Abstract

The objective of this project is to build a model using neural networks that can take a audio file of a bird call and accurately predict the bird species. The model was trained using data composed of 12 bird species each with varying number of samples. Two models were trained using convoluted neural networks, CNN. The first model is a binary classification model trained to predict whether a sample is a Western Meadowlark or a Mallard duck. The second model is a multiclass classification model trained to distinguish a bird call as one of the 12 bird species. The binary model was able to differentiate between the two species at an accuracy rate of 93.3%. The multiclass model had a modest accuracy rate of 62.67% To test our model three audio files were also converted into spectrograms. There were multiple birds present in the audio files. The model predicted the presence of a Western Meadowlark, the Stellar's Jay, and a mallard at the highest probabilities in the first audio file. For the second audio file a Stellar's Jay was predicted and in the third audio file the Western Meadowlark, the Stellar's Jay, and a mallard had the highest probability of being present.

Introduction

The goal of this project is to build a reliable model that can predict bird calls. There are numerous potential applications for such a model. It could be used by hobby bird watchers. Ornithologist could make use of such a model to evaluate ecological conditions, the presence of bird species could serve as indicator for the health of a biosphere. It can be used to monitor bird populations, or track migration patterns. One study used classification of bird sounds as an early warning method of forest fires [7].

To develop this model, convolution neural networks will be deployed. The dataset used to train the model is a subset of a larger dataset from Xeno-canto [8]. Xeno-canto houses crowd sourced bird sounds. There are 12 species in this dataset. The 12 species include the American crow, barn swallow, black-capped chickadee, blue jay, dark-eyed junco, house-finch, mallard. Northern flicker, red-winged blackbird, Stellar's jay, western meadowlark, and the white-crowned sparrow. The bird calls are stored as spectrograms, a visual representation of the audio sample. It shows the strength or loudness of different frequencies over time. Each spectrogram in the dataset is an 'image' of the 2-second segment of the birdcall. The dimension of this spectrogram is 343 (time) x 256 (frequency).

Background

What is a neural network?

At its simplest, neural network takes an input vector of *p* variables and builds a nonlinear function to predict the response variable. To understand the structure of a neural network this 28 X 28 image of the hand drawn number 9 will be used.

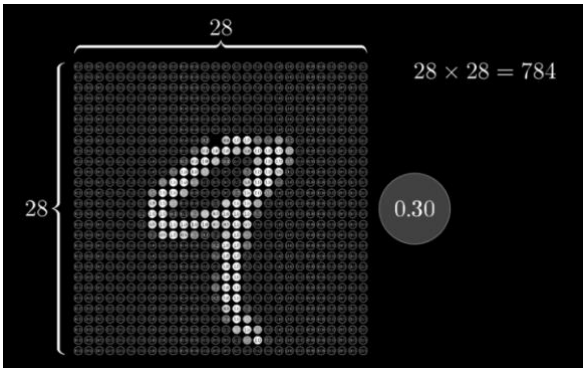

Figure 1 28 X 28 image (3blue1brown)

There are 784 pixels in this image. Each pixel has a value between 0.0 to 1.0. Black is represented by 0.0 and white 1.0. Each one of these pixels will be an input into our neural network.

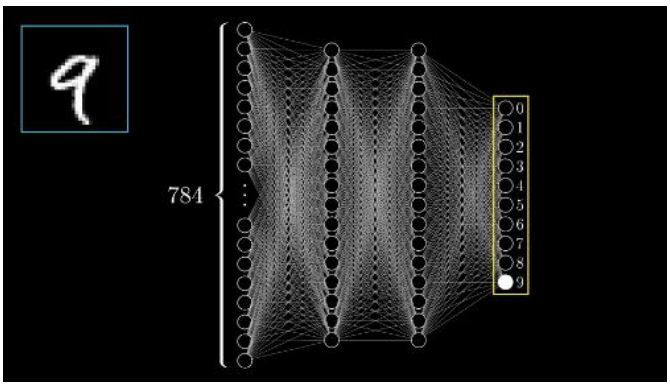

Figure 2 Neural Network (3blue1brown)

