achieved 82.97% accuracy with an AUC score of 0.911, effectively distinguishing bird vocalizations from ambient noise. For species classification, the EfficientNetB3 model demonstrated superior performance with 90.73% accuracy across 41 bird species, outperforming the CNN-LSTM model's 74.73% accuracy. The system employs Mel-Spectrograms and Mel-Frequency Cepstral Coefficients (MFCC) for feature extraction, with genetic algorithm optimizing model hyperparameters. Implementation includes a mobile application that enables users to record bird sounds, visualize audio waveforms, and map identified species' locations. The system's high accuracy and practical application demonstrate its potential for both scientific research and citizen science initiatives in avian conservation.

Keywords−*Bird Species Classification, Bioacoustics, Convolutional Neural Network, EfficientNet, LSTM, ResNet, Mel-Spectrogram, MFCC, Genetic Algorithm*

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABBREVIATIONS

AUC	Area Under the Curve
CCE	Categorical Cross-Entropy
CNN	Convolutional Neural Network
DCNN	Deep Convolutional Neural Network
DCT	Discrete Cosine Transform
DFT	Discrete Fourier Transform
DGT	Discrete Gabor Transform
FFT	Fast Fourier Transform
FPR	False Positive Rate
GA	Genetic Algorithm
GPS	Global Positioning System
GRU	Gated Recurrent Network
IDE	Integrated Development Environment
LSTM	Long Short-Term Memory
MAP	Mean Average Precision
MBConv	Mobile Inverted Bottleneck Convolution
MFCC	Mel-Frequency Cepstral Coefficients
MLP	Multilayer Perceptron
ResNet	Residual Network
RNN	Recurrent Neural Network
ROC	Receiver Operating Characteristics
SE	Squeeze and Excitation
SQL	Structured Query Language
STFT	Short-Time Fourier Transform
TPR	True Positive Rate
WSOLA	Waveform Similarity OverLap Add

CHAPTER 1
INTRODUCTION

Globally, the avian kingdom is vast, with over 11,000 species, showing nature's complexity and evolutionary skill. This number, from the International Ornithological Committee as of April 2023, shows the great bird diversity, with each species having its own role and story.[1]

In Nepal, a country known for its rich nature and varied ecosystems, from the lowland Terai to the high Himalayas, there are more than 887 bird species, according to the Himalayan Nature organization. This is over 8% of the world's known bird species, a big number given Nepal's small size.[2]

Many of these species are endangered due to habitat loss, climate change, and human activities. The National Red List of Nepal's Birds identifies 168 nationally threatened bird species, including 68 Critically Endangered, 38 Endangered, and 62 Vulnerable species, as detailed in a publication by the Journal of Threatened Taxa.[3]

The situation of these endangered species shows the need for conservation efforts. Technologies such as audio recognition provide new methods for identifying and monitoring bird populations. By analyzing bird sounds and pictures, researchers can better understand species distribution, behavior, and threats. These technologies not only help conserve endangered species but also support broader biodiversity research.

1.1 Problem Statement

In Nepal, a hotspot of avian biodiversity, accurately identifying and classifying bird species, particularly those that are endangered, is a critical yet complex task. Traditional observation methods are limited by the vast geographical and ecological diversity of the region, making it challenging to monitor and protect these birds effectively. The necessity for precise identification is paramount for conservation efforts aimed at maintaining ecosystem balance. To address this, there is a pressing need for a method that can overcome these constraints by leveraging advanced technologies capable of distin-

guishing between the myriad of bird calls and songs, as well as visual markers through image classification. Such a method promises to automate the identification process, enhancing accuracy and efficiency in monitoring endangered species.

1.2 Objective

To develop and implement an integrated technological solution that utilizes advanced audio recognition technique for the accurate identification and monitoring of bird species in Nepal, with an emphasis on endangered species.

1.3 Application Scope

1. **Conservation Efforts:**
   This system will enhance conservation by enabling accurate monitoring of bird populations, helping track endangered species and take a step towards habitat protection.
2. **Biodiversity Monitoring:**
   Automated identification will aid biodiversity monitoring by processing large datasets, helping detect species distribution on bird communities.
3. **Ecological Research:**
   Researchers can use the system to study bird migration, and habitat use, providing crucial data for modeling ecosystems and understanding ecological interactions.
4. **Environmental Education and Awareness:**
   Integrated into educational programs, this tool will raise public awareness about biodiversity and conservation, engaging students and scientists in bird identification.
5. **Bird viewing:**
   Bird enthusiasts will benefit from this system as it will enhance bird watching experiences by providing instant identification of bird species

1.4 Features

1. **Species Identification Using Audio:**

   The app allows users to record bird sounds in real-time using their device's microphone or upload pre-recorded audio files. Advanced noise filtering techniques isolate bird calls from background noise, and sound wave analysis helps in identifying distinct frequency patterns. Machine learning algorithms, trained on a vast database of bird calls, match the recorded sound to identify the bird species accurately.

2. **Mapping Identified Bird Habitat:**

   The app tags the location of identified birds using GPS, providing detailed habitat information typical of each species. Integrating with mapping services, it displays bird sightings on an interactive map, generating heat maps to show species density and distribution. Additionally, it tracks and visualizes bird migration patterns over time, helping users understand seasonal movements.

3. **Provide Description About the Birds:**

   For each identified bird species, the app offers detailed profiles that include scientific and common names, physical descriptions, and conservation status. It also provides audio and visual media for reference, along with information on the bird's behavior, diet, and typical habitats, enriching the user's understanding of the species.

1.5 Feasibility Study

Before implementation of project design, the feasibility analysis of the project must be done to move any further. The feasibility analysis of the project gives an idea on how the project will perform and its impact in the real world scenario. So, it is of utmost importance.

1.5.1 Economic Feasibility

Our system is economically achievable as a result of the development of several tools, libraries, and frameworks. Since all the software required to construct it is free and readily available online, this project is incredibly cost-effective. Only time and effort are needed to create a worthwhile, genuinely passive system. The project doesn't come at a substantial cost. From an economic standpoint, the project appears successful in this sense.

1.5.2 Technical Feasibility

The software needed to implement a project can be downloaded from a wide variety of online resources. Technically speaking, the project is feasible as the necessary software is easily available. We were able to learn the information we required for the project through a variety of online sources, including classes. All the libraries and data are accessible online for free because this project does not require any licensing costs. It is technically possible if one has the necessary information and resources.

1.5.3 Operational Feasibility

The project aims to enhance bird species identification through audio classification, making bird sound recognition more accessible and efficient. This solution is particularly beneficial for ornithologists, bird watchers, and environmental researchers who require accurate and quick identification of bird species based on their calls. The project leverages advanced audio signal processing and deep learning algorithms to classify bird sounds, ensuring high accuracy and reliability. Given the widespread availability of mobile devices and recording equipment, the project is operationally feasible, as it can be easily integrated into existing workflows and tools used by bird enthusiasts and professionals. By providing an efficient method for bird sound classification, the project supports a sizable community interested in avian studies and conservation, ensuring practical applicability and ease of use.

1.6 System Requirements

1.6.1 Development Requirements

Table 1.1: Development Requirements

Software Requirements	**Hardware Requirements**
Programming Language: Python, Dart, Javascript	RAM: >= 8 GB Megapixels
Design Tools: Figma, Canva, Draw.io	CPU: i5 10th
Libraries: Librosa, PyAudio, Pytorch	GPU: P100 (Recommended)
Framework: Flutter, Django RestFramework	Storage: >= 50 GB
IDEs: VSCode, Android Studio	

1.6.2 Deployment Requirements

Table 1.2: Deployment Requirements

Software Requirements	**Hardware Requirements**
Android: >= 10	RAM: > 4 GB
Read/Write FileSystem	Storage: >= 20 GB
Internet Accessibility	Recording Quality >= 256 Kbps, 48 KHz
Database: MySQL	

CHAPTER 2
LITERATURE REVIEW

This literature review explores the progression of methodologies and technologies in the field, with a particular focus on the use of Convolutional Neural Networks (CNN) and Long Short-Term Memory (LSTM) networks for audio-based bird species identification. The review also examines the challenges associated with dataset quality and diversity, and the innovative strategies employed to address these issues, providing a comprehensive overview of the current state of research and future directions in avian bioacoustics.

2.1 Related Works

BirdNET[4] is a cutting-edge research platform developed through collaboration between the K. Lisa Yang Center for Conservation Bioacoustics at the Cornell Lab of Ornithology and the Chair of Media Informatics at Chemnitz University of Technology. Its primary aim is to detect and classify bird sounds using machine learning technologies, serving both experts and citizen scientists in their efforts to monitor and protect bird populations.

BirdNET can identify around 3,000 of the world's most common bird species, with plans to expand this number. Features such as a live submissions map and a Twitter bot are included to engage the community and share real-time data. The project is supported by donations and collaborations, offering opportunities for researchers and developers to contribute to its growth. BirdNET serves as an invaluable tool for bird enthusiasts, conservationists, and biologists alike, providing innovative solutions for large-scale acoustic monitoring and contributing to the conservation and understanding of avian biodiversity.

The BirdCLEF 2023 competition on Kaggle is a significant data science challenge that falls under the broader LifeCLEF initiative, aimed at pushing the boundaries of species identification and biodiversity monitoring through technological innovation. This par-

ticular competition focuses on the development of machine learning models that can identify bird species based on audio recordings. It presents a complex and realistic challenge due to the diversity of the audio recordings, which are collected from various environments and feature a wide range of bird species.

2.2 Related Research

The study[5] "Audio Classifier for Automatic Identification of Endangered Bird Species of Nepal" focuses on using deep learning techniques to identify endangered bird species from audio recordings. The dataset, collected from xeno-canto.org, comprises 2215 audio recordings of 41 bird species, 38 of which are endangered. This dataset was expanded to 6733 recordings through 10-second audio splitting and Gaussian noise augmentation, with 5407 recordings used for training, 639 for validation, and 687 for testing. The methodology involved handling imbalanced class distribution through data augmentation, employing Mel spectrograms and Mel-Frequency Cepstral Coefficients (MFCCs) for feature extraction, and developing a custom Convolutional Neural Network (CNN) model and an EfficientNet model. The hyperparameters of these models were optimized using a genetic algorithm. The Mel spectrograms were created using Short-Time Fourier Transform, converting amplitudes to decibel scale, and applying Mel filter banks to the spectrograms. Similarly, MFCCs were derived by framing the audio signals, applying Discrete Fourier Transform, logarithmic scaling, Mel scaling, and Discrete Cosine Transform. The EfficientNet architecture utilized compound scaling for network depth, width, and resolution. The findings indicated that the proposed approach achieved satisfactory results in classifying the bird species. Model I, using Mel Spectrogram and EfficientNet, achieved an F1-score of 79%, while Model II, using Mel Spectrogram and Custom CNN, achieved 64%, and Model III, using MFCC and EfficientNet, achieved 72%. However, limitations include the relatively small dataset size and the need for further enhancement in model robustness and accuracy.

Sevilla et al.[6] introduced an innovative approach to bird sound classification with their study 'Audio Bird Classification with Inception-v4 extended with Time and Time-Frequency Attention Mechanisms'. The datasets employed include various bird sounds,

prominently from the BirdClef2017 challenge, consisting of 1500 bird species recordings. The methodology revolves around treating bird sound classification as an image classification problem through transfer learning. The Inception-v4 model, initially pretrained on ImageNet, was adapted to process time-frequency representations of bird sounds by converting these sounds into RGB images using three log-spectrograms generated via fast Fourier transform at different scales (128, 512, 2048 bins).The findings demonstrate that the model, termed 'Soundception', integrates time and time-frequency attention mechanisms effectively, significantly improving classification accuracy. The results highlight Soundception's outstanding performance, achieving a mean average precision (MAP) of 0.714 in classifying 1500 bird species, 0.616 MAP for background species, and 0.288 MAP for soundscapes with time-codes, making it the top model in the BirdClef2017 challenge across multiple tasks. However, limitations include the incomplete convergence of the model due to computational constraints and the extensive GPU resources required for training. The paper concludes with a discussion on future improvements, such as exploring different scalable optimizations and incorporating stacked GRU layers for better audio-to-image representation learning, underscoring the potential of transfer learning from advanced image classification models to acoustic domains.

The study, conducted by Chandu B et al.[7], outlines a robust methodology for identifying bird species from audio recordings, leveraging a combination of meticulously curated datasets and machine learning techniques. The dataset was manually compiledThe working mechanism for bird identification from audio utilizes the dataset compiled from Xeno Canto, housing a diverse collection of avian vocalizations. Our methodology revolves around the utilization of EfficientNet, a state-of-the-art convolutional neural network architecture known for its balance between accuracy and efficiency. The primary objective of this study is to achieve high levels of accuracy in identifying a broad spectrum of 41 distinct bird species. Here lies the detailed explanation of the methodology, from the working mechanism to model training. from both local recordings and online resources such as xeno-canto.org, which apart from bird songs also contains ambient noise and human voices to simulate real-world con-

ditions. Pre-processing techniques including pre-emphasis, framing, silence removal, and reconstruction were applied to the audio clips to enhance the relevant frequency components and eliminate unnecessary noise, ensuring the purity of the dataset. Spectrograms of these processed clips were generated and used as input for training a convolutional neural network (CNN), specifically AlexNet, chosen for its high accuracy in image classification tasks. Through transfer learning, AlexNet was adapted to recognize bird species from the spectrograms, achieving a classification accuracy of 97% in controlled environments. However, recognizing the variability of real-world conditions, the researchers retrained the model with datasets containing ambient noise, achieving a real-time classification accuracy of 91%. Despite these promising results, the study acknowledges limitations such as the relatively small size of the dataset and the need for further tuning of performance parameters to improve robustness.

'An Ensemble of Convolutional Neural Networks for Audio Classification' [8] presents a comprehensive study on CNN classification using different architectures, data augmentation techniques, and audio signal representations, aimed at enhancing audio classification tasks across various datasets. The study employs three datasets: BIRDZ, CAT, and ESC-50, each offering unique challenges in audio classification. The methodology involves training five convolutional neural networks (CNNs) with four audio representations combined with six different data augmentation methods, resulting in thirty-five subtypes of ensembles. The audio representations include techniques such as the Discrete Gabor Transform (DGT), Waveform Similarity OverLap Add (WSOLA), and Phase Vocoder. The data augmentation methods encompass procedures like short spectrogram augmentation, random time shift, and frequency masking. The CNN architectures are pre-trained models fine-tuned with these augmented datasets to boost classification accuracy. The findings reveal that the ensemble method outperforms standalone networks, achieving 97% accuracy on the BIRDZ dataset, 90.51% on the CAT dataset, andThe working mechanism for bird identification from audio utilizes the dataset compiled from Xeno Canto, housing a diverse collection of avian vocalizations. Our methodology revolves around the utilization of EfficientNet, a state-of-the-art convolutional neural network architecture known for its balance between accuracy

and efficiency.The primary objective of this study is to achieve high levels of accuracy in identifying a broad spectrum of 41 distinct bird species. Here lies the detailed explanation of the methodology, from the working mechanism to model training. 88.65% on the ESC-50 dataset. The study also highlights that the best-performing CNNs are VGG16 and VGG19, with DGT as the most effective signal representation. However, the study acknowledges limitations, such as the computational cost of training ensembles and the variability in performance across different augmentation techniques.

'Analysis of bird call datasets sourced from Xeno-Canto'[9] comprisies of 72,172 samples from 264 bird species in 16-bit wav format with a 16 kHz sampling rate. The methodology involved preprocessing the audio data to filter out low-frequency noise and normalize signal amplitude, followed by generating Mel Spectrograms and Mel-Frequency Cepstral Coefficients (MFCCs) as inputs for deep learning models. The Mel Spectrograms were produced using discrete Fourier transform (DFT), and the MFCCs were derived by applying discrete cosine transform (DCT) to the Mel Spectrogram. The study employed various metrics to evaluate the performance of these methods, including ROC analysis to visualize model effectiveness. Findings indicated that the proposed models showed significant promise in identifying bird species from their calls, with improvements in classification accuracy compared to previous approaches. However, limitations were noted, including potential biases in the dataset due to uneven sample distribution across species and the challenge of background noise affecting signal quality. Future work suggested enhancing noise reduction techniques and exploring more sophisticated neural network architectures to further improve model robustness and accuracy.

The study[10] by Pahuja et al. conducted an in-depth analysis of bird species recognition through acoustic monitoring, utilizing a robust dataset of bird sound samples, meticulously annotated and validated for accuracy. The dataset, referred to as SD, comprises multispecies bird sound recordings, each labeled with species name and sample ID, along with corresponding metadata, providing a comprehensive foundation for model training and evaluation. Methodologically, the research employed a

spectrogram-based feature extraction approach, leveraging Short-Time Fourier Transform (STFT) to capture the intricate temporal and spectral characteristics of bird sounds. This was followed by the application of a Multilayer Perceptron (MLP) classifier to distinguish between different bird species. The findings reveal that the proposed model achieved high recognition accuracy, with some species being identified with perfect precision, recall, and accuracy (100%), though the performance varied across species, with a few showing lower recognition rates (86.9%) and precision/recall values ranging between 5075%. The results demonstrated an overall classification accuracy of 96%, with cross-validation accuracy standing at 81.4%, highlighting the model's robustness yet indicating room for improvement in generalizability across diverse datasets. Despite the promising results, the study acknowledges several limitations, including the variability in recognition accuracy among different species and the potential influence of environmental noise on model performance. Future work is suggested to explore feature and model fusion techniques, integrate the model with cloud-based systems for real-time recognition, and expand the dataset to include a broader range of bird species to enhance the model's applicability and accuracy in practical scenarios.

The dataset used in this study[11] comprises recordings labeled by species from California and Nevada, USA. It includes 91 species, with 30 audio samples per species, amounting to a total of 2,730 MP3 files. The methodology adopted involves three main steps: pre-processing, feature extraction, and deep learning modeling. For feature extraction, MFCCs were obtained using the Python library python_speech_features, with parameters such as sample rate, 13 cepstrum coefficients, 26 filterbank filters, and an FFT size of 512. Mel spectrograms were extracted using the Librosa library, employing parameters such as a sample rate, an FFT window size of 2048, a hop length of 512, and 128 Mel bands. In the deep learning modeling, CNNs and LSTMs were compared for their effectiveness in classifying bird sounds. CNNs demonstrated superior training accuracy with Mel spectrogram features, achieving 99.05% and 98.76% accuracy for 3-second and 1.5-second spectroThe working mechanism for bird identification from audio utilizes the dataset compiled from Xeno Canto, housing a diverse collection of avian vocalizations. Our methodology revolves around the utilization of EfficientNet,

a state-of-the-art convolutional neural network architecture known for its balance between accuracy and efficiency.The primary objective of this study is to achieve high levels of accuracy in identifying a broad spectrum of 41 distinct bird species. Here lies the detailed explanation of the methodology, from the working mechanism to model training.grams, respectively. In contrast, LSTMs achieved lower training accuracies of 75.85% and 73.29% under similar conditions.These results highlight the superior ability of CNNs to leverage the spatial and frequency-related patterns in Mel spectrograms for accurate bird species classification.

Finally, the study[12], 'Acoustic Bird Detection with Deep Convolution Neural Network', presents a comprehensive study of DCNNs pretrained on ImageNet which have been used highly effective for bioacoustic classification, with preprocessing and augmentation playing crucial roles in model performance. The preprocessing pipeline typically involves applying a shallow high-pass filter (Q = 0.707) with a 2 kHz cutoff, resampling to 22,050 Hz, and extracting around 4-second audio chunks, which are transformed into mel spectrograms with 310 mel bands (160-10,300 Hz). Further processing includes removing extreme frequencies, normalizing power spectrograms to decibel units, resizing to 224*224, and converting grayscale spectrograms to RGB for compatibility with ResNet. To improve generalization, data augmentation techniques such as jittering chunk duration, extracting chunks from random positions, adding noise from unrelated audio files, and applying random amplitude scaling are employed. Additional augmentations in the frequency domain include frequency shifting/stretching, piecewise time/frequency resizing, and color jittering (brightness, contrast, saturation, hue). Among these, the most effective methods are noise addition from random files, piecewise time and frequency stretching, and time interval dropout, which enhance robustness by simulating real-world variations in bird vocalizations.

2.3 Research Gap Analysis

2.3.1 Dataset Diversity and Representation

Current State: Existing datasets on bird species in Nepal, such as those from xeno-canto and local ornithological surveys, often lack comprehensive coverage of endangered species. These datasets are typically biased towards more common species, resulting in an underrepresentation of rare and endangered birds.

Gap: There is a critical need for more extensive and balanced datasets that include a wider range of endangered bird species. This would involve targeted data collection efforts in diverse habitats across Nepal, ensuring that the unique vocalizations and behaviors of endangered species are adequately captured.

2.3.2 Handling Environmental Noise

Current State: Studies like those by Sevilla et al.[6] and Lasseck et al.[12] have highlighted the challenges posed by environmental noise in bird sound classification. While some methods, such as noise addition and filtering, have been employed, the impact of noise on model performance remains a significant issue.

Gap: More advanced noise reduction and signal processing techniques are required to enhance the accuracy of bird sound classification in noisy environments. This includes developing models that can effectively distinguish bird calls from background noise and other overlapping sounds.

