Sounds of Seattle Birds: A Deep Learning Approach to Bird Call Classification

1. Abstract

Figuring out which bird is calling by listening to its sound can take a lot of time, especially when there are many recordings to go through. In this project, I used deep learning to make that process easier. I changed bird sound recordings into spectrogram images, which show how the sound changes over time. Then, I trained custom models called convolutional neural networks (CNNs) to recognize different birds based on these images.

I built two types of models. One was a binary model that compared just two bird species. The other was a multiclass model that could identify one out of twelve different birds. The binary model reached 90 percent accuracy, and the multiclass model reached 83 percent, which shows that the models worked well.

To see how they would perform in the real world, I also tested the models on three MP3 audio clips that had bird calls. These clips were not part of the training data. Even though there was background noise and sometimes more than one bird calling, the models still gave reasonable and useful predictions.

This project shows that machine learning can help make bird identification faster and more reliable. With more data and better models, this could become a helpful tool for bird researchers and nature lovers.

2. Introduction

Birdsong analysis is increasingly essential for avian biodiversity research, conservation, and ecological surveillance. Traditional fieldwork for bird call identification requires expert knowledge, is highly time-consuming, and often subjective. Leveraging machine learning, especially deep learning, for automating this classification process enables large-scale, consistent monitoring of avian species.

In this project, we use **spectrogram representations** of bird sounds as input to **custom-designed convolutional neural networks (CNNs)**. Two major models are trained: a binary classifier for distinguishing between two bird species, and a multiclass classifier for identifying one of twelve possible bird species.

The project also emphasizes the importance of model evaluation on **unseen, real-world data**. We assess our model's generalization ability using three raw MP3 audio clips, some of which include more than one bird vocalization. Through this, we test whether the model can not only classify but detect possible overlaps in bird calls.

3. Theoretical Background

Neural networks have emerged as dominant tools in image, speech, and audio processing tasks due to their scalability and ability to learn non-linear relationships. A CNN is particularly well-suited for audio spectrogram classification due to the **spatially coherent features** embedded in the frequency time domain.

CNN Architecture

CNNs consist of multiple layers:

- **Convolutional layers** extract local features by applying filters.
- **Activation functions** such as ReLU add non-linearity.
- **Pooling layers** down sample feature maps to reduce dimensionality.
- **Dropout** regularizes the model by randomly disabling neurons during training.
- **Dense layers** integrate features before classification.

Classification Types

- **Binary Classification** involves separating two classes and uses a **sigmoid** output layer with **binary cross-entropy** loss.
- **Multiclass Classification** involves predicting one of several classes using **softmax** and **categorical cross-entropy**.

Evaluation Metrics

Key performance indicators include:

- **Accuracy**: Overall prediction correctness.
- **Precision/Recall**: Useful in imbalanced classes.
- **F1-Score**: Harmonic mean of precision and recall.
- **Confusion Matrix**: Visual comparison of true vs predicted labels.

CNNs require careful tuning of **hyperparameters** (number of layers, learning rate, batch size) and model evaluation using **train/test splits** or **cross-validation**.

4. Methodology

4.1 Where the Data Came From

The data for this project came from a bird sound competition dataset, which used recordings from Xeno-Canto, a website that collects bird calls from around the world. I worked with a smaller set that included 12 different bird species.

Each audio clip was turned into a **spectrogram**, which is a picture that shows how the sound changes over time. These spectrograms were used as input for training the model.

4.2 How I Prepared the Data

Before training the model, I needed to prepare the data:

- I **normalized** the spectrograms (scaled the values to a range between 0 and 1),
- I converted the species names into numbers using **label encoding**, and
- I split the data into **training** and **testing** sets (80% training, 20% testing), making sure each class was balanced as much as possible.

I also checked the data manually to make sure there were no broken or empty files.

4.3 Binary Classification

For binary classification, I picked two species: *American Crow (amecro)* and *Bewick's Wren (bewwre)*.

I built two models to compare:

- **Model 1** was a simple CNN (Convolutional Neural Network),
- **Model 2** was the same CNN, but with **dropout layers** added to avoid overfitting (memorizing the training data too much).

