Report for automated detection of bird species

Mullapudi Surya Karthikeya(210001041)
G Aakash(210001015)
Mahendraker Gaurav(210001036)

April 20, 2024

1 Introduction

In recent years, advancements in artificial intelligence (AI) and machine learning have revolutionized various fields, including environmental conservation and biodiversity monitoring. One such innovative application is the automated detection and classification of bird species using sound recordings. Birds play a crucial role in ecosystems as indicators of environmental health, yet traditional methods of monitoring, such as visual surveys, can be labor-intensive, time-consuming, and prone to human error. Leveraging AI technologies to analyze the acoustic signatures of bird vocalizations offers a promising solution to these challenges.

The distinctive vocalizations produced by different bird species serve as unique identifiers, allowing for the development of AI models capable of accurately detecting and classifying birds based solely on their calls or songs. This approach enables researchers, conservationists, and wildlife managers to efficiently gather data on bird populations across diverse habitats and geographical regions, facilitating informed decision-making for conservation efforts.

This introduction sets the stage for exploring the significance of AI-powered automated detection systems in bird species monitoring, highlighting their potential to enhance our understanding of avian ecology, track population trends, and inform conservation strategies in an increasingly human-impacted world.

2 Problem Statement

The project involves automating the detection of bird and frog species in tropical soundscape recordings. The audio clips we are working with contain multiple species, making this problem a multilabel classification task using audio data. Our objective is to develop a model that can accurately identify the probability of ach of the species being present in the given audio file.

3 Dataset Description

The dataset for our project comprises audio files along with a CSV file that provides detailed descriptions of these audio files. The CSV file contains several columns essential for our analysis, including `recording_id` (a unique identifier for each recording), `species_id` (a unique identifier for each species present in the recording), `songtype_id` (a unique identifier for the type of song), `t_min` (the start second of the annotated signal), `f_min` (the lower frequency of the annotated signal), `t_max` (the end second of the annotated signal), `f_max` (the upper frequency of the annotated signal), and `is_tp` (an indicator of whether the label is from the train_tp or train_fp file, available in TFRecords only).

Each audio file in the dataset can potentially contain multiple species, with each species having different time ranges and frequency ranges within the audio clip. Additionally, our dataset includes frequency ranges for each species, which are derived from the available information.

To identify a species within an audio file, we need to focus on the given time and frequency range information provided in the CSV file. This involves analyzing the audio signals within the specified time intervals and frequency bands to determine the presence and identity of each species in the recording.

