Bird Call Detection Final Report

Group 1: Lucas Morgan, Tianyuan (TY) Shao

June 3, 2022

1 Introduction

1.1 Problem Statement

We are concerned with monitoring rare and endangered bird species and because of the difficulty of finding ample amounts of audio recordings, our goal is to correctly classify these particular species given minimal training data. We are given 1 minute clips and must determine for every 5 second interval whether the specified bird is calling in that 5 second interval with True or False values.

1.2 Background

Many bird species in Hawaii are becoming extinct. This has negative effects on the overall food chains of the islands as extinction of certain species has a domino effect in which other predator or prey species may also go extinct. Scientists are enacting conservation efforts to attempt to preserve the natural balance of the ecosystem. They must take into account the impacts of their efforts and how certain species may react in response.

Audio recordings are utilized to conduct population monitoring and remain a popularized choice in means of evaluation because they are especially useful in areas that are not easily accessible where many of the rare bird species inhabit. However, analyses of audio recordings heavily require manual labor by trained experts who must sit through the recordings and tediously annotate them in order to identify various species. This is where neural networks can come into play as they can be incorporated into this evaluation process and trained to identify specific bird species.

2 Exploratory Data Analysis

2.1 Initial Observations

After conducting some initial data analyses, we noted that our dataset contains 14852 audio recordings of inconsistent, varying lengths. Some recordings are only 3 seconds long while others may persist for durations of longer than a minute. Additionally, there are some recordings that include several secondary species labels along with a primary species label. Recordings with these labels indicate that there are additional bird species that are calling in the background. There are no missing data or recordings within our dataset.

2.2 Bird Species Label Analysis

There are a total of 152 different species labels within our dataset which will correspond to the number of classes our model will have. The distribution of the recordings for each species label generally makes sense after examining the associated bird names. There are the most recordings (500) for common sandpiper, eurasian skylark, house sparrow, barn owl, mallard, and dunlin birds. These are all relatively common birds that are prevalent in many areas throughout the world, not just exclusive to Hawaii. There is not too much concern about the endangerment of these species since their populations are stable as shown through the multitude of recordings. Species that have few recordings are considered to be rare species and are in critical endangerment status. For example, the maui parrotbill and akohekohe, which have only one and two recordings respectively, are known to be critically endangered species that reside on high elevation mountain forests of Maui.

2.3 Potential Challenges Faced

Our data consists of audio recordings which introduces various challenges. Audio data in general can be very sparse. In our case, the specified bird is not making sounds at each second of the recording. It may be hard to pinpoint the exact positions in the recording the bird may be calling. There may also be some unaccounted noise in our audio recordings. While listening to some audio recordings, we encountered one in which there was

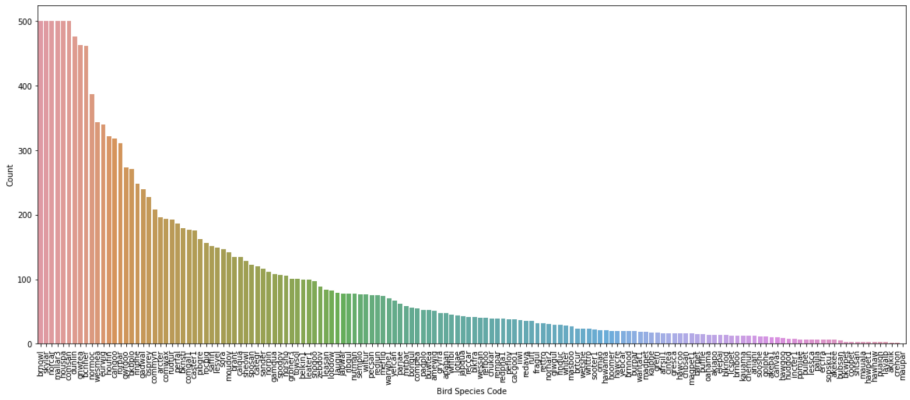

Figure 1: Countplot of Bird Species Labels

a fly buzzing near the recording device. This sort of noise is unrelated to our problem of interest. Our recordings are also of various lengths which may potentially cause our network to pick up features that are unrelated to the actual calls of the birds themselves. If some birds had longer recordings, our model may associate audio recordings of longer lengths with that associated bird. These are both features unrelated to the bird's call itself that we do not want our model to pick up on.

