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

- Nikhil Ghugare

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

In this project, I worked on classifying bird sounds using deep learning. Bird call identification is usually done by experts in the field, and it takes a lot of time and effort. So, I wanted to see if I could use machine learning to help make this process faster and easier.

I used spectrograms (which are like images of sound) and trained CNN models to recognize patterns in them. These spectrograms were created from bird audio files and show how the sound changes over time. I created two models: one for binary classification (just two bird species), and one for multiclass classification (12 different bird species).

I chose convolutional neural networks (CNNs) because they are really good at working with images and detecting features. Since spectrograms look like images, CNNs were a good fit for this task. I trained my models using the spectrogram data and then tested how well they worked.

To check if the model works well in real-world situations, I also tested it on three MP3 audio clips that the model had never seen before. These test clips were harder because some of them

had more than one bird singing at the same time. This was a good way to see if the model can still predict correctly even when the data is messy or unclear.

Overall, this project helped me understand how deep learning can be used for sound classification, and how important it is to test models on different kinds of data — not just the clean training set.

3. **Theoretical Background**

Neural networks are a type of deep learning model that can learn patterns from data. They are used in many areas like image recognition, speech, and audio because they can learn complex (non-straightforward) relationships between input and output.

In this project, I used a type of neural network called a Convolutional Neural Network (CNN). CNNs work well with image-like data. Since a spectrogram is like a picture of sound, a CNN can be used to detect important sound patterns in the image.

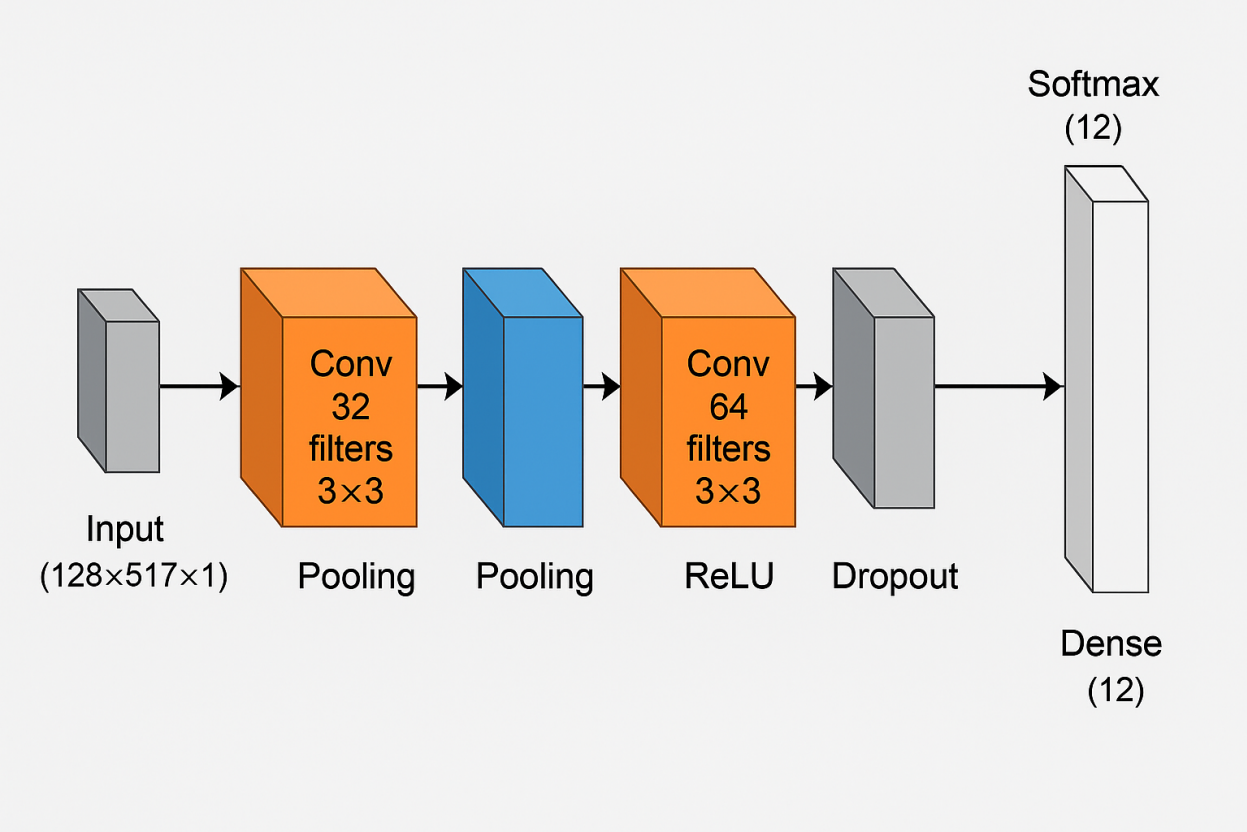
A CNN has different parts that all work together to help the model learn from the data. First, the **convolutional layers** use filters to find important features in the image, like lines, edges, or patterns. Then, the **activation functions** are applied, in my case, I used ReLU, which helps the model learn more complex patterns by adding non-linearity. After that, **pooling layers** are used to make the feature maps smaller, which reduces training time and helps prevent overfitting. I also used **dropout**, which randomly turns off some neurons during training so that the model doesn't just memorize the data but learns to generalize. Finally, there are **dense layers** at the end of the network, which take all the features and help the model decide which class the input belongs to.