2.3.3 Real-Time and Multi-Bird Sound Recognition

Current State: Most existing models, including those by Gautam et al.[5] and Chandu B et al.[7], focus on identifying single bird species from audio recordings. However, real-world scenarios often involve multiple birds vocalizing simultaneously.

Gap: There is a need for models capable of real-time processing and recognizing multiple bird species in a single recording. This requires advancements in source separation techniques and the development of algorithms that can handle complex soundscapes.

2.3.4 Integration with Mobile and Cloud-Based Systems

Current State: Some studies, such as those by BirdNET and BirdCLEF, have explored the integration of bird sound classification models with mobile and cloud-based systems. However, these implementations often face challenges related to internet dependency and real-time processing.

Gap: Developing offline capabilities and optimizing models for real-time processing on mobile devices would enhance the usability and accessibility of bird sound classification systems. This includes creating lightweight models that can run efficiently on mobile hardware.

2.3.5 Advanced Feature Extraction Techniques

Current State: Studies have employed various feature extraction techniques, such as Mel Spectrograms and MFCCs, as seen in the works of Sevilla et al.[6] and Pahuja et al.[10]. While these methods have proven effective, there is potential for further enhancement.

Gap: Exploring additional feature extraction techniques and combining multiple features could improve the performance of bird sound classification models. This includes investigating new representations of audio signals that capture more nuanced aspects of bird vocalizations.

2.3.6 Integration with Conservation Efforts

Current State: While some research has explored the use of bioacoustic monitoring for conservation, there is limited integration of these technologies into practical conservation strategies for endangered bird species in Nepal.

Gap: Developing systems that can be seamlessly integrated into conservation programs is essential. This includes creating user-friendly tools for field researchers and conservationists, as well as establishing protocols for using bioacoustic data to inform conservation actions and policies.

CHAPTER 3
METHODOLOGY

This chapter describes the overall system including the bird sound detection and bird species classification.

3.1 System Diagram

The usecase diagram for Bird species identification from Audio and Image is given below:

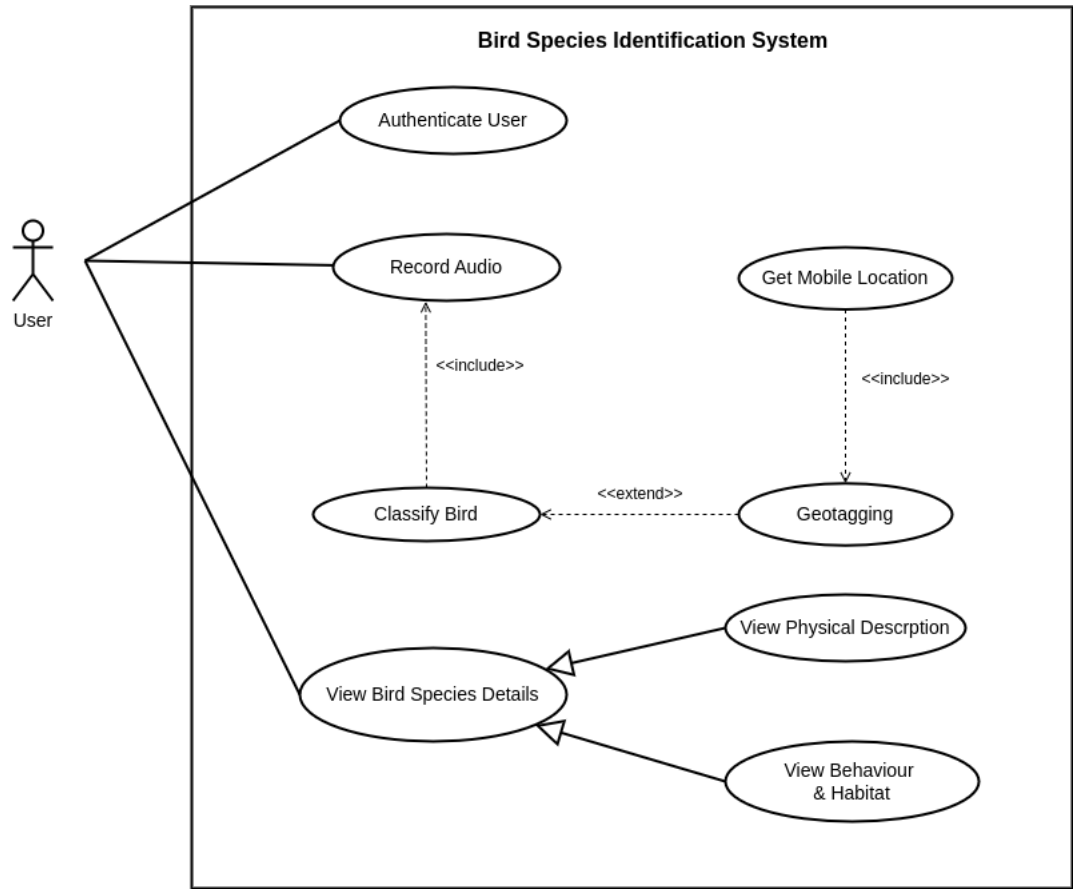

Figure 3.1: Usecase Diagram for FeatherFind

3.2 System Overview

Our project leverages a mobile application to capture audio recordings, which are then processed to identify bird species. The mobile app records the sound and visualizes it

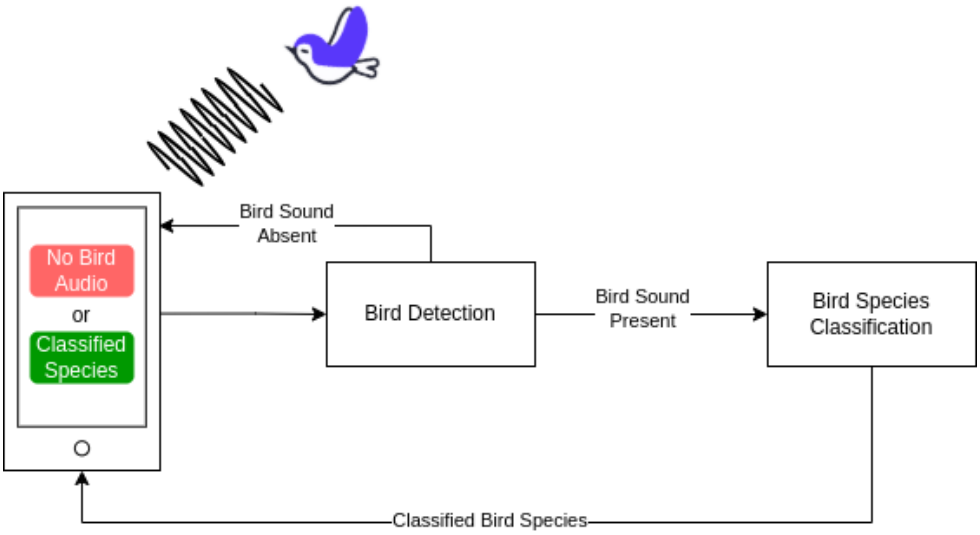

Figure 3.2: System Overview

with waveforms during the recording process. Once the audio is captured, it is sent to our backend system for further analysis.

This model determines whether the recorded audio contains bird sounds. If bird sounds are detected, the audio is then forwarded to the bird species classifier model. The classifier identifies the specific bird species present in the recording. If the bird sound detection model does not detect any bird sounds, the recorded audio is labeled accordingly, indicating the absence of bird sounds.

Upon successful classification, the system maps the identified bird species to the location where the audio was recorded, utilizing the device's GPS data. This approach ensures that only relevant audio recordings are processed for species classification, enhancing the accuracy and efficiency of the system. The integration of Django, Django Rest Framework, and MySQL provides a robust and scalable backend infrastructure to support the application's functionality as shown in Figure 3.2.

3.3 Working Mechanism for Bird Sound Detection

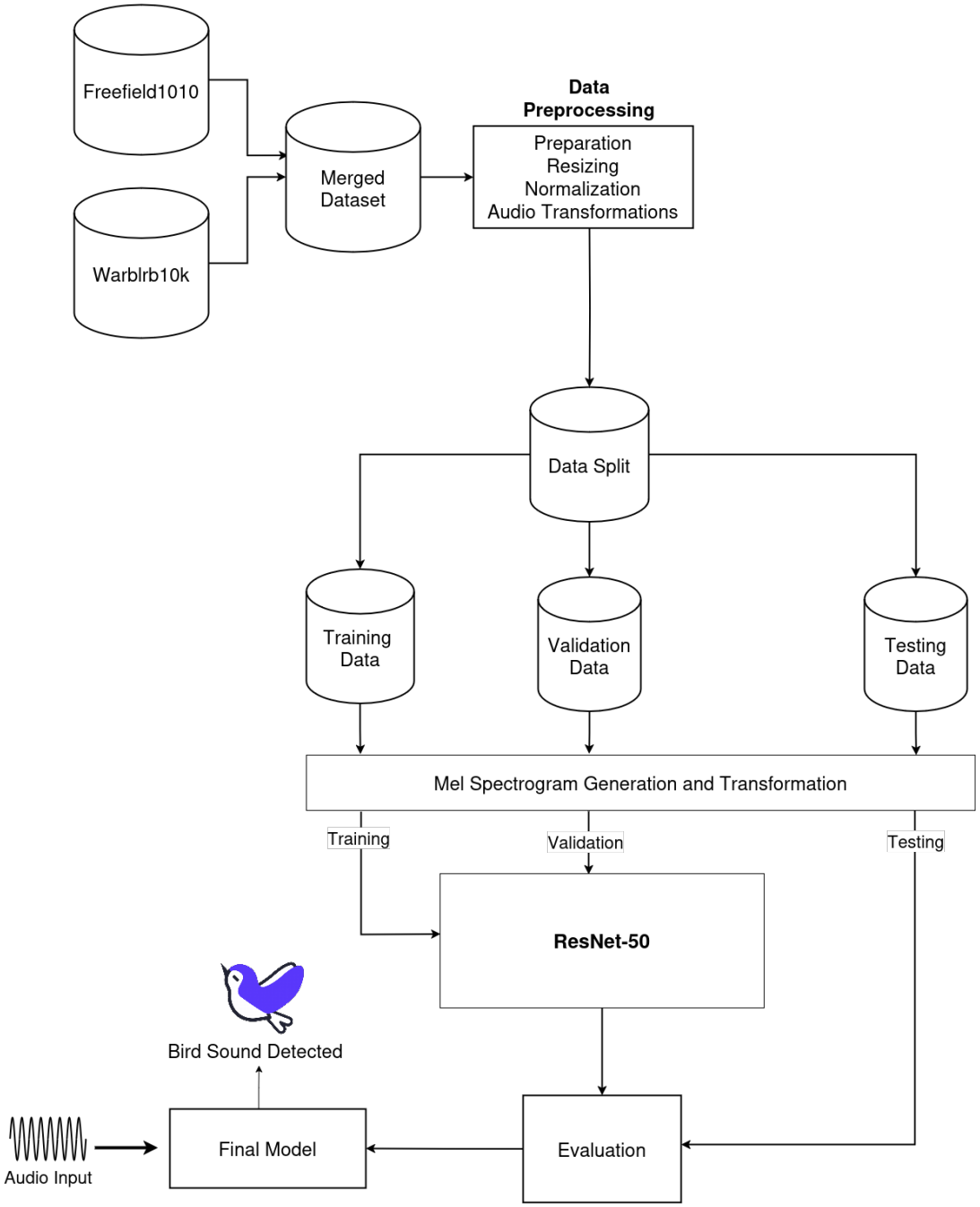

Figure 3.3: Block diagram for the working mechanism for Bird Detection

The bird sound detection process represents the initial filtering stage of our audio classification system, determining whether a given audio clip contains bird vocalizations. The mechanism follows a series of steps as shown in Figure 3.3.This module uses a modified ResNet-50 architecture, pre-trained on ImageNet and fine-tuned for binary classification of bird sounds. By converting audio inputs into mel spectrograms, the model can effectively distinguish bird vocalizations from ambient noise and other.

3.3.1 Data Collection

The dataset used for the Bird Sound Detection Model includes two primary sources:

- **Field recordings, worldwide (”freefield1010”):** This collection consists of 7,690 excerpts from field recordings around the world, gathered by the FreeSound project and standardized for research. The dataset is diverse in location and environment and has been annotated for the presence or absence of birds. The distribution of this dataset is shown in Figure 3.4.

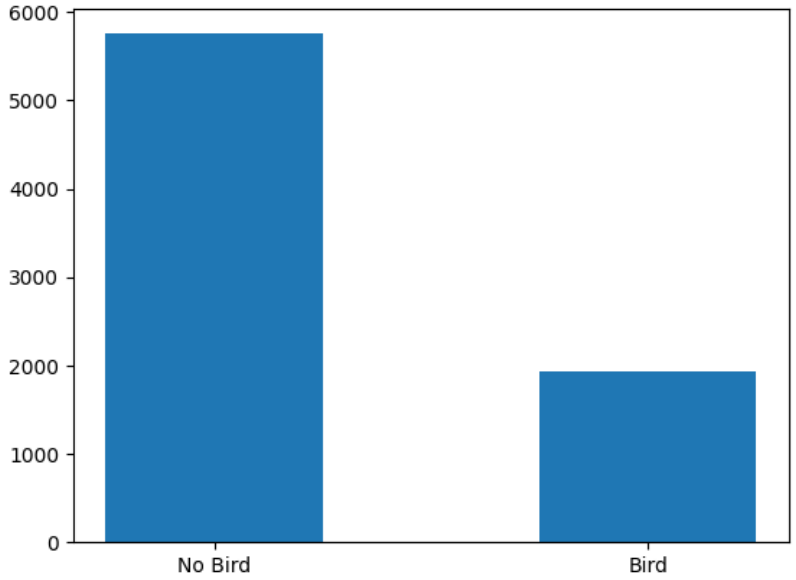

Figure 3.4: Dataset distribution for freefield1010

- **Crowdsourced dataset, UK (”warblrb10k”):** This dataset includes 8,000 smartphone audio recordings around the UK, crowdsourced by users of Warblr, the bird recognition app. The audio covers a wide distribution of UK locations and environments and includes weather noise, traffic noise, human speech, and even human bird imitations. The distribution of this dataset is shown in Figure 3.5.

Instead of using the datasets separately, we merged them to create a single dataset comprising 15,690 audio recordings—7,710 without bird sounds and 7,980 with bird sounds. The merged dataset was then partitioned into training, testing, and validation sets in an 80:10:10 ratio, ensuring a comprehensive and balanced dataset for model training and evaluation.

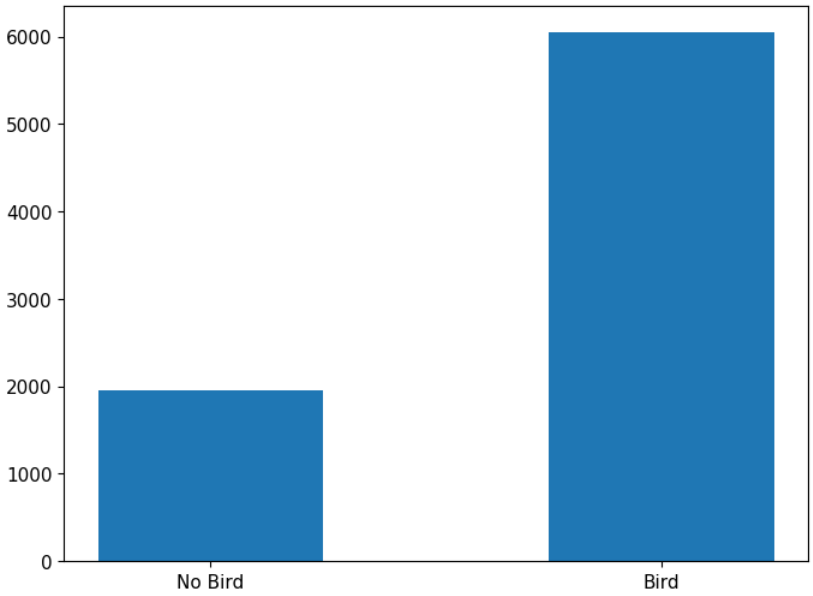

Figure 3.5: Dataset distribution for warblrb10k

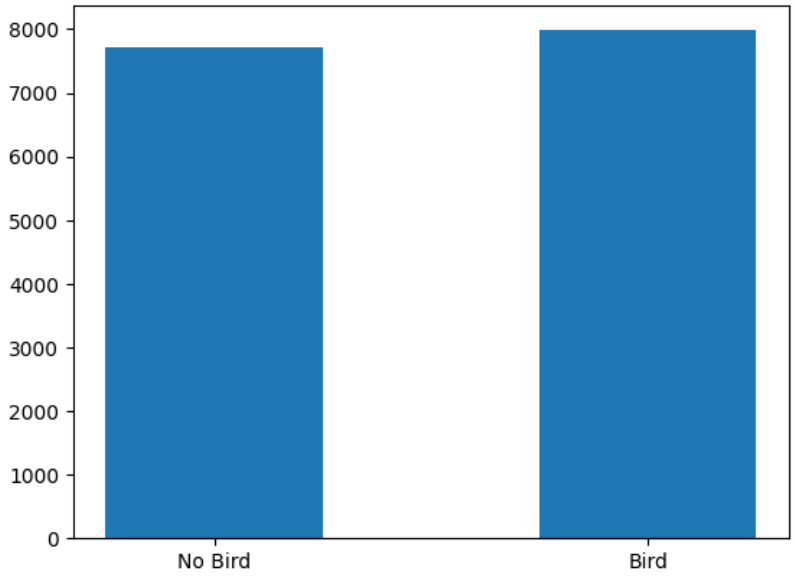

Figure 3.6: Distribution of the merged dataset

3.3.2 Data Transformation

To enhance the robustness and generalization of our bird sound detection model, we applied a series of transformations on both the raw audio signals and their corresponding Mel Spectrograms. These transformations include:

- **Add Gaussian Noise (Raw Audio):**

  This transformation adds Gaussian noise to the raw audio signal. The noise is

generated with a specified mean and standard deviation, and it is added to the audio signal with a certain probability. This helps in making the model robust to noisy environments.

$$\text{noise} = \mathcal{N}(\mu, \sigma^2) \tag{3.1}$$

- **Random Volume Scaling (Raw Audio):**

  This transformation scales the volume of the raw audio signal by a random factor within a specified range. The scaling is applied with a certain probability, making the model resilient to variations in recording levels.

$$\text{audio} = \text{audio} \times \text{gain} \tag{3.2}$$

- **Time Stretch (Raw Audio):**

  This transformation stretches or compresses the time axis of the raw audio signal by a random factor within a specified range. The transformed audio is then converted into a Mel Spectrogram, which helps the model to handle variations in speed.

$$\text{audio} = \text{TimeStretch}(\text{audio}, \text{rate}) \tag{3.3}$$

- **Frequency Masking (Mel Spectrogram):**

  This transformation masks a portion of the frequency bins in the Mel Spectrogram. The maximum number of frequency bins to be masked is specified, and the masking is applied with a certain probability. It encourages the model to focus on different frequency components.

$$\text{mel\_spec}[f : f + \text{max\_mask}] = 0 \tag{3.4}$$

- **Time Masking (Mel Spectrogram):**

  This transformation masks a portion of the time frames in the Mel Spectrogram. The maximum number of time frames to be masked is specified, and the masking is applied with a certain probability. This helps the model to focus on various temporal features.

$$\text{mel\_spec}[:, t : t + \text{max\_mask}] = 0 \tag{3.5}$$

- **Random Content Mixing (Mel Spectrogram):**

This transformation mixes the Mel Spectrogram with another randomly selected noise Mel Spectrogram. The mixing ratio is chosen randomly within a specified range, and the mixing is applied with a certain probability. This aids in making the model robust to background noises.

$$\text{mel\_spec} = (1 - \text{mix\_ratio}) \times \text{mel\_spec} + \text{mix\_ratio} \times \text{noise\_spec} \tag{3.6}$$

- **Time Interval Dropout (Mel Spectrogram):**
  This transformation randomly drops intervals of time frames in the Mel Spectrogram. The number of intervals and the width of each interval are chosen randomly within specified ranges, applied with a certain probability. This helps the model to cope with missing or corrupted audio segments.

$$\text{mel\_spec}[:, t : t + \text{width}] = 0 \tag{3.7}$$

3.3.3 Feature Extraction

The Mel Spectrogram transforms raw audio signals into a time-frequency image that captures the essential spectral characteristics required for bird sound detection. As demonstrated by Lasseck et al.[12], this approach efficiently distinguishes recordings containing bird vocalizations from those without. For a comprehensive explanation of the computation, please refer to the 3.5.

3.3.4 Model Architecture

The bird sound detection model utilizes a pre-trained ResNet-50 architecture as a feature extractor, which has been modified to process mel spectrogram representations of audio signals. This section provides a detailed explanation of the ResNet-50 architecture and the modifications applied to adapt it for bird sound detection.