Naturally, because some of the bird species are more rare than others, we have fewer training data for those species than the bird species that are more prevalent. We are more interested in being able to identify those species than the more common ones because the goal of our project is to be able to identify the calls of rare bird species for monitoring.

2.4 Sound Event Detection

Sound event detection is a form of an audio classification problem and also a type of audio tagging problem. The goal of audio tagging problems is to decipher the label the signal corresponds to. However, sound event detection problems take it one step further with the objective of recognizing the specific temporal instances different sounds are active within an audio signal along with the corresponding label for that signal. Our problem at hand is a sound event detection problem since we are determining the presence or absence of a specified bird at various segments within our overall audio recording input.

3 Preprocessing

3.1 Signal Preprocessing

The primary signal preprocessing technique we implemented was Additive White Gaussian Noise. Intuitive in its name, this technique is used to add some noise to our signal. It is additive in the sense that it can be added to preexisting noise within the signal and white in reference to the power being uniformly distributed across all frequencies. Additionally, the overall probability distributions for the noise will follow a Gaussian distribution with zero mean.

This is a common and popular technique used to generate a known amount of noise within our signals. The overall purpose is to mimic the effect of random processes for each signal that you would expect to find in real-world environments. Also, since we will be converting signals into spectrograms using Fourier transforms, Additive White Gaussian noise allows for easy calculations because mathematically, the Fourier transform of Gaussian noise is identical to itself. We then normalized across each entry by dividing by the maximum of each sample from each signal. Adding this sort of noise to our audio signals is good because in changing our original audio samples with varying signal to noise ratios, we are able to evaluate the performance of our model given noisy data as we have discussed previously in the case where we heard the fly buzzing in the background of the recording.

3.2 Mel Spectrograms

Mel Spectrograms are a state of the art tool used in audio classification problems, more specifically in our case, sound event detection. Spectrograms are visual representations of a spectrum of frequencies of a signal throughout time. Mel Spectrograms, on the other hand, are spectrograms scaled in a non-linear manner that replicates the human hearing.

We will be training our model on Mel Spectrograms rather than normal spectrograms because not all of the frequencies in our problem hold equal importance. We are more concentrated on distinguishing between the higher frequency pitches of bird call sounds which generally are more challenging for the human ear to distinguish between. All of our data comes in the form of audio signals. We apply short-time Fourier transforms to create a spectrogram and afterwards, ensure that the y-axis is converted into the Mel-scale.

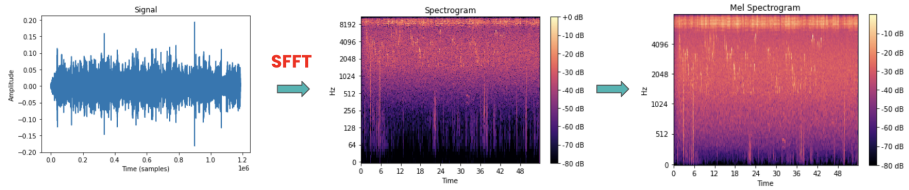

Figure 2: Derivation of Mel Spectrograms

3.3 Spectrogram Preprocessing

To process our spectrograms, we utilized two libraries in particular: Librosa and Albumentations.

Librosa is a Python library that allows for audio analysis and manipulation. We utilized Librosa to directly convert our audio files into Mel Spectrograms. We first loaded the audio files into our Dataset class as time-series NumPy arrays and then cropped or padded the samples to force them into equal lengths. For recordings that were longer than our specified duration, we specified a random offset to start the sample. We then used that standardized NumPy array to compute a corresponding Mel Spectrogram. Since the goal of our project was to solve a sound event detection problem at 5 second intervals of an audio recording, we also specified the number of samples to be taken from each training data as 160,000 which was the result of 32,000 samples taken every second for 5 seconds.

Additionally, Albumentations is a Python library allowing for image augmentations with OpenCV and NumPy functions. Adding image augmentations to our Mel Spectrograms would improve the overall performance of our model given our limited training data. We applied some of the following transformations to our spectrograms with random probability except for Normalize (done for every image): HorizontalFlip, Cutout, and CourseDropout. The Cutout and CoarseDropout transformations randomly cover a region or regions of the input with rectangles in order to improve generalization so that our model learns to pay attention to the entirety of the image.