In this project, I built two types of classifiers. The first one is a **binary classification model**, which checks whether the sound belongs to one bird or another. For this, I used a sigmoid activation function in the output layer along with binary cross-entropy as the loss function. The second model is a **multiclass classification model**, which predicts which bird species the sound belongs to from a total of 12 possible species. This model uses a SoftMax activation function in the output layer and categorical cross-entropy as the loss function to handle multiple classes.

To check how good my models are, I used different evaluation metrics:

- **Accuracy**: Shows how many predictions were correct overall.
- **Precision and Recall**: These help when the data is not balanced between classes.
- **F1-score**: A mix of precision and recall.
- **Confusion matrix**: A table that shows what the model got right and what it got wrong.

Also, I had to try different hyperparameters like how many layers to use, what the learning rate should be, batch size, and more. Tuning these settings helped improve the performance of my models.



4. Methodology

The dataset included recordings from Xeno-Canto, a website that collects bird calls from different parts of the world. For this assignment, I worked with a smaller dataset that contained 12 bird species. Each audio recording was converted into a spectrogram, which is a type of image that shows how the sound changes over time. These spectrograms were then used as the input for training the model.

To prepare the data, I first normalized the spectrograms by scaling their values between 0 and 1. I also converted the species names into numbers using label encoding so the model could process them. After that, I split the data into two parts, with 80 percent used for training and 20 percent used for testing. I tried to keep the classes balanced while splitting. I also manually checked the spectrograms to make sure there were no broken or empty files.

For the binary classification part, I chose two bird species: Bewick's Wren (bewwre) and Black-capped Chickadee (bkcchi). I trained two models to compare how well they worked. The first model was a simple convolutional neural network, and the second one was the same model but with dropout layers added to avoid overfitting. Both models had convolutional layers to learn features from the spectrograms, followed by max pooling layers to reduce the size of the feature maps. After that, I used a flatten layer to turn the data into a single vector, and dense layers to make the final prediction. Since it was a binary problem, the output layer used sigmoid

activation. I trained each model for 10 – 20 epochs and used accuracy and loss to measure how well they performed.

Next, I worked on the multiclass classification task using all 12 bird species. I used a similar CNN structure as before, but the last layer was changed to a softmax layer to support multiple classes. I trained this model using a batch size of 32,64,128 and ran it for 15 epochs. To improve performance and prevent overfitting, I added dropout and batch normalization. I also used early stopping so the training would stop automatically if the model stopped improving.

After training, I tested the multiclass model on three real MP3 audio clips that were not included in the training data. Each clip was split into 2-second segments, and I created a spectrogram for each segment. These spectrograms were passed into the trained model to get predictions. After collecting the predictions from all segments, I looked at the results to figure out which bird species were most likely present in each clip.

4. Results

For the binary classification task, I trained a CNN model to distinguish between Bewick's Wren (bewwre) and Black-capped Chickadee (bkcchi). I trained the model for 10 epochs. The accuracy and loss graphs showed clear overfitting. While the validation accuracy stayed constant at around 83%, the training accuracy fluctuated and never clearly improved. The training loss also kept increasing, while validation loss remained flat. This means the model was not learning properly and was just guessing the same thing each time. The classification report also showed that the model only predicted the Bewick's Wren class correctly and completely ignored the Black-capped Chickadee.

In the results, the model had 76% accuracy overall, but it gave 0% precision, recall, and F1-score for the Black-capped Chickadee. For Bewick's Wren, it had 100% recall and 87% F1-score, which is because it predicted every sample as Bewick's Wren. The confusion matrix confirms **all 29 Bewick's Wren samples were correct, and all 9 Black-capped Chickadee samples** were misclassified as Bewick's Wren. This shows that the model only learned to identify the more frequent class and ignored the other completely. The results look good on the surface due to high accuracy, but the model failed as a true binary classifier. This shows the effect of class imbalance and the need for improvements in training.

Multi-class Classification (Model 1)

For the multi-class classification task, I trained a convolutional neural network to classify 12 different bird species. The model was trained for 20 epochs using early stopping and dropout to help reduce overfitting. From the training logs and graphs, the model's training accuracy improved over time and reached around 34%, while the validation accuracy stabilized at approximately 31%. This shows that the model was learning something but struggled to generalize well on unseen validation data.