The layers between the input layer and the output layer are the hidden layers. The output layer will each have a value between 0.0 and 1.0 representing a probability, all neurons in the output layer will add up to one. Figure 2 shows 2 hidden layers with 16 neurons each. Each connection

from the input layer to a neuron in the first hidden layer is assigned a weight (Figure 4). Figure 4 shows a highlighted area where the shaded green pixels represent weights that are positive, the red pixels are negative to represent edges, and all other pixel outside the highlighted region would be zero. Taking the weighted sum of all pixels then would give us essentially the added sum of the pixels in the highlighted area. An activation function is then applied to this weighted sum.

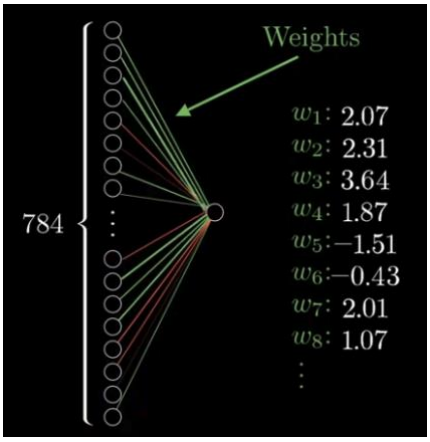

Figure 3 Weights (3brown1blue)

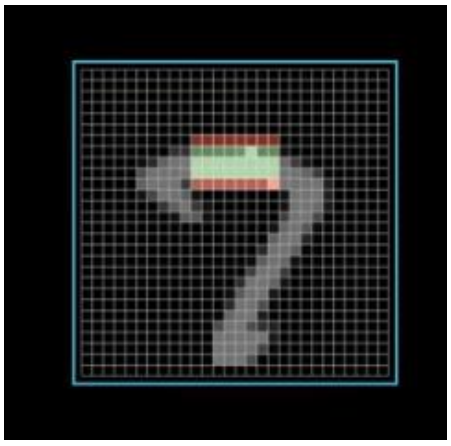

Figure 4 Weights (3brown1blue)

A value called the bias can be added to this weighted sum to make this neuron active or inactive. Each neuron in the layer will have their own set of weights and biases. A chosen activation function is applied to all neurons in this layer. The activation function is an equation that determines the output of that hidden layer and determines the activation of that neuron. Their values can range from -1 to 1 or 0 to 1.

Examples of activation functions is the Sigmoid, ReLu, leaky ReLu and Softmax, all of which are used in the models. For the Sigmoid function, it takes on values 0 to 1. It is the same function used in logistic regression to produce probabilities between 0 and 1, it is used for binary classification models. The ReLu function is the most widely used function for convolution neural networks. It has a threshold of zero, it is a non-linear piece-wise function that is nearly linear, which means that it can preserve the properties of linear models [2]. The Softmax function is used for multi-class logistic regression, it produces a probability distribution for all classes in the output layer, all adding up to 1.

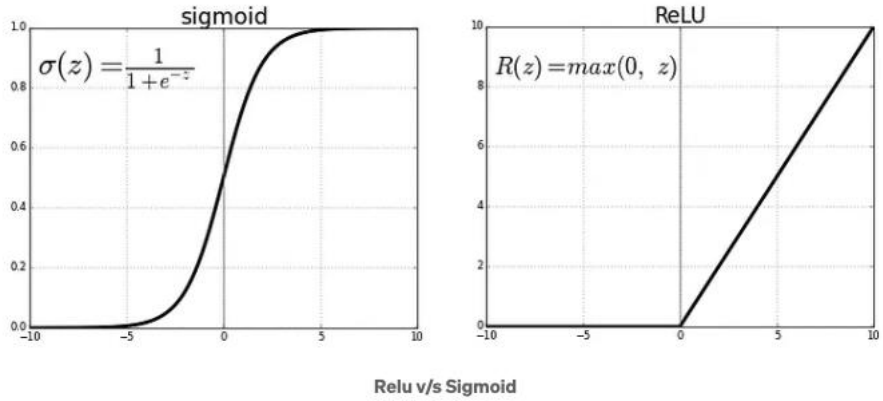

Figure 5 (Medium[1])

A leaky ReLu is like the traditional ReLu in that it sets negative values to zero, except that it allows for a non-zero gradient for negative inputs [1]. The output of this layer becomes the input of the following layer, hence why it is called *feed forward network*. Training a model consists of changing the weights and biases to find the optimal values.