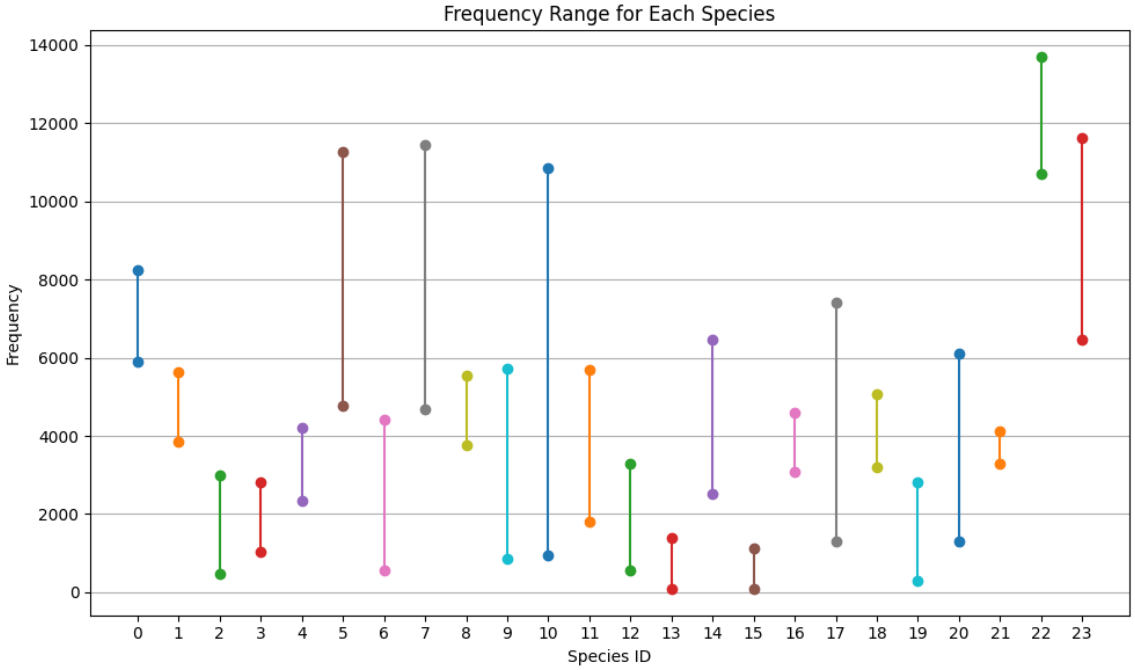

Figure 1: Frequency Range of each Species

4 Data Collection

The data was collected across 749 sites in the island of Puerto Rico mainly from the El Yunque National Forest(around 50 percent) and consists of frequencies in the range of 93.8 Hz to 13687.5 Hz. Audio Moth Recorders were used to collect acoustic data. Recorders were placed on trees at the height of 1.5 m and programmed to record 1 min of audio every 10 min for a total of 144 recordings per day at a sampling rate of 48 kHz. A small portion (10 percent) of recordings were sampled at 44.1 kHz. In total, 97,900 1-min soundscape recordings were weakly or fully annotated with species-specific time-frequency bounding boxes. 78 percent of recordings were collected in 2019, 11 percent in 2018, and 11 percent collected between 2015 and 2018. Most recordings were collected in the months March and April, a period of high acoustic activity.

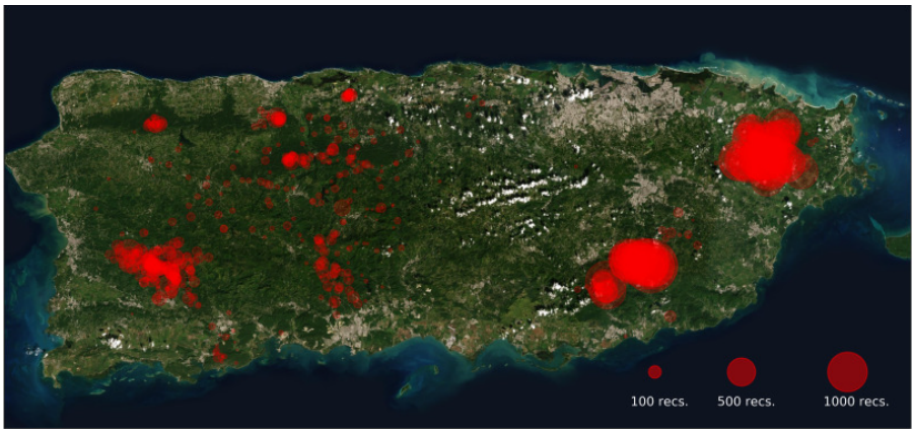

Figure 2: Data Distribution

Spatial distribution of recordings used for training and testing is given above. Circles represent recorder sites with diameter indicating the number of recordings used. Most recordings came from El Yunque National Forest (large cluster in upper right) and the test recordings were sampled from this area.

Training data was generated using template matching. The data was then classified into true and false positives by expert consultation. The data consists strong labelling of the events and the output model is expected to give a weak labelling ie just the probability of the bird being present. No detail about when the bird is detected is required to be provided.

5 Data Preprocessing

The data preprocessing steps involve several key operations to prepare the audio data for analysis. Firstly, each audio file is loaded using the `librosa` library, and the sample rate is set to a specific value (48000 Hz in this case) to ensure

consistency across the dataset. The audio is then divided into shorter segments of 5 seconds each to facilitate processing.