3.3.4.1 ResNet-50: A Detailed Overview

ResNet-50, or Residual Network-50, is a deep convolutional neural network (CNN) designed to overcome the vanishing gradient problem that commonly arises in deep networks. It achieves this through the use of residual learning, where shortcut connections (skip connections) allow gradients to bypass certain layers, ensuring efficient training even in very deep architectures.

i) **Architecture of ResNet-50** ResNet-50 consists of multiple convolutional layers arranged into residual blocks. These blocks introduce identity mappings that help the model learn transformations efficiently. The network follows a hierarchical structure where initial layers focus on low-level features such as edges and textures, while deeper layers extract high-level representations.

   The architecture is composed of four main stages, each containing bottleneck residual blocks. Each block consists of three convolutional layers: a 1×1 convolution for dimensionality reduction, a 3×3 convolution for feature extraction, and another 1×1 convolution to restore dimensionality. By utilizing these bottleneck layers, ResNet-50 maintains computational efficiency while preserving representational power.

   Another key component of ResNet-50 is batch normalization, which standardizes feature maps at each layer, leading to stable gradient flow and faster convergence. Additionally, ReLU activation functions introduce non-linearity, allowing the network to learn complex patterns in the data.

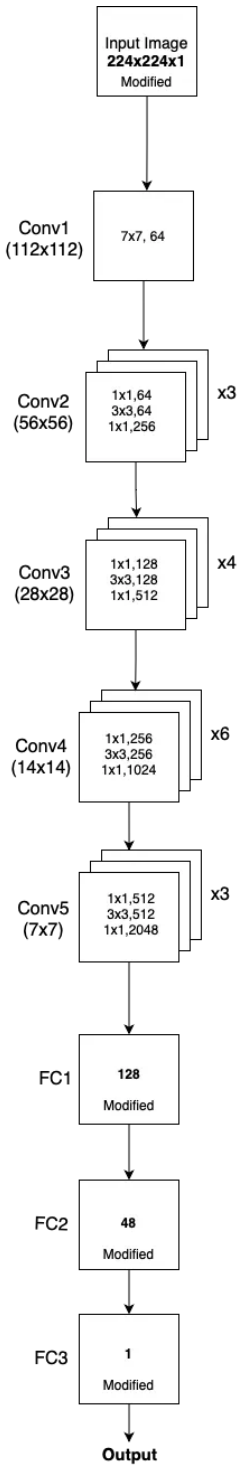

Figure 3.7: Modified ResNet-50 architecture for Bird Sound Detection.

ii) **Components of ResNet-50 Architecture**

The ResNet-50 model consists of several esential components that contribute to its powerful feature extraction and classification capabilities:

– **Convolutional Layers:** These layers apply convolution operations to extract spatial features from input images. Early layers capture basic features

(edges, textures), while deeper layers learn more abstract patterns.

- **Batch Normalization:** This technique normalizes activations at each layer to stabilize training, improve convergence, and prevent internal covariate shifts.
- **ReLU Activation:** A non-linear activation function that introduces sparsity and helps the network learn complex representations.
- **Residual Blocks:** The defining characteristic of ResNet, these blocks contain skip connections that allow gradients to bypass certain layers, mitigating the vanishing gradient problem.
- **Bottleneck Layers:** Each residual block consists of three convolutional layers: a 1×1 convolution to reduce dimensionality, a 3×3 convolution to extract features, and another 1×1 convolution to restore the original depth.
- **Global Average Pooling:** Reduces the dimensionality of feature maps before passing them to the final classification layer, making the network more efficient.
- **Fully Connected Layer:** The final layer of the network that performs classification based on extracted features.
- **Softmax/Sigmoid Activation:** Applies a probability distribution to the output for classification, with softmax used for multi-class classification and sigmoid used for binary classification.

iii) **Adaptation for Bird Sound Detection**

While ResNet-50 is originally designed for image classification, it can be adapted for processing spectrograms, which are visual representations of sound frequencies over time. In this model, several modifications have been applied to tailor ResNet-50 for the bird sound detection task (see Figure 3.7):

- **Input Modification:** Since spectrograms are typically represented as grayscale images with a single channel, the original ResNet-50 architecture, which expects three-channel RGB images, has been adjusted to accept single-channel inputs. This allows the network to effectively process spectrogram data without requiring redundant channels.
- **Feature Extraction:** The pre-trained ResNet-50 model, trained on large-scale image datasets, is used as a feature extractor. The earlier layers remain

unchanged, as they capture fundamental patterns useful for various classification tasks. These convolutional layers are frozen, meaning their weights are not updated during training. This strategy ensures that the model benefits from pre-learned visual features while focusing its learning on the final classification task.

- **Modified Classification Layer:** The original fully connected classification layer of ResNet-50 has been replaced with a custom classification head optimized for binary classification (presence or absence of bird sound). This new layer stack introduces additional non-linearity and dropout mechanisms to prevent overfitting, ultimately improving the model's ability to generalize to new data.

iv) **Training and Generalization**

By freezing the early convolutional layers and training only the newly added classification layers, the model efficiently learns to distinguish bird sounds from other environmental noises. This transfer learning approach significantly reduces the training time and the amount of labeled data required, making the model robust in real-world scenarios.

The final model is designed to process mel spectrogram inputs and produce a probability score indicating the presence of bird sounds. The use of residual learning allows for deep feature extraction, making the model highly effective in distinguishing between subtle variations in audio patterns.

v) **Training Process**

The training process for the bird sound detection model was carefully designed to ensure optimal learning and generalization. The process incorporated several key components and strategies:

- **Training Parameters:**
  - Number of epochs: 20
  - Batch size: 32
  - Initial learning rate: 0.001
  - Weight decay: 0.0001
- **Loss Function:** Binary Cross-Entropy Loss (BCE) was chosen as the loss function due to its effectiveness in binary classification tasks. For input

predictions $\hat{y}$ and true labels y, the BCE loss is calculated as:

$$L_{BCE} = -\frac{1}{N}\sum_{i=1}^{N}[y_i \log(\hat{y}_i) + (1 - y_i)\log(1 - \hat{y}_i)] \tag{3.8}$$

where N is the number of data samples.

- **Optimization:**
  - The Adam optimizer was employed with the following parameters:
    - Learning rate: 0.001
    - Weight decay: 0.0001
  - Learning rate scheduling was implemented using ReduceLROnPlateau with:
    - Reduction factor: 0.1
    - Patience: 3 epochs
- **Monitoring and Validation:**
  - Training and validation losses were monitored each epoch
  - Learning rate adjustments were made based on validation loss plateaus
  - Early stopping was implemented to prevent overfitting
  - Model performance was evaluated on the test set after training completion

The key hyperparameters used for training are summarized in Table 3.2.

Table 3.1: Training Hyperparameters

Hyperparameter	**Value**
Number of Epochs	20
Batch Size	32
Initial Learning Rate	0.001
Weight Decay	0.0001
Loss Function	Binary Cross-Entropy (BCE)
Optimizer	Adam
Learning Rate Scheduling	ReduceLROnPlateau
Reduction Factor	0.1
Patience	3 epochs
Early Stopping	Patience = 3 epochs

3.3.5 Evaluation

To assess the effectiveness of our model in distinguishing between the target classes, we analyzed its performance using the Receiver Operating Characteristic (ROC) curve. Following the benchmark established by Lasseck et al.[12] for the freefield1010 and warblrb10k datasets, the Area Under the Curve (AUC) metric was chosen as the primary evaluation criterion.

The ROC curve provides a comprehensive visualization of the model's discriminative ability by plotting the True Positive Rate (TPR) against the False Positive Rate (FPR) at various classification thresholds. For our binary classification task:

- **True Positive Rate (Sensitivity):**

$$TPR = \frac{TP}{TP + FN} \tag{3.9}$$

  Measures the proportion of actual bird sounds correctly identified.
- **False Positive Rate (1 - Specificity):**

$$FPR = \frac{FP}{FP + TN} \tag{3.10}$$

  Represents the proportion of non-bird sounds incorrectly classified as bird sounds.

The Area Under the ROC Curve (AUC) provides a single scalar value representing the model's overall classification performance. An AUC of 1.0 represents perfect classification, while 0.5 indicates random chance performance.

The ROC curve analysis reveals several important aspects of our model's performance:

- **Threshold Independence:** The ROC curve demonstrates the model's performance across all possible classification thresholds, providing a more robust evaluation than single-threshold metrics.
- **Class Imbalance Handling:** Unlike accuracy, the ROC curve remains effective even when dealing with imbalanced datasets, making it particularly suitable for

real-world bird sound detection scenarios.

- **Operating Point Selection:** The curve enables the selection of optimal decision thresholds based on specific requirements for sensitivity versus specificity in deployment scenarios.

The high AUC score indicates that our model effectively separates bird sounds from background noise and other environmental sounds, making it reliable for real-world applications in bird sound detection.

3.4 Working Mechanism for Bird Species Classification

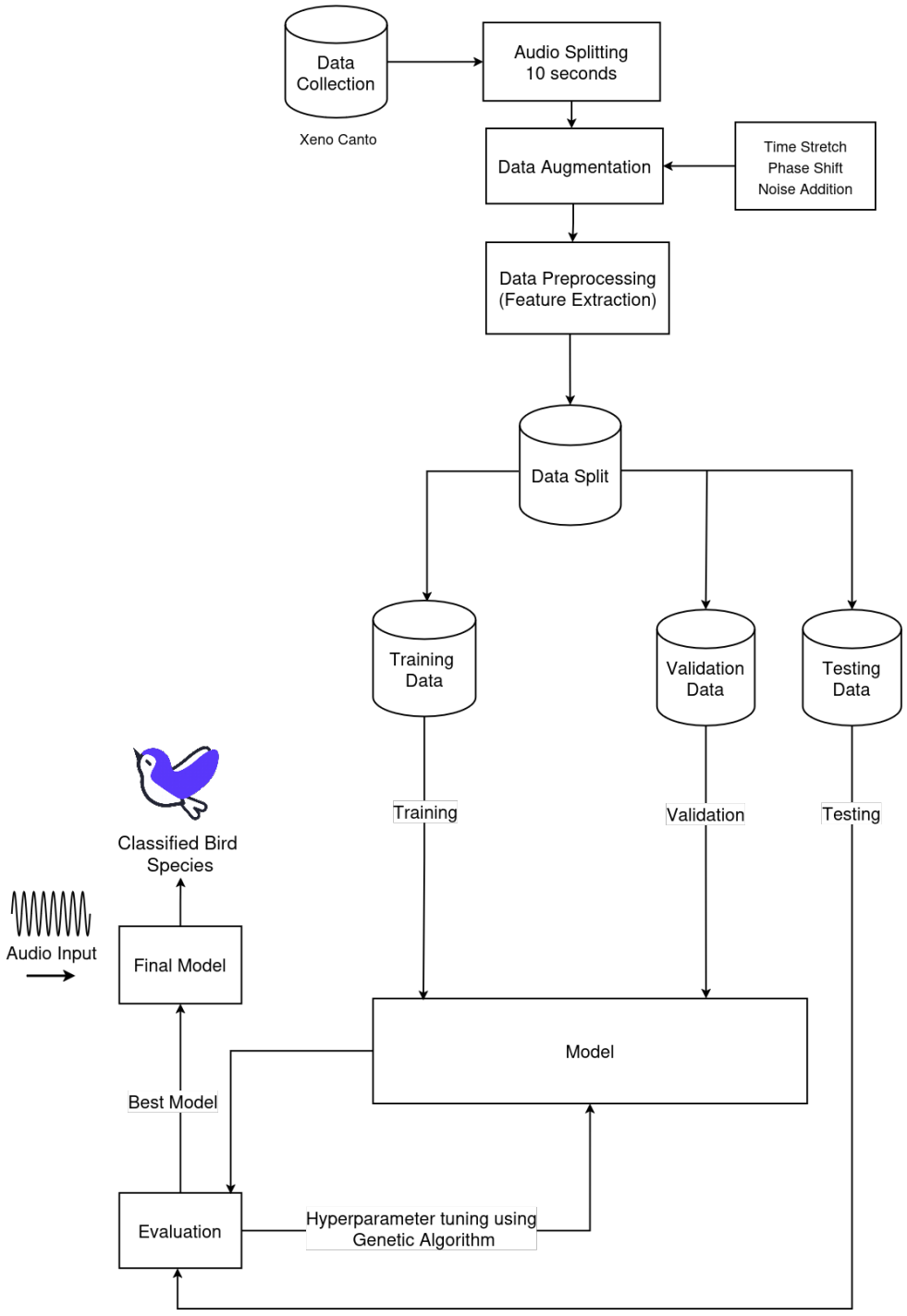

Figure 3.8: Block diagram for the working mechanism for Bird Species Identification

The working mechanism for bird identification from audio utilizes the dataset compiled from Xeno Canto, housing a diverse collection of avian vocalizations. Our methodology revolves around the utilization of EfficientNet, a state-of-the-art convolutional neural network architecture known for its balance between accuracy and efficiency.

The primary objective of this study is to achieve high levels of accuracy in identifying a broad spectrum of 41 distinct bird species. Here lies the detailed explanation of the methodology, from the working mechanism to model training.

3.4.1 Data Collection

For the bird species classification model, we scraped audio data from Xeno-canto (*xeno-canto.org*), a global repository of bird vocalizations contributed by ornithologists and birding enthusiasts. Since the dataset was highly imbalanced, we applied a series of pre-processing and augmentation techniques to ensure a more uniform distribution across bird species.

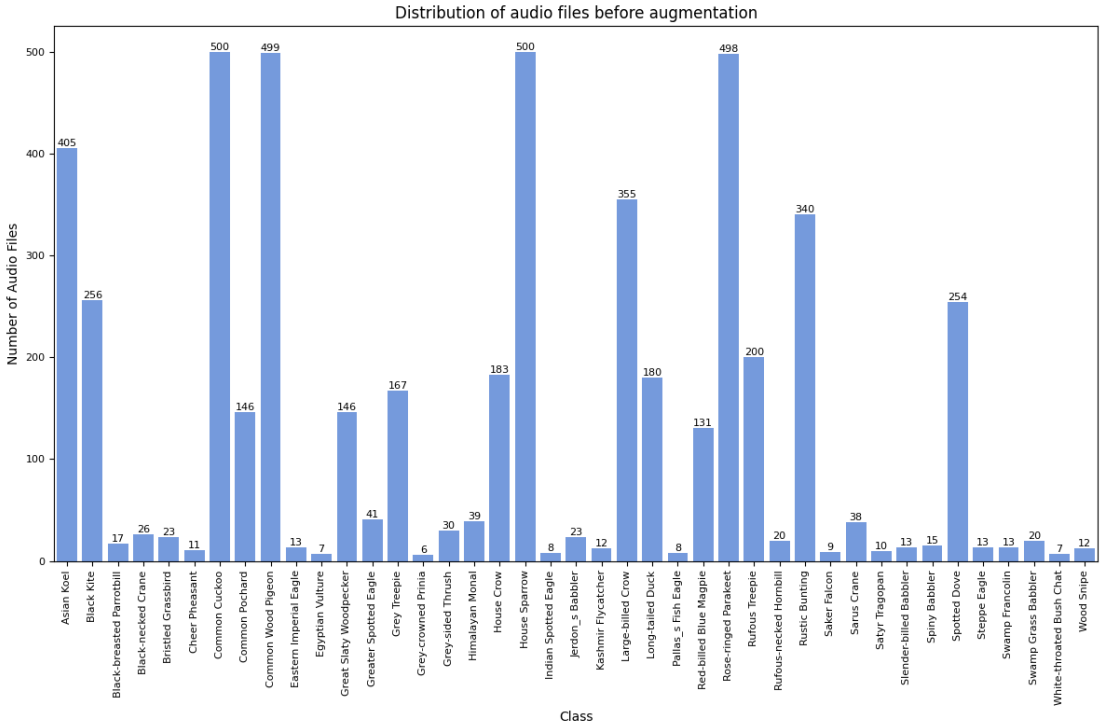

Figure 3.9: Dataset before performing augmentation

3.4.2 Data Augmentation

Initial analysis of the collected data revealed significant class imbalance, with some species having over 400 recordings while others had fewer than 50. Additionally, the audio recordings varied in length. To address these issues:

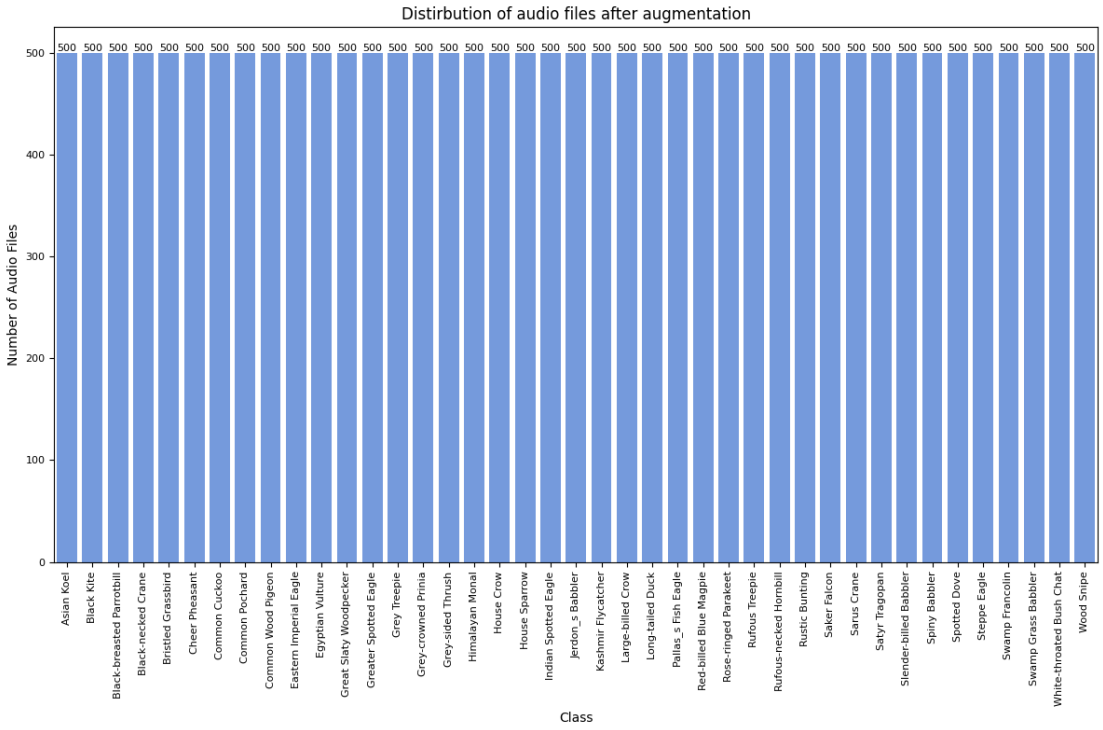

Figure 3.10: Dataset after performing augmentation

- **Audio Clipping:**
  - Recordings were clipped into segments of 10 seconds each.
  - Clips shorter than 5 seconds were discarded to avoid blank audio segments and insufficient data.
  - Clips between 5 and 10 seconds were padded with silence at the end to standardize their length to 10 seconds.
- **Augmentation Techniques:** To balance the dataset, augmentation techniques such as time stretching, phase shifting, and noise addition were applied. The parameters were varied to ensure diversity in the augmented data.
  - Each bird class was augmented to contain exactly 500 audio clips.

3.4.3 Data Preprocessing

The audio recordings were transformed into Mel Spectrograms for further analysis.

- **Conversion Details:**
  - Audio files were converted using a sample rate of 32,000 Hz.
  - The spectrograms were generated with a Hanning window and 48 Mel bands.

- **Dataset Expansion:**
  - Each audio file was converted into corresponding Mel Spectrogram images for CNN based models and MFCC vectors for LSTM based model.
  - The resulting dataset was divided into training, validation, and testing sets for model development.

3.4.4 Feature Extraction

The Mel Spectrogram was employed to transform raw audio recordings into a two-dimensional time-frequency representation, effectively converting sound into an image-like format for processing by convolutional neural networks. This method captures both the temporal and spectral nuances of bird calls, allowing the classifier to discern distinguishing features among various species. As demonstrated in the work of Sevilla et al.[6], such time-frequency representations enhance classification accuracy by emphasizing key frequency components inherent in bird vocalizations.

For a more detailed discussion on the Mel Spectrograms, please refer to the 3.5.

3.4.5 Model Architecture

The bird call classification model is based on EfficientNetB3, a deep convolutional neural network (CNN) designed for optimal efficiency and accuracy. EfficientNetB3 achieves this through **compound scaling**, which uniformly scales network depth, width, and resolution to maximize performance while minimizing computational cost.