Convolution Neural Network

CNNs for short, can be effectively used for image classification. For this project, the spectrograms or 'image' of a sound will be used to train the CNN models. It mimics how human classify images by taking local features of that image. Figure 6 shows the breakdown of an image into it features.

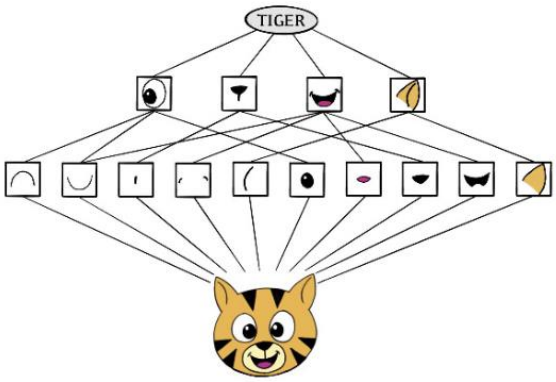

Figure 6 Features of an Image (ISLR)

A convolution layer in a CNN is made up of is made up of several convolution filters. A filter can blur, sharpen, or detect edges to find certain features in the image. Take Figure 7 for example. The filter helps to distinguish certain features that are similar to the filter.

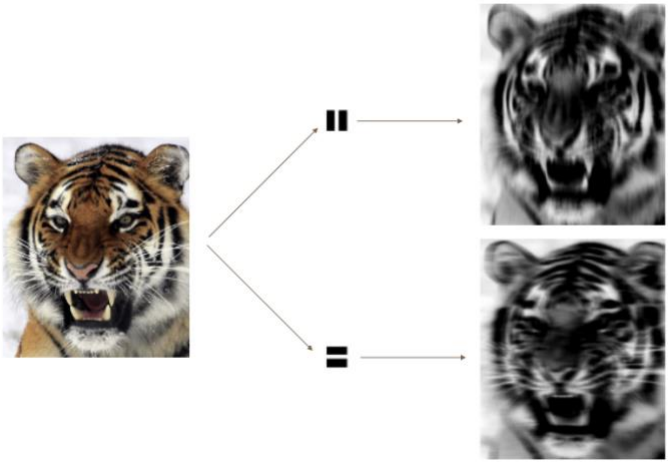

Figure 7 Features highlighted by filter (ISLR)

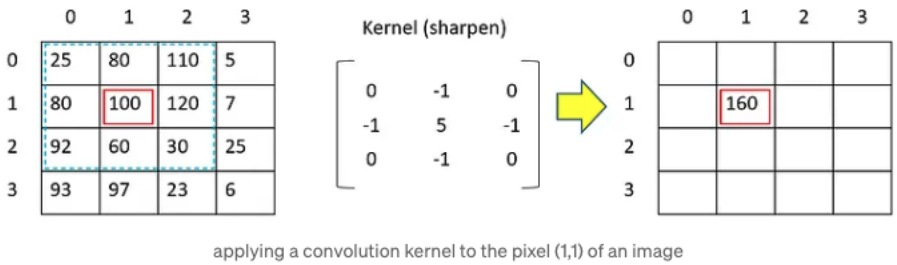

Figure 8 (Medium [4])

Matrix multiplication is used to determine the new values of an image. Each filter as you can see will produce a new matrix to better detect features.

Tuning and Hyperparameters

Besides activation function selection, for the CNN models there are several ways to tune a model. As mentioned before one way to tune a model is to add hidden layers and specifying the number of filters, which are akin to the neurons previously described. The size of the filter can also be adjusted. Larger filters can increase accuracy by capturing more information in one output pixel. Smaller pixel size also makes sense because useful features are usually local, and it would make sense to take only a few pixels at a time [5]. When adding layers there is an increase in filters because with each layer there are new combinations of features and patterns that can be captured.