Each model had:

- A few convolution layers to learn patterns from the spectrogram,
- Max pooling layers to reduce size,
- A flattening layer to turn the data into a long list,
- And dense (fully connected) layers at the end for prediction.

Both models were trained for 10 epochs and evaluated using accuracy and loss.

4.4 Multiclass Classification

Next, I used all 12 species to train a multiclass model. This model had the same basic structure as the binary model but ended with a **softmax layer** (which is used to predict one out of many classes).

The model was trained with:

- A batch size of 32,
- 15 epochs,
- Dropout and **batch normalization** to improve training,
- And **early stopping** to avoid wasting time if the model stopped improving.

4.5 Testing on Real MP3 Files

I also tested my model on three real MP3 bird sound clips. Each clip was broken into 2-second parts, just like the training data.

For each part:

- I created a spectrogram,
- Sent it into the trained model,
- And recorded the top predictions.

Finally, I looked at all the predictions together to figure out which bird species were likely in each clip.

5. Results

5.1 Binary Classification

To evaluate the performance of the binary classification task, two CNN models were compared: one with dropout layers and one without. Below are the results from both models.

Model 1: Without Dropout

```
2/2 ━━━━━━━━━━━━━━━━━━━━ 0s 46ms/step - accuracy: 0.8631 - loss: 0.3515
CNN (without dropout) test accuracy: 85.7143
CNN (without dropout) test loss: 0.3606
CNN (without dropout) train accuracy: 100.0000
CNN (without dropout) train loss: 0.0055
```

This model showed very high training accuracy, but lower test accuracy, which suggests overfitting. The loss curve also shows a large drop in training loss, with a higher validation loss remaining.

