Spotting the Mockingjay: Learning to Detect Bird Sounds Amidst Background Noise

Taneisha Arora*
Department of Statistics
University of California, Irvine
Irvine, CA 92617
arorat@uci.edu

Thanasi Bakis*
Department of Statistics
University of California, Irvine
Irvine, CA 92617
abakis@uci.edu

Theja Krishna*
Department of Computer Science
University of California, Irvine
Irvine, CA 92617
takrishn@uci.edu

Bryon Tjanaka*
Department of Computer Science
University of California, Irvine
Irvine, CA 92617
btjanaka@uci.edu

Abstract

The Bird Audio Detection (BAD) Challenge was a DCASE 2018 challenge in which participants sought to develop models that detected recordings with bird sounds. The participants were given 3 development datasets and assessed on 3 evaluation datasets. In this project, we examine the differences in detection performance, as measured by AUC, when using different sets of features to perform this challenge. To this end, we train four convolutional models that use either MFCC (Mel Frequencey Cepstral Coefficients), Spectral Contrast, Chromagram, or all 3 of these features to perform the detection. Not surprisingly, we find that the model that uses all features typically performs the best, but its performance is similar to that of using MFCC alone, suggesting that the Spectral Contrast and Chromagram features do not provide any information that MFCC does not.[1]

1 Introduction

Bird audio detection has applications ranging from acoustic wildlife monitoring, such as assessing wildlife health in Chernobyl, to pre-filtering for larger, more computationally expensive tasks, such as audio classification of bird species. A bird audio detector can also function as an automatic filter for crowd-sourced libraries like Xeno-Canto, reducing the human effort required to sort through audio files containing bird sounds.

This challenge of detecting bird sounds in audio files has been put forth several times in the past (1), leading to dozens of proposed solutions (2). Currently, state of the art approaches such as (3) use Mel-band spectrograms as their primary feature. They then run these spectrograms through convolutional neural networks to produce their results. For example, in the DCASE 2018 Bird Audio Detection challenge, participants were asked to determine the presence of birds based on sound. Participants were given three development datasets to train their models on and then assessed on three evaluation datasets. The top contestant for this challenge was able to obtain a 95% AUC on the development datasets, and scored an 89% AUC on the test datasets by using the Mel-band spectrogram as their primary feature (4).

*Equal contribution. Order determined alphabetically.

[1]Code: https://github.com/btjanaka/hhh

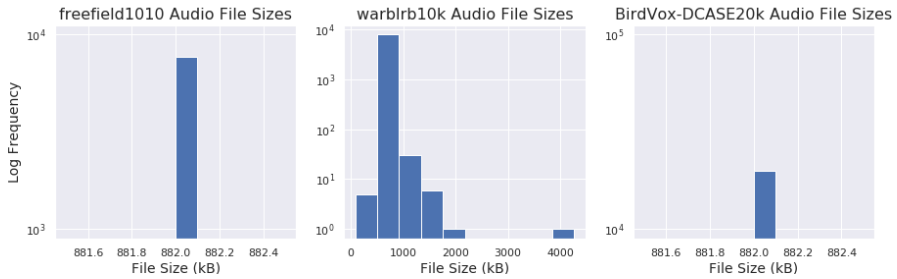

Figure 1: Distribution of audio file sizes for the freefield1010, warblrb10k and BirdVox-DCASE20k dataset. Note our use of a log scale, as the vast majority of files had the same size.

In this project, we seek to analyze how feature selection affects detection results. To this end, we create models that utilize one or all of 3 audio features – MFCC, Spectral Contrast, and Chromagram. We then compare the performance of these models on the development datasets from the DCASE 2018 challenge. Ultimately, we find that using all features typically produces the best results, but not much better than using MFCC alone.

2 Data

Data for this project originates in the DCASE 2018 Bird Audio Detection Challenge (1). The six datasets provided with the challenge were split into two categories: development and evaluation. Each of the three development datasets came with a collection of audio files with a corresponding CSV file of labels. The evaluation datasets, on the other hand, did not come with labels. Due to the lack of labels for the evaluation datasets, we relied on the three development datasets for training and testing our models.