Pooling layers can be added to the model. A pooling layer condenses larger images into summary images, the max pooling option will summarize non-overlapping of pixels within an image. If there is one pixel with a high number in the block of pixels, the output will reflect the presence of that high value pixel in the resulting reduced image.

Kernel regularization can be utilized to prevent overfitting so that the model can be generalized. It adds a penalty term to the loss function. The loss measures the discrepancy between the model's prediction and the true value. The two common types are L1, or lasso, and L2, ridge regularization. The two common loss functions used for the models are binary cross-entropy loss, for binary models. Categorical cross-entropy loss, for multi-class classification. Another regularization method is to specify a dropout rate at each layer. The dropout rate specifies the number of outputs that are to be randomly ignored for each epoch. The learning curves can be useful in determining if the model is overfitting.

An epoch is completed when the data has passed through an algorithm. The following epoch will learn from the previous epoch an adjust accordingly. To determine what is the optimal number of epochs we look to the training loss and validation loss after testing a chosen number of epochs. Using the learning curves, where both the training and validation loss lines have plateaued will indicate the optimal number of epochs. For each epoch a chosen batch size will be used to train the model. For example, in the bird call dataset there are 400 samples in the training set, a batch size or 100 will use the first 100 samples to train, then with the next 100, and so on.

Finally, there are several optimizers that can be chosen when compiling your model. Optimizers are algorithms that changes the weights and learning rates to reduce loss. For these model RMSprop will be utilized.

Methods

Binary Classification Model

For the binary classification model, spectrograms for the Western Meadowlark and the Mallard were used. There was a total of 72 samples, 36 Western Meadowlark, and 36 Mallard samples. The 72 samples were split into a training set of 57 and a test set of 15. A convolution neural network was used with two layers first layer had 32 filters with convolution filter size was set at 3x3. The activation function used for this layer is ReLu. A ridge regularization was applied at a strength of 0.02. The max pooling layer specified pool size of 2,2. The second lay had 64 filter, convolution filter size of 3x3, ReLu activation, and ridge regression of 0.02. After flattening the output of the previous layers into a 1D vector, a fully connected layer is applied with 64 neurons using a ReLu activation for this layer. A dropout
of 20% is executed before the final layer with a single output neuron and sigmoid activation function. A batch size of 5 and an epoch size of 15 was utilized.

Multiclass Classification Model

For the multiclass model, all 579 samples from the 12 bird species were used. The samples were split to create a training set composed of 463 samples and a test set of 116 samples. There are four 2D convolution layers, with filters starting at 32 doubling at each layer, the last 2D convolution layer had 256 filters. Filter size was set 3x3 for each layer. A L2 regularizer was used, the penalty term set at 0.01 and the dropout rate is 30%. ReLu activation function was used at each layer. The output of the last 2D convolution layer was flattened, passed through a fully connected layer with 64 neurons, 30% of that output was dropped and passed through the last layer with an Softmax activation function to classify the 12 different bird species.

Other models were also tested, models that varied in number of layers, l2 and l1 penalty terms, dropout rates, and optimizer. Below are examples of other models tested that achieved lower accuracy rates.

Model accuracy: .5845

```
# Define the CNN
model_multi4 = Sequential([
    Conv2D(32, (3, 3), activation='relu', input_shape=(256, 343, 1)),
    MaxPooling2D(pool_size=(2,2)),
    Dropout(0.1),
    Conv2D(filters=64, kernel_size=(3,3),activation='relu', kernel_regularizer = l2(0.1)),
    MaxPooling2D(pool_size=(2,2)),
    Dropout(0.1),
    Conv2D(filters=128, kernel_size=(3,3), activation='relu', kernel_regularizer = l2(0.1)),
    MaxPooling2D(pool_size=(2,2)),
    Dropout(0.1),
    Conv2D(filters=256, kernel_size=(3,3),activation='relu', kernel_regularizer = l2(0.1)),
    MaxPooling2D(pool_size=(2,2)),
    Dropout(0.1),
    Flatten(),
    Dense(64, activation='relu'),
    Dropout(0.4),
    Dense(12, activation='softmax')
])

# Compile
model_multi4.compile(optimizer='adam',
              loss='categorical_crossentropy',
              metrics=['accuracy'])

# Train
history = model_multi4.fit(X_train, y_train, epochs=35, batch_size=50, validation_split=0.2)

# Evaluate
test_loss, test_acc = model_multi4.evaluate(X_test, y_test)
print('Test accuracy:', test_acc)
```