Next, the audio data is converted into a Mel spectrogram using the librosa.feature.melspectrogram function. The choice of a Mel spectrogram is motivated by its ability to approximate the human auditory system's frequency response, making it suitable for analyzing audio signals in a manner similar to human hearing.

To focus on relevant time intervals within each audio segment, time cropping is applied. This involves determining the center of the segment based on the annotated start and end times and then extracting a 5-second segment centered around this point. If the segment exceeds the 5-second duration, it is trimmed accordingly.

Frequency masking is another crucial preprocessing step where frequencies outside the specified range (`f_min` to `f_max`) for each species are masked out. This is achieved by creating a boolean mask based on the frequency range and applying it to the Mel spectrogram. Frequencies outside the specified range are set to the minimum value, effectively masking them from further analysis.

These preprocessing steps ensure that the audio data is transformed into a format suitable for machine learning algorithms to extract meaningful features and accurately classify species based on their acoustic signatures.

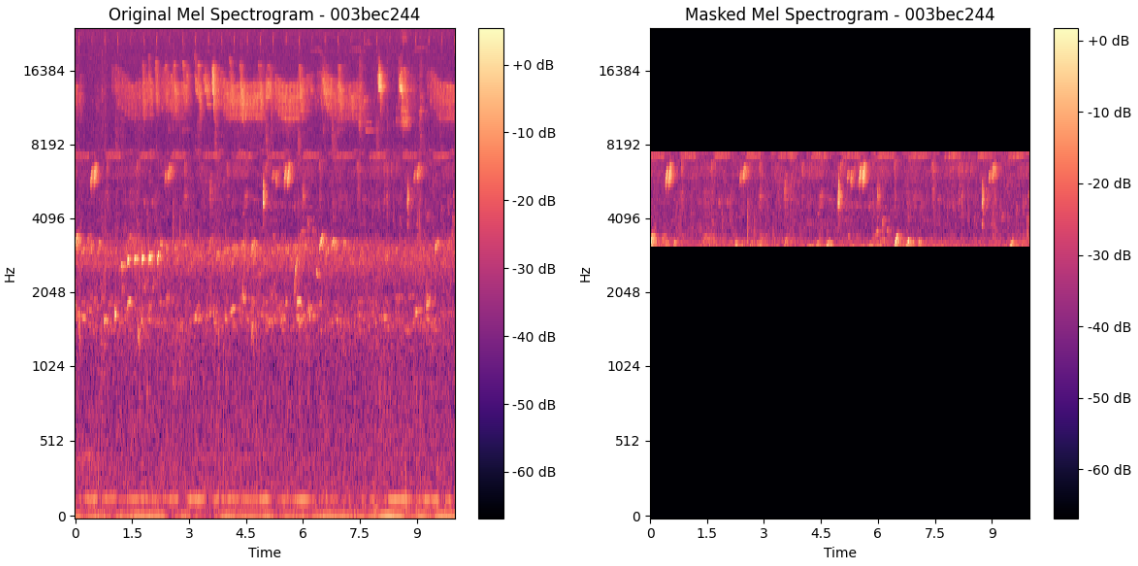

Figure 3: Preprocessing on Dataset

6 Algorithm and Model pipeline

6.1 Model Formulation

To implement a deep learning model, all the files were loaded using Librosa and then sliced accordingto the time at which a species is heard (i.e., let's say audio files is of 60 s and a species is heard in the audio at5-8 s then audio signal at that particular time was separated to use it for further steps). The audio data preprocessing pipeline incorporates essential augmentation techniques to enhance model robustness. Alongside initializing parameters, such as augmentation probabilities and durations, the pipeline segments audio files based on species presence, utilizing 5-second intervals. Augmentation, applied with a 50% probability, includes addition of audio segments from various recordings and temporal augmentation through artificial beat insertion, fostering rhythm pattern diversification. Moreover, techniques like Gaussian and pink noise addition, time shifting, and volume control modulations improve data diversity. Frequency and time masking further enrich feature extraction, contributing to a more resilient and adaptable deep learning model for audio classification.