2.1 Sources

The development datasets came from three different sources, meaning that our training data was not homogeneous. The datasets were created as follows:

1. **freefield1010**: A collection of 7690 field recordings from around the world accrued as part of the FreeSound project (5). All audio files in this collection were standardized for data-mining in audio archive and soundscape research.
2. **warblrb10k**: A collection of 8000 audio recordings crowd-sourced from users of the Warblr bird recognition app. These recordings cover a large part of the United Kingdom. Being crowdsourced, the sound clips include urban noises like human speech and traffic in addition to background noises from more natural environments.
3. **BirdVox-DCASE-20k**: These audio clips were collected by six different ROBIN autonomous recording units positioned around Ithaca, New York on the night of September 23rd, 2015. All sound files in this dataset are 10 seconds long, and the labelled set has anywhere from 10 to 100 mislabelings.

2.2 Exploratory Data Analysis

Before beginning the modeling process, we sought to develop a holistic view of our datasets. From a cursory glance at the metadata associated with our development datasets, we found that the audio files from freefield1010 and BirdVis-DCASE-20k were all 10-second (882 KB) long .wav files with a sampling rate of 44.1 kHz. The crowd-sourced audio clips collected from the Warblr app, however, were variable length, as suggested by the inconsistent audio file sizes, which ranged from 84 KB to 4270 KB. Distributions of the file sizes for each of the datasets are shown in Figure 1.

After gaining a preliminary understanding of the nature of the audio clips, we took a closer look at the label files associated with each of the three development datasets. To ensure that our data was balanced and had a comparable number of examples for clips with and without bird sounds. As can

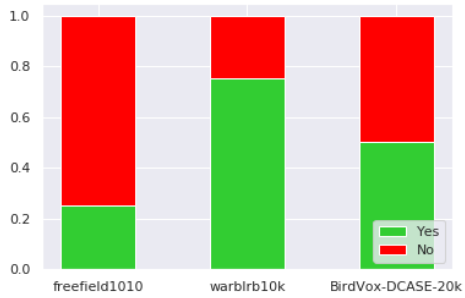

Figure 2: Proportion of Audio Clips With Bird Sounds across the Development Datasets.

be seen from Figure 2, $\sim$25% of the clips in freefield1010 had bird sounds, $\sim$75% of the clips in warblrb10k had bird sounds, and $\sim$50% of the clips in BirdVox-DCASE20k had bird sounds. Hence, while the data was skewed for the freefield1010 and warblrb10k datasets, overall, the data across all three development datasets was fairly balanced.

2.3 Data Preprocessing

Given a majority, but not all, of the audio files across the three development datasets were about 10 seconds long, we decided to cut the larger audio files and pad the shorter ones to convert them to 10-second long clips. Digitizing these 10-second long clips yielded an $N \times 450$ matrix for every audio file, where N was the number of channels determined by the signal processing technique used to convert the audio file to a digital representation.

3 Methods

3.1 Feature Extraction

Here we describe the three audio features analyzed in our project: MFCC, Spectral Contrast, and Chromagram. Each of these features originate in digital signal processing. Visualizations of these features for a single audio file may be found in Figure 3.

3.1.1 Mel-Frequency Cepstral Coefficients

A highly common feature representation of audio data is the set of Mel-Frequency Cepstral Coefficients (MFCC). Broadly speaking, the cepstrum is the rate of change of the frequencies in the audio, but not particularly in the time domain nor the frequency domain. (6) From a musicality perspective, we interpret the MFCC feature to give a broad overview of the audio itself, particularly representing the timbre of the sound, i.e. the inherent frequency-based characteristics of the voices in the sound.

Briefly, the calculation of the MFCC proceeds as follows: the signal undergoes a Fourier transform; the spectrum undergoes two transformations, a conversion to Mel scale frequencies and a log magnitude transformation; and the resulting spectrum undergoes a discrete cosine transform. The Mel scale is designed to emphasize the frequencies most audible to the human ear.

3.1.2 Spectral Contrast

Spectral contrast is another feature representation of audio that is often believed to work well both in tandem with and in place of the MFCC (7). Broadly speaking, it is the difference in intensities of the spectral peaks and valleys throughout the audio track. From a musicality perspective, we interpret the spectral contrast feature to give a measurement of the noisiness or variety in the sound throughout the track.

The calculation of the spectral contrast is similar to that of the MFCC, but with a few notable changes, including: the use of octave sub-bands for frequencies instead of the Mel scale and the use of the Karhunen-Loeve transform in place of the cosine transform.