Figure 9 Test Model Example

Model accuracy: 0.606

```
# Define the CNN
model_multi4 = Sequential([
    Conv2D(32, (3, 3), activation='relu', input_shape=(256, 343, 1)),
    MaxPooling2D(pool_size=(2,2)),
    Dropout(0.1),
    Conv2D(filters=64, kernel_size=(3,3),activation='relu', kernel_regularizer = l2(0.2)),
    MaxPooling2D(pool_size=(2,2)),
    Dropout(0.1),
    Conv2D(filters=128, kernel_size=(3,3), activation='relu', kernel_regularizer = l2(0.2)),
    MaxPooling2D(pool_size=(2,2)),
    Dropout(0.1),
    Flatten(),
    Dense(64, activation='relu'),
    Dropout(0.4),
    Dense(12, activation='softmax')
])

# Compile
model_multi4.compile(optimizer='adam',
              loss='categorical_crossentropy',
              metrics=['accuracy'])

# Train
history = model_multi4.fit(X_train, y_train, epochs=35, batch_size=50, validation_split=0.2)

# Evaluate
test_loss, test_acc = model_multi4.evaluate(X_test, y_test)
print('Test accuracy:', test_acc)
```

Figure 10 Test Model Example #2

Spectrograms

A second testing set was also used to test model accuracy, this was separate from the testing set used to train the model. The audio files used were three mp3 files of bird sounds. The

audio files varied in length. To preprocess these audio file, they were first converted into spectrograms slowed down to the 22050 Hz. The audio files were split into 2 second spectrograms. The first audio file produced 11 spectrograms, the second file produced 2 spectrograms, and the third file produced 6 spectrograms.

Results

Binary

The binary model was able to differentiate between the meadowlark and mallard with an accuracy rate of 93.3%. It predicted the mallard at a rate of 100% and the meadowlark at a rate of 80%. Figure 9 is the confusion matrix for the prediction.

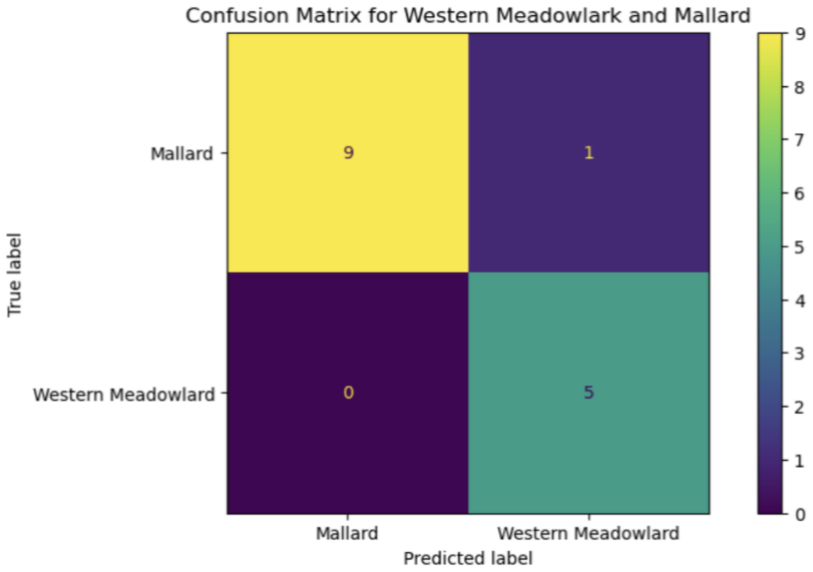

Figure 11 Binary Classification Model

Below are the learning plots for this model. The epoch number was chosen based on where the loss and validation loss begin to plateau.