3.4.5.1 EfficientNetB3

EfficientNetB3 is part of the EfficientNet family, which employs a balance between deeper layers, wider networks, and higher input resolutions. This architecture integrates several enhancements, including **Mobile Inverted Bottleneck Convolution (MBConv) blocks**, **squeeze-and-excitation (SE) modules**, and the **Swish activation function**. These components contribute to better feature extraction and gradient flow, ensuring robust classification performance.

i) **Architecture of EfficientNetB3**

EfficientNetB3 follows a hierarchical feature extraction process, where early layers detect fundamental patterns (edges, textures), while deeper layers extract more complex features. The architecture consists of multiple MBConv blocks that improve computational efficiency by using **depthwise separable convolutions** and **linear bottlenecks**. Additionally, SE modules dynamically recalibrate channel-wise feature responses, enhancing the model's ability to focus on important frequency patterns in the mel spectrogram. The key architectural components of EfficientNetB3 are summarized below:

- **Convolutional Layers:** Extract spatial features from input spectrograms. Early layers capture basic patterns such as edges and textures. Deeper layers learn abstract audio representations.
- **MBConv Blocks:** .Mobile Inverted Bottleneck Convolution blocks enhance efficiency. Utilize depthwise separable convolutions to reduce computational cost. Incorporate linear bottlenecks to maintain feature representation.
- **MBConv Blocks:** Mobile Inverted Bottleneck Convolution blocks enhance efficiency by using depthwise separable convolutions and linear bottlenecks.
- **Squeeze-and-Excitation (SE) Modules:** Improve feature recalibration by dynamically adjusting channel-wise feature responses, allowing the network to focus on relevant frequency patterns.
- **Swish Activation:** A smooth, non-monotonic activation function that enhances gradient flow and optimization stability.
- **Batch Normalization:** Standardizes activations at each layer to stabilize training and improve convergence.
- **Global Average Pooling:** Reduces dimensionality of feature maps before passing them to the classification layer.
- **Dropout Layer:** A dropout of 0.3 is applied to prevent overfitting by randomly deactivating neurons.
- **Fully Connected Layer:** Converts extracted features into classification outputs.
- **Softmax Activation:** Probability distributions for multi-class classification.

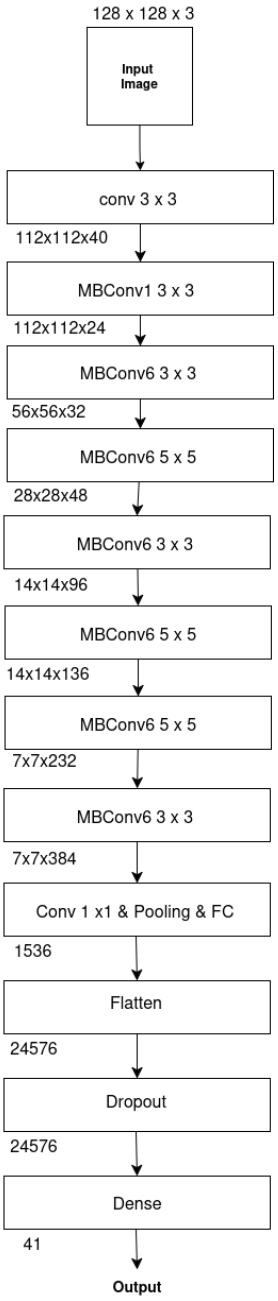

Figure 3.11: EfficientNetB3 architecture for Bird Sound Detection.

ii) **Adaptation for Bird Call Classification**

While EfficientNetB3 is originally designed for image classification, it has been adapted for bird call classification by processing mel spectrograms as input. The key modifications include:

- **Input Modification:** The model accepts spectrogram inputs with dimensions $128 \times 128 \times 3$, where three channels correspond to different spectrogram augmentations or frequency bands.

- **Feature Extraction:** The EfficientNetB3 layers pre-trained on ImageNet are used as a feature extractor. The convolutional layers are **frozen** to retain general visual features while training the classification head on bird call data.
- **Custom Classification Head:** The original fully connected layers are replaced with a new classification head consisting of a **Flatten** layer, **Dropout** layer (0.3), and a **Dense** layer. A softmax activation function is applied to generate class probabilities.

iii) **Training and Generalization**

The model is trained using a transfer learning approach, where only the classification layers are updated while the pre-trained EfficientNetB3 layers remain frozen. This reduces training time and improves generalization.

iv) **Training Process**

- **Initial Training Parameters:**
  - Number of epochs: 30
  - Batch size: 4
  - Image size: $128 \times 128 \times 3$
  - Initial learning rate: 0.0001
- **Loss Function:** Categorical Cross-Entropy(CCE) was chosen as the loss function since it is well-suited for multi-class classification tasks. The Categorical Cross-Entropy is calculated as:

$$L_{CCE} = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{C} y_{i,j}\log(\hat{y}_{i,j}) \tag{3.11}$$

  where N is Number of data samples and C is number of Classes.
- **Optimization:**
  - The Adam optimizer was employed with the following parameters:
    - Learning rate: 0.0001
  - Learning rate scheduling was implemented using an exponential decay function after the first five epochs.
- **Regularization:**
  - Dropout (0.3) was applied to prevent overfitting.

    - L2 regularization ($\lambda = 1e^{-4}$) was used for additional weight regularization.
  - **Monitoring and Validation:**
    - Training and validation losses were monitored each epoch.
    - Learning rate adjustments were made using an exponential decay function.
    - Early stopping was implemented with a patience of three epochs to prevent overfitting.
    - Model checkpointing was applied to save the best-performing model based on validation loss.

The key hyperparameters used for training are summarized in Table 3.2.

Table 3.2: Initial Training Hyperparameters

Hyperparameter	**Value**
Number of Epochs	30
Batch Size	4
Input Image Size	$128 \times 128 \times 3$
Initial Learning Rate	0.0001
Loss Function	Categorical Cross-Entropy
Optimizer	Adam
Learning Rate Scheduling	Exponential Decay (after 5 epochs)
Dropout Rate	0.3
L2 Regularization	$1e^{-4}$
Early Stopping	Patience = 3 epochs

The final model efficiently classifies bird calls by leveraging deep feature extraction from EfficientNetB3 while incorporating a custom classification head optimized for spectrogram data. Through transfer learning and regularization techniques, the model generalizes well to new audio samples, achieving high classification accuracy.

3.4.5.2 CNN+LSTM Model

The CNN+LSTM architecture is designed for bird call classification using MFCC (Mel-Frequency Cepstral Coefficients) instead of mel spectrogram images. Unlike EfficientNetB3, which processes image representations of audio, this model extracts both spatial and temporal features by combining convolutional layers with recurrent layers.

i) **Architecture of CNN+LSTM**

This model follows a two-stage feature extraction process:

- **Convolutional Feature Extraction:** Convolutional layers extract local frequency features from MFCC matrices, capturing patterns in short time frames.
- **Temporal Modeling with LSTM:** A Long Short-Term Memory (LSTM) layer processes sequential dependencies in the extracted features, enhancing temporal understanding.

The key architectural components of the CNN+LSTM model are as follows:

- **Convolutional Layers:** Three convolutional layers extract spatial features from MFCC input. Each convolutional block includes:
  - **Conv2D:** Filters of sizes (3×3) with ReLU activation.
  - **Batch Normalization:** Normalizes activations for stable training.
  - **MaxPooling2D:** Reduces feature map dimensions by pooling along the time axis.
- **Time-Distributed Flattening:** Flattens convolutional feature maps while preserving temporal dimensions for LSTM processing.
- **LSTM Layer:** A 256-unit LSTM layer captures long-range temporal dependencies in bird call sequences.
- **Fully Connected Layers:** Two dense layers process the LSTM output:
  - A 64-unit dense layer with ReLU activation and dropout (0.3) for regularization.
  - An output dense layer with softmax activation for classification.

ii) **Adaptation for Bird Call Classification**

The CNN+LSTM model is tailored to classify bird calls based on MFCC matrices instead of image-based spectrograms. The key modifications include:

- **Input Modification:** The model accepts MFCC matrices with dimensions $(13 \times 626 \times 1)$, where 13 represents MFCC coefficients, and 626 represents time frames.
- **Sequential Feature Extraction:** CNN layers extract spatial frequency information, while the LSTM layer captures sequential dependencies in bird calls.
- **Custom Classification Head:** Includes fully connected layers with dropout

to reduce overfitting.

iii) **Training and Generalization**

The model is trained using a combination of early stopping, learning rate scheduling, and model checkpointing to ensure optimal performance.

iv) **Training Process**

- **Initial Training Parameters:**
  - Number of epochs: 30
  - Batch size: 16
  - Learning rate: 0.0001
  - Loss function: Categorical Cross-Entropy
- **Optimization:**
  - Adam optimizer with an initial learning rate of 0.0001.
  - Learning rate scheduling using an exponential decay function after the first five epochs.
- **Regularization:**
  - Dropout (0.3) applied to prevent overfitting.
- **Monitoring and Validation:**
  - Training and validation losses monitored per epoch.
  - Early stopping with a patience of 3 epochs.
  - Model checkpointing to save the best-performing model based on validation loss.

The key hyperparameters used for training are summarized in Table 3.3.

Table 3.3: Initial Training Hyperparameters for CNN+LSTM

Hyperparameter	**Value**
Number of Epochs	30
Batch Size	16
Input Shape	(13, 626, 1)
Learning Rate	0.0001
Loss Function	Categorical Cross-Entropy
Optimizer	Adam
Learning Rate Scheduling	Exponential Decay (after 5 epochs)
Dropout Rate	0.3
Early Stopping	Patience = 3 epochs

The final CNN+LSTM model effectively classifies bird calls by integrating deep convolutional feature extraction with temporal modeling, ensuring high accuracy while leveraging MFCC representations.

3.4.6 Evaluation Metrics

To assess the performance of the proposed models, the following evaluation metrics were used. These metrics provide a comprehensive understanding of the models classification ability by analyzing both correct predictions and class-wise performance.

- **Accuracy:** Accuracy measures the overall correctness of the model by calculating the ratio of correctly classified samples to the total number of samples.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{3.12}$$

- **Recall:** Recall (also known as sensitivity or true positive rate) measures the model's ability to correctly identify positive instances.

$$Recall = \frac{TP}{TP + FN} \tag{3.13}$$

- **Precision:** Precision quantifies the proportion of correctly predicted positive in-

stances out of all predicted positive instances.

$$Precision = \frac{TP}{TP + FP} \tag{3.14}$$

- **F1-score:** The F1-score is the harmonic mean of precision and recall, providing a balanced measure when there is an uneven class distribution.

$$F1\text{-}score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{3.15}$$

3.5 Feature Extraction

Feature extraction was a critical step where the audio clips were transformed into a format that were fed into machine learning models. We used two primary techniques for feature extraction: Mel Spectrograms and Mel-Frequency Cepstral Coefficients (MFCC).

3.5.1 Mel Spectrogram

A spectrogram is a visual representation of the spectrum of frequencies in a sound signal as they vary with time. It is generated by applying the Short-Time Fourier Transform (STFT) to the audio signal, followed by mapping the frequencies to the Mel scale, which better represents human auditory perception. This transformation provides insight into how the frequency content of the signal changes over time.

1. **Short-Time Fourier Transform (STFT):** The STFT is computed by dividing the signal into overlapping frames, applying a window function, and then performing the Fourier Transform on each frame.

$$X(n) = \sum_{m=0}^{N-1} x(m) \cdot w(n - m) \tag{3.16}$$

   where $x(m)$ is the audio signal and $w(n)$ is the window function.
2. **Mel Scale Transformation:** The frequency axis of the spectrogram is then mapped

to the Mel scale using triangular filter banks.

$$M(f) = 2595 \log_{10} \left(1 + \frac{f}{700}\right) \tag{3.17}$$

where $M(f)$ is the Mel frequency, and f is the frequency in Hz.

3. **Log Scaling and Normalization:** Convert Mel spectrogram to a logarithmic scale:

$$S_{\text{log}}(m, n) = \log \left(S_{\text{mel}}(m, n)\right) \tag{3.18}$$

where $S_{\text{log}}(m, n)$ is the log-scaled Mel spectrogram, and $S_{\text{mel}}(m, n)$ is the Mel spectrogram before log transformation.

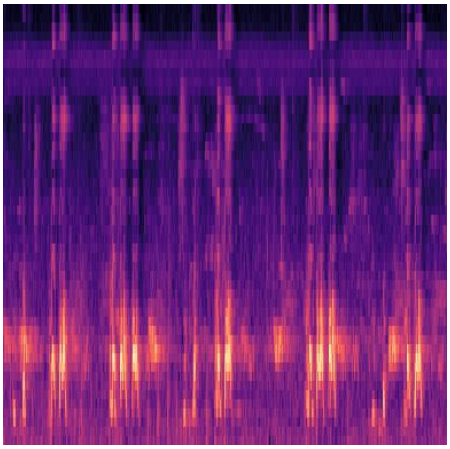

Figure 3.12: Mel Spectrogram image for a sample audio

3.5.2 Mel-Frequency Cepstral Coefficients (MFCC)

MFCCs are coefficients that collectively describe the short-term power spectrum of a sound signal. The process of obtaining MFCCs involves several steps:

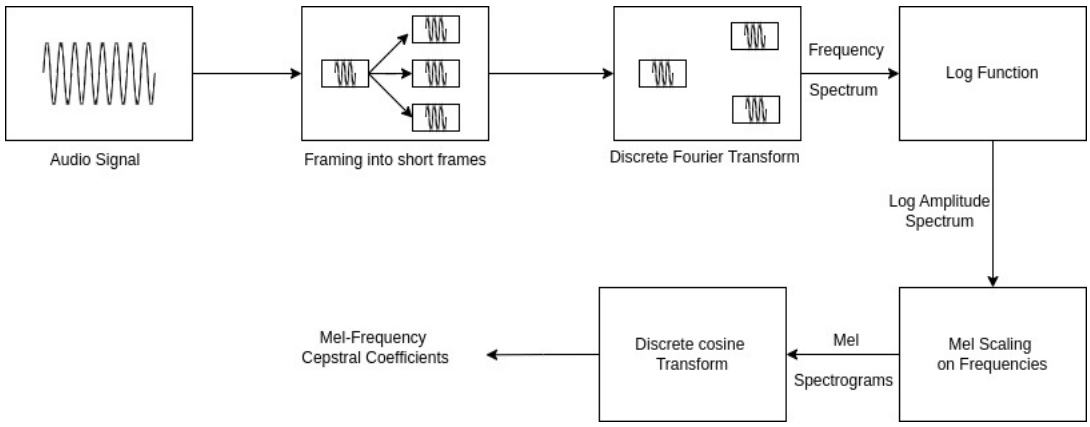

Figure 3.13: Feature Extraction Using Spectrogram and MFCC

1. **Framing:** Divide the audio signal into short frames.
2. **Discrete Fourier Transform (DFT):** Convert each frame to the frequency domain.

$$X(k) = \sum_{n=0}^{N-1} x(n) \cdot e^{-j\frac{2\pi}{N}kn} \tag{3.19}$$

3. **Log Function:** Apply a logarithm to the amplitude spectrum.

$$S_{\log}(k) = \log(|X(k)|) \tag{3.20}$$

4. **Mel-Scaling:** Map the frequencies to the Mel scale, which better represents how humans perceive sound.

$$f_{\text{mel}} = 2595 \cdot \log_{10}(1 + \frac{f}{700}) \tag{3.21}$$

5. **Discrete Cosine Transform (DCT):** Convert the Mel spectrum to the cepstral domain, yielding the MFCC features.

$$C(n) = \sum_{k=0}^{K-1} S_{\text{mel}}(k) \cdot \cos\left(\frac{\pi n(k+0.5)}{K}\right) \tag{3.22}$$

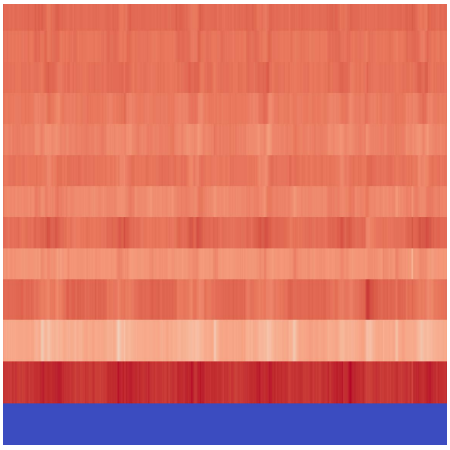

Figure 3.14: MFCC image for a sample audio

3.6 Hyperparameter Optimization using Genetic Algorithm

Genetic algorithms are a powerful method for optimizing hyperparameters in machine learning models. Genetic Algorithm have proven to significantly improve the performance metrics of the CNN model instead of using hand tuned approach for hyperparameters. This section outlines the steps involved in using GAs for hyperparameter optimization[13].

- **Encoding the Hyperparameters**
  - Hyperparameters are represented as a chromosome, where each hyperparameter is a gene in the chromosome.
  - For example, in a neural network, a chromosome might include genes for the learning rate, number of layers, number of neurons per layer, and activation functions.
- **Initial Population**
  - An initial population of chromosomes is generated randomly, with each chromosome representing a different set of hyperparameters.
- **Fitness Function**
  - A fitness function is defined to evaluate the performance of each set of hyperparameters.
  - This typically involves training the model with the given hyperparameters and measuring its performance on a validation set.
- **Selection**

  - Selection involves choosing the best-performing chromosomes to serve as parents for the next generation.
  - Various selection methods can be employed, such as tournament selection, roulette wheel selection, or rank-based selection.
- **Crossover (Recombination)**
  - Crossover combines pairs of parent chromosomes to produce offspring for the next generation.
  - This is done by swapping segments of parent chromosomes to create new chromosomes, thereby combining features of both parents.
- **Mutation**
  - Mutation introduces random changes to some of the genes in the offspring chromosomes.
  - This helps maintain genetic diversity in the population and allows the algorithm to explore a broader search space.
- **Replacement**
  - The current population is partially or entirely replaced with the new generation of chromosomes, ensuring that better solutions are carried forward while allowing for exploration of new possibilities.
- **Termination**
  - The process of selection, crossover, mutation, and replacement is repeated until a termination criterion is met.
  - This could be a set number of generations, convergence of fitness scores, or achieving a satisfactory performance level.
- **Best Solution**
  - The best chromosome at the end of the process represents the optimal or near-optimal set of hyperparameters for the model.

3.7 Algorithms Used

3.7.1 Mel Spectrogram

Algorithm 1 Mel Spectrogram Extraction

Input: Audio signal $x(t)$.
Output: Mel spectrogram S_{mel}.
Apply STFT to generate spectrogram $S(f, t)$.
Convert amplitudes of $S(f, t)$ to dB scale, obtaining $S_{\text{dB}}(f, t)$.
Convert frequencies to Mel scale.
    Choose number of mel bands N_{mel}.
Construct mel filter banks.
    Convert f_{min} and f_{max} of $S_{\text{dB}}(f, t)$ to Mel scale.
    Divide Mel scale range into N_{mel} intervals.
    Convert center frequencies of Mel bands back to Hertz.
    Round center frequencies to nearest bins.
    Design triangular band pass filters for each Mel band.
    Apply mel filter banks to $S_{\text{dB}}(f, t)$ to obtain S_{mel}.

3.7.2 Mel-Frequency Cepstral Coefficients Extraction

Algorithm 2 MFCC Extraction

Input: Audio signal $x(t)$
Output: MFCC coefficients.
Frame the signal into short frames.
Apply a window function to the frames.
Apply DFT to generate the frequency spectrum of each frame.
Apply logarithm to the spectrum to get log amplitude spectrum.
Perform Mel scaling using filter banks to get Mel spectrogram.
Apply DCT to the Mel spectrogram to get MFCCs.

3.7.3 Genetic Algorithm

Algorithm 3 Genetic Algorithm for Hyperparameter Optimization

1: Initialize the population with random hyperparameters.
2: **for** generation = 1 to N **do**
3: Evaluate the fitness of each individual in the population.
4: Select individuals to be parents based on their fitness scores.
5: Generate offspring through crossover.
6: Apply mutation to the offspring.
7: Replace the old population with the new generation.
8: **end for**
9: Return the best solution found.

3.7.4 Fitness Function

The fitness function evaluates the performance of hyperparameters by training and validating the model:

Algorithm 4 Fitness Function

1: Train the model with given hyperparameters.
2: Evaluate the model's performance on a validation set.
3: **return** Performance metric (e.g., accuracy, F1 score).

3.7.5 Mapping Location of Bird in Map

Mapping the location of a bird in an application using **flutter_map** and **geolocator** involves using the classified bird data to pinpoint its location on a map. This is particularly useful for bird watchers and researchers to track bird sightings.

Algorithm 5 Mapping Location of Bird in Map

1: Initialize the Map widget.
2: Record the bird audio.
3: Get the classified bird species and captured location.
4: Convert the location data to latitude and longitude coordinates.
5: Add a marker to the map at the specified coordinates.
6: Update the map view to center on the new marker.

3.8 Software Development Model

This project will be using an incremental methodology since it offers a functioning prototype at an early stage of development. The requirements and scope of the project can be altered as necessary by studying the prototype. The rationale for the preference for this software development strategy is the flexibility offered by adopting the incremental technique. In this paradigm, the project goes through several releases or iterations prior to its official release.