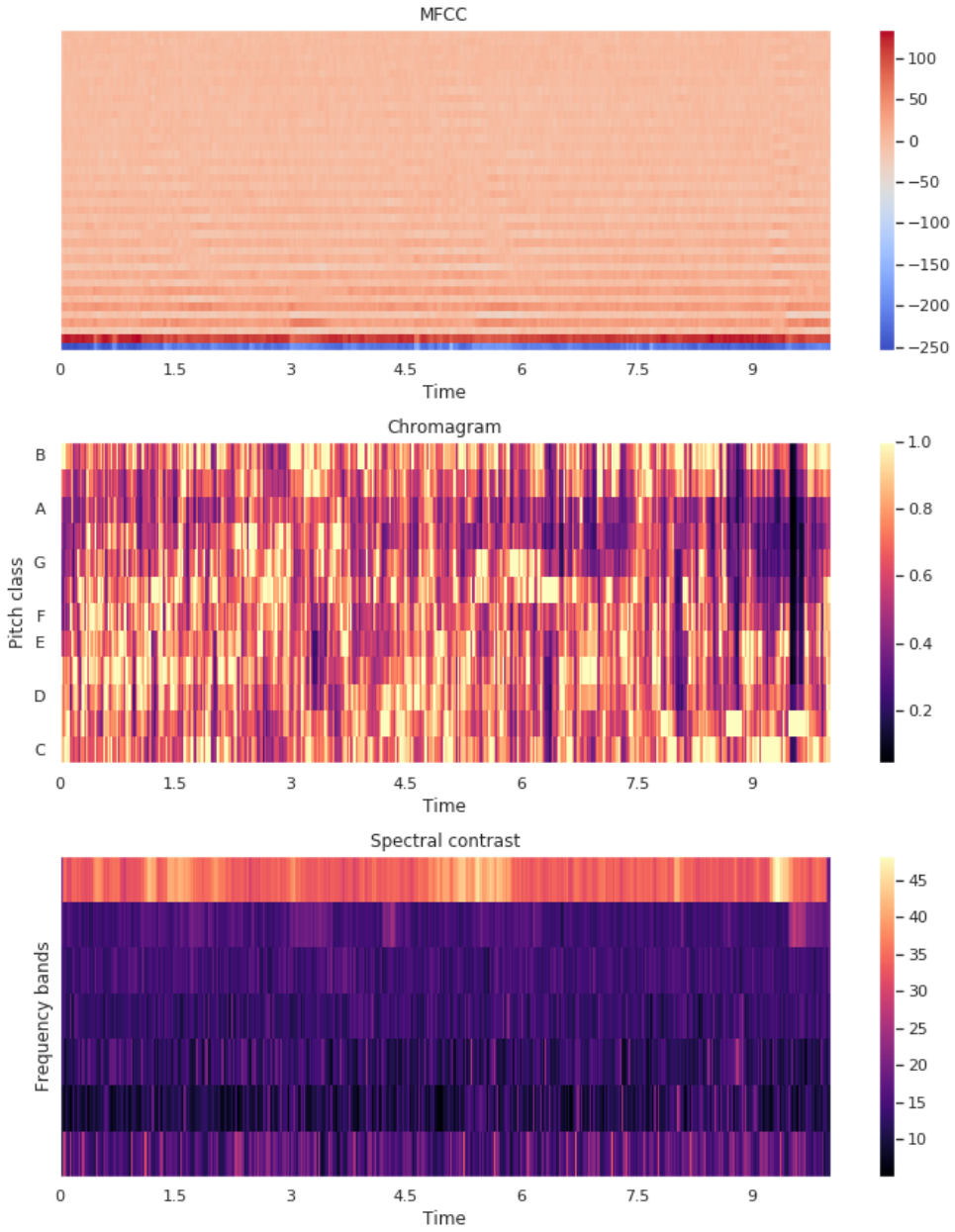

Figure 3: An example of the audio features of a single audio track.

3.1.3 Chromagram

The third and final feature we used in our experiments was the chromagram, which measures the intensities of the sound divided into the twelve distinct semitones (C, C#, D, . . .) across all octaves. (8) From a musicality perspective, we interpret the chromagram very similarly to this definition, measuring how much each note is present throughout the audio.

