Deep Learning for Bird Sound Classification

GitHub: https://github.com/Prateekmax21/Deep-Learning-for-Bird-Sound-Classification

By Prateek Pagare

Abstract

This report incudes a application of convolutional neural networks Deep Learning to classify bird species using audio spectrograms around seattle university campus. Using a dataset of 12 bird species of various lengths provided in a preprocessed HDF5 format , We developed two models: a binary model, and a multi-class model to identify all 12 species, The binary model achieved 90.2% validation accuracy with a mean absolute error of 0.0976, while the multi-class model reached 37.3% accuracy when classifying across all 12 species. We also tested our final multi-class model on three external audio clips and presented predictions using spectrogram visualization and top predict scores. Through this work, we explored key neural network concepts such as activation functions, overfitting, class imbalance, and model evaluation. All models were trained on spectrograms derived from the first few seconds of each audio clip using mel-frequency scales.

Introduction

The aim of this project is to develop and evaluate convolutional neural network (CNN) models capable of identifying bird species based on their audio calls. Birdsong classification offers a meaningful challenge in machine learning, especially when working with real-world, imbalanced, and acoustically complex datasets. This assignment focuses on birds commonly found around Seattle, with the goal of building a model that can accurately classify species from short sound clips.

The dataset used in this project originates from the Birdcall competition data, which compiles field recordings from the Xeno-Canto community an open-access, crowd-sourced archive of bird vocalizations. For this assignment, A cleaned and preprocessed subset of 12 species is provided in HDF5 format, with each clip transformed into a 128×517 mel-spectrogram after resampling to 22,050 Hz

In this project, we explore two classification problems. First, a binary classification model distinguishes between American Crow and Spotted Towhee.The selected species (see Figure 1) include familiar birds such as the American Crow and House Sparrow, with notable variation in

call type, duration, and recording quality. Second, a multi-class model attempts to classify among all 12 bird species. We also apply the final multi-class model to three mystery test clips and analyze the predictions visually and statistically. Throughout this work, we engage with deep learning concepts such as overfitting, dropout, softmax activation, and class imbalance handling, all implemented using Python and TensorFlow.

Figure 1: Photographs of the 12 bird species used in this study

Theoretical Background

Deep Learning

Deep learning is a specialized area of machine learning that uses layered models called neural networks to learn patterns directly from raw data. These models are particularly effective when working with complex data like images, text, or sound. Unlike traditional methods, which often rely on manual feature selection, deep learning automatically extracts important features as part of the training process. In this project, we apply deep learning to analyze bird calls, using visual representations of audio called spectrograms.

Neural Networks

a neural network is a mathematical system designed to recognize patterns. Inspired by the structure of the human brain, it consists of layers of small computational units called nodes. Layers can have as many as a dozen number of nodes or millions, depending on how complex neural networks will be required to uncover hidden patterns, it uses layers made of simple

processing units called nodes (or neurons). These nodes work together to learn patterns from data, just like our brain might learn to recognize a bird call after hearing it many times.

The architecture of a neural network (Fig 2)consists of three layers such as the input layer, the hidden layer, and the output layer.

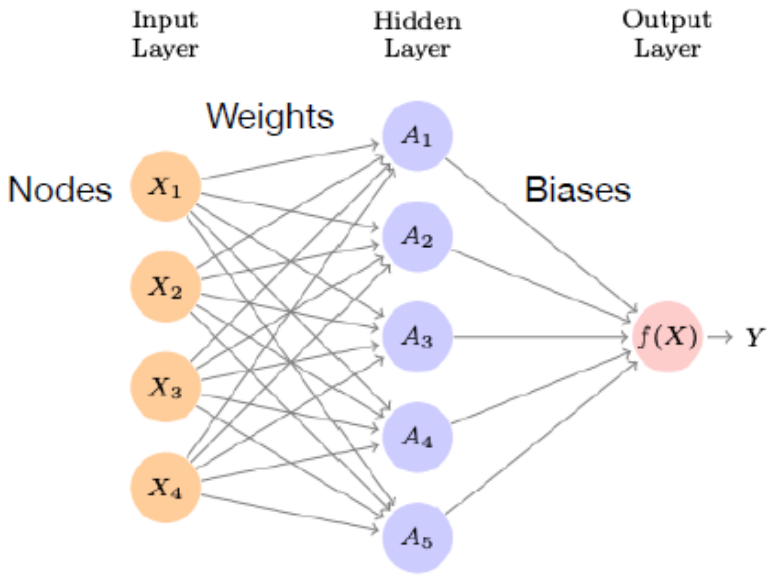


Fig 2 :- Neural Network Architecture.

The input layer of a neural network is the first layer It receives input information from external sources. The input data is available in the form of text, numbers, images, or audio files. Input is also a linear combination of data observations.

$$z_k = w_{k0} + \sum_{j=1}^{p} w_{kj} X_j$$

each neuron receives inputs (like numbers from an image or sound data). It doesn't just take them as-is. Instead, it multiplies each input XjX_jXj by a weight wkjw_{kj}wkj, which tells the neuron how important that input is. Then it adds up all these products and includes a small constant called a bias (written here as wk0w_{k0}wk0).

Hidden layers are middle layers in neural networks. There can be one or more hidden layers in a neural network depending on how deep the training data is used to uncover patterns. This layer extracts relevant patterns from input data and transfers them to the next layer for analysis, this layers are ideal for performing mathematical computations on input data.

$$A_k = g(z_k) = \frac{1}{1 + e^{-z_k}}$$

we have a linear combination, we need to help the network learn more complex relationships. That's done using a nonlinear activation function, like sigmoid function, This function squashes the number into a range between 0 and 1

The output layer consists of rigorous mathematical computations carried out by the hidden layer to obtain results based on these calculations. Additionally, it serves as a final output for bringing the information learned by the neural network. It is also a linear combination of the activation functions.

$$f(X) = \beta_0 + \sum_{k=1}^{K} \beta_k A_k$$

Finally, in the output layer, we combine the outputs (activations) from the hidden neurons, Here, the network adds up the activations from all the hidden layer neurons, each multiplied by a weight.

Activation Functions

Activation functions play a crucial role in neural networks. They are mathematical tools that determine whether a particular neuron should be "activated" or not. In simpler terms, they decide how much of the input signal should be passed forward through the network. Without activation functions, a neural network would behave like a simple linear model and would not be able to learn complex patterns.

One of the main jobs of an activation function is to introduce **non-linearity** into the model. Real-world data, like audio signals or images, are often highly non-linear, meaning their patterns are not simple straight lines. By applying non-linear transformations to the inputs, activation functions enable the network to learn more flexible and powerful relationships.

The most commonly used activation functions are sigmoid and RELU functions.

The sigmoid function (Fig 3) was one of the earliest activation functions used in neural networks. It compresses any real-valued number into a range between 0 and 1. This property makes it useful for binary classification problems, where we want the output to represent the probability of belonging to a certain class. However, the sigmoid function has some drawbacks. When the input is very large or very small, tsigmoid function produces very small gradients, where the network stops learning effectively.