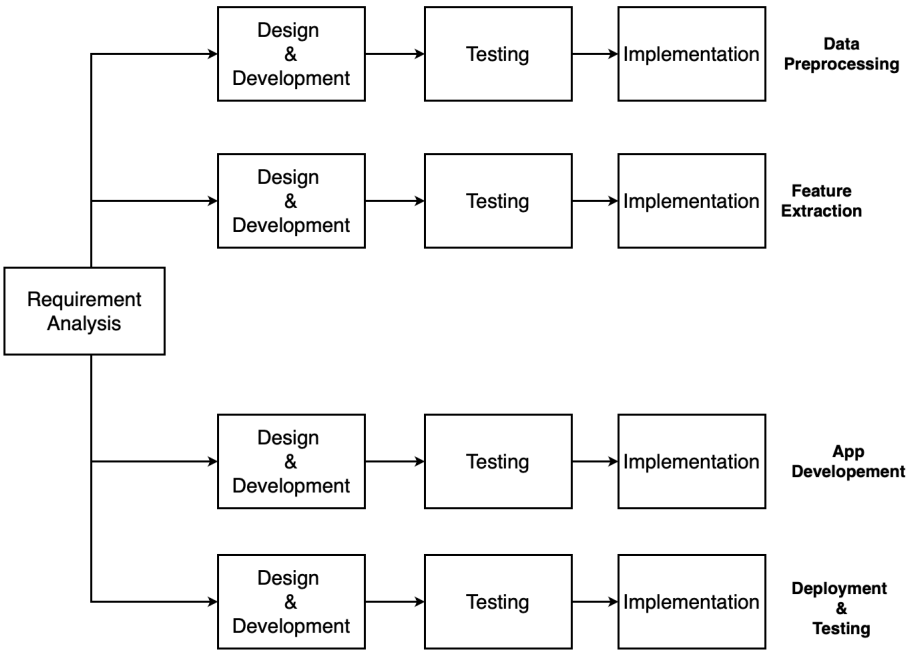

Figure 3.15: Incremental Model for development of FeatherFind

CHAPTER 4
RESULTS, ANALYSIS, AND COMPARISIONS

In the course of our project on Bird Species Classification from Audio, we have accomplished several key tasks to progress our analysis and model development:

4.1 Bird Sound Detection

Figure 4.3 shows the training progress over 15 epochs. The model shows consistent improvement in both accuracy and loss metrics during training and validation phases.

The training process exhibited steady improvement, with the model achieving final metrics of:

- Training accuracy: **84.20%** and validation accuracy: **84.10%**
- Training loss: **0.394** and validation loss: **0.412**
- Training F1-score: **0.585** and validation F1-score: **0.615**

The close alignment between training and validation metrics indicates good generalization without overfitting. The loss curves show consistent convergence, while accuracy and F1-scores demonstrate steady improvement throughout the training process.

The confusion matrix in Figure 4.4 shows the classification performance on the test dataset. From the total of **1621** samples, the model correctly classified **621** samples as non-bird sounds (True Negatives) and 680 samples as bird sounds (True Positives). There were 107 false positives and **160** false negatives. The model performance metrics are summarized in Table 4.1.

Table 4.1: Performance Metrics for Bird Sound Detection Model

Metric	**Value (%)**
Accuracy	82.97
F1-score	83.59
Precision	80.95
Recall	86.40

Figure 4.5 shows the Receiver Operating Characteristic (ROC) curve for the bird sound

Figure 4.1: Accuracy curves

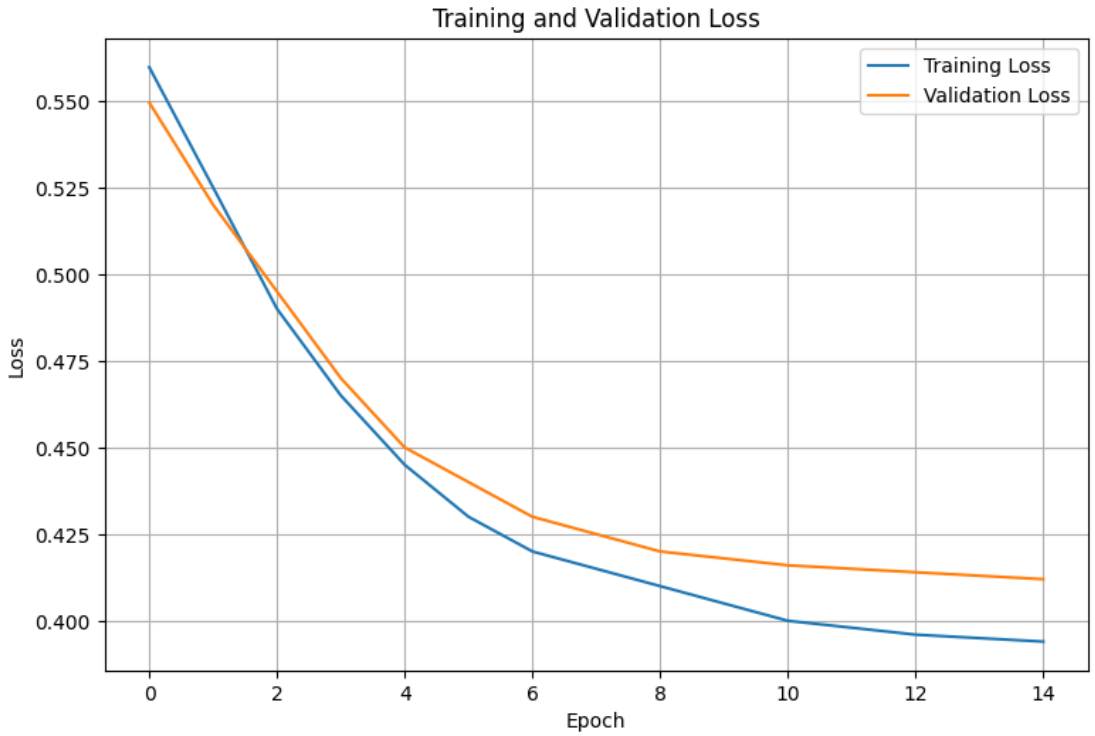

Figure 4.2: Loss curves

Figure 4.3: Training and validation curves for Bird Sound Detection Model

detection model. The curve illustrates the model's ability to discriminate between bird and non-bird sounds across different classification thresholds.

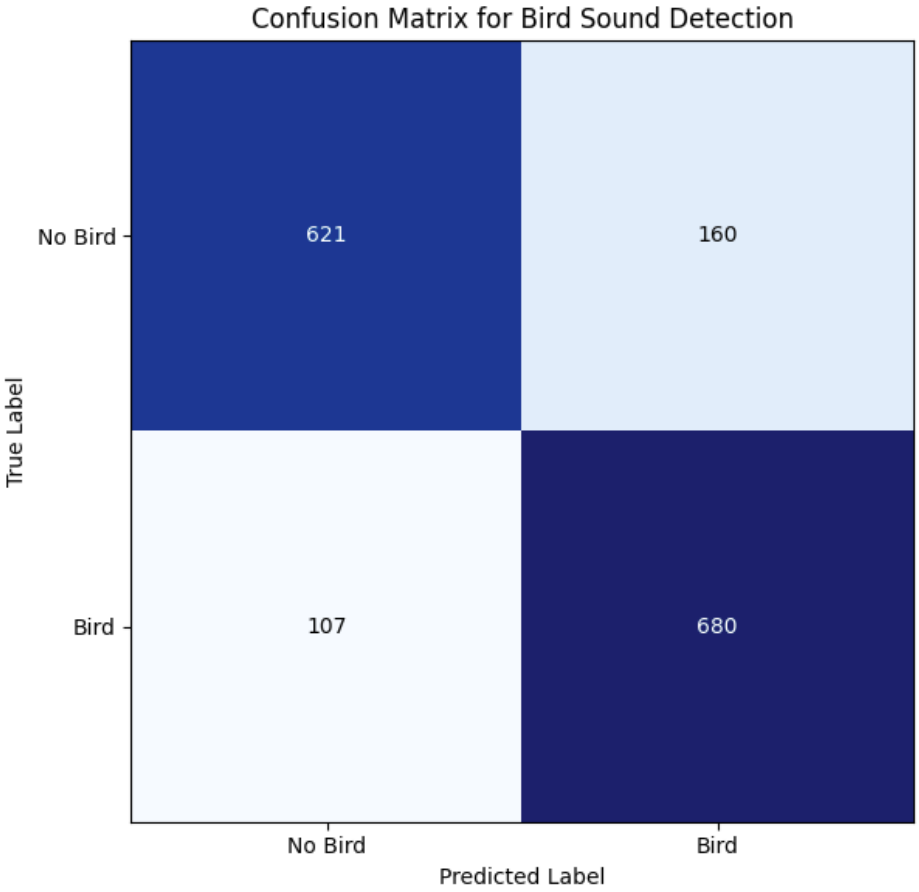

Figure 4.4: Confusion Matrix for Bird Sound Detection Model

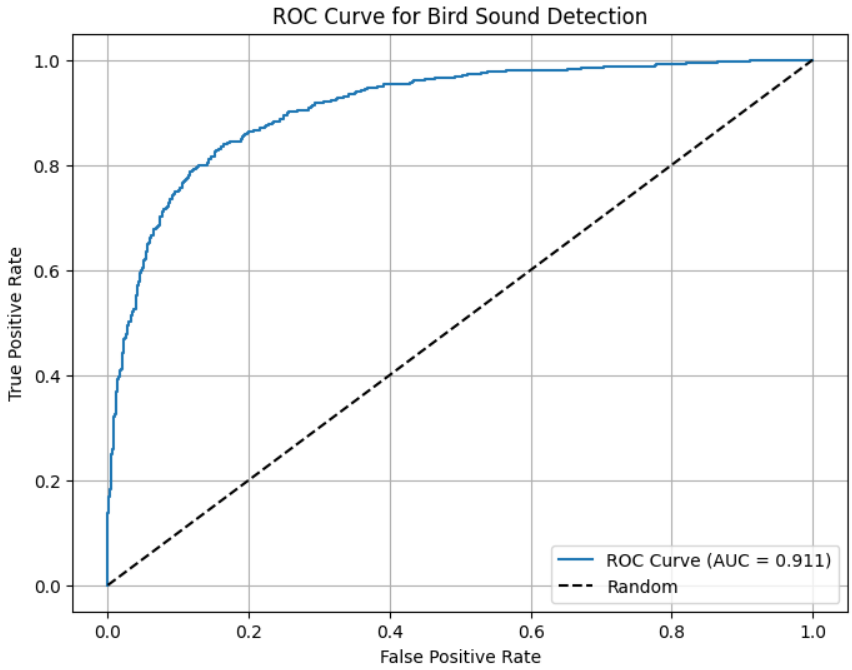

Figure 4.5: ROC Curve of the detection model

The Area Under the Curve (AUC) score of **0.911** indicates excellent discriminative ability, as it is significantly higher than the random chance baseline of **0.5**.

4.2 Bird Species Classification

The below figures show the outcomes of Bird Species Classification based on two different architectures i.e CNN-LSTM and EfficientNetB3, evaluated on our dataset of 41 bird species.

4.2.1 Classification using CNN-LSTM

The CNN-LSTM model achieved an accuracy of **74.73%** with a cross-entropy loss of **1.0171** across all 41 bird species. The confusion matrix shows stronger diagonal elements indicating good classification ability for most species, though there is some confusion between similar-sounding birds. The training and validation curves show the

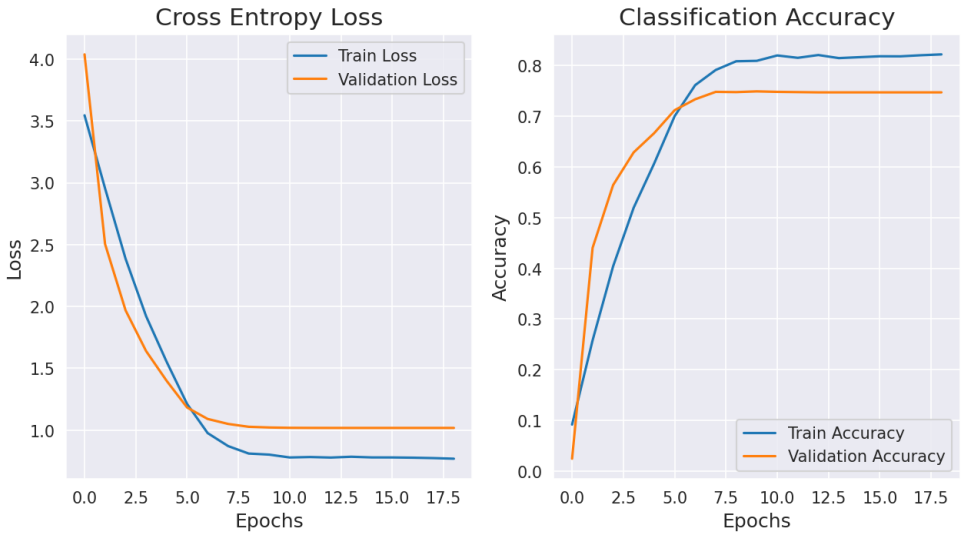

Figure 4.6: CNN-LSTM Model Evaluation

model achieved a training accuracy of **80.76%** and validation accuracy of **74.73%**, with corresponding losses of **0.8025** and **1.0171** respectively.

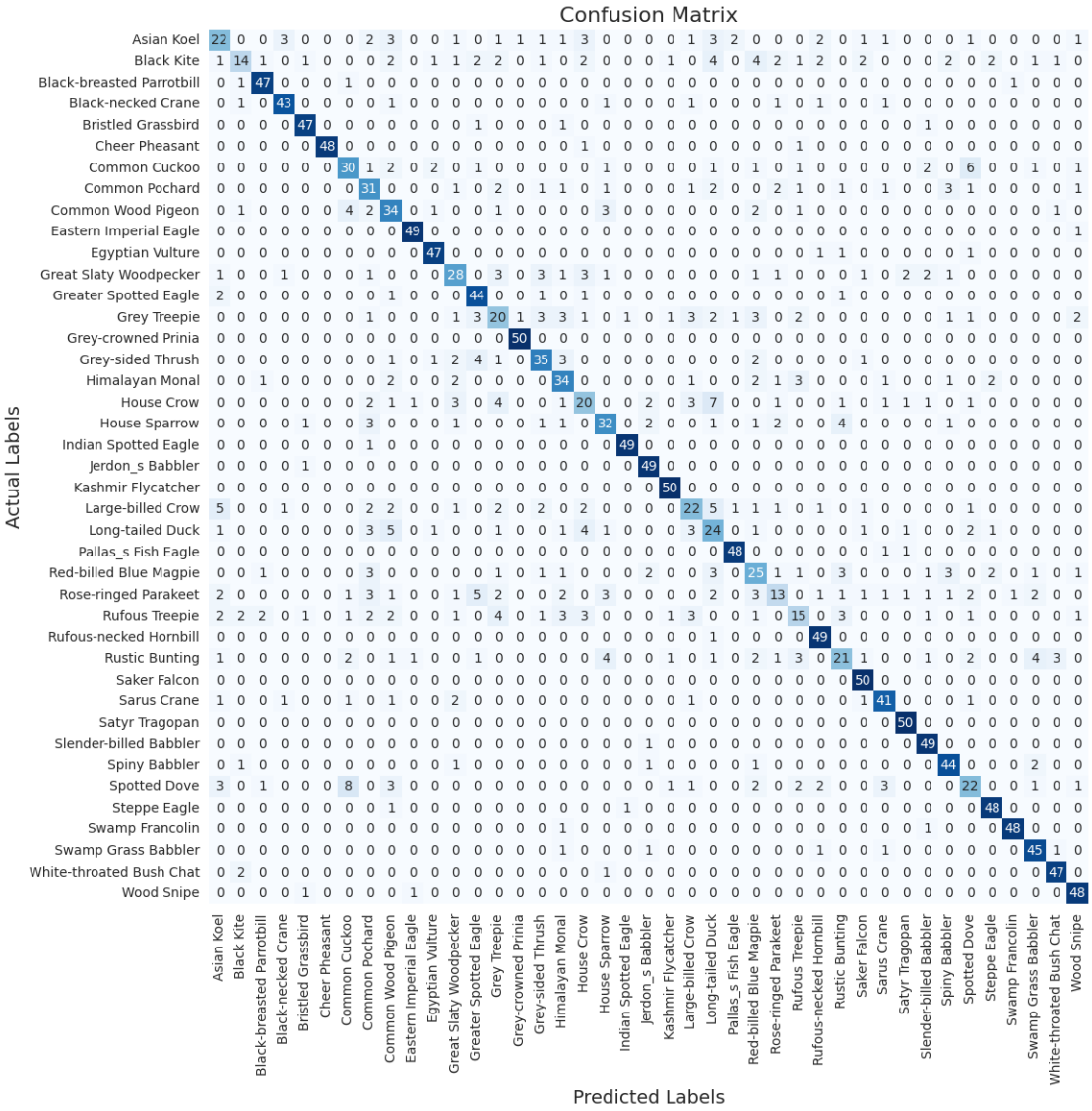

Figure 4.7: CNN-LSTM Confusion Matrix

The confusion matrix visualizes the model's predictions across all 41 classes, with darker diagonal elements indicating correct classifications and lighter off-diagonal elements showing misclassifications between species.

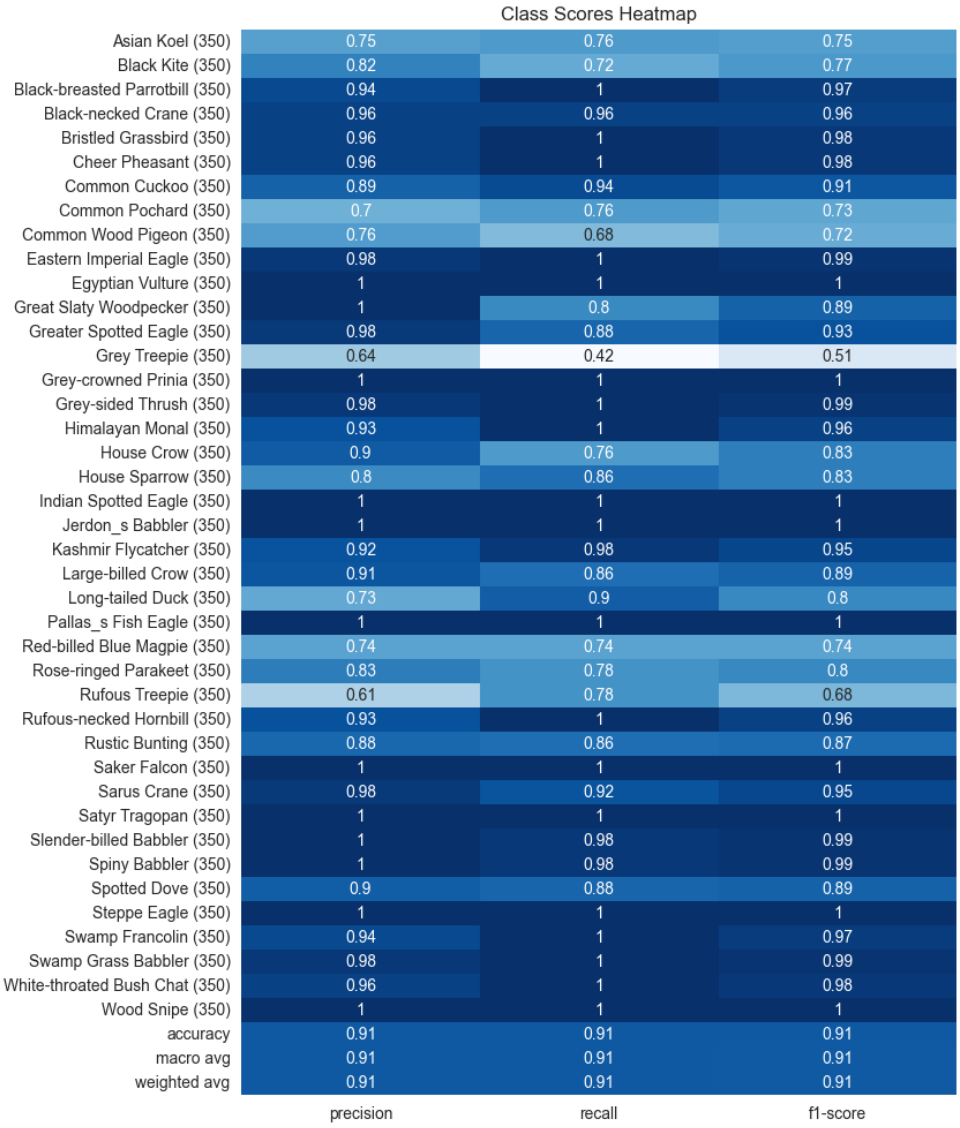

Figure 4.8: CNN-LSTM Classification Report

The classification report shows detailed metrics for each species, with an average precision of **73%**, recall of **75%**, and F1-score of **73%** across all classes.

4.2.2 Classification using EfficientNetB3

The EfficientNetB3 model demonstrated superior performance with an accuracy of **90.73%** with a cross-entropy loss of **0.4154**, significantly outperforming the CNN-LSTM architecture. The confusion matrix reveals excellent discrimination ability across most species, with minimal misclassifications between different bird calls.

