CS 598 MLSP Project: Birdsong classification

Meha Kumar
University of Illinois
mehagk2@illinois.edu

Tara Vijaykumar
University of Illinois
tgv2@illinois.edu

1 Introduction

Identifying a bird by its song is a useful and necessary task for ecologists and ornithologists interested in knowing the population of a bird species in a particular area. Birdsong is typically classified by hand by expert ornithologists. However, this is a taxing, time consuming process. The Cornell Lab of Ornithology has released a dataset of labeled birdsongs as a Kaggle competition. In this paper, we will classify birdsongs using baseline and more recent, sophisticated models. We will also perform exploratory data analysis to better understand the dataset.

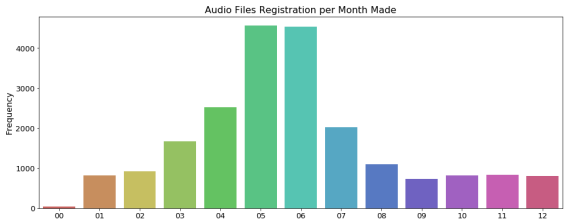

Figure 1: The month each recording was made in.

2 Dataset

The data includes files with non-bird sounds and/or multiple birds in the same recording. There are 264 unique bird species labeled across about 21,000 samples of less than 100 seconds each. A csv file with additional information is provided, but we decided not to use it so that we can practice our audio-classification skills. The other reason is that the csv file did not include these values The plots below were created using that csv file, to give a better idea of the types of birds recorded.

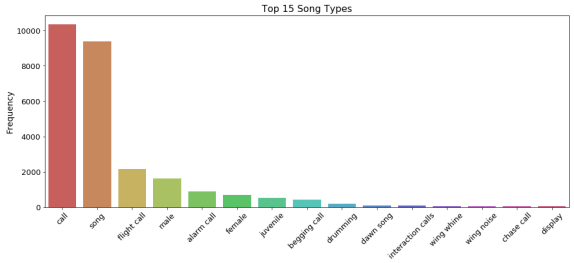

Figure 2: The type of bird call recorded in each recording.

3 Feature Extraction

The choice of representation is integral to audio classification tasks. A few examples of representations (aka, possible features for the models) are listed below, along with images of these representations.

- Spectrogram: generated by applying a series of Fourier Transforms, these plots lie in the time-frequency domain.
- Chromagram: separates audio into 12 pitch classes, or chroma, as understood from the

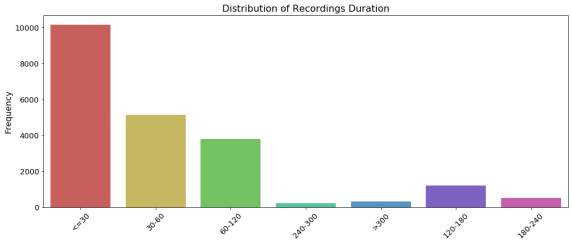

Figure 3: The duration of each recording.

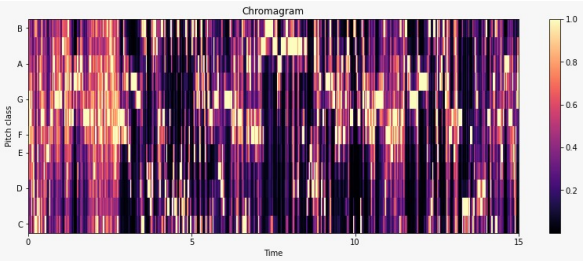

Figure 4: Chromagram

field of music. Sounds fall into the same chroma if they differ by about an octave to the human ear. These plots lie in the time-chroma domain.

- Root-mean-square energy: given a discrete audio signal, RMS-Energy is defined mathematically as:

$$\sqrt{\frac{1}{N}\sum_{n}|x(n)|^2}$$

This formula essentially produces an outline of the raw waveform, giving information about the loudness of the input signal over time.