3.2 Model Architectures

In order to compare the effects of each feature on the classification performance both individually and simultaneously, we designed three separate model architectures to handle the various combinations

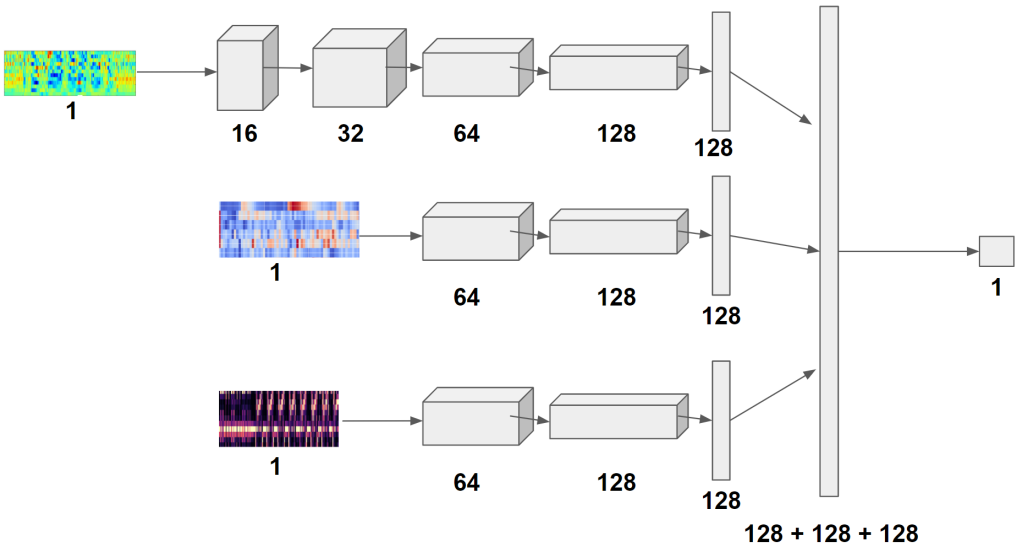

Figure 4: The MultifeatureModel architecture.

of differently-sized features that we engineered. These architectures – "ConvNet", "BabyConvNet", and "MultifeatureModel" – are discussed in the following sections.

3.2.1 ConvNet

We first designed a model to perform classification using just the MFCC feature. This architecture, titled "ConvNet", is a basic convolutional neural network that feeds its 40x450 MFCC input through four layers of filtering and max pooling, with each layer gradually increasing the number of filters until we have an output size of 2x26 with 128 channels. We then perform global average pooling to receive a single vector of length 128. Between each convolutional layer, dropout is performed with probability 0.2. Finally, we use a linear transfer with a sigmoid activation to receive the estimated probability that the input audio contained bird sounds.

3.2.2 BabyConvNet

We needed to slightly modify the ConvNet architecture to allow spectral contrast or chroma to be used as features, as the dimensions of these feature matrices were not as large as those of MFCC and thus could not undergo as many max pools as the MFCC. In this "BabyConvNet" architecture, we use only the last two layers of filtering and max pooling from ConvNet, instead of the previous four. All other details remain consistent.

3.2.3 MultifeatureModel

Finally, in order to build a network that would allow all the features to be used simultaneously, we created an architecture that unifies the above two techniques. The "MultifeatureModel" first removes the linear transfer and sigmoid activation from the ConvNet and BabyConvNet architectures. It then takes the 128-length output vectors from each model and concatenates them to form one longer vector, which is then passed through the linear transfer and sigmoid activation to receive our final estimated probability. Figure 4 shows our final architecture.

3.3 Training

All models were trained with batch size 16 for 100 epochs using the Adam optimizer and binary cross-entropy loss. We applied a 80-20 train-test split to our data. All training was done using a CUDA-enabled NVIDIA GTX 1650. With these hyper-parameters and hardware, training took 30-60 minutes per model.

3.4 Software

We used the following software in our project:

- **PyTorch**: An open source machine learning framework that accelerates the path from research prototyping to production deployment. We used PyTorch to design and train our neural networks.
- **scikit-learn**: Simple and efficient tools for predictive data analysis. We used scikit-learn to perform a train-test split of our data and calculate AUC.
- **librosa**: A Python package for audio and music signal processing. We used librosa to engineer the MFCC, spectral contrast, and chroma features
- **TensorBoard**: TensorFlow's visualization toolkit, providing the visualization and tooling for machine learning experimentation. We used TensorBoard to visualize the training metrics of our models.

4 Results

Table 1 shows the results for models trained to detect bird sounds with different features. Specifically, we use our ConvNet for MFCC, BabyConvNet for the Spectral Contrast and Chromagram, and MultifeatureModel for All. For each of the development datasets from the DCASE 2018 challenge, we present the final AUC from training on 80% of that dataset and testing on the remaining 20%. For plots of the training and testing AUC over all 100 epochs of training on each dataset, please refer to Appendix A.

We find that the MultifeatureModel, trained with All features, performs the best (i.e. has a test AUC closest to 1) on the freefield1010 and warblrb10k datasets, while the model trained with only MFCC performs best on the BirdVox-DCASE-20k dataset. In general, however, there does not seem to be much difference in performance between the models trained with All and MFCC.