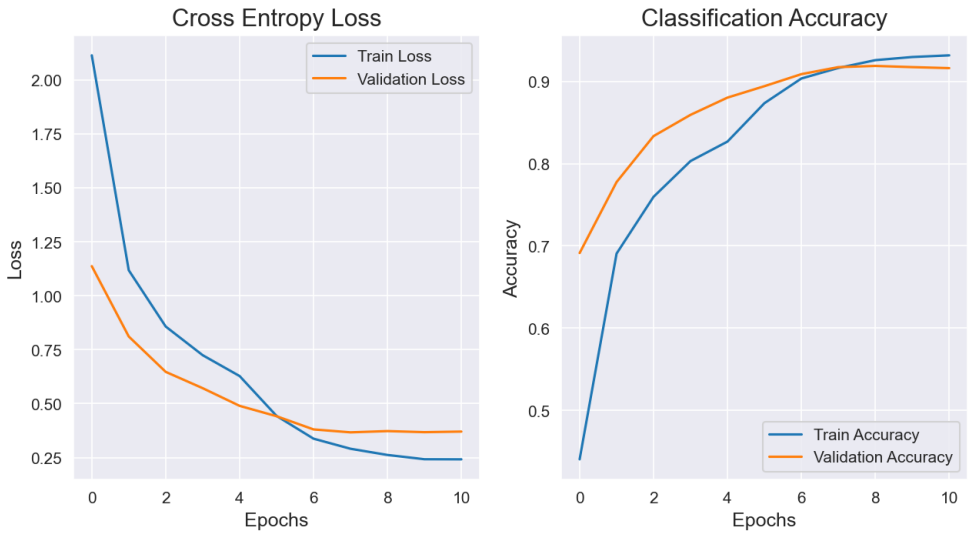

Figure 4.9: EfficientNetB3 Model Evaluation

The training and validation curves demonstrate strong performance with a training accuracy of **93.32%** and validation accuracy of **91.61%**, while maintaining low loss values of **0.2336** and **0.3706** respectively.

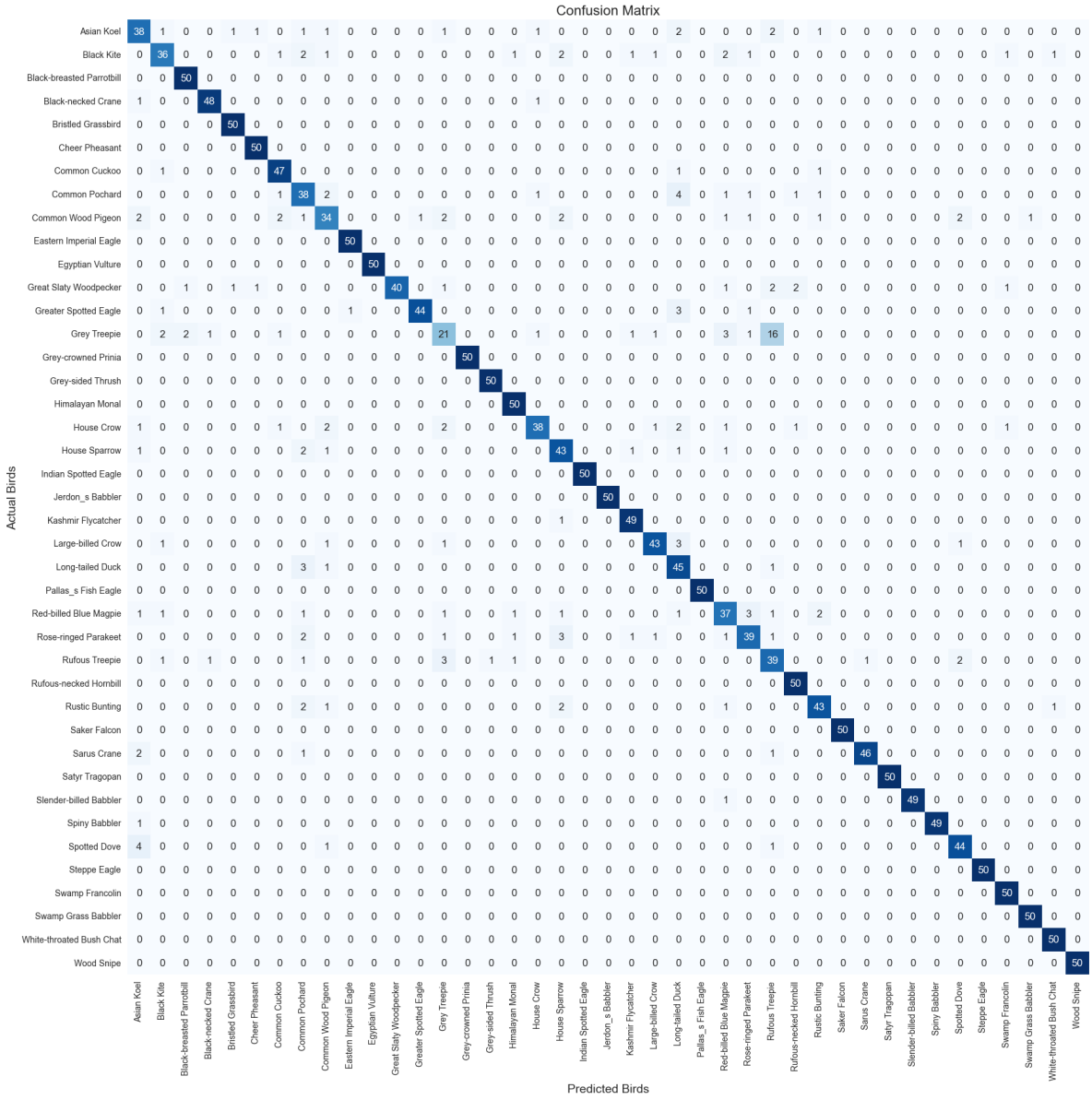

Figure 4.10: EfficientNetB3 Confusion Matrix

The confusion matrix shows strong diagonal elements indicating excellent classification accuracy across most species, with minimal confusion between different bird calls.

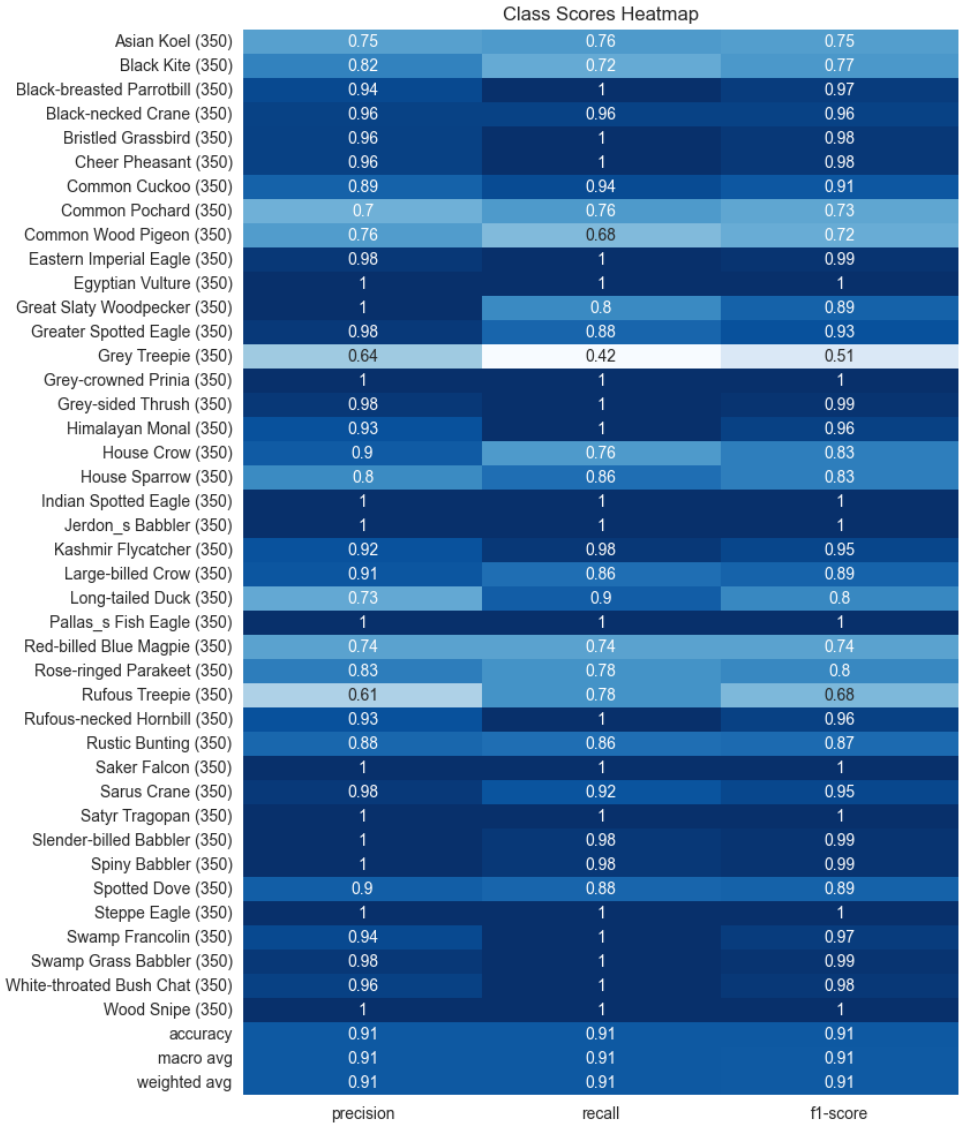

Figure 4.11: EfficientNetB3 Classification Report

The classification report shows detailed metrics for each species, with an average precision, recall and f1-score of **91%** across all classes.

4.3 Hyperparameter Optimization Using Genetic Algorithim

To optimize the performance of our model, we used genetic algorithm for hyperparameter tuning. This approach helped us systematically explore and identify the most effective hyperparameter settings for our model, including network architecture parameters like learning rates, batch sizes, images sizes, dropout and others. The genetic algorithm enabled a more efficient and comprehensive search for optimal configurations.

4.3.1 Hyperparameters for CNN-LSTM

For the CNN-LSTM model, GA was applied to tune hyperparameters that impact both convolutional feature extraction and model efficiency. The key hyperparameters optimized includes Batch Size, Learning Rate, Number of Epochs, Dropout Rate, LSTM Units and Dense Units. The genetic algorithm was executed over multiple generations, evolving hyperparameter configurations towards optimal performance.

4.3.1.1 Hyperparameter Search Space

The genetic algorithm optimized the following hyperparameters:

Hyperparameter	Search Values
Batch Size	{16, 32, 64}
Learning Rate	{1e-4, 1e-4, 1e-3}
Epochs	{20, 30, 40}
Dropout Rate	{0.3, 0.4, 0.5}
LSTM Units	{64, 128, 256}
Dense Units	{64, 128, 256}

Table 4.2: Hyperparameter search space for CNN-LSTM

4.3.1.2 Genetic Algorithm Process

The GA followed an evolutionary approach, iterating over multiple generations to refine hyperparameters. The process consisted of:

- **Population Size:** 3 (each generation consists of 3 hyperparameter sets)
- **Generations:** 5 (iterative evolution of hyperparameters)

- **Mutation Rate:** 10% (randomly modifying one parameter)
- **Crossover:** Parents were selected based on validation accuracy, and offspring were created by combining their hyperparameters.

4.3.1.3 Results for CNN-LSTM

Over five generations, the algorithm refined hyperparameters, improving model accuracy and minimizing overfitting. The following images illustrate training and validation performance trends for different hyperparameter configurations across generations.

The Following Graphs shows the training and validation accuracies and loss for the respective hyperparameters used for CNN-LSTM.

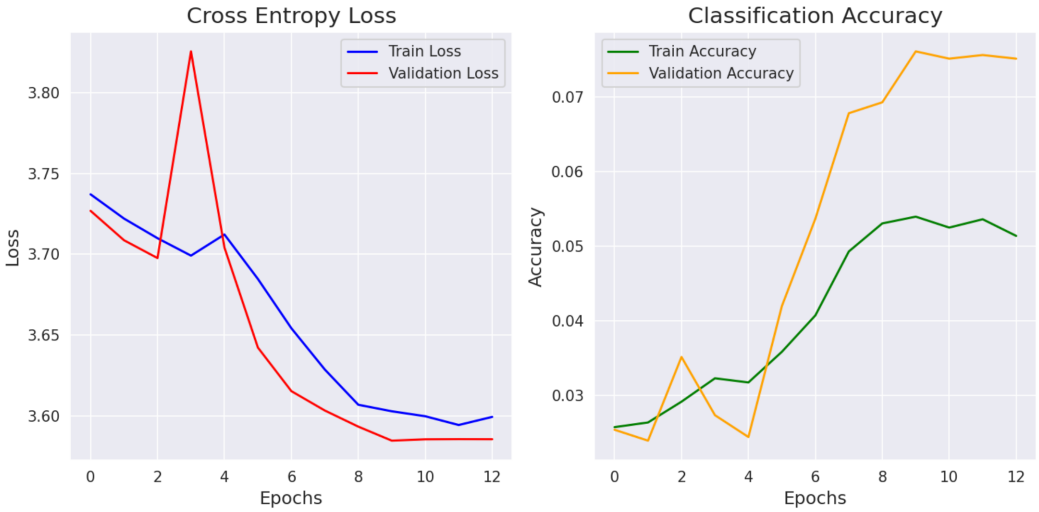

Figure 4.12: Accuracy and Loss for CNN-LSTM in generation 1

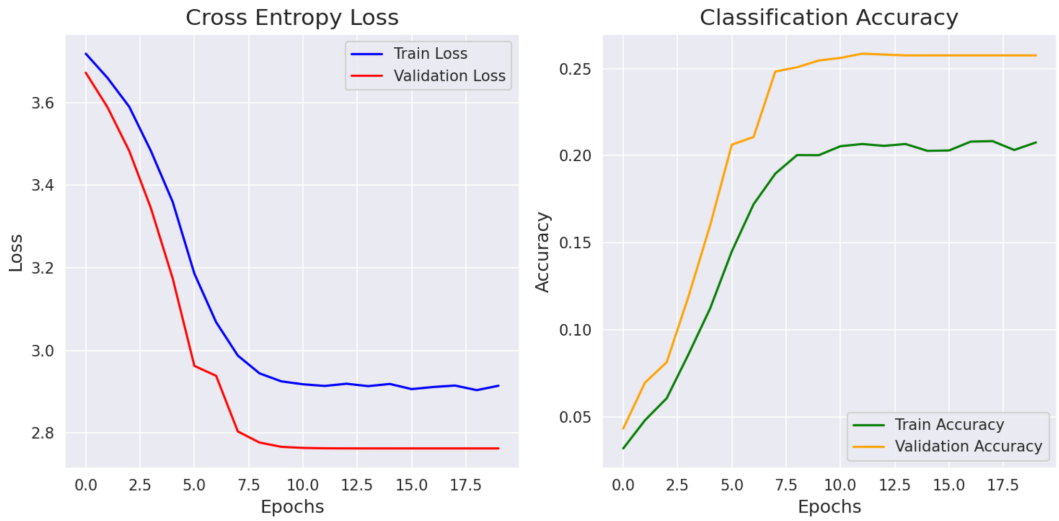

Figure 4.13: Accuracy and Loss for CNN-LSTM in generation 2

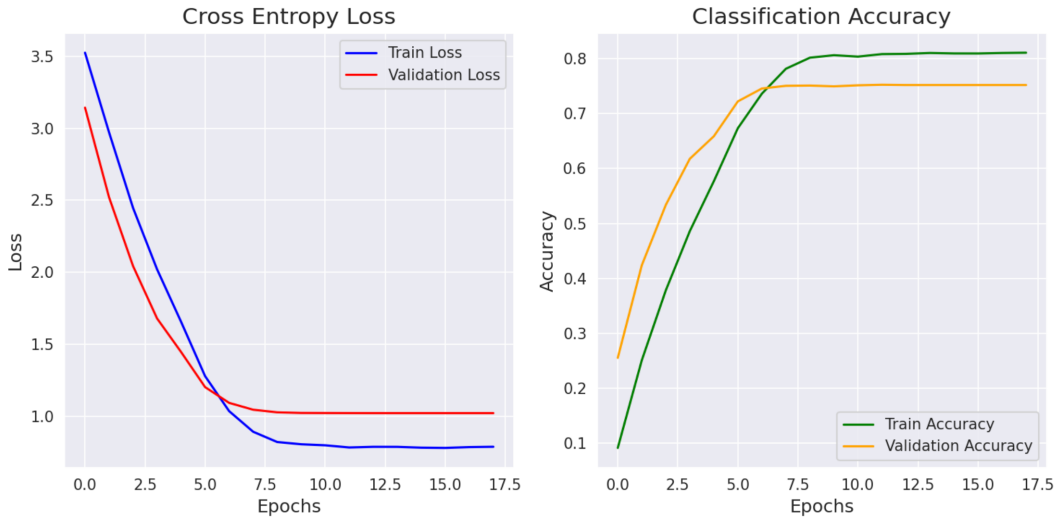

Figure 4.14: Accuracy and Loss for CNN-LSTM in generation 3

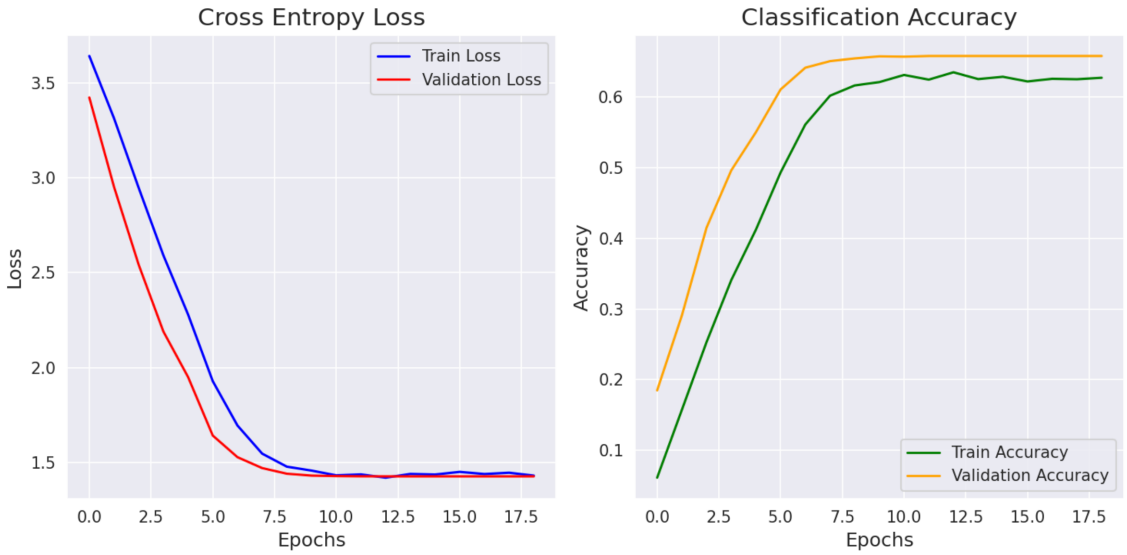

Figure 4.15: Accuracy and Loss for CNN-LSTM in generation 4

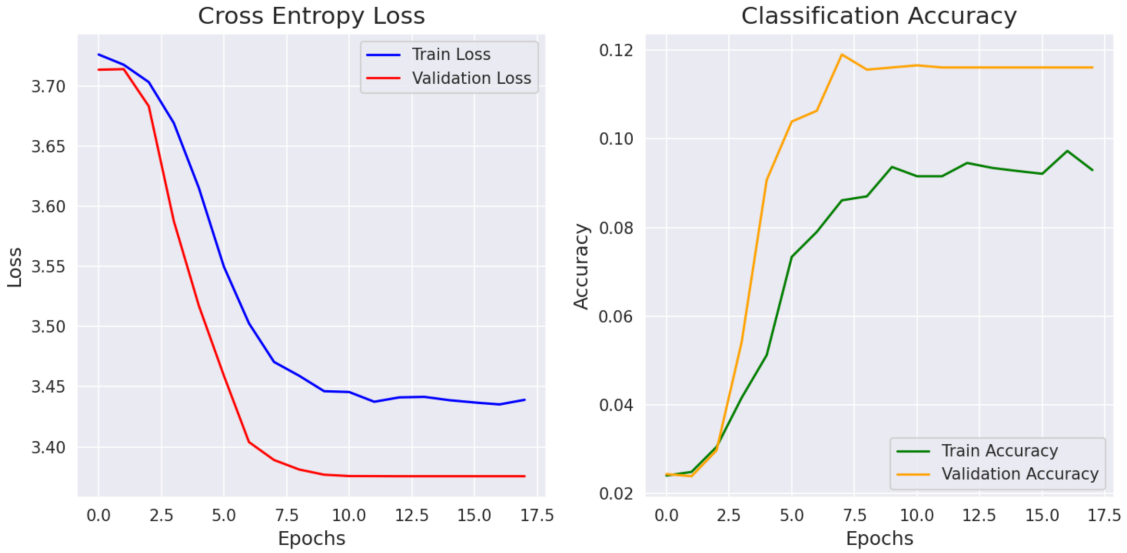

Figure 4.16: Accuracy and Loss for CNN-LSTM in generation 5

The following table demonstrates different hyperparameter configurations and their corresponding test accuracies.