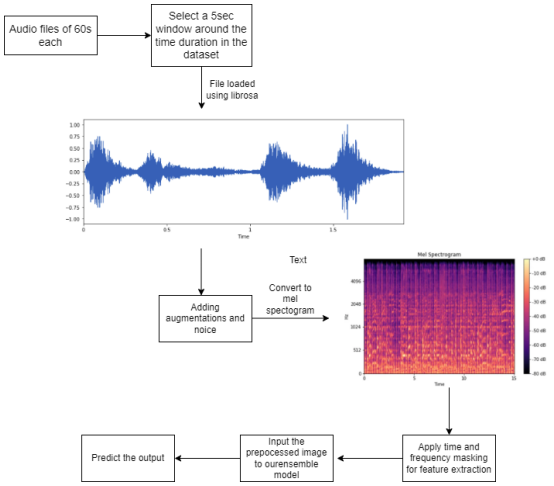

Figure 4: Model pipeline

6.2 Deep Learning Models

Here, we discuss various deep learning models utilized for audio classification tasks.

6.2.1 CNN (Convolutional Neural Network)

The Convolutional Neural Network (CNN) is a powerful deep learning architecture widely used for various computer vision and audio classification tasks. It is

particularly well-suited for tasks where the input data has a grid-like topology, such as images and spectrograms.

In the context of audio classification, CNNs operate on spectrogram representations of audio signals. Spectrograms provide a 2D representation of the audio signal's frequency content over time, making them ideal inputs for CNNs.

CNNs typically consist of multiple layers, including convolutional layers, pooling layers, and fully connected layers. The convolutional layers apply filters to the input spectrogram, extracting spatial features through convolutions. The pooling layers downsample the feature maps, reducing their spatial dimensions while retaining important features. Finally, the fully connected layers perform classification based on the learned features.

One of the key advantages of CNNs is their ability to automatically learn hierarchical representations of data. Lower layers of the network capture low-level features such as edges and textures, while higher layers learn more abstract features relevant to the task at hand.

Training a CNN for audio classification involves feeding labeled spectrogram data into the network and optimizing its parameters using gradient-based optimization algorithms such as stochastic gradient descent (SGD) or Adam.

CNNs have demonstrated impressive performance in various audio classification tasks, including speech recognition, music genre classification, and environmental sound classification. However, designing an effective CNN architecture requires careful consideration of factors such as network depth, filter sizes, and regularization techniques to prevent overfitting and maximize performance.

6.2.2 MobileNet

MobileNet is a lightweight convolutional neural network (CNN) architecture designed by Google. It aims to provide efficient and fast inference on mobile and embedded devices with limited computational resources. MobileNet achieves this by using depthwise separable convolutions, which separate the spatial and channel-wise filtering processes, reducing the number of parameters and computational cost while maintaining high accuracy. MobileNet models come in different versions (e.g., MobileNetV1, MobileNetV2), with improvements in performance and efficiency in subsequent versions.

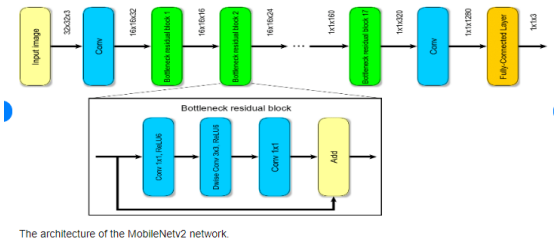

Figure 5: Architecture of a MobilenetV2 model

6.2.3 EfficientNet

EfficientNet is a family of CNN architectures introduced by Google's AutoML team. It employs a novel compound scaling method that uniformly scales network depth, width, and resolution with fixed scaling coefficients to achieve state-of-the-art performance with significantly fewer parameters compared to traditional models. This approach optimizes the trade-off between model size and accuracy, making EfficientNet models highly efficient and adaptable to different resource constraints.