3.4 Data Augmentation

SpecAugment is a data augmentation technique from the TorchLibrosa library. Unlike the previous augmentation techniques we incorporated, this transformation is directly applied to the spectrogram image when they are fed into our neural network. The three main augmentations that SpecAugment consists of are time warping, time masking, and frequency masking. The goal of adding these augmentations is to make our model more robust to variations in time directions and loss of frequency information [4]. Out of the three augmentations, time warping is the most computationally expensive since time and frequency masking have effects similar to Cutout and CoarseDropout.

We wanted to include multiple transformations and augmentations to the images of our Mel Spectrograms because we were concerned with the limited training data associated with the rarer birds of the classes we were primarily interested in identifying. Adding these augmentations would further help improve the generalization of our model. However, adding too many may also be a concern in that our model will start to underfit the data if it generalizes too much. If that appears to be the case, then we should make note to remove some of the augmentations we applied onto our data.

4 Model Architecture

4.1 PANNs Background

PANNs are large-scale pretrained audio neural networks for audio pattern recognition. They are a family of CNNs trained on the AudioSet (ImageNet equivalent for audio data) dataset with contains over 5000 hours of audio recordings and 527 classes. As mentioned in the original paper [2], mean average precision (mAP) was used to compare model performances and it was discovered that generally CNNs with more layers had higher mAP scores. With significantly more layers and parameters than its CNN counterparts, CNN14 had the highest mAP score of 0.431 compared to CNN6's 0.343 and CNN10's 0.380. The original paper also proposed a model

Wavegram-Logmel_CNN which takes both a waveform and a Mel Spectrogram as the input features and has a mAP score of 0.439. It uses around 81 million parameters, which is similar to CNN14, yet has a slightly higher mAP score.

PANNs typically outperform state of the art models for audio tagging and sound event detection problems which is why we decided to base our general model architecture after it. However, one challenge is that the AudioSet dataset only provides weak labels that indicate the presence or absence of a certain sound. It does not provide information about the location of the sound which is a major part of sound event detection tasks.

PANN's architecture produces two outputs: Framewise Output and Clipwise Output. A framewise output contains both the time and class, whereas a clipwise output only produces the possible classes for each clip. We use clipwise output to train our model because our training data does not give time information for each label. It only provides the audio clip and the birds that are calling within it.

4.2 Attention-Based Multiple Instance Learning

Multiple Instance Learning is a supervised machine learning technique and a popular framework for sound event detection problems. It involves predicting an overall bag label based on the discovery of instances that trigger the bag label [1]. In our case, we produced a list of potential birds that each 5 second interval of our test samples potentially belonged to. If the specified bird (target variable) is contained within the list of potential birds, we predicted that the audio sample belongs to the bird. If not, we rejected that it was the specified bird calling. Our attention block consists of a one dimensional convolutional attention layer along with a one dimensional convolutional classifier layer. We applied a softmax activation after our input was fed into the attention layer along with a sigmoid activation onto our input with the classifier layer. The clipwise output y is computed using the general weighted average formula of our outputs of the attention and classifier layers.

4.3 Loss Function

Since we are making binary predictions of whether or not a specified bird is calling in each 5 second interval of the audio recording, we will be using a form of binary cross entropy as our loss function. In particular, we implemented a Binary Cross Entropy Focal Loss function based off the built-in BCEWithLogitsLoss() from Pytorch. BCEWithLogitsLoss is more numerically stable than the regular Binary Cross Entropy loss function followed by a Sigmoid layer because it combines the Sigmoid and BCELoss in one class so the operations are combined into one layer. It is important in our particular problem to consider using Focal Loss because of the limited training examples of the rarer birds which will overwhelm the normal cross entropy loss. We would like to emphasize the examples that our model predicts incorrectly so that it can learn the calls of the rare birds over time in a process called Down Weighting.

In implementing our BCE Focal Loss function, we incorporated Gamma, a focusing parameter, and Alpha, a weighting factor. When the focusing parameter is equivalent to 0, it is essentially the same as Binary Cross Entropy loss. As it is increased, the modulating factor, a factor that scales the loss, reduces the contribution of the losses from the examples that are more easily classified (the common birds). The weighting factor, typically a value between 0 and 1 when associated with positive classes, is another common method for addressing class imbalance since its addition makes it a balanced binary cross entropy. According to the authors of the paper [3], it was discovered through experiments that the best values for the focusing parameter and weighting factor were 2 and 0.25 respectively, so we used those values as well as the hyperparameters in our function.