Features	Model	Test AUC		
		freefield1010	warblrb10k	BirdVox-DCASE-20k
MFCC	ConvNet	0.8236	0.9148	**0.8800**
Spectral Contrast	BabyConvNet	0.7653	0.8938	0.7156
Chromagram	BabyConvNet	0.7399	0.8203	0.7924
All	MultifeatureModel	**0.8425**	**0.9263**	0.8635

Table 1: Test AUC from training models to detect sounds from various features in each dataset. "All" refers to using all three of the other features. Models with the best performance are bolded.

5 Discussion

It is not surprising that the MultifeatureModel trained with All features has the best AUC on most of the datasets. With more features, the model has more information from which to detect a bird sound. However, the fact that the models with All and MFCC have such close performance suggests that MFCC is the primary feature on which decisions are based. Perhaps the information conveyed by the Spectral Contrast and Chromagram features is encapsulated by the information in the MFCC, especially since these features are smaller than the MFCC. We cannot say that they provide *no* information, as the models trained with each of those features achieved respectable results.

Another issue to consider is that the Spectral Contrast and Chromagram features were passed through different architectures than the MFCC. That is, since the BabyConvNet was smaller than the ConvNet, Spectral Contrast and Chromagram may not have been processed well enough, meaning they could not learn a function complex enough to take advantage of these features.

6 Conclusion

In this project, we analyze models trained to detect bird sounds using various audio features. We find that using all three of the features we analyze – MFCC, Spectral Contrast, and Chromagram – typically results in the best performance in terms of test set AUC on each of our three datasets, but using MFCC alone does not make much difference.

These results suggest several routes for future exploration. For the model, we can develop more complex convolutional architectures to further process our features, in order to see if more complex models eliminate the differences between features. In a similar vein, we can also experiment with entirely different architectures; for instance, we could replace the convolutional architecture for each feature with an RNN that recurs over time slices, either one slice at a time or with a sliding window. Meanwhile, we could examine the effect of data augmentation on feature selection – (4) inserts random chunks from other audio files into each audio file. Finally, on the implementation side, we could look into utilizing more powerful computing resources such as Google Cloud.

Acknowledgments

This project was completed as part of the Spring 2020 offering of CS 172B: Neural Networks and Deep Learning taught by Pierre Baldi at the University of California, Irvine. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the instructor.

References

[1] D. Stowell, M. Wood, Y. Stylianou, and H. Glotin, "Bird detection in audio: a survey and a challenge," *CoRR*, vol. abs/1608.03417, 2016.

[2] D. Stowell, M. D. Wood, H. Pamuła, Y. Stylianou, and H. Glotin, "Automatic acoustic detection of birds through deep learning: The first bird audio detection challenge," *Methods in Ecology and Evolution*, vol. 10, no. 3, pp. 368–380, 2019.

[3] T. Grill and J. Schlüter, "Two convolutional neural networks for bird detection in audio signals," in *2017 25th European Signal Processing Conference (EUSIPCO)*, pp. 1764–1768, 2017.

[4] M. Lasseck, "Acoustic bird detection with deep convolutional neural networks," tech. rep., DCASE2018 Challenge, September 2018.

[5] D. Stowell and M. D. Plumbley, "An open dataset for research on audio field recording archives: freefield1010," 2013.

[6] A. V. Oppenheim and R. W. Schafer, "From frequency to quefrency: A history of the cepstrum," *IEEE signal processing Magazine*, vol. 21, no. 5, pp. 95–106, 2004.

[7] D.-N. Jiang, L. Lu, H.-J. Zhang, J.-H. Tao, and L.-H. Cai, "Music type classification by spectral contrast feature," in *Proceedings. IEEE International Conference on Multimedia and Expo*, vol. 1, pp. 113–116, IEEE, 2002.

[8] A. Wankhammer, P. Sciri, and A. Sontacchi, "Chroma and mfcc based pattern recognition in audio files utilizing hidden markov models and dynamic programming," in *Proceedings of the 12th International Conference on Digital Audio Effects (DAFx-09)*, pp. 401–407, 2009.

[9] V. Lostanlen, J. Salamon, A. Farnsworth, S. Kelling, and J. P. Bello, "Birdvox-full-night: a dataset and benchmark for avian flight call detection," in *Proc. IEEE ICASSP*, April 2018.