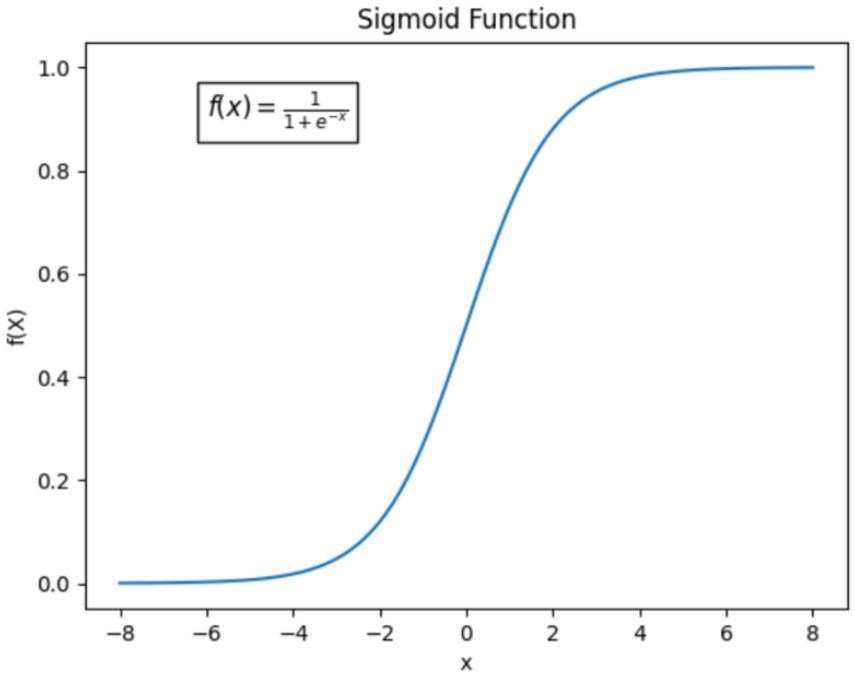


Fig 3:- Sigmoid Function

ReLU (Rectified linear unit | Fig 4) is a common choice for hidden layer activation in deep learning, especially for hidden layers. It is very simple: it returns the input value if it is positive, and zero otherwise. This makes it computationally efficient and helps to reduce issues like vanishing gradients. However, ReLU can sometimes lead to “dead neurons” , neurons that never activate for any input if their value stays negative.

$$f(x) = \max(0, x)$$

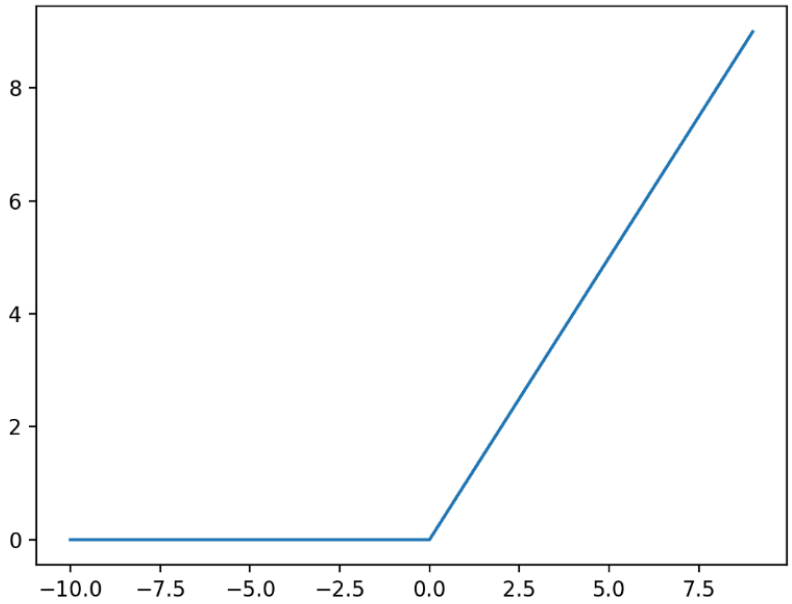


Fig 4:- RELU Activation

While ReLU is great for hidden layers, the softmax function (Fig 5) is commonly used in the output layer of a classification model, especially when there are multiple classes. Softmax turns a list of scores into probabilities that sum to 1, making it easier to interpret which class the model thinks is most likely. Each probability reflects how confident the model is about a particular class being correct.

$$\sigma(z)_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}}$$

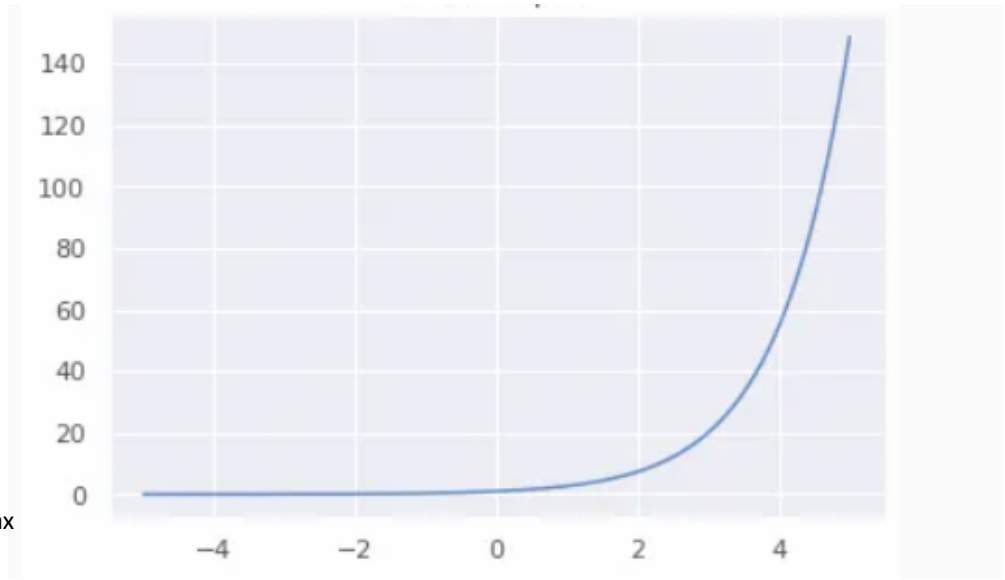


Fig 5:- Softmax

Loss Function

To train a neural network, we first need a way to measure how well or poorly it's doing. This is where a loss function comes in. Think of the loss function as a scorekeeper, it checks how far the model's predictions are from the correct answers. Mostly used Loss are binary_crossentropy and categorical_crossentropy. The bigger the mistake, the higher the loss. Our goal during training is to keep reducing this number until the model gets better at making accurate predictions.

Optimizer

Once the loss is calculated, an optimizer comes into play. It tries to find the best possible settings (called weights) inside the model that would reduce the loss. And most popular optimizers is called Adam and RMSprop, which is smart enough to adjust itself as training goes on. It learns faster and more smoothly compared to basic methods.

Batch Size

Instead of updating the model after seeing the entire dataset at once, we break the data into smaller groups called batches. For example, if we use a batch size of 64, the model learns from 64 examples at a time. Going through the entire dataset once is called an epoch. We usually run many epochs so the model can slowly improve over time.