- Spectral Centroid: calculated as the mean of each frame of the magnitude spectrogram. These plots are in the time-amplitude domain.
- Spectral bandwidth: similar to a weighted standard deviation, these plots are in the time-amplitude domain.
- Spectral Rolloff: the frequency below which at least x percent of the energy of a spectrogram lies. We used $x = 0.85$. These plots are in the time-frequency domain.
- Zero-crossing rate: for each second, this metric computes the number of times the waveform crosses the x-axis.
- Mel Frequency Cepstral Coefficients (MFCC): after computing the *cepstrum* of a signal (the inverse fourier transform of the log signal spectrum), the MFCC represents it in the Mel-scale, which is more proportionate to human perception than traditional linear scales.

All of these measures involve a time axis. The clips are all of varying lengths, which means that some processing is needed to obtain uniform-length

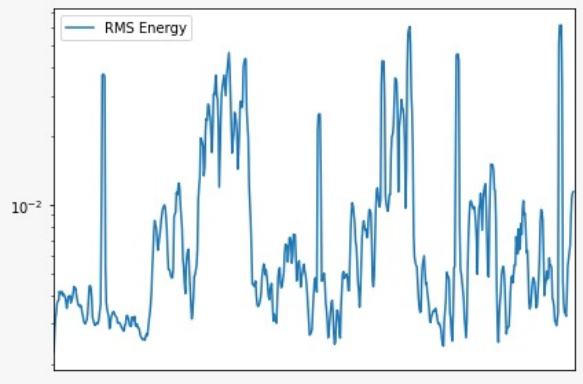

Figure 5: Root-mean-square energy

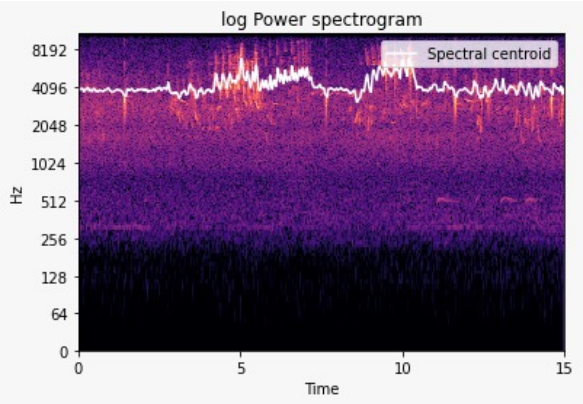

Figure 6: Spectral Centroid

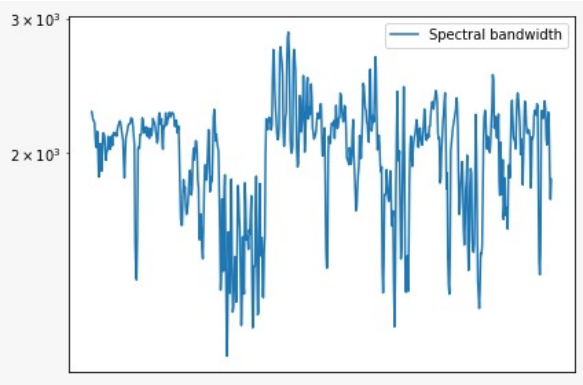

Figure 7: Spectral bandwidth

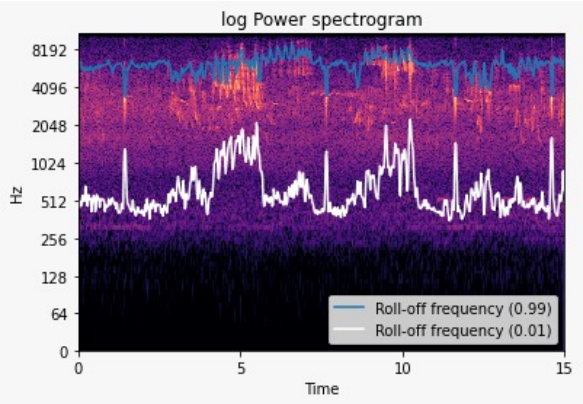

Figure 8: Spectral rolloff

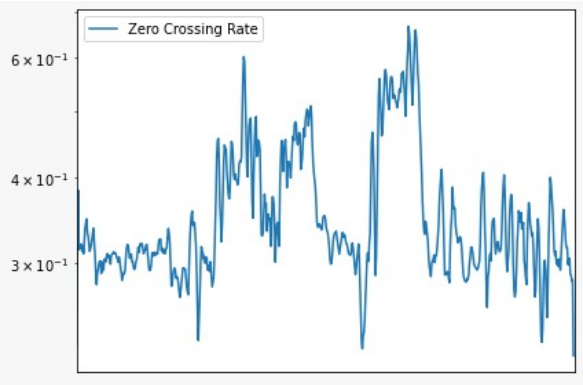

Figure 9: Zero-crossing rate

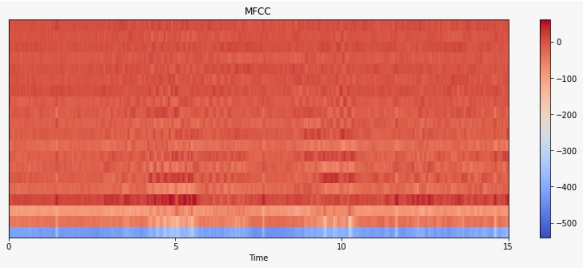

Figure 10: MFCC

input data before feeding the data into models. Another issue is that we have limited computing power, so reducing the dimension of data may be useful. We attempted two methods to account for our lack of computing power and the varying lengths of the recordings. These metrics are separate approaches, ie. we did not trim to 15 seconds and then average metrics.

1. Trim recordings: we trimmed recordings to a random 15 seconds each. We chose 15 seconds because bird songs tend to be fairly long, so using one second (or even one frame of a spectrogram, as in class) would not be enough to identify the song.