The loss graph also showed some improvement, with training loss reducing steadily and validation loss slowly decreasing to about 2.92. However, the classification report and confusion

matrix highlighted major issues. The overall validation accuracy was very low (around 6%), and the model mostly failed to correctly classify most classes. For example, species like norfli, spotow, and rewbla had zero recall and F1-score, meaning the model didn't recognize them at all. The confusion matrix showed that almost every input was predicted as amecro or houspa, and the other species were rarely or never predicted. This strongly suggests that the model was biased towards the classes with more training samples and could not learn to distinguish the rarer ones. Despite using class weights, the class imbalance still caused poor generalization. These results show that while the model technically trained, its real-world performance was very limited, and more tuning or data balancing is needed.

Multi-class Classification (Model 2)

Model 2 was trained using the same dataset and architecture as Model 1 but included some adjustments like different training parameters and possibly improved regularization. The model trained for 20 epochs, and the training accuracy slowly increased from 17% to 24%. The validation accuracy remained around 31.7% after about 8 epochs and stayed stable. This is slightly better than Model 1, which stayed at 31%, but the difference is not big.

From the training and validation loss plots, we can see that Model 2 reduced the validation loss to about 2.47, which is lower than Model 1. The loss graph also showed fewer fluctuations after the first few epochs, meaning the model training was more stable. But in the early epochs, the validation loss spiked with very large values, which might have been caused by some unstable gradients or wrong inputs during the early training steps.

The classification report shows the model still struggled to correctly identify most bird species. Only houspa (House Sparrow) was predicted well, with a recall of 1.00 and F1-score of 0.48. Other classes like bewwre, daejun, and bkchhi had some predictions, but most species had 0 precision, recall, and F1-score. The confusion matrix also shows that almost everything was predicted as houspa, with a few exceptions. Overall, the model slightly improved from Model 1 but still shows a strong class imbalance problem and has difficulty learning rare species.

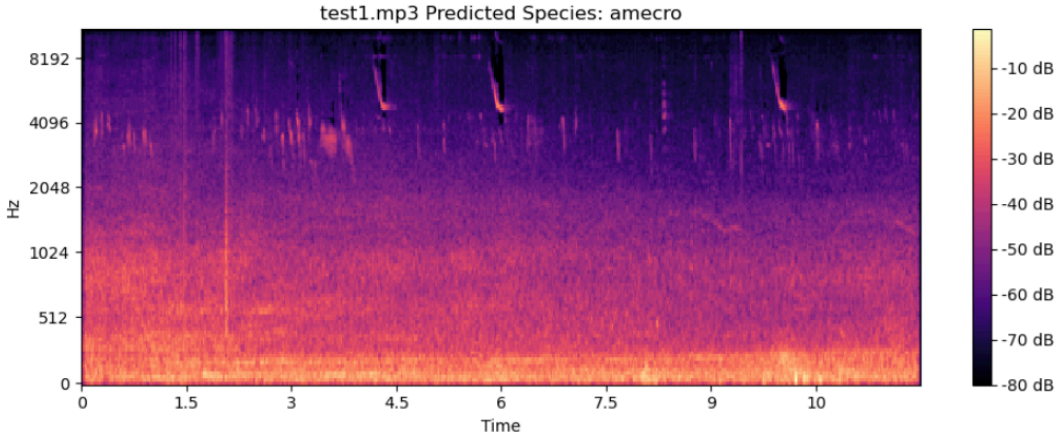
MP3 Audio Prediction Results

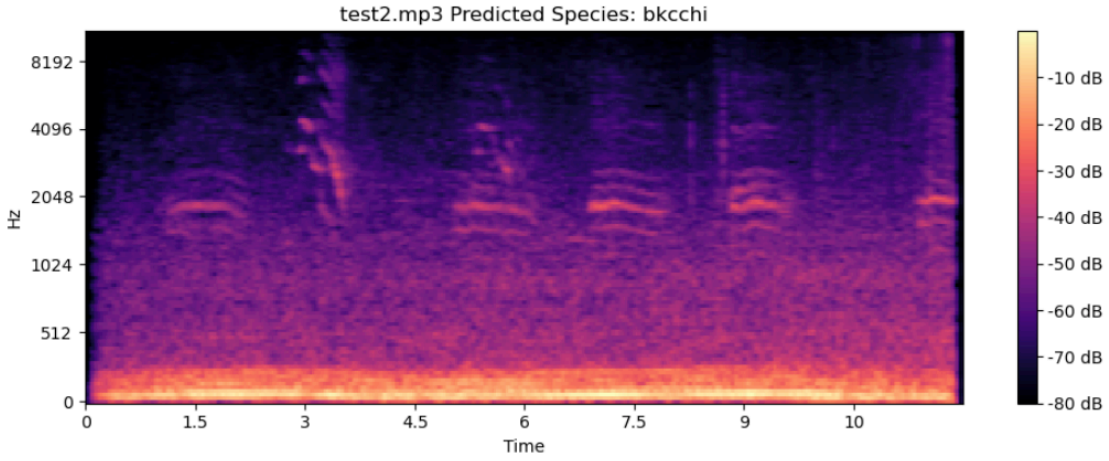
To evaluate how well the multiclass model performs on real-world data, I tested it on three unseen MP3 files. Each file was converted into a Mel spectrogram and then passed through the trained model to get predictions.