This process of computing loss, adjusting the model using an optimizer, and repeating over multiple epochs is what helps the model learn to recognize patterns, in our case, the unique sound patterns of different bird species.

Convolutional Neural Network(CNN)

It (Fig 6) is a class of neural networks used to process data with a grid-like topology, such as images and audio spectrograms, Unlike traditional neural networks where each input is directly connected to every neuron in the next layer, CNNs focus on local patterns by scanning small filters (also called kernels) across the input.

CNNs typically have three layers: a convolutional layer, a pooling layer, and a fully connected layer.

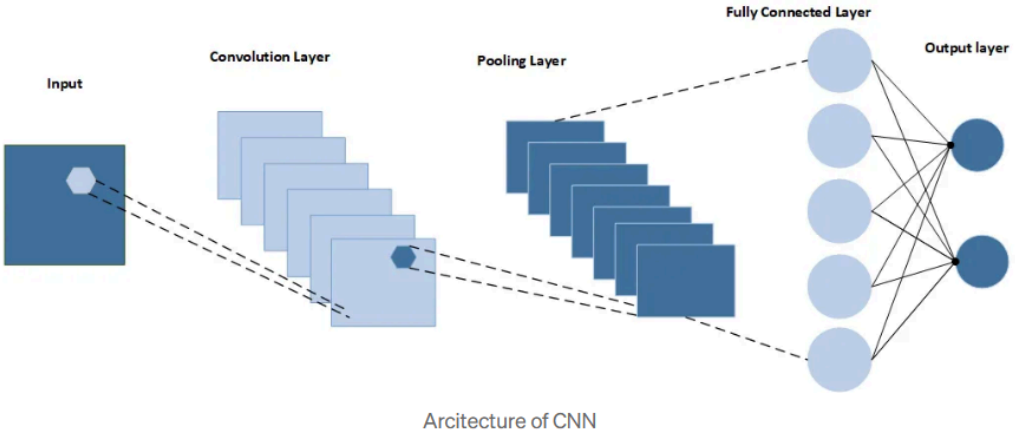


Fig 6:- CNN Architecture(reference)

At the heart of a CNN is the **convolutional layer**. This layer applies filters that slide across the input data (like a tiny window moving across a larger picture). Each filter is trained to detect certain patterns, such as edges, curves, or textures, in the input. The result of this operation is called a feature map, which highlights where those patterns appear. For example, a filter might detect the horizontal stripes in a spectrogram that correspond to repeated notes in a bird call.

Mathematically, this is done using a dot product between the filter and overlapping regions of the input data. If a region matches the filter well, it returns a high value. The deeper the network, the more complex features it can learn.

After convolution, we often use a **pooling layer** (Fig 7) to reduce the size of the data. This makes the model faster and less sensitive to small shifts or noise. The most common type is max pooling, which looks at small patches of the feature map (like 2×2 grids) and keeps only the largest value. This helps the model focus on the most important parts of the signal.

Pooling layers act like a summary, compressing the important information while reducing unnecessary details. They help prevent overfitting and make the model more robust to slight changes in the input.

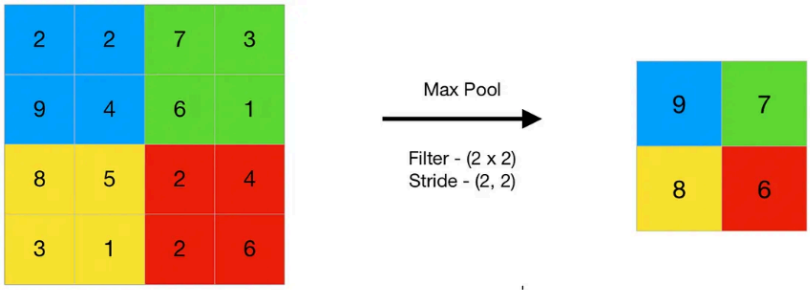


Fig 7:- Pooling Operation(Source)

At the end of a CNN, we flatten the output of the last pooling layer into a one-dimensional vector. This is passed to one or more fully connected layers, standard neural network layers where every input is connected to every output. These layers act as a classifier, combining all the features the earlier layers have extracted and deciding what the final output should be (e.g., which bird species the sound belongs to).