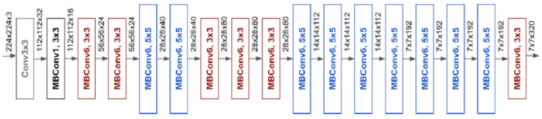

Figure 6: Architecture of a Efficientb0 model

6.2.4 ResNet

Residual Network (ResNet) is a deep CNN architecture proposed by Microsoft Research. It introduces residual connections, also known as skip connections, which enable training of very deep neural networks by mitigating the vanishing gradient problem. ResNet architectures come in various depths, such as ResNet-18, ResNet-50, ResNet-101, and ResNet-152, with deeper models having more layers. ResNet has been widely adopted in various computer vision tasks and benchmarks, achieving state-of-the-art performance on tasks like image classification, object detection, and semantic segmentation.

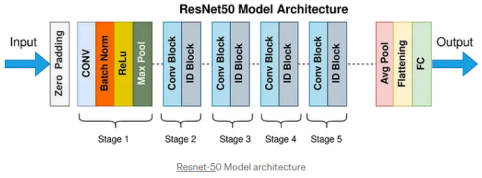

Figure 7: Architecture of a ResNet model

6.2.5 Proposed Ensemble Model Fusion of EfficientNet, MobileNetV2, and ResNet50

In this section, we present an ensemble model that leverages the strengths of three prominent convolutional neural network architectures: EfficientNet, MobileNetV2, and ResNet50. The ensemble model aims to combine the individual predictions of these models to enhance the accuracy and robustness of bird sound classification.

The ensemble model architecture begins by incorporating EfficientNet, MobileNetV2, and ResNet50 as the backbone networks. Each of these models is capable of producing predictions for the specified number of classes (n_class), which in our case is 24.

To integrate these models into the ensemble, we concatenate their output layers, resulting in a combined feature representation of size $24 \times 3 = 72$. We then introduce an additional dense layer with 24 neurons to process this concatenated feature representation and generate the final predictions for the 24 target classes.

The ensemble model architecture follows a straightforward yet effective approach, combining the predictions from three distinct models to form a comprehensive understanding of the input bird sound samples. By leveraging the complementary strengths of EfficientNet, MobileNetV2, and ResNet50, the ensemble model aims to achieve superior performance in sound classification tasks.

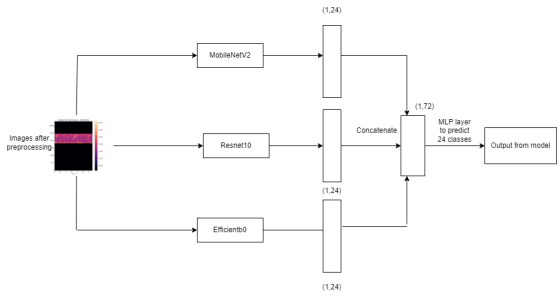

Figure 8: Ensemble model architecture

6.3 Proposed Algorithm

6.3.1 Preprocessing

1. Load audio files using Librosa.
2. Slice the audio files based on the time at which a species is heard.
3. Segment audio files into 5-second intervals.
4. Apply essential augmentation techniques with a 50% probability:
   - Add audio segments from various recordings.
   - Insert artificial beats to diversify rhythm patterns.
   - Add Gaussian and pink noise.
   - Perform time shifting and volume control modulations.
   - Convert the audio data to mel spectogram
   - Apply frequency and time masking to enrich feature extraction.

6.3.2 Model Implementation

1. Pass preprocessed data to the input layer of the ensemble model.
2. Load pre-trained EfficientNet, MobileNetV2, and ResNet50 models.
3. Modify the classification heads of each model to output the specified number of classes.
4. Define an ensemble model architecture that combines the predictions of the three models.
5. Create an `EnsembleModel` class with three backbone networks and an output layer.
6. Implement the `forward` method to concatenate the outputs of the backbone networks and pass them through the output layer.
7. Apply dropout with a dropout rate of 0.3 to avoid overfitting.