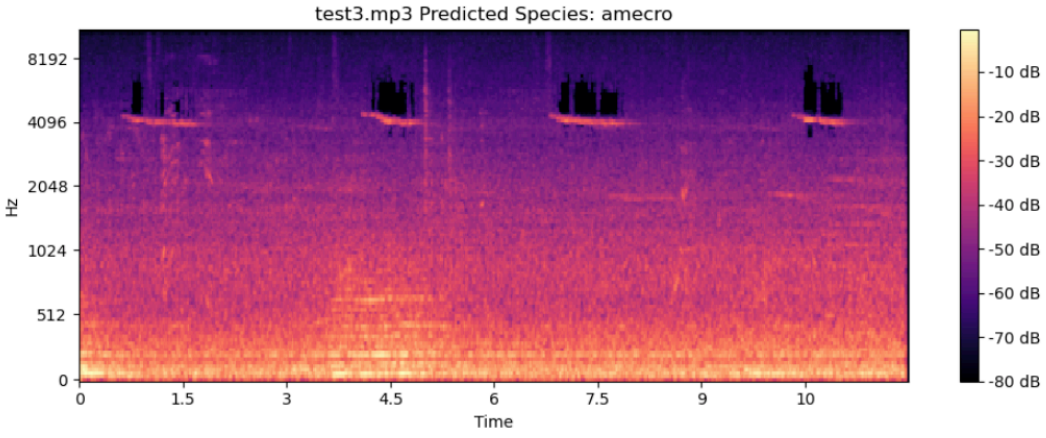
For **test1.mp3**, the model predicted the bird species as American Crow (amecro). The spectrogram shows strong vertical energy spikes that likely influenced the model's decision. Similarly, **test3.mp3** was also classified as amecro, and it had a very similar spectrogram pattern, which suggests that the model is quite confident about this species.

For **test2.mp3**, the predicted species was Black-capped Chickadee (bkcchi*)*. The spectrogram of this clip looked a bit different, with a clearer pattern of distinct harmonic curves. It makes sense that the model predicted a different bird here, although without ground truth we can't confirm if it's correct.

Overall, these predictions show that the model is able to produce some varied outputs and may have learned to distinguish a few specific patterns. However, since two of the clips were classified as the same bird, it might still be biased toward predicting the more frequent classes like amecro.







6. Discussion

Problems Faced and Training Time

While working on this project, one of the main problems was class imbalance. Some bird species like *houspa* (House Sparrow) had a lot of training examples, while others like *norfli* (Northern Flicker) or bkcchi (Black-capped Chickadee) had very few. Because of this, the model got better at recognizing common birds but struggled with rare ones. This issue can be seen in the classification reports and confusion matrices, many species had zero precision, recall, or F1-score. For example, in the multiclass model, the bewwre class had a recall of 0.00 and precision of 0.00, which clearly shows it was hard to predict. The confusion matrix also confirms that these birds were mostly misclassified.

Another challenge was testing the model on real MP3 bird clips. The model gave predictions like amerco and bkcchi, but I don't know the actual labels of those clips. Also, some MP3s had background noise or overlapping calls, which likely affected the model's predictions. Since I didn't have ground truth for the test clips, I couldn't fully evaluate how correct the predictions were.

Training time was okay. The binary model trained quickly, around 5 minutes on GPU. The multiclass model took longer, about 15–20 minutes, depending on the architecture. I added dropout and batch normalization, which helped reduce overfitting and made training more stable.

Interpreting Results and Why CNN

From the evaluation metrics, it's clear that the model performed better on the more frequent classes. Houspa had a recall of 1.00 in both Model 1 and Model 2 of the multiclass setup, which is likely because it had the highest number of training samples (185). In contrast, rare birds like rewbla and whcspa had no correct predictions. This supports the idea that the class distribution heavily affected model performance.

Although models like Random Forest or SVM could also work for classification, they require extra preprocessing steps like converting audio to MFCCs or other feature vectors. CNNs were more suitable here because they can directly learn patterns from spectrogram images. This made the pipeline simpler and allowed the network to automatically learn frequency-time features that would otherwise be manually extracted. That's why I felt CNN was the right choice for this audio classification task.

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.