Methodolody

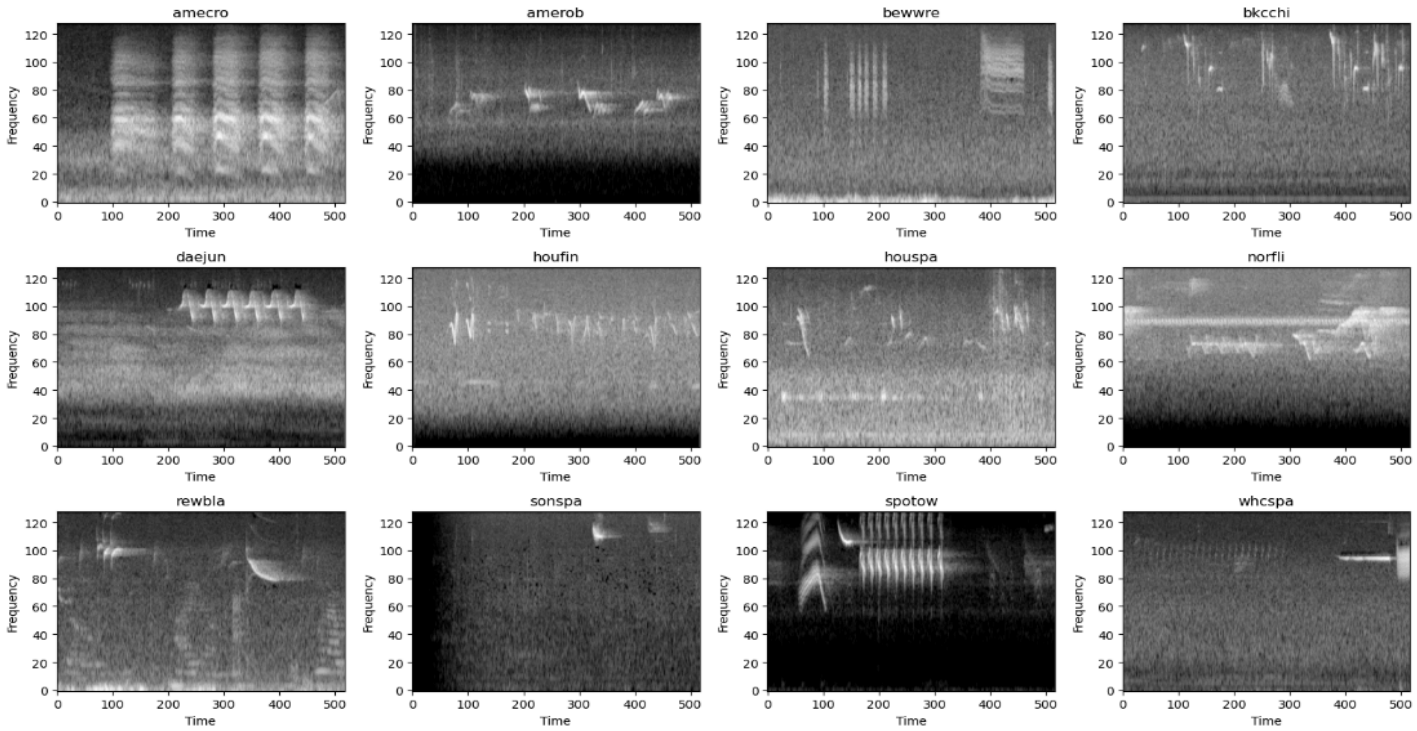
Data Preprocessing

The dataset for this project consists of preprocessed audio spectrograms representing the calls of 12 bird species commonly found in the Seattle area. These spectrograms were extracted from original mp3 clips collected through the Xeno-Canto community and processed into HDF5 format for ease of modeling. Each recording was resampled to 22,050 Hz, and a 2-second segment with audible bird calls was selected. From these segments, mel-spectrograms were generated, producing sample “images” of shape (samples, 128, 517, 1). As shown in Figure 8, each bird species exhibits distinct frequency and time patterns in its spectrogram. However, the number of samples per species varies considerably, creating a class imbalance that required attention during modeling.

Sample Birds Name	Samples per species
American Crow	66
American Robin	172
Bewick’s Wren	144

Sample Birds Name	Samples per species
American Crow	66
American Robin	172
Black-capped Chickadee	45
Dark-eyed Junco	125
House Finch	84
House Sparrow	630
Northern Flicker	37
Red_winged Blackbird	187
Song Sparrow	263
Spotted Towhee	137
White-crowned Sparrow	91

Fig 8:- Sample data Spectrograms



Binary Classification Model: American Crow vs. Spotted Towhee

We choose American Crow and Spotted Towhee , We extracted spectrogram data for only these two species from the full dataset of 12 classes. The spectrograms represent short audio clips as two-dimensional images, Each sample was labeled: 0 for American Crow and 1 for Spotted Towhee. These labels were then converted into a format suitable for neural networks using one-hot encoding, a common preprocessing step for classification tasks.

Preprocessing the data split the data into 80% training and 20% validation sets, ensuring that the relative class proportions were preserved through stratified sampling. This helped prevent any imbalance between species in the training and testing phases.

We then implement our Binary Convolutional model:

- First Convolutional Layer: This layer applied 32 small filters (or kernels) of size 3×3 across the input spectrogram. These filters detect local patterns to avoid overfitting and to keep computational cost low and used ReLU as activation function that is suitable for extracting low-level features from the input shape of 128 x 517 pixels

- Max Pooling Layer: After the first convolution, a max pooling layer was applied. This step reduces the size of the data while preserving the most important features.

- Dropout Layer: To avoid overfitting, we used a dropout layer that randomly disables 20% of the neurons during training.

- Second Convolutional and Pooling Layer: We repeated the same structure, this time with 64 filters, to extract more complex patterns. Another pooling step followed to downsample again.

At this stage, the 2D data was flattened into a 1D vector to prepare it for the final classification layers. and a dense layer a fully connected layer with 64 neurons with sigmoid activation for features extraction then We compiled the model using the Adam optimizer, which automatically adjusts the learning rate during training, helping the model converge efficiently. The loss function used was binary cross-entropy, appropriate for two-class problems. The model was trained for 20 epochs with a batch size of 16 using the training and validation sets. Then we calculated the plots , validation loss and accuracy of the model, along with its confusion matrix.

Multi-Class Classification Model: Identifying 12 Bird Species

After successfully training a binary model for two specific birds, we expanded our task to develop a multi-class classification model capable of identifying all 12 bird species included in the dataset. This more challenging setup required the model to recognize a broader variety of call patterns while dealing with a naturally imbalanced distribution of samples across species.

Data preprocessing include o train the model, we used all available spectrograms and encoded their labels using one-hot encoding, a method that converts each species label into a vector where only one position is set to 1 , and the rest are 0. The dataset was split into 80% training and 20% testing using stratified sampling to maintain consistent class proportions.

Since the number of spectrograms varied widely between species (from as few as 37 to over 630), we calculated class weights to help the model treat each class fairly. These weights assign more importance to underrepresented classes during training, helping mitigate the effects of class imbalance.

Then we implement the Multi-class Model:

- We started by specifying an input layer (128 x 517 x 1). Two Conv2D layers with 32 and 64 filters are employed to extract features, each using a 3x3 kernel size and ReLU activation.
- Max Pooling Layer: With a 2x2 pool size, Reduces the spatial resolution of the spectrograms, allowing the model to focus on dominant features while reducing computational cost.
- Flatten Layer: This layer flattens the extracted feature maps into a 1D vector to feed into dense layers.
- Dense Layer: A fully connected layer with 256 neurons was added, allowing the model to combine features for final classification. A Dropout layer (rate 0.5) , SoftMax activation function was used here to reduce overfitting by randomly deactivating some neurons during training.

The model was compiled using the RMSprop optimizer with a learning rate of 0.001, which works well for deep networks by adapting learning rates during training. We used categorical cross-entropy as the loss function, which is standard for multi-class classification problems.

The model was trained for 20 epochs with a batch size of 64, using both the training and validation data. To handle the class imbalance, we included the previously computed class

weights during training. Then we calculated the validation loss and accuracy of the model, along with its confusion matrix for both the models.

Predictions on External Test Data Using the Multi-Class Model

To evaluate how well our trained multi-class CNN model performs on unseen audio around seattle university campus, we tested it on three test bird audio clips provided. These audio files were not labeled and served as an external validation set. To prepare them for prediction, we applied the same preprocessing steps used during training: loading the clips, extracting audible sections of the waveform, and converting them into mel-spectrograms of consistent dimensions (128×517). Each spectrogram was then passed through our pretrained multi-class model to produce predicted probability scores for all 12 bird species. For each clip, we averaged predictions across all time segments and selected the top three most likely species based on the highest probabilities. Additionally, we checked whether multiple species might be present in a single clip by inspecting if two or more prediction scores exceeded a defined threshold (15.4%). This helped us flag potential overlaps of bird calls in noisy or complex recordings. The final results were compiled into a summary table showing the top predicted species for each clip, their confidence scores, clip duration, and whether multiple birds were likely present.

Results

Binary Classification Model

After evaluated our binary convolutional neural network (CNN) model trained to distinguish between American Crow and Spotted Towhee using spectrograms. After 20 epochs of training, the model achieved a validation accuracy of **87.8%** and a validation loss of **0.3338**, indicating good generalization performance without signs of severe overfitting.

The classification report is summarized in Table 1. The model achieved a high precision of 1.00 for Anerican Crow but a lower recall of 0.62, while Towhee achieved a recall of 1.00 and precision of 0.85. This implies that the model was better at identifying Towhee correctly but occasionally confused Crow calls as Towhee.