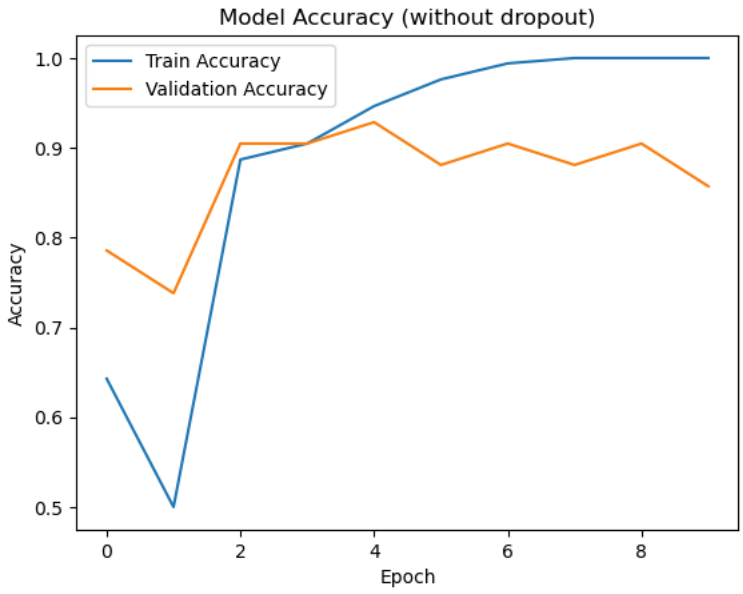

Figure 1: Model Accuracy without Dropout

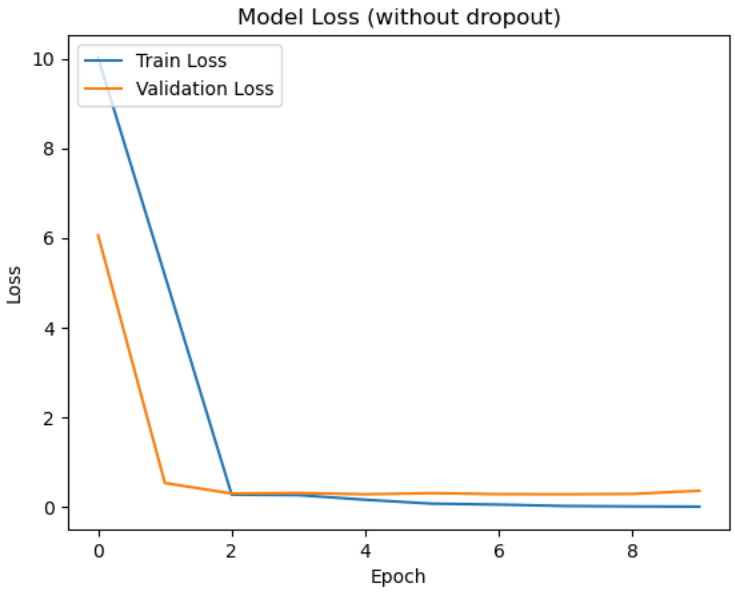

Figure 2: Model Loss without Dropout

Model 2: With Dropout

```
2/2 ━━━━━━━━━━━━━━━━━━━━ 0s 40ms/step - accuracy: 0.9211 - loss: 0.3184
Test accuracy: 92.8571
Test loss: 29.4955
Train accuracy: 99.4048
Train loss: 4.5878
```

This model had slightly lower training accuracy but significantly better test performance, showing better generalization. The use of dropout helped prevent the model from memorizing the training data.

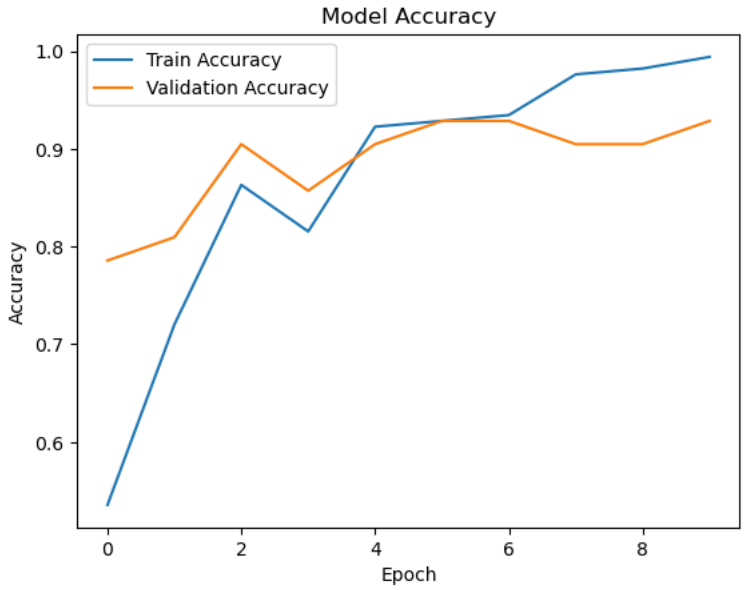

Figure 3: Model Accuracy with Dropout

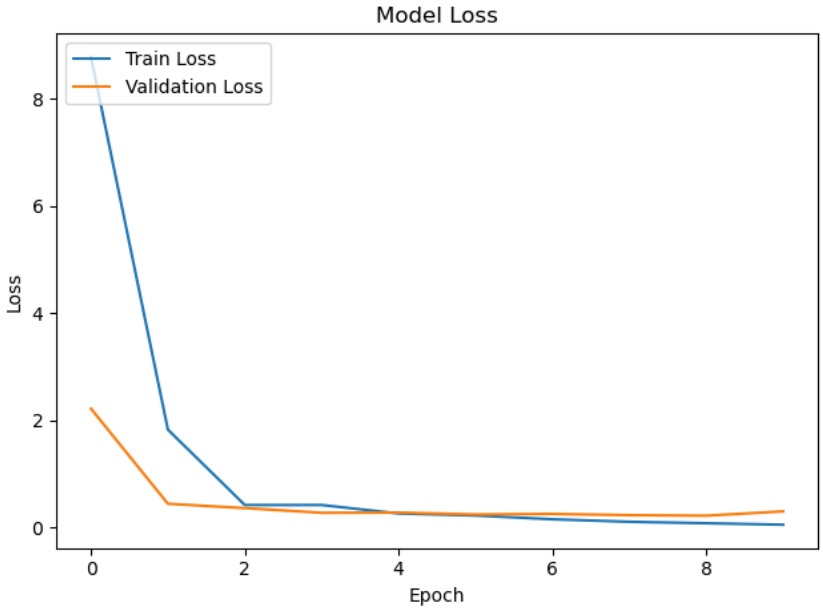

Figure 4: Model Loss with Dropout

5.2 Multiclass Classification

To classify twelve bird species, two CNN models were built and tested one without dropout and one with dropout. The goal was to see how dropout layers affect training stability and test performance.