[10] Y. Su, K. Zhang, J. Wang, and K. Madani, "Environment sound classification using a two-stream cnn based on decision-level fusion," *Sensors*, vol. 19, p. 1733, 04 2019.

Appendix

A AUC vs. Training Epochs

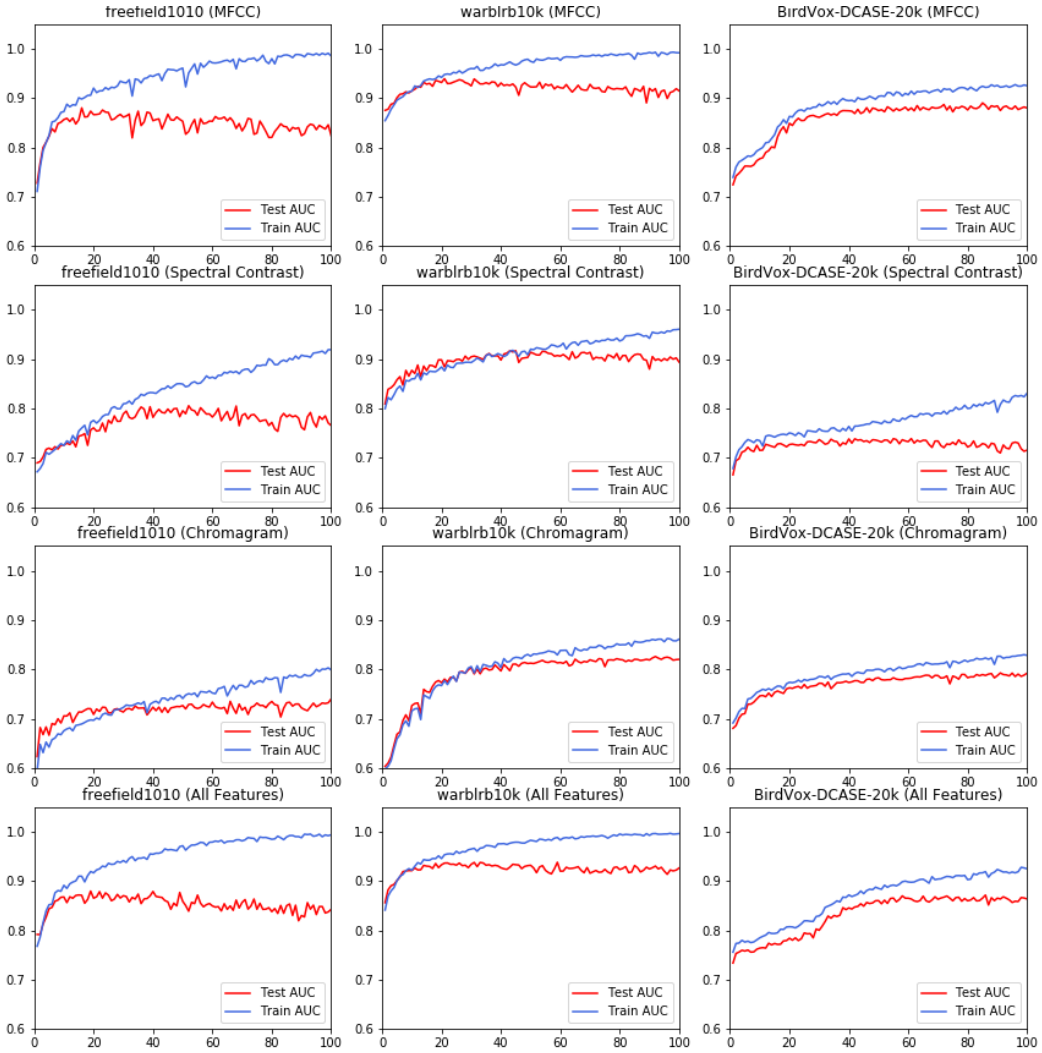

Figure 5: Test and Train AUC across training epochs for each of our experiments.

B Team Member Contributions

Taneisha Arora researched alternate digital representations for audio files and the effect of combining features on the model, created some visualizations for initial EDA, contributed to making the presentation and writing the report.

Thanasi Bakis researched techniques for analyzing audio data, implemented parts of the models, developed slides for feature engineering, and wrote and edited sections of the report.

Theja Krishna trained and evaluated all models and collected results, developed slides for results and discussion, wrote and edited sections of the report.

Bryon Tjanaka researched past work on the Bird Audio Detection Challenge, implemented the PyTorch models, presented the project to the class, and wrote and edited sections of the report.