Class	Precision	Recall	F1-Score	Support
Anerical Crow	1.00	0.62	0.76	13
Spotted Towhee	0.85	1.00	0.92	28
Accuracy			0.88	41
Macro Avg	0.92	0.81	0.84	41
Weighted Avg	0.90	0.88	0.87	41

Table 1:- Classification Report

Figure 9: Accuracy Performance of Binary Model, the model steadily improved in accuracy, and the validation accuracy closely followed the training trend, suggesting consistent learning and Fig 10: Loss Performance of Binary Model , loss curves further support this, as the validation and training losses both decreased over time.

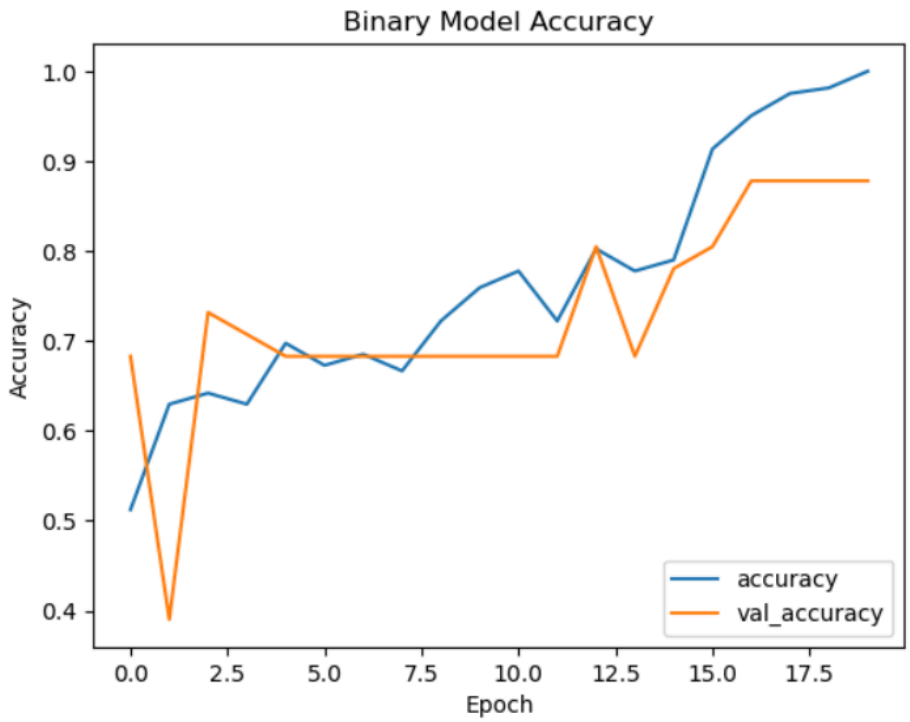


Fig 9:- Accuracy Performance of Binary Model

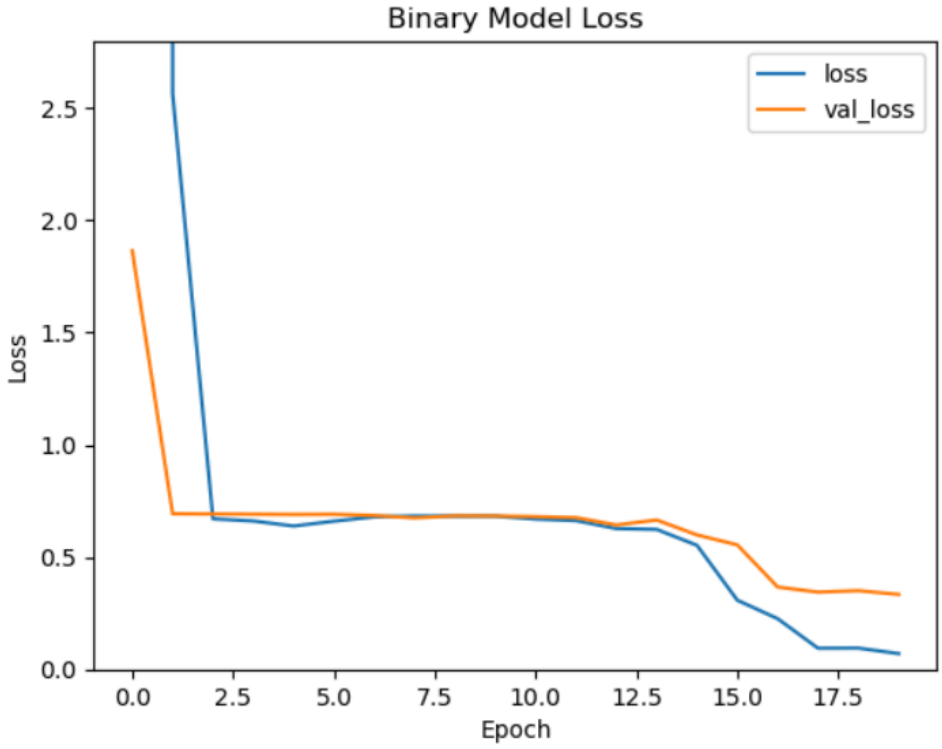


Fig 10:- Loss Performance of Binary Model

The following is the confusion matrix for the binary model to classify the two selected bird species.

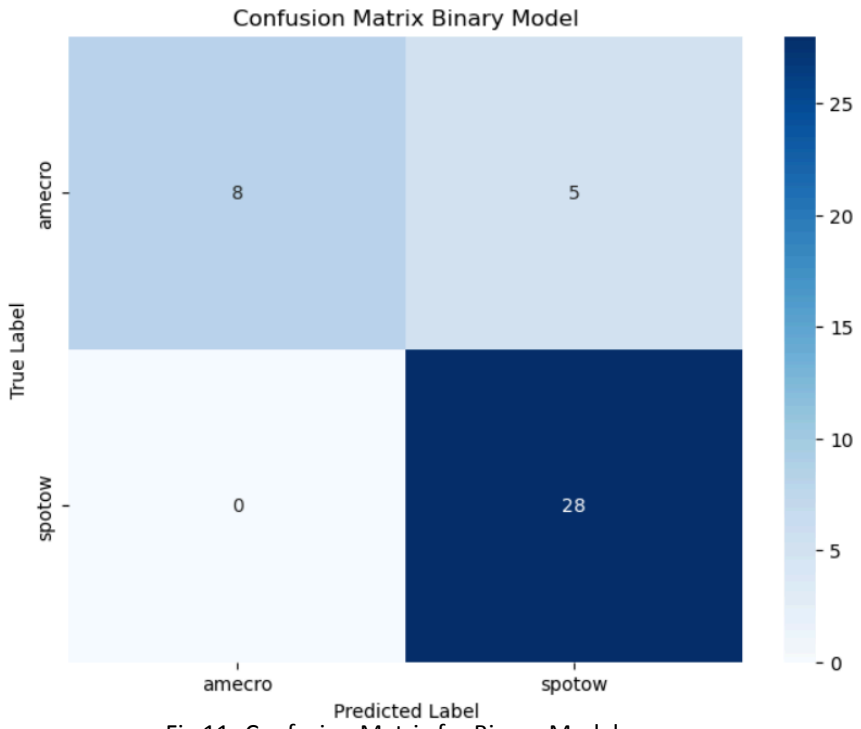


Fig 11:-Confusion Matrix for Binary Model

all 28 samples of Spotted Towhee but misclassified 5 out of 13 American Crow samples. This imbalance in misclassification highlights a minor bias toward the Towhee class, possibly due to greater variability or fewer training samples for Crow.