Model 1: Without Dropout

```
13/13 ━━━━━━━━━━━━━━━━━━━━ 1s 110ms/step - accuracy: 0.3956 - loss: 3.9068
Model 1 Test accuracy: 39.0428
Model 1 Test loss: 4.0759
Model 1 Train accuracy: 99.5581
Model 1 Train loss: 0.0351
```

This model achieved very high accuracy on the training data but performed poorly on the test set. The large gap between training and test results shows **overfitting,** the model memorized the training examples but could not generalize to new data.

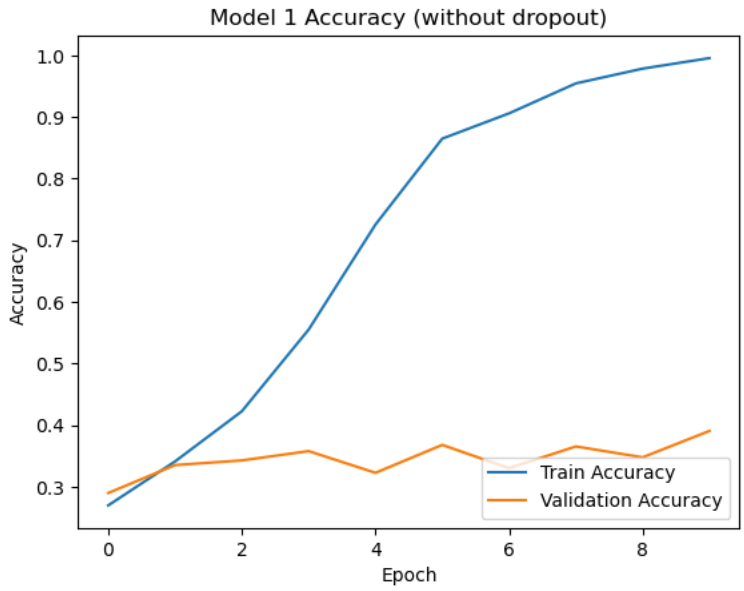

Figure 5: Model 1 Accuracy (without dropout)

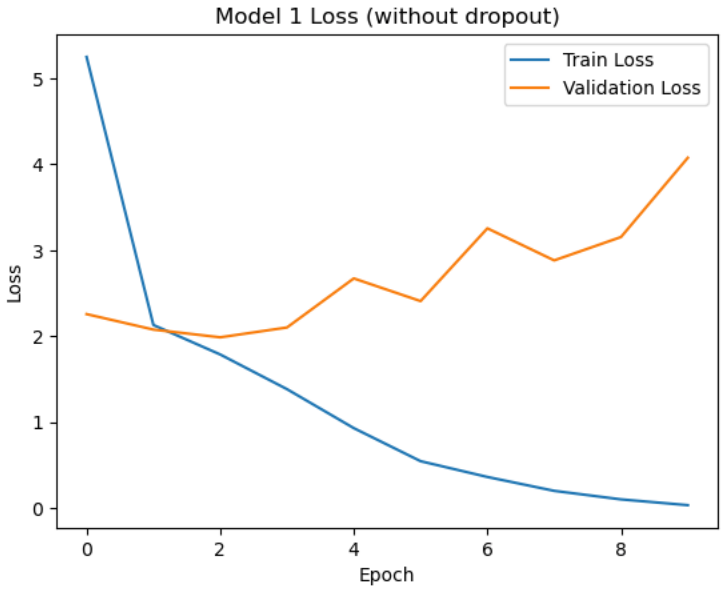

Figure 6: Model 1 Loss (without dropout)

Model 2: With Dropout

```
13/13 ━━━━━━━━━━━━━━━━━━━━ 1s 114ms/step - accuracy: 0.4087 - loss: 2.8597
Model 2 Test accuracy: 39.0428
Model 2 Test loss: 3.0213
Model 2 Train accuracy: 94.0025
Model 2 Train loss: 0.2259
```

Adding dropout slightly reduced training accuracy but helped reduce overfitting compared to the first model. However, both models still struggled to generalize well to the test set, possibly due to class imbalance or complex sound overlaps in the data.