2. Average metrics: we distilled each metric (except for the spectrogram) into a single number by taking a simple average over the time domain. For the MFCC, we took the average of each of the 20 coefficients to obtain 20 numbers. Although we are undoubtedly losing a lot of information (literally all the information from the time domain), we wanted to try this rough representation and see how much accuracy we could get.

The final features we used were the spectrograms of 15-second recordings and the average metrics.

4 Simple models on data subset

First, baseline models were developed and run on the various extracted features. The following results in Table 1 and Table 2 were run on only five bird classes, using an 80-20 train-test split, due to limited computing power. Running on a limited subset of the data allowed us to get some quick intuition on what features seemed to perform well before training and testing using all of the data.

We also ran this set of classifiers on each of the average features individually. This would allow us to see if one particular feature is responsible for more of the classification accuracy than others. This means we ran these classifiers on a dataset with the average spectral centroid (a single number) for each sample, and similarly for each of the average features. For MFCCs, we ran these classifiers with a dataset consisting of all 20 average coefficients for each sample. However, the results indicated that the set of all averaged features was more informative than any feature on its own. The highest accuracy from these experiments was

Classifier	Accuracy
Nearest Neighbors	**0.593**
Linear SVM	0.339
RBF SVM	0.492
Gaussian Process	0.508
Decision Tree	0.457
Random Forest	0.389
Multilayer Perceptron	0.559
AdaBoost	0.271
Naive Bayes	0.458
QDA	0.119

Table 1: Test accuracy scores for various classifiers on 15-second spectrograms of bird songs from 5 bird species.

Classifier	Accuracy
KNN Classifier	0.573
Linear SVM	0.680
RBF SVM	0.267
Gaussian Process	0.253
Decision Tree	0.600
Random Forest	0.533
Neural Net	**0.760**
AdaBoost	0.507
Naive Bayes	0.493
QDA	0.547

Table 2: Test accuracy scores for various classifiers on the average metrics described earlier. Features were appended to form a single vector for each sample.