Finally, we also report the Mean Absolute Error (MAE) of the model's predictions as 0.1220, indicating that on average, the model's predicted class probabilities were close to the true class labels.

Multi-class Classification Model

The final trained model achieved a **validation accuracy of 38.3%** and a **validation loss of 4.43** after 20 training epochs. As shown in Figure 12, training accuracy rose consistently through epochs, peaking above 90%, while the validation accuracy plateaued much earlier, never exceeding 42%. This gap indicates some level of overfitting, where the model performs well on the training data but struggles to generalize to unseen validation data.

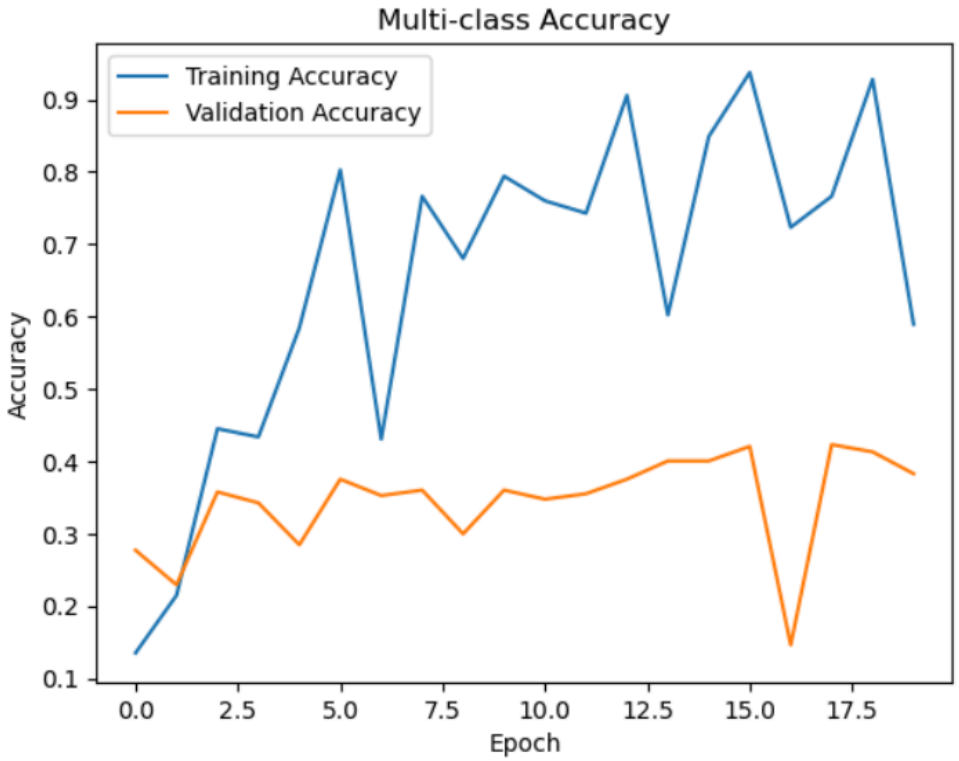


Fig 12:-Accuracy Performance Multi-class Model

The training and validation loss trends (see Figure 13) further confirm this. The training loss fluctuated sharply across epochs due to class imbalance and complex species patterns, while the validation loss remained relatively higher and less stable, even spiking after epoch 10.