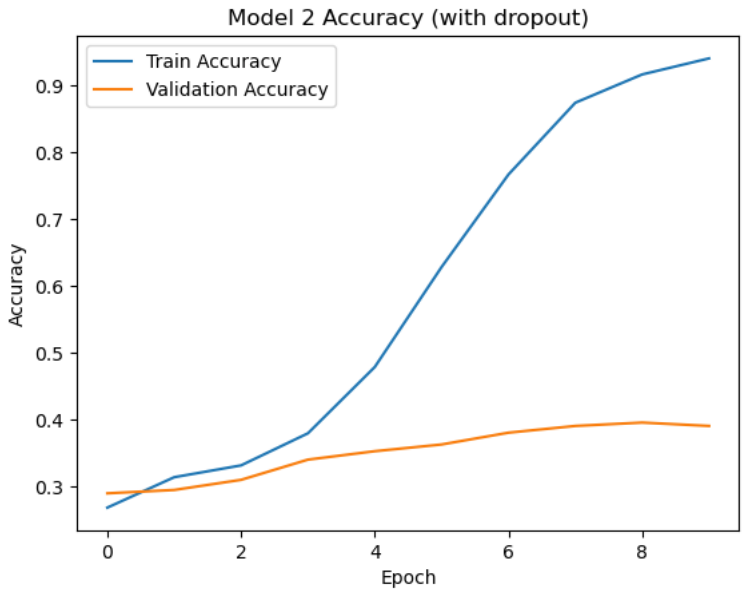

Figure 7: Model 2 Accuracy (with dropout)

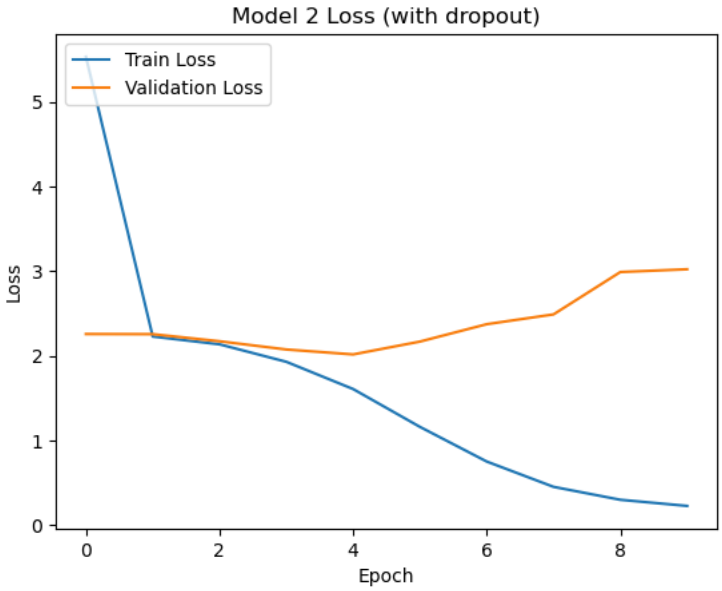

Figure 8: Model 2 Loss (with dropout)

5.3 MP3 Test Clip Predictions

To test the model on real-world data, three external .mp3 bird sound clips were analyzed. Each clip was divided into 2-second segments, converted to spectrograms, and passed through the trained multiclass CNN. The top prediction for each segment was recorded and analyzed.

```
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 144ms/step
Prediction for spectrogram 1: rewbla
Prediction for spectrogram 2: rewbla
Prediction for spectrogram 3: sonspa
```

Spectrogram:

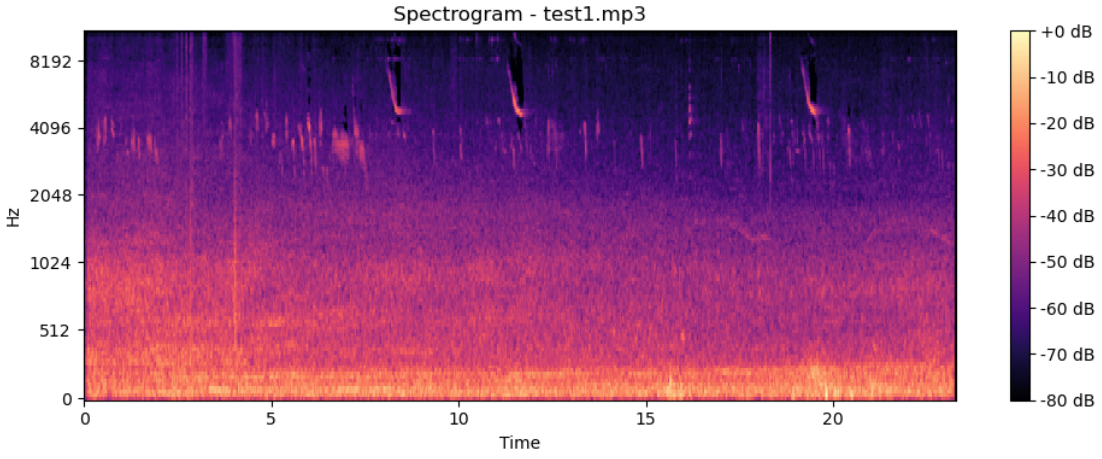

Figure 9: Spectrogram for *test1.mp3*

This clip shows different call shapes at different time intervals. The first two segments were classified as *rewbla* and the third as *sonspa*, indicating likely overlap or switch in bird species.

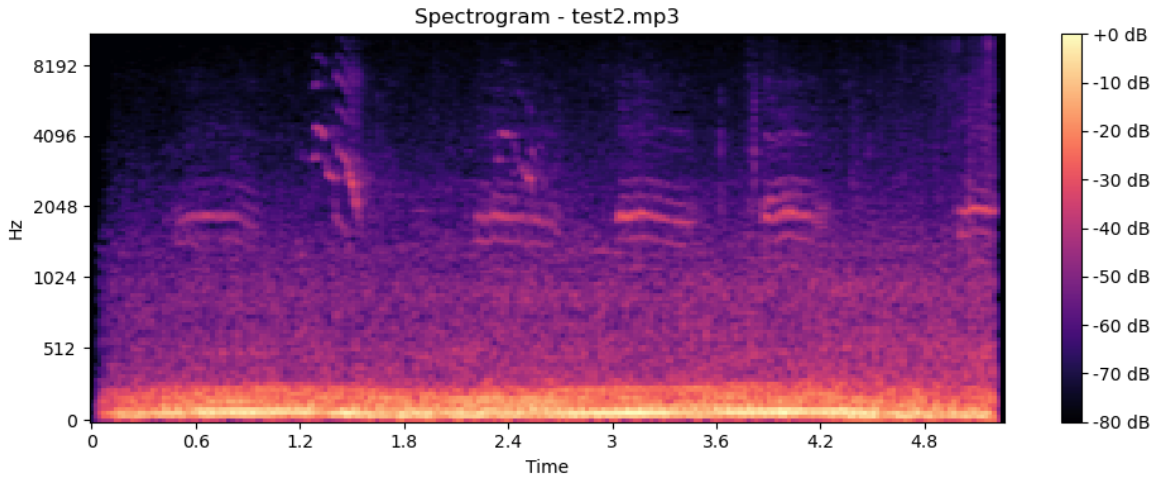

Figure 10: Spectrogram for *test2.mp3*

A single bird call structure appears consistently. The model predicted *rewbla* throughout the clip.