4.4 Model Backbones

4.4.1 MobileNetV2:

We were first interested in incorporating MobileNetV2 because the number of parameters is much smaller than CNN14 or ResNet38. MobileNetV2 has 4 million parameters, whereas CNN14 has 80 million parameters. Since this architecture produces a relatively decent mAP considering the number of parameters it has, we would like to establish the MobileNet backbone as our baseline. Additionally, we chose V2 over V1 here as they are very similar in performance, yet V2 has around 700,000 parameters less than V1.

4.4.2 EfficientNet B0:

Many have also claimed that for this particular competition, the EfficientNet model backbone has also performed quite well. Since EfficientNet uses MobileNetV2's blocks as its core building blocks but with additional layers, we also wanted to try to incorporate this model as our backbone to compare its performance with that of MobileNetV2. However, we found it challenging to truly compare the two models because it was taking too long to train EfficientNet for as many epochs as we did for MobileNetV2. We ultimately felt that it would be

an unfair comparison if we were to observe the results of two models that were trained for a differing number of epochs and focused our resources on training MobileNetV2.

4.4.3 CNN14 & ResNet38:

For this competition and previous ones, CNN14 is also a very strong performing model. We wanted to test out both this model as well as ResNet38 because they have a similar number of model parameters and maP scores. Its strong performance is also backed by the PANNs paper as it contains the highest mAP of all the other CNN models it was compared against. However, the number of parameters again made these models quite unfeasible for us to train in a timely manner with our limited computational resources.

5 Evaluation Metric

5.1 F1-Score:

We are using the F1 score as our main evaluation metric for our problem at hand. Because of our imbalanced classes and the distribution of training samples for our labeled bird species, we prefer to use this evaluation metric over accuracy. For our problem, it is of utmost importance that we are able to correctly identify the rare bird species calls because those birds, in particular, are the ones that the scientists are trying to monitor in regards to the effects of their conservation efforts.

6 Conclusion

We used MobileNetV2 as our backbone and produced a 0.6 F1-Score on our validation set over 40 epochs. In total, the running time for training our model took upwards of 30 hours. Considering that we were limited by our hardware and that MobileNetV2 has only about 700,000 parameters, it was quite unfeasible for us to train on various other models such as EfficientNet or ResNet.

From observing various other discussions and experiments, we saw that models using various versions of EfficientNet along with additional augmentation techniques were able to produce a 0.7 F1-score, meaning that perhaps if we were able to train our model using some version of EfficientNet as a backbone instead, we may also see a performance increase. These observations ultimately align with the findings from the PANNs paper stating that models with more layers and parameters tend to outperform ones that have significantly fewer parameters. The obvious drawback of a more complex model, as we have seen, is that it utilizes more computational resources to train.

Ultimately, producing an F1-Score of 0.6 along with our loss function being Binary Cross Entropy Focal Loss indicates that we are doing a relatively good job of predicting the rarer/endangered species of birds. Given the unbalanced dataset of bird species, we felt that we met our objectives of trying to identify these particular species of interest rather than just focusing on the accuracy of our model and correctly identifying as many birds as we can.

7 Contributions

Both Lucas Morgan and Tianyuan Shao equally contributed to the project. Together, we researched the topics and ideas that are explored for this problem and implemented the code together. Lucas conducted the training of the model on his personal device and made small corrections to errors in the code while Tianyuan detailed most of the project presentation slides and report.

References

[1] Maximilian Ilse, Jakub M. Tomczak, and Max Welling. "Attention-based Deep Multiple Instance Learning". In: (2018). DOI: https://arxiv.org/pdf/1802.04712.pdf.

[2] Qiuqiang Kong et al. "PANNs: Large-Scale Pretrained Audio Neural Networks for Audio Pattern Recognition". In: (2020). DOI: https://arxiv.org/pdf/1912.10211.pdf.

[3] Tsung-Yi Lin et al. "Focal Loss for Dense Object Detection". In: (2018). DOI: https://arxiv.org/pdf/1708.02002v2.pdf.

[4] Daniel S. Park et al. "SpecAugment: A Simple Data Augmentation Method for Automatic Speech Recognition". In: (2019). DOI: https://arxiv.org/pdf/1904.08779.pdf.