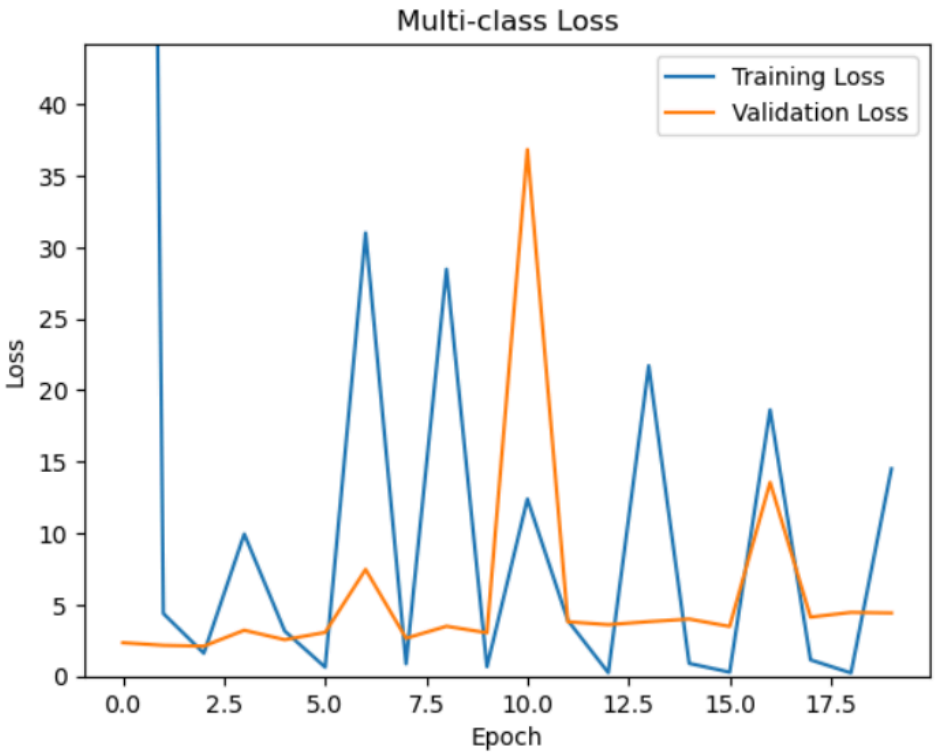


Fig 13:- Loss Performance Multi-class Model

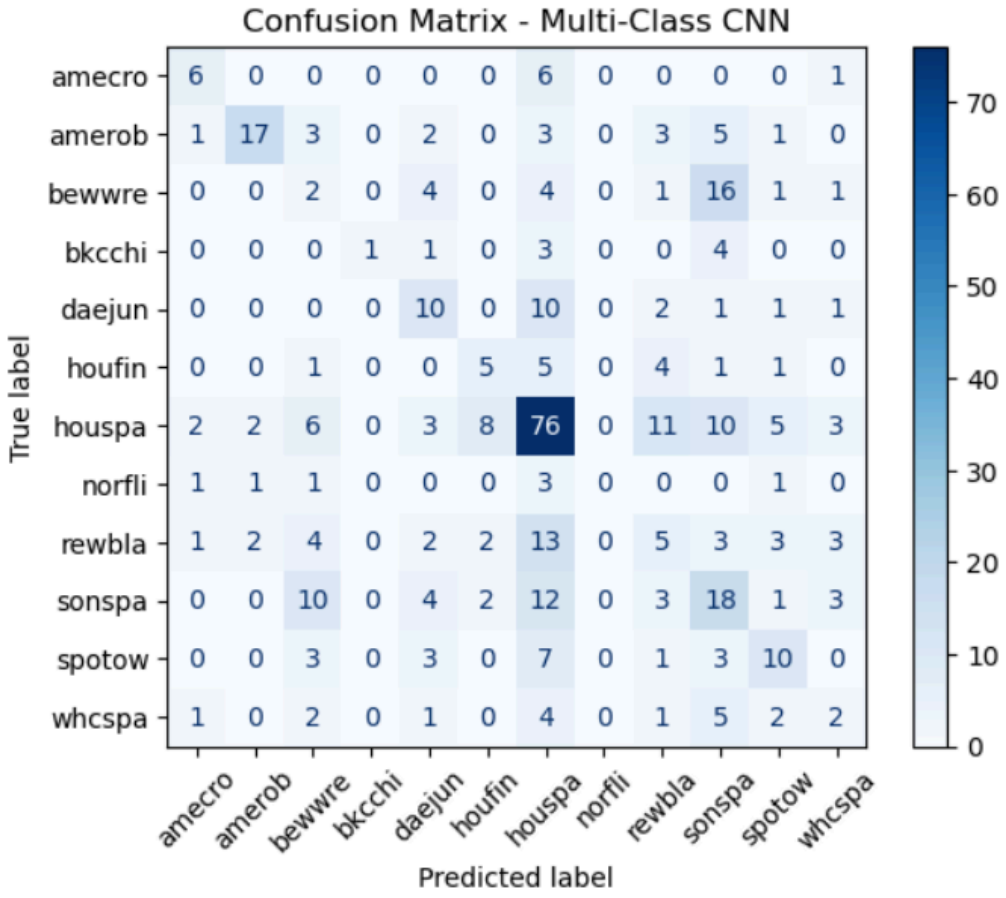


Fig 14:- Confusion Matrix

To better understand where the model was successful and where it struggled, we visualized the confusion matrix in Figure 14 above and generated a detailed classification report.

The confusion matrix shows strong performance in identifying some common species such as House Sparrow (houspa), where the model correctly classified 76 out of 126 instances, and American Robin (amerob), with 17 correct predictions out of 35. However, less frequent or acoustically ambiguous species like Northern Flicker (norfli) and White-crowned Sparrow (whcspa) were poorly classified, with near-zero correct predictions.

Below is a summary table of precision, recall, and F1-score for each species:

Species	Precision	Recall	F1-score	Support
American Crow (amecro)	0.50	0.46	0.48	13
American Robin (amerob)	0.77	0.49	0.60	35
Bewick's Wren (bewwre)	0.06	0.07	0.07	29
Black-capped Chickadee (bkcchi)	1.00	0.11	0.20	9
Dark-eyed Junco (daejun)	0.33	0.40	0.36	25
House Finch (houfin)	0.29	0.29	0.29	17
House Sparrow (houspa)	0.52	0.60	0.56	126
Northern Flicker (norfli)	0.00	0.00	0.00	7

Red-winged Blackbird (rewbla)	0.16	0.13	0.14	38
Song Sparrow (sonspa)	0.27	0.34	0.30	53
Spotted Towhee (spotow)	0.38	0.37	0.38	27
White-crowned Sparrow (whcspa)	0.14	0.11	0.13	18

Table 2:- Classification Report

Overall accuracy: 38.3%

Macro F1-score: ~29%

Weighted F1-score: ~38%

The model had difficulty with species that had fewer examples or whose spectrograms closely resembled others, and this Overfitting is evident due to the large gap between training and validation accuracy Imbalanced class distribution significantly affected model generalization despite using class weights.

Predictions on External Test Clips

After training our final multi-class CNN model, we tested it on three test bird audio recordings.

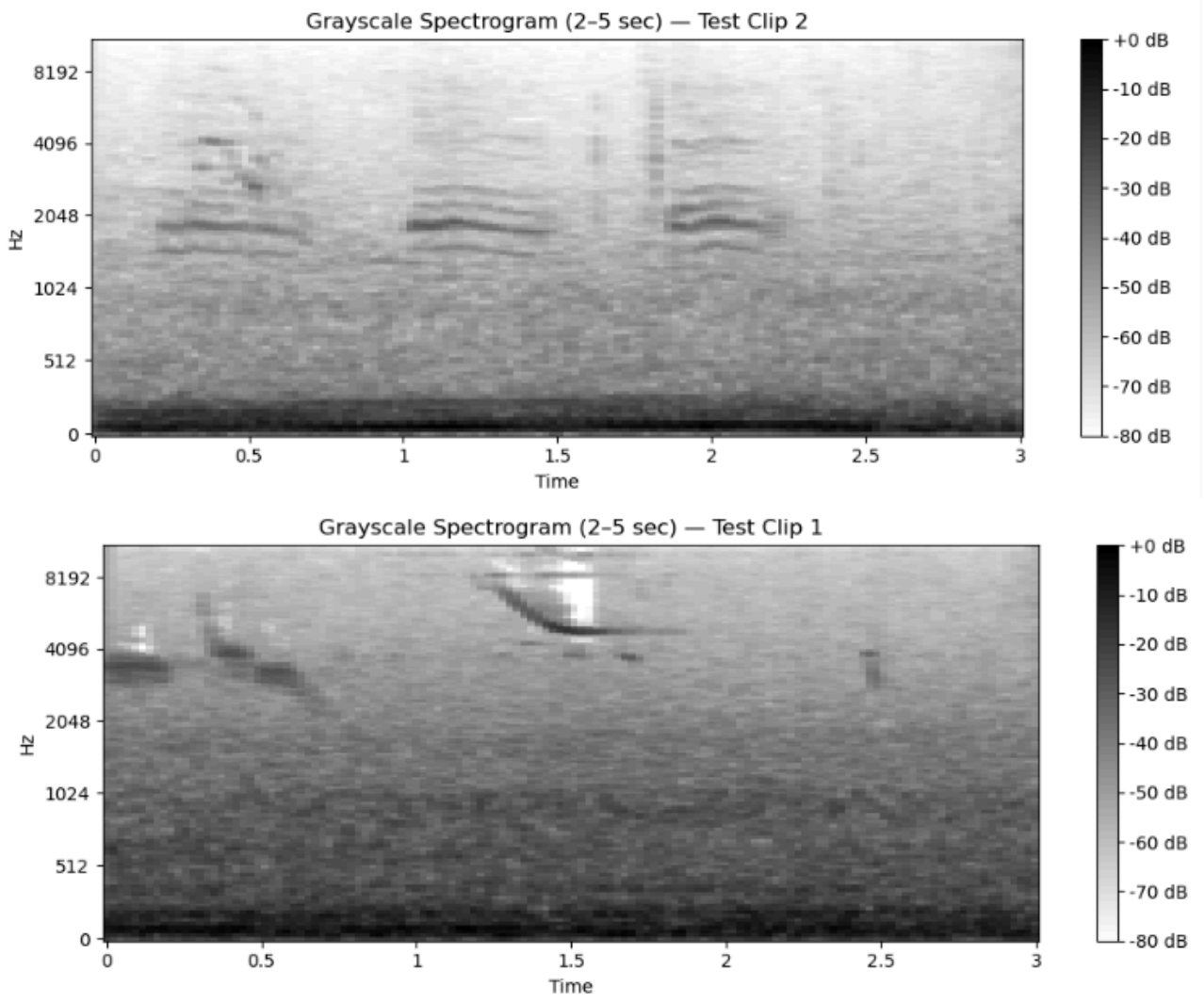
For Test Clip 1, the model predicted Northern Flicker (norfli) as the most likely species, with a confidence score of 19.4%, followed closely by American Crow (amecro) at 15.8% and House Sparrow (houspa) at 12.5%. Since more than one species had a score exceeding 15.4%, this clip is likely to include overlapping calls from multiple birds.

