Bird Audio Classification

Authors: Emily Duniec, Shimon Dasgupta

Introduction

Identifying species of birds by their song is a unique challenge that scientists face when they are attempting to monitor bird populations. In this project, we sought to classify the songs made by different species of birds. Bird songs and sounds in general are made up of many different features such as the hertz of the song and its amplitude. You can also visualize the waves of a song through spectrograms, chronograms, and its chroma features. Convolutional neural networks (CNNs) are designed to automatically and adaptively learn spatial hierarchies of features through their layers. In the context of bird songs, CNNs can learn to recognize various patterns in audio that correspond to different bird calls without manual feature engineering. We want to test and see if a CNN can learn to classify birds based on the chroma features and chronogram of their songs.

Methodology

Online dataset - https://www.kaggle.com/datasets/vinayshanbhag/bird-song-data-set

Our dataset consisted of over 5000 recordings of bird songs from 5 different bird species. The length of each audio clip is around 3 seconds. We chose this dataset because it had enough entries to provide accurate results but wasn't too large where our models would take too long to train. The audio quality is also very good and has minimal background noise that could confuse the model.

Extraction:
We used TensorFlow and Keras for model construction and training. The python audio library Librosa was employed for audio signal processing and feature extraction.

Chroma Features:
Chroma features were extracted to represent the harmonic content of the audio signals. The recordings were resampled to 16 kHz to reduce computational demands.

Hop Length: A hop length of 256 samples was chosen to achieve a finer temporal resolution, which is important for capturing the rapid changes in bird songs.

FFT Window: A 1024-sample FFT window was used to compute the short-time Fourier transform. This window size was selected to balance between frequency resolution and time resolution.

Feature Calculation: librosa.feature.chroma_cqt was utilized for chroma feature calculation because of its ability to provide a more natural representation of harmonic content at various tempos.

Kaggle online model - https://www.kaggle.com/code/collinpfeifer/5-bird-audio-classifier-model/input

We used the kaggle dataset as a reference for working with librosa. The actual model they trained was very bad and overfitted so we created our own model using chroma features and chronograms.

Model Architecture

The first model used mel-spectrograms to classify different audio. The process began by splitting the training data into training and test sets using `train_test_split`, with 25% of the data reserved for testing. The class labels were then converted to a one-hot encoded format using `to_categorical` from Keras, based on the 5 species included in the data. The mel-spectrogram arrays were normalized using TensorFlow's `normalize` utility to help in model training by scaling the input features. These arrays were then wrapped into TensorFlow datasets, which are batched and prefetched to optimize loading times during training.

The first model architecture consists of several layers designed to extract features from the input spectrograms. The input layer was configured to accept an input shape of 128x130, which corresponds to the dimensions of the mel-spectrogram. This was followed by a reshape layer to ensure the input is correctly formatted for the convolutional layers. The first convolutional layer had 64 filters of size 8x8 with ReLU activation, followed by batch normalization and a max pooling layer to reduce spatial dimensions. A second convolutional layer with 16 filters of size 2x2 also used ReLU activation. After convolutional layers, the data was flattened into a one-dimensional array and passed through a dropout layer with a rate of 0.5 to mitigate overfitting. The network included two dense layers; the first has 128 neurons with ReLU activation, and the final output layer has 5 neurons corresponding to the number of classes, using softmax activation for multi-class classification..

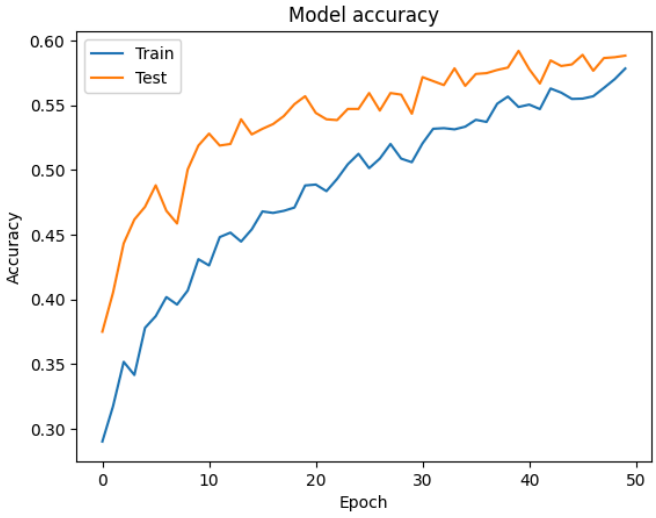
Similarly, a second CNN was developed using chroma feature data instead of spectrograms, data was split into training and test sets with a 25% held out for testing. This data was one-hot

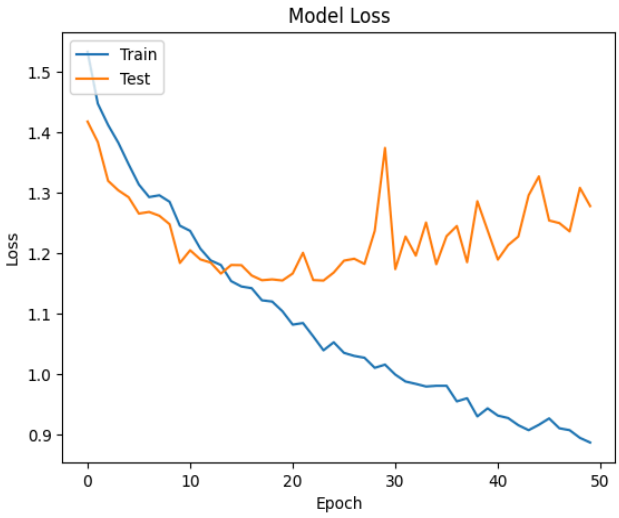
encoded, normalized, and again formatted into TensorFlow datasets. The CNN architecture was adapted to the 12x130 chroma input dimensions and included nearly identical layers for convolution, batch normalization, max pooling, flattening, and dropout to the first model. The network concluded with a dense layer for feature interpretation and a softmax output layer corresponding to the class labels.

Both models were compiled with the Adam optimizer and categorical crossentropy for loss, the models also tracked recall, precision, and accuracy as performance metrics and were trained over 50 epochs.

Results - What were your results? However you evaluated your system, put the results here.

Loss and Accuracy for Chroma Feature Model





Chroma Extraction:

The chroma extraction model consistently returned an accuracy around 70%. Looking at the epoch graphs we can see that the model learns very well after the initial epochs but slowly evens out from an accuracy perspective. The loss function reflects the same but slightly worse lose in the testing data than training. Either way its clear that the model is going in the right direction and its results are relatively impressive.

Spectrogram Model:

The Spectrogram Model yielded an accuracy of around 60% which is slightly lower than the Chroma Extraction model. This could be explained because Spectrograms represent more complex and higher frequencies of data which is benefited by larger data sets.

Conclusions - What did you conclude from your results? Is there anything else your team learned from the experience?

Based on the results of our model we can conclude that while it is possible for a CNN to classify bird species based on their song to a certain extent. A 70% accuracy for the chroma feature model and 60% for spectrogram mode in identifying the species out of 5 options shows the model has learned the differences to a certain extent. If the model learned nothing then the accuracy would be an expected ⅕ or 20%. We believe that the model accuracies could be further improved if we lowered the number of species included or increased the amount of data. Around 1000 samples per species is pretty low for the task we gave the model. We also would like to investigate different audio features to train the model on.