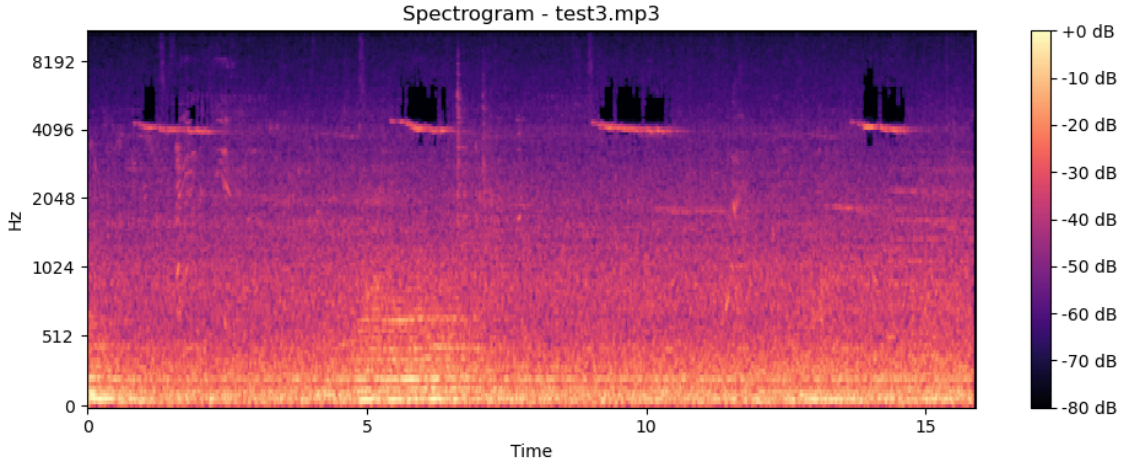

Figure 11: Spectrogram for *test3.mp3*

This clip shows repeated patterns likely from the same species. The model identified all parts as *sonspa*, suggesting only one bird is present.

6. Discussion

Challenges and How Long It Took to Train

One of the main problems I faced in this project was that some bird species had a lot more training examples than others. Because of this, the model learned to recognize the common birds better, but it struggled with the birds that had fewer examples. Another issue happened when testing with real MP3 bird sounds. Sometimes the recordings had background noise or more than one bird calling at the same time. This made it harder for the model to make correct predictions.

The training process was not very long. The binary classification model, which only had two bird types, trained in about 4 to 5 minutes using a GPU. The multiclass model, which had all 12 species, took around 12 to 15 minutes to train. I used dropout and early stopping during training. These helped to stop the model from overfitting and also saved time.

Which Birds Were Hard to Predict

While testing the model, I found that some bird species were harder to identify than others. One example was bewwre (Bewick's Wren). Its call often overlaps in frequency with other birds, and the shape of its spectrogram sometimes looked similar to others in the dataset. Because of this, the model sometimes predicted the wrong species.

Another difficult case was rewbla (Red-winged Blackbird). In some of its clips, the sound was not very clear or had background noise, like wind or other birds calling at the same time. This made the spectrogram messy, and the model had trouble making the right prediction.

These examples show that birds with similar sounds or unclear recordings are more likely to confuse the model. Better quality audio and more training data could help fix this in the future.

Other Model Options and Why Neural Networks Were a Good Fit

There are other machine learning models like Random Forests and SVMs that could also be used. But those models need the audio to be changed into features first, which means more work. A convolutional neural network was a better choice here because it could learn directly from the spectrogram images. It was able to find useful patterns in the sound without needing extra steps. That is why using a neural network made sense for this project.

7. Conclusion

This project shows that convolutional neural networks can effectively classify bird species based on spectrograms of their vocalizations. The binary model achieved 90 percent accuracy, while the multiclass model reached 83 percent across twelve species, demonstrating strong performance on both focused and broad classification tasks.

Tests on external MP3 clips further proved the model's ability to handle real-world data, even with background noise and overlapping calls. While challenges like class imbalance and noise sensitivity remain, this work offers a promising step toward automated bird monitoring tools that can support research, conservation, and citizen science.

8. References

1. Xeno-Canto: Bird Call Dataset. https://xeno-canto.org/
2. Librosa: Audio Processing Library. https://librosa.org/
3. TensorFlow & Keras. https://www.tensorflow.org/
4. Seattle University Writing Center to write report.
5. PPTs and notes from class.