In Test Clip 2, the top predicted species was American Crow at 19.4%, with Northern Flicker and Song Sparrow (sonspa) also appearing in the top three but with slightly lower confidence.

However, only one prediction exceeded the 15.4% threshold, suggesting that only one primary species is present in this clip.

Test Clip 3 again had Northern Flicker as the top species at 21.3%, with American Crow and House Sparrow also predicted at moderate confidence levels. also considered to contain overlapping species sounds.

with Clip 1 and Clip 3 lasting around 12 seconds and Clip 2 being shorter at about 5 seconds. The grayscale spectrograms (Figure 15) show distinctive frequency bands that align with the vocal patterns of the predicted species.



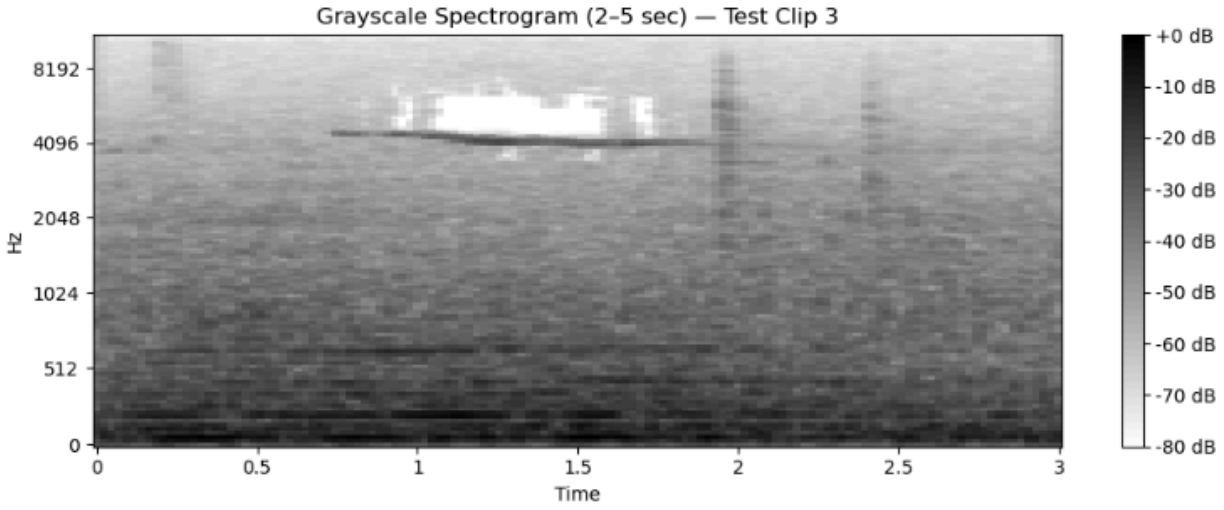


Fig 15:- Test Data Spectrogram

Discussion

Overall, the binary classification model performed strongly, distinguishing between American Crow and Spotted Towhee with 87.8% validation accuracy and a low mean absolute error (0.122). In contrast, the multi-class CNN model, which was designed to identify all 12 bird species, reached a lower validation accuracy of 38.3%. Despite its challenges, this model still managed to learn meaningful acoustic patterns for several species and offered useful predictions when applied to mystery clips.

One key limitation we faced was the computational cost of training models, especially the multi-class network. Training for 20 epochs often took over 28 minutes per run on a standard laptop, compared to under 2-3 minute for the binary model. This reflects the complexity of working with deep networks and large, high-dimensional input data. At times, training processes crashed or ran slowly,. Additionally, class imbalance posed a challenge. Some species had over 150 training samples, while others had fewer than 37, making it harder for the model to generalize fairly across all classes.

The confusion matrix for the multi-class model revealed that certain species were especially difficult to classify. When we reviewed their spectrograms, we noticed overlapping time-frequency structures, many of these birds produce short, high-frequency chirps that look visually similar in grayscale spectrograms.

In our predictions on external test audio, the model identified Northern Flicker and American Crow most confidently, while also occasionally detecting overlap with House Sparrow. The fact that multiple birds were likely present in two out of three clips might be possible,

As an alternative, we could have explored other machine learning models such as Support Vector Machines (SVMs) or Random Forests, particularly by using handcrafted features like average frequency, spectral entropy, However, these models would have required extensive feature engineering and would not scale well with the full spectrogram input. CNNs, by contrast, can learn features automatically and efficiently from raw pixel data, making them particularly suitable for image-like inputs such as spectrograms.

Conclusion

In this project, we developed and evaluated convolutional neural network models to classify bird species based on audio recordings of their calls. Our binary classification model, trained to distinguish between American Crow and Spotted Towhee, achieved strong performance with a validation accuracy of 87.8% and a mean absolute error of 0.122. The multi-class model, designed to identify all 12 bird species, performed more modestly with a validation accuracy of 38.3%. While the multi-class model showed signs of overfitting and struggled to distinguish among acoustically similar species, it was still able to make reasonable predictions on real-world test clips and flagged likely multi-species audio segments effectively. Overall, this study offers insight into how neural networks can aid environmental monitoring and biodiversity conservation efforts.

References

- CNN Architecture : https://www.upgrad.com/blog/basic-cnn-architecture/ and https://medium.com/@samina.amin/understanding-convolutional-neural-networks-cnn-9174c5f16108
- Pooling in CNN : https://medium.com/@abhishekjainindore24/pooling-and-their-types-in-cnn-4a4b8a7a4611
- *An Introduction to Statistical Learning with Applications in Python* (ISLP). Springer. https://hastie.su.domains/ISLP/ISLP_website.pdf.download.html
- Xeno-Canto**.** A community-driven database of bird calls and songs: https://www.xeno-canto.org

- Neural Networks https://www.geeksforgeeks.org/neural-networks-a-beginners-guide/
- Librosa: https://librosa.org/doc/latest/index.html
- Activation Functions in Neural Network: https://www.geeksforgeeks.org/activation-functions-neural-networks/
- Complete Guide to Neural Networks, by Frederik vom Lehn, https://medium.com/advanced-deep-learning/complete-guide-to-neural-networks-7eccbc3bbd80
- What is Deep Learning , https://cloud.google.com/discover/what-is-deep-learning
- Lecture Notes and In class Labs keras.

Appendix

GitHub Repository for Code and Data :
https://github.com/Prateekmax21/Deep-Learning-for-Bird-Sound-Classification