7 Evaluation Metrics

7.1 Cross Entropy Loss

Cross-entropy loss, also known as log loss, quantifies the difference between two probability distributions. In classification tasks, it measures the dissimilarity between the predicted probability distribution and the actual distribution of the target labels. It is commonly used as a loss function in scenarios where the output can be interpreted as probabilities, such as multi-class classification problems. Cross-entropy loss penalizes the model more heavily for making confident incorrect predictions, making it suitable for training models to output well-calibrated probabilities.

$$\text{Cross-Entropy Loss} = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{M} y_{ij}\log(p_{ij}) \tag{1}$$

$$\text{softmax}(x_i) = \frac{e^{x_i}}{\sum_{j=1}^{N} e^{x_j}} \tag{2}$$

7.2 LWLRAP

The Label-Weighted Label Ranking Average Precision (LWLRAP) is an evaluation metric commonly used in multi-label classification tasks. It measures the quality of the ranking of the predicted labels for each sample.

Let L be the set of true labels for a sample, P be the set of predicted labels, and S be the corresponding scores or probabilities assigned to each label by the classifier.

For each sample, the LWLRAP is calculated as follows:

$$\text{LWLRAP} = \frac{1}{|L|} \sum_{l \in L} \frac{|L_l|}{\min(|L|, |P|)} \sum_{p \in P} S_{lp} I(p \in L_l)$$

Where: - $|L|$ is the number of true labels for the sample. - $|L_l|$ is the number of labels in L that are ranked higher than label l in the predicted list P. - $I(p \in L_l)$ is an indicator function that returns 1 if label p is in the set L_l of labels ranked higher than label l, and 0 otherwise.

The LWLRAP ranges from 0 to 1, where a higher value indicates better performance in ranking the true labels higher in the predicted list.

8 Experimentation and Results

We have tried three models ResNet, EfficientNet, and MobileNet, the accuracy is represented by the table below.
The Ensembled model gave the best results as the model generalized well after training it for 10 epochs.

One of the key reasons for utilizing K-fold cross-validation(k=5) was its ability to mitigate overfitting. Overfitting occurs when a model learns to memorize the training data excessively, leading to poor performance on unseen data. By training on diverse subsets and validating on separate folds, K-fold cross-validation helps the model generalize better to new, unseen data.

The technique is particularly beneficial when dealing with limited datasets, as it maximizes the use of available data for both training and validation. This ensures that the model's performance estimates are more robust and less biased.

Moreover, K-fold cross-validation can be combined with hyperparameter tuning methods like grid search or random search to find the optimal model configuration that generalizes well.

In summary, our adoption of K-fold cross-validation was instrumental in improving the robustness and generalization of our machine learning model by effectively reducing overfitting and providing more reliable performance estimates.

The ensemble model, evaluated using K-fold cross-validation(k=5), demonstrated a validation loss of 0.162 and a training loss of 0.154, showcasing not only its strong performance during training but also its robust generalization ability across unseen data.

Model	**Lwlrap score**
Ensembled ResNet—EfficientNet—MobileNet	0.925
ResNet	0.911
EfficientNet	0.907
MobileNet	0.891

Table 1: Comparison of Model Accuracies

9 Conclusion

Few people show interest these days in becoming a forest ranger. With the scarcity of experts in this area. In this project, multiple models were tried to classify the distinct species of tropical rainforest with the audio signal. The ensembled showed the best results. The model is created in such a way that it can give almost real-time prediction and does not require high-end GPU to work, making it a lot easier to productionize the model. This ensembled model showed good performance with 0.925 LWLRAP score and 0.162 multiclass log losses but it can surely be improved with the use of widely popular algorithm like Attention Mechanism.If more data can be collected for more types of species, this model can be retrained so that it can recognize a large variety of species.Our Model is deployed on https://streamlit.io/.

Github Repository https://github.com/cse210001015/CS-354-Final-Project.

10 References

- Intelligent Audio Signal Processing for Detecting Rainforest Species Using Deep Learning
- CNN architectures for large-scale audio classification problem (Stoyan, S.,R. Kwon)