S.N	Batch	LR	Epochs	Dropout	LSTM Units	Dense Units	Test Accuracy (%)
1	64	0.001	40	0.5	128	64	0.761
2	16	0.001	30	0.3	256	64	25.76
3	**16**	**0.0001**	**20**	**0.3**	**256**	**64**	**75.17**
4	16	0.0001	20	0.4	256	64	65.80
5	16	0.001	30	0.3	65	64	11.61

Table 4.3: Hyperparameter configurations and test accuracies for CNN-LSTM

4.3.1.4 Conclusion for CNN-LSTM

The best model achieved a Test Accuracy of **75.17%** with a **Batch Size of 16**, **Learning Rate of 0.0001**, **Epochs = 20**, **Dropout Rate of 0.3**, **LSTM Units = 256**, and **Dense Units = 64**. These results highlight the effectiveness of the Genetic Algorithm in systematically optimizing hyperparameters, leading to improved learning efficiency and minimized overfitting.

4.3.2 Hyperparameters for Efficient-Net

For EfficientNet, GA was used to fine-tune hyperparameters that directly affect feature extraction and classification efficiency. The optimized parameters includes Batch Size, Learning Rate, Number of Epochs, Dropout Rate,Regularization and Dense Units.

4.3.2.1 Hyperparameter Search Space

The genetic algorithm optimized the following hyperparameters:

Hyperparameter	Search Values
Batch Size	{4, 8, 16}
Image Size	{(128, 128), (128, 128), (224, 224)}
Learning Rate	{1e-4, 1e-4, 1e-3}
Epochs	{20, 20, 20}
Dropout Rate	{0.3, 0.2, 0.4}
L2 Regularization	{1e-4, 1e-4, 1e-3}

Table 4.4: Hyperparameter search space for the EfficientNetB3

4.3.2.2 Genetic Algorithm Process

The GA followed an evolutionary approach, iterating over multiple generations to refine hyperparameters. The process consisted of:

- **Population Size:** 3 (each generation consists of 3 hyperparameter sets)
- **Generations:** 5 (iterative evolution of hyperparameters)
- **Mutation Rate:** 10% (randomly modifying one parameter)

- **Crossover:** Parents were selected based on validation accuracy, and offspring were created by combining their hyperparameters.

4.3.2.3 Results for EfficientNetB3

Over five generations, the algorithm refined hyperparameters, leading to improved validation accuracy. The following figures illustrate training and validation performance trends for different hyperparameter configurations across generations.

The Following Graphs shows the training and validation accuracies and loss for the respective hyperparameters used for EfficientNetB3.

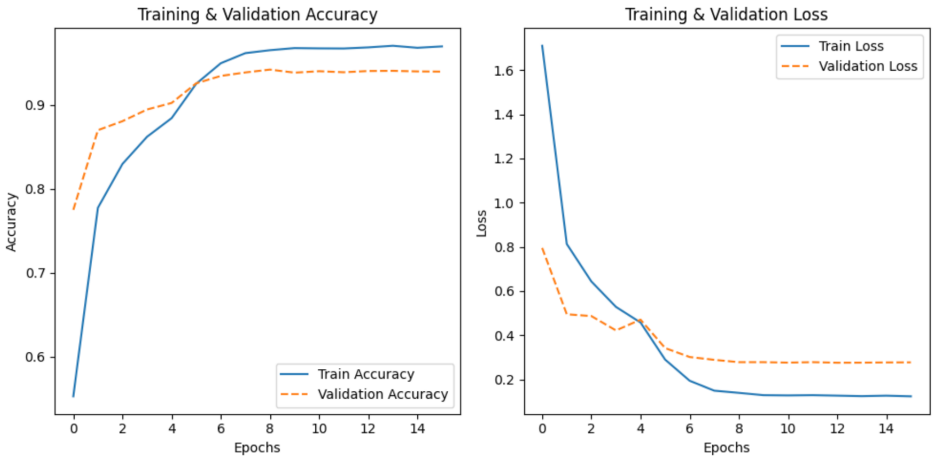

Figure 4.17: Accuracy and Loss for EfficientNetB3 in generation 1

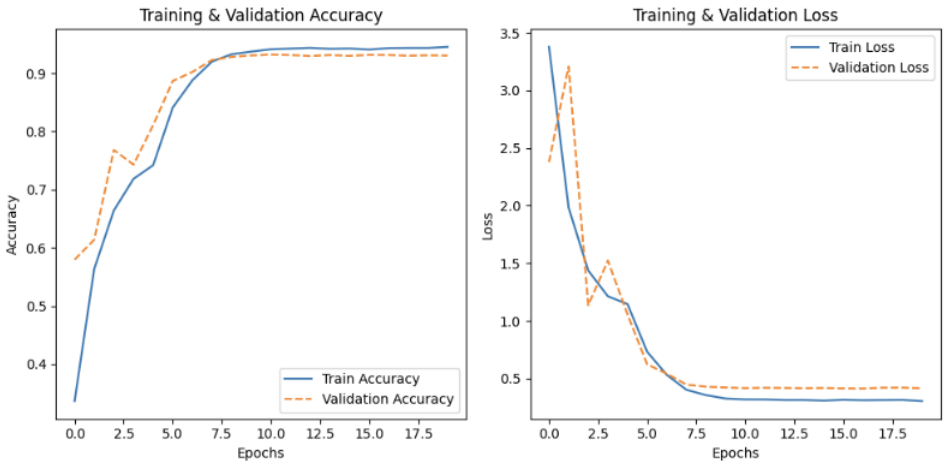

Figure 4.18: Accuracy and Loss for EfficientNetB3 in generation 2

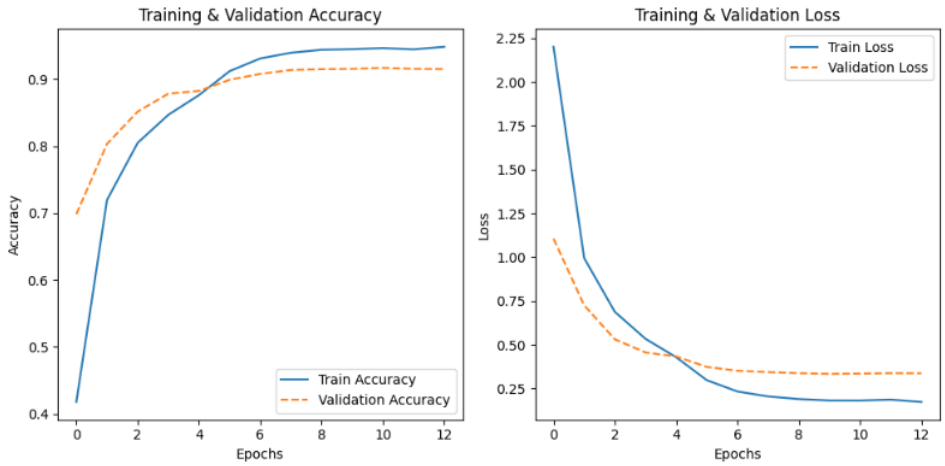

Figure 4.19: Accuracy and Loss for EfficientNetB3 in generation 3

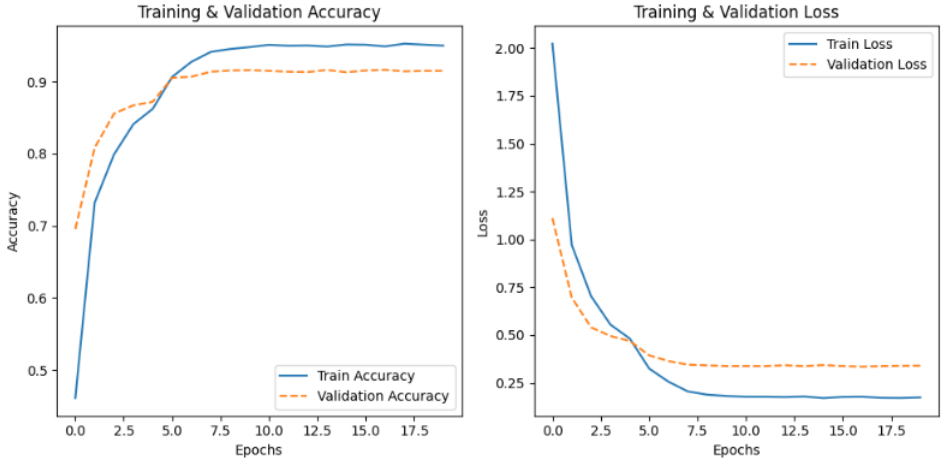

Figure 4.20: Accuracy and Loss for EfficientNetB3 in generation 4

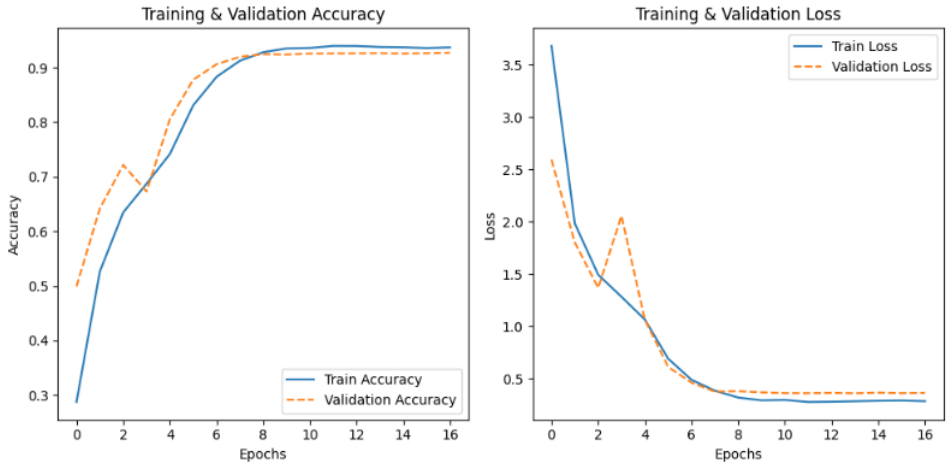

Figure 4.21: Accuracy and Loss for EfficientNetB3 in generation 5

The following table demonstrates different hyperparameter configurations and their corresponding test accuracies.

SN	Batch	Img Size	LR	Epochs	Dropout	L2	Test Accuracy (%)
1	**8**	**224x224**	**0.0001**	**20**	**0.2**	**1e-4**	**93.37**
2	8	128x128	0.001	20	0.4	1e-4	92.54
3	16	128x128	0.0001	20	0.3	1e-4	90.68
4	8	128x128	0.0001	20	0.2	1e-4	91.02
5	8	128x128	0.001	20	0.2	1e-4	92.54

Table 4.5: Hyperparameter configurations and test accuracies for Efficient-NetB3

4.3.2.4 Conclusion for EfficientNetB3

The best model achieved a Test Accuracy of **93.37%** with a **Batch Size of 8**, **Image Size of 224x224**, **Learning Rate of 0.0001**, **Epochs = 20**, **Dropout Rate of 0.2**, and **L2 Regularization = 1e-4**. These results highlight the effectiveness of the Genetic Algorithm in systematically optimizing hyperparameters, leading to improved model performance and reduced overfitting.

4.4 Mobile Application

The mobile application allows the user to record the audio, visualize it with waveforms during recording, before sending the actual audio data to the server for processing. The user can also use the application to add their current location to the map.The homepage of the application allows the user to view the birds along with the audio if they are available in FeatherFind's database. The map page in FeatherFind allows the user to view all the birds that has been located using the application and view their location in the maps embedded in the application alone with the detailed description of the birds.

For the audio recognition the user can record the audio in the application and as the audio is being recorded the app visualizes the audio in appropriate waveform. When the user stops recording the audio, the user is prompted with a confirmation dialog, which can be used to process the audio and recognize it.

4.5 Web Application

The web application developed for our project is designed to facilitate the recording, identification, and mapping of bird species using a combination of React for the frontend and Django for the backend. It consists of two primary sections: Record and Map, each serving a distinct yet interconnected purpose in the study and visualization of bird calls.

In the Record section, users can capture bird sounds directly through the web application. The interface provides a canvas displaying a spectrogram, which visualizes the audio frequencies of the recorded sound over time. Once the recording is complete, the data is processed by the backend, where the system analyzes the audio and determines the corresponding bird species. The backend then returns the predicted species and its confidence score, and the web application provides users with an option to visit the respective Wikipedia page for more information about the identified bird. This feature enhances the user experience by integrating automated species recognition with an educational component.

The Map section utilizes OpenStreetMap with the Leaflet library to present a detailed visualization of bird sightings within the user's region. The application retrieves previously detected bird locations from an SQL database, displaying them as markers on the map. Additionally, it allows users to identify clusters of similar bird species based on past records, enabling pattern recognition and ecological study. A search functionality is also included, enabling users to look up specific bird species and view their recorded sightings. This mapping capability helps in tracking bird populations over time and provides valuable insights into their distribution.

By combining real-time audio analysis with geospatial visualization, the web application serves as a comprehensive tool for bird enthusiasts and researchers alike. The integration of React and Django ensures a seamless and efficient user experience, while the SQL database facilitates structured storage of bird detection data for future reference and analysis.

CHAPTER 5
CONCLUSION AND FUTURE IMPROVEMENTS

The project on bird species identification from audio recordings has yielded promising results, demonstrating the efficacy of advanced machine learning techniques in bioacoustic analysis. Our approach involved the development of two primary models: a bird sound detection model based on a modified ResNet-50 architecture and a bird species classification model utilizing EfficientNetB3 and CNN-LSTM architectures. The bird sound detection model achieved an impressive accuracy of 82.97%, with an AUC score of 0.911, indicating its strong ability to distinguish bird sounds from background noise. This model's performance underscores the importance of robust feature extraction and data augmentation techniques in handling diverse and noisy audio environments. On the other hand, the bird species classification model, particularly the EfficientNetB3 architecture, demonstrated superior performance with an accuracy of 90.73% and an F1-score of 91%. This high level of accuracy highlights the model's capability to effectively classify a wide range of bird species based on their vocalizations. The use of genetic algorithms for hyperparameter optimization further enhanced the models' performance, showcasing the potential of evolutionary algorithms in fine-tuning complex neural networks. We found that Mel-Spectrograms are best suited for CNN-based architectures, while MFCCs are more effective for RNN-based models. Additionally, the EfficientNetB3 model outperformed the CNN-LSTM model, indicating that compound scaling in neural networks can significantly improve classification accuracy. Overall, this project represents a significant step forward in the application of machine learning to avian bioacoustics, offering valuable tools for conservation efforts, biodiversity monitoring, and ecological research.

5.1 Limitations

- **Dataset Limitations:** While the dataset provides a solid foundation, it has limitations in species diversity, environmental noise, and recording conditions. These factors affect the model's accuracy when applied to new or less-represented bird species. Overrepresentation of certain species has led to biased predictions, mak-

ing identification less reliable for species with fewer training samples.

- **Multiple Bird Sounds:** The system is designed to recognize a single dominant bird call in an audio recording. When multiple birds vocalize simultaneously, the model may misclassify calls or overlook background species, reducing accuracy in complex soundscapes.
- **Internet Dependency:** The prediction process requires an active internet connection since the model runs on a cloud-based service. Limited or no connectivity may prevent users from accessing real-time predictions.

5.2 Future Improvements

- **Enhanced Dataset:** Expanding the dataset to include more diverse bird species and recording conditions will improve the model's generalization and accuracy. Collaborating with ornithologists and bird enthusiasts to gather more data can help achieve this goal.
- **Multi-Bird Sound Recognition:** Developing algorithms to handle multiple overlapping bird calls in a single recording will enhance the system's robustness in real-world scenarios. Techniques such as source separation and advanced signal processing can be explored.
- **Offline Functionality:** Implementing offline capabilities in the mobile application will allow users to record and analyze bird sounds without requiring an internet connection. This can be achieved by integrating a lightweight version of the model directly into the app.
- **User Feedback Integration:** Incorporating user feedback mechanisms to validate and improve predictions can enhance the system's accuracy over time. Users can provide corrections or additional information about the identified species, contributing to continuous model improvement.
- **Real-Time Processing:** Optimizing the model for real-time processing will enable instant identification of bird species during field observations. This can be particularly useful for bird watchers and researchers in remote locations.

REFERENCES

[1] I. O. Union, "Ioc world bird list," https://www.worldbirdnames.org/new/updates/, 2024, accessed: 2024-06-04.

[2] H. Nature, "National red list of nepal's birds," https://www.himalayannature.org/works/projects/national-red-list-of-nepals-birds/, 2024, accessed: 2024-06-03.

[3] C. Inskipp, H. S. Baral, T. Inskipp, A. P. Khatiwada, M. P. Khatiwada, L. P. Poudyal, and R. Amin, "Nepal's national red list of birds," *Journal of Threatened Taxa*, vol. 9, no. 1, pp. 9700–9722, 2017.

[4] K. L. Y. C. for Conservation Bioacoustics, "Birdnet: Analyzing bird sounds using machine learning," https://birdnet.cornell.edu/, 2025, accessed: 2024-06-03.

[5] R. Gautam, B. Khatiwada, B. P. Subedi, N. Duwal, and K. C. Dahal, "Audio classifier for automatic identification of endangered bird species of nepal," 2023.

[6] A. Sevilla and H. Glotin, "Audio bird classification with inception-v4 extended with time and time-frequency attention mechanisms." *CLEF (Working Notes)*, vol. 1866, pp. 1–8, 2017.

[7] B. Chandu, A. Munikoti, K. S. Murthy, G. Murthy, and C. Nagaraj, "Automated bird species identification using audio signal processing and neural networks," in *2020 International Conference on Artificial Intelligence and Signal Processing (AISP)*. IEEE, 2020, pp. 1–5.

[8] L. Nanni, G. Maguolo, S. Brahnam, and M. Paci, "An ensemble of convolutional neural networks for audio classification," *Applied Sciences*, vol. 11, no. 13, p. 5796, 2021.

[9] H. Wang, Y. Xu, Y. Yu, Y. Lin, and J. Ran, "An efficient model for a vast number of bird species identification based on acoustic features," *Animals*, vol. 12, no. 18, p. 2434, 2022.

[10] R. Pahuja and A. Kumar, "Sound-spectrogram based automatic bird species recognition using mlp classifier," *Applied Acoustics*, vol. 180, p. 108077, 2021.

[11] S. Carvalho and E. F. Gomes, "Automatic classification of bird sounds: using mfcc and mel spectrogram features with deep learning," *Vietnam Journal of Computer Science*, vol. 10, no. 01, pp. 39–54, 2023.

[12] M. Lasseck, "Acoustic bird detection with deep convolutional neural networks." in *DCASE*, 2018, pp. 143–147.

[13] A. Reiling, W. Mitchell, S. Westberg, E. Balster, and T. Taha, "Cnn optimization with a genetic algorithm," in *2019 IEEE National Aerospace and Electronics Conference (NAECON)*, 2019, pp. 340–344.

ANNEX

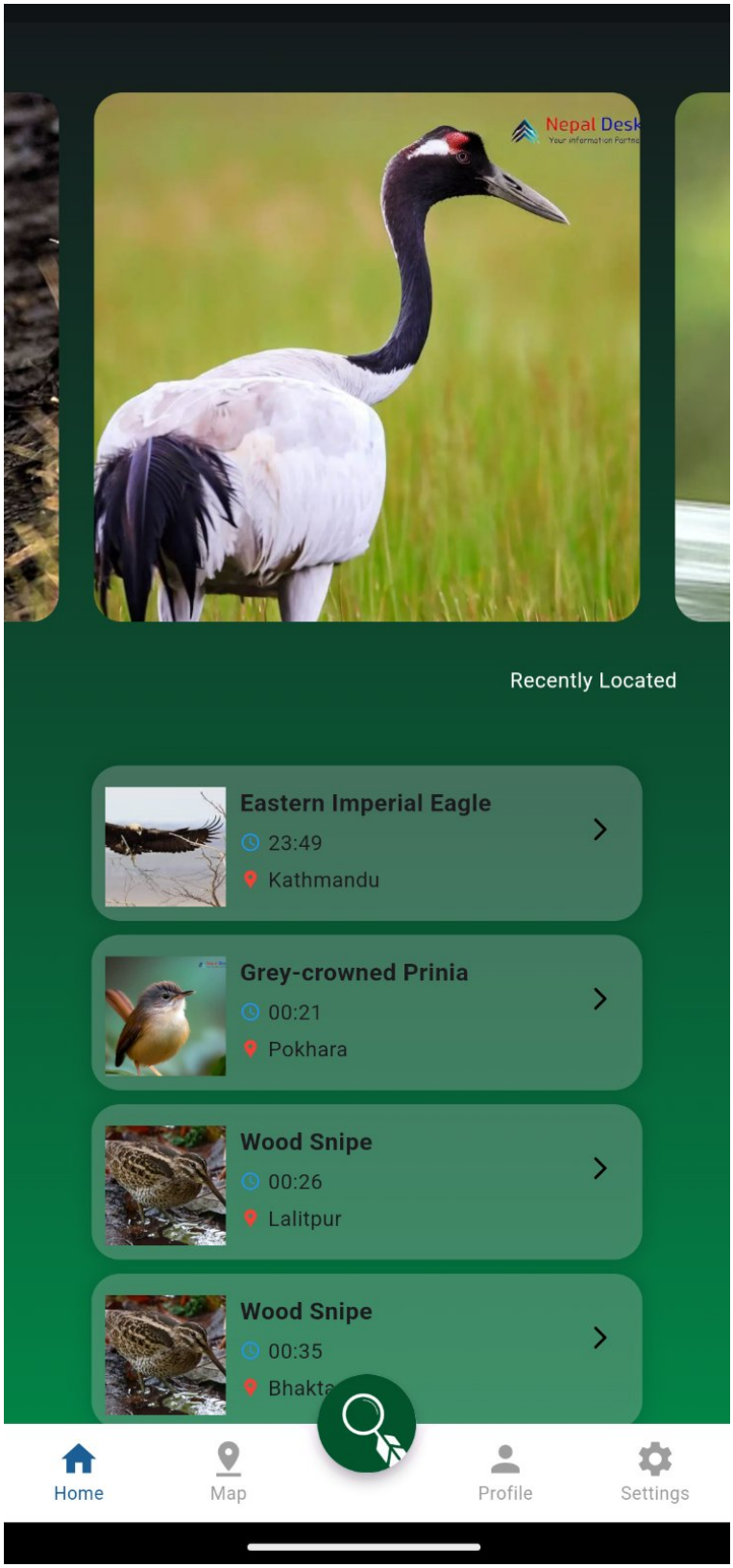

Figure 5.1: Home page of the mobile app.