Hyperparameter	Value
Hidden Layers	3
Nodes in hidden layer 1	256
Nodes in hidden layer 2	128
Nodes in hidden layer 3	64
Activation function	ReLu
Optimizer	Adam
Loss	Cat. Cross Entropy
Epochs	20
Batch Size	50
Train accuracy	0.9902
Test accuracy	0.8701

Table 3: Hyperparameters used for the feed-forward neural network trained on five classes of bird species. Train and test accuracies are provided.

the MFCC using a Neural Net, reporting accuracy 0.680.

Seeing as the highest test accuracy was given by the using all averaged features, we moved forward in our classification efforts using this data representation. This was a surprising find, as this representation has no time information. Possible reasons the spectrogram representation failed to give better performance are:

- 15 seconds was not enough: the recordings contain background noise and are not guaranteed to have the bird in question singing continuously for the whole clip. Even though we clipped silence from the beginning and end, the random 15 second clip we extracted may or may not have had enough information to identify the bird.

- Dimensionality too high: the dimensionality may have needed to be reduced using PCA or NMF in order for the models to perform well.

5 More sophisticated model on data subset

After determining which feature set to use, we created a deep neural architecture to classify the bird species. By comparing our sophisticated model to the baselines from the previous section, we aimed to determine if our chosen feature set *could* achieve better performance than our baselines indicated. The model parameters for this feed-forward neural network, along with its performance, are shown in Table 3.

We considered the gain from 0.760 from the basic classifiers to 0.870 from the deep neural network to be significant enough to continue with this feature set.

6 Sophisticated model on more bird species

The next step is running a more sophisticated model on more bird species. We were able to load 148 bird species' recordings at once. This is less than the original 264 birds, but significantly more than five. This should give us a better idea of the final performance of our model.

We trained a similar feed-forward neural network, with more structured hyperparameter search. The parameters changed were the number of hidden layers (2 to 5), the number of nodes in each hidden layer (64 to 512), and the batch size (50 to 500). We also included dropout layers between each layer with probability 0.5 to prevent overfitting. After starting hyperparameter search with 100 epochs each, however, we realized the performance was absolute garbage. None of the hyperparameters boosted the accuracy much above 10%, even on the training data, and increasing the number of epochs was not helpful.

Curiously, when dropout layers were deleted, the training accuracy quickly approached that of the five-bird models (around 0.99). It is interesting, but not unexpected, that the test accuracy on five birds still remained quite high, while on 148 birds it dropped to about 10%. Doing our initial work with overfitting prevention measures would have been a smarter choice.

7 Future Work

There are many avenues for improvement in this work. First, better feature extraction could be performed. Audio recordings could be split into segments of equal length, say n seconds long, such that each recording is represented by $length\_of\_recording/n$ segments. Then, a classifier could be trained on those segments. A final prediction would be attained by taking the majority vote from all the predictions for all the segments for each audio file. This way, we avoid losing information as we did when we clipped only one 15 second segment.

In addition to this improved clipping algorithm, further testing of time-domain preserving representations would be needed. For example, we could test the spectrogram, mel-spectrogram, and MFCC. Dimensionality reduction could be helpful at this stage as well. We did not use traditional dimensionality reduction because our averaging technique seemed to perform well, but certainly these methods are worth exploring.

Another way to improve performance would be source separation. However, monaural source separation with no labeled data is a difficult problem. We fiddled with NMF to see what kind of separation we could achieve, but because we also don't know how many sources of audio there are in each recording, we received some truly awful sounds back from this algorithm.

8 Conclusions

In hindsight, spending most of our time developing on such a small subset of the data was a poor choice. It is not surprising that the small feature set of about 30 total integers per sample did not perform well on discriminating between 148 different bird songs. Overall, however, we were able to practice what we've learned about audio processing and classification tasks.

The code for our project can be found here: https://github.com/mehaKumar/birb