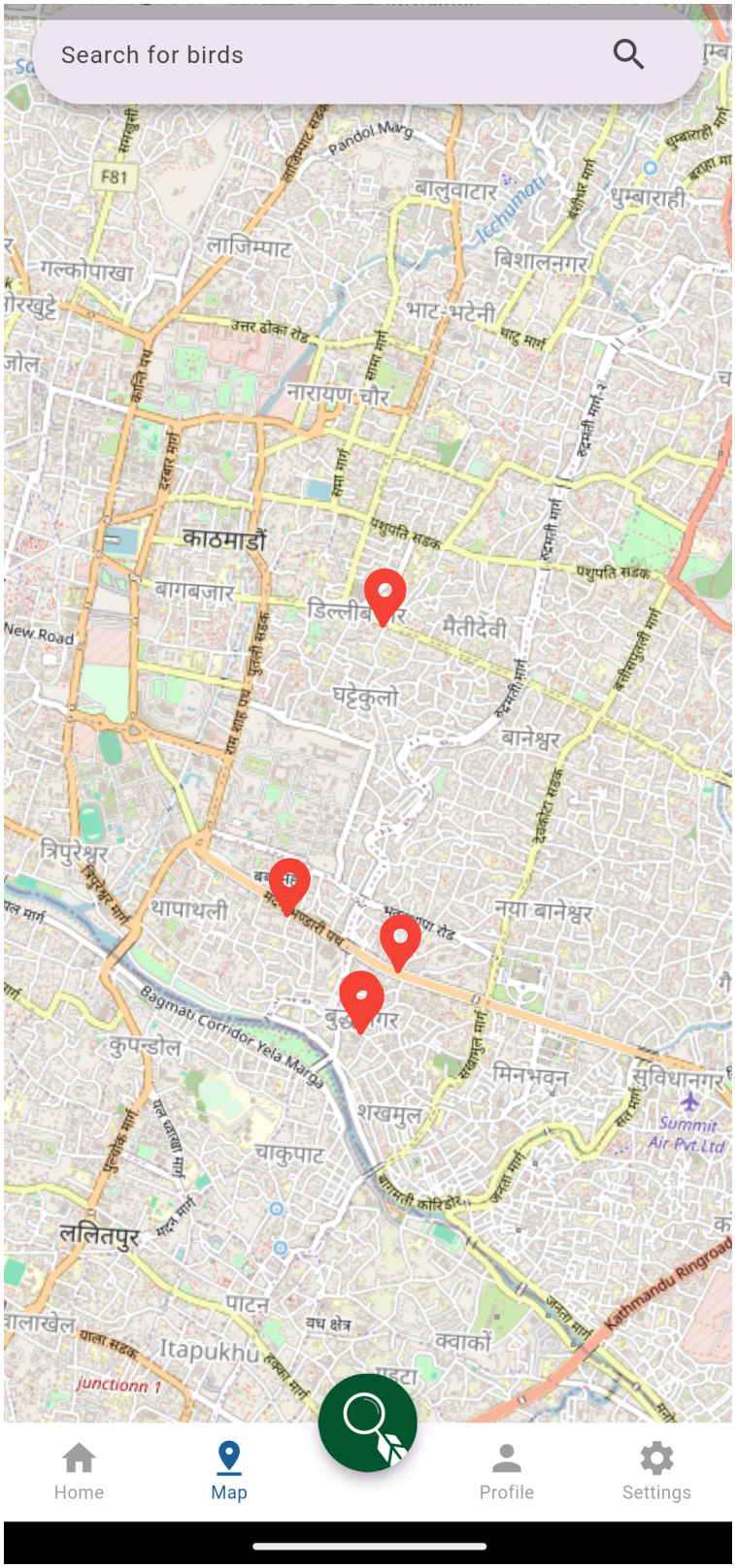

Figure 5.2: Mapped birds.

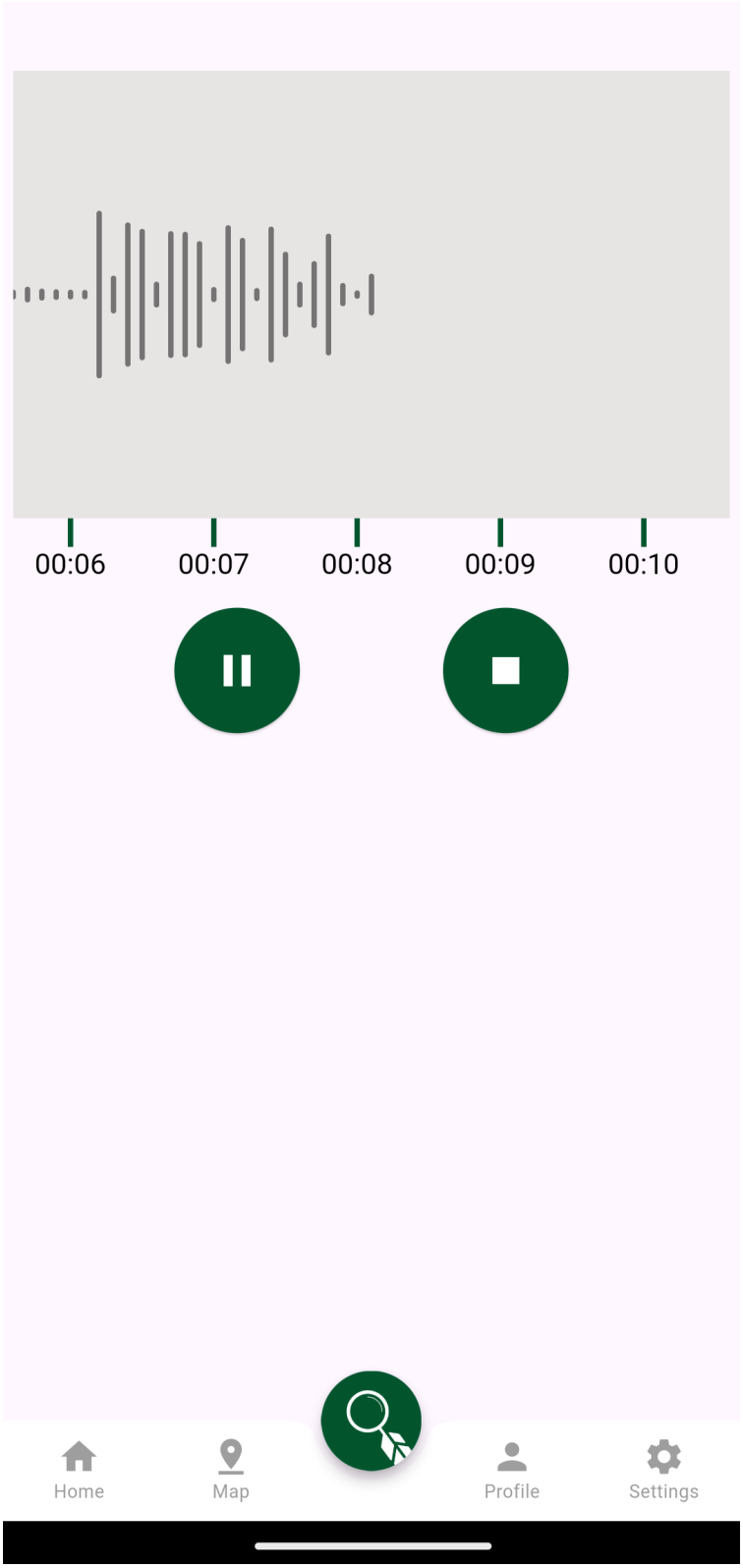

Figure 5.3: Bird call recording using the mobile app.

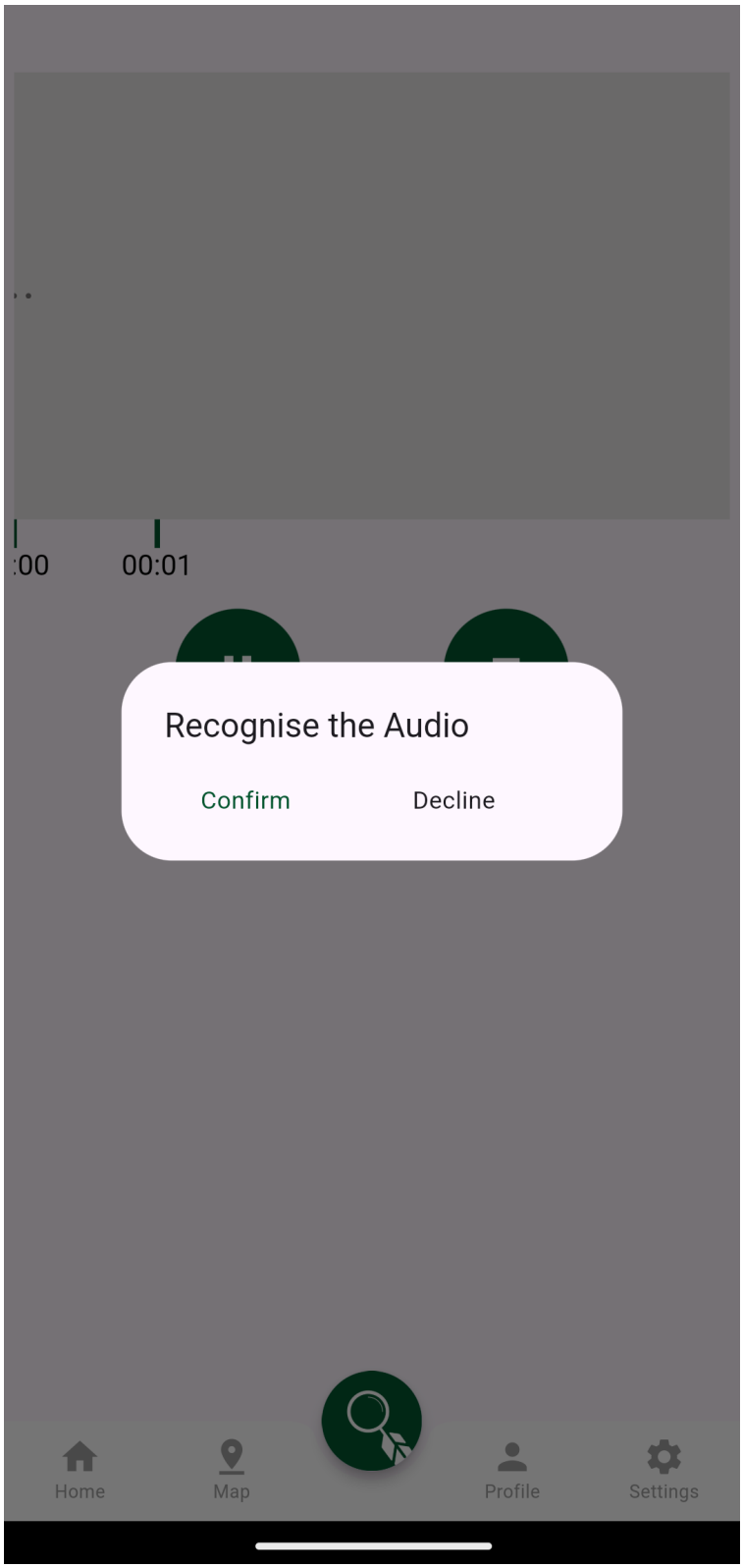

Figure 5.4: Confirmation for audio recognition using the mobile app.

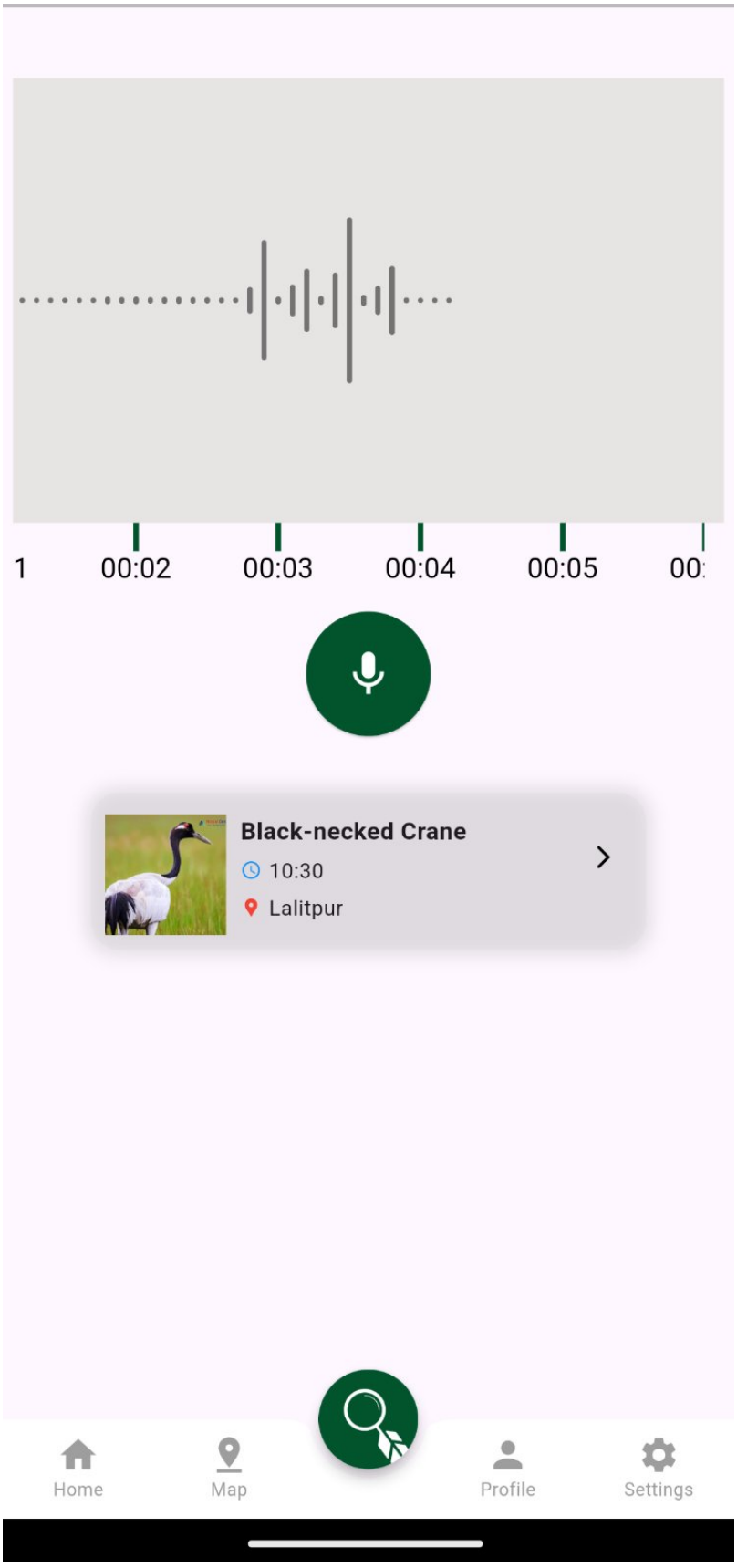

Figure 5.5: Bird species identification using the mobile app.

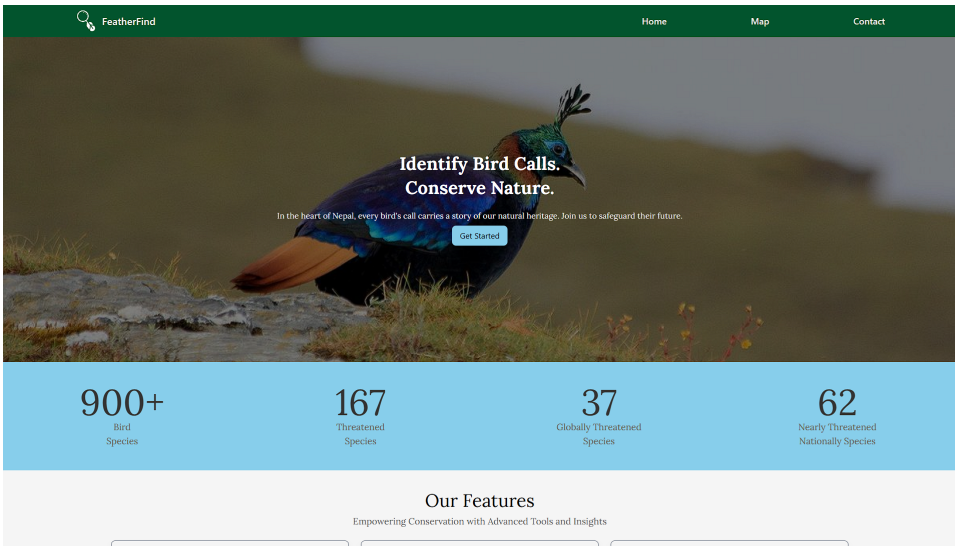

Figure 5.6: Home page of the web app.

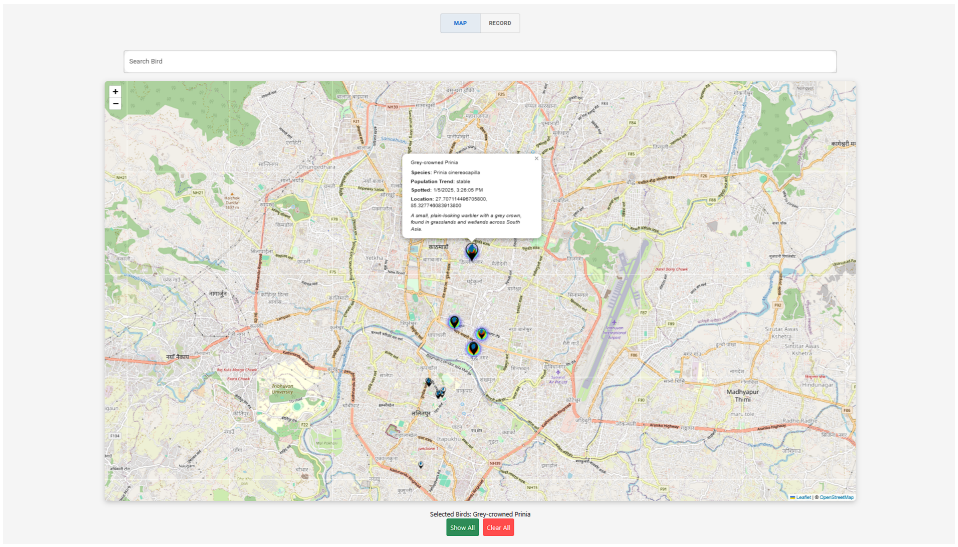

Figure 5.7: Map page of the web app.

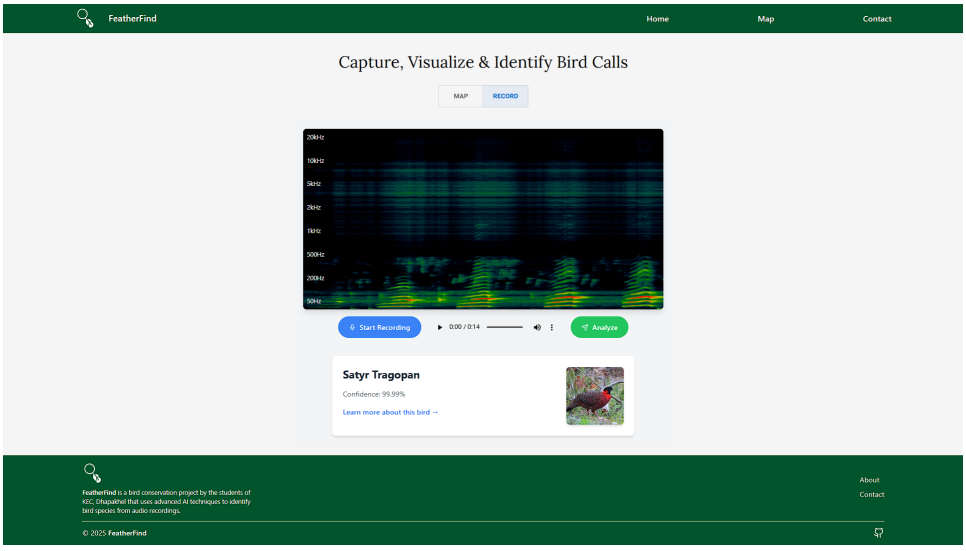

Figure 5.8: Record page of the web app.