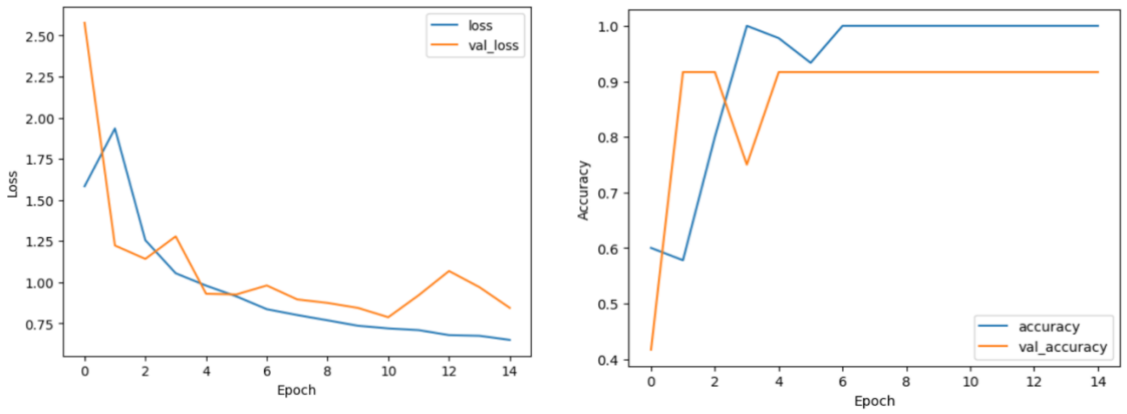

Figure 12 Learning plots

Multiclass Classification Model

The most effective model had the following parameters, 2 convolution layers, first had 32 filters, the second 64, and the third128 filters. All filters were set to size 3x3. The activation function used was ReLu and ridge regression with a penalty term of .1 at each layer. A 2x2 max pooling layer was used after each convolution layer, and there was a 10% dropout. The output is then flattened and then passed through fully connected layer using the ReLu, dropout of 40% and the final layer uses a Softmax function outputting probabilities for all 12 species. There were 35 epochs and a batch size of 50.

The numbers correspond to these bird species:

0. American crow
1. Barn swallow
2. Black-capped chickadee
3. Blue jay
4. Dark-eyed junco
5. House-finch
6. Mallard
7. Northern flicker
8. Red-winged blackbird
9. Stellar's jay
10. Western meadowlark
11. white-crowned sparrow

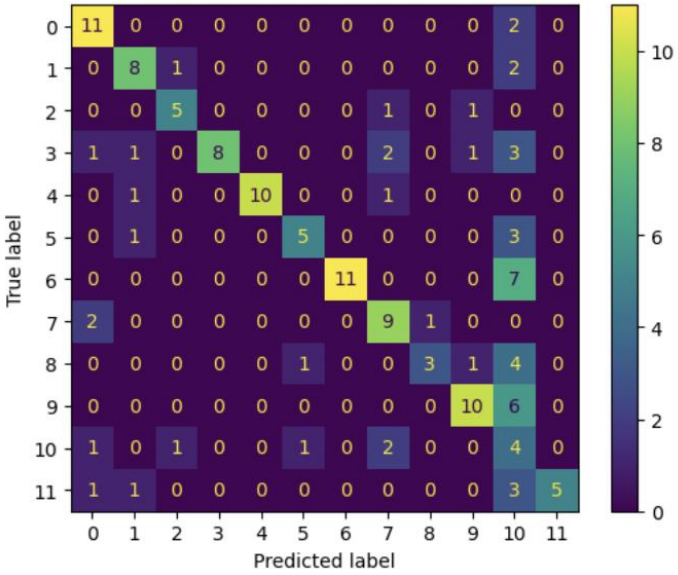

Figure 13 Multi-Class Model

This model had an accuracy rate of .6267. This model was used to predict the second test set of 3 unknown bird species.

3 Test Spectrograms

The following are the results of using the multiclass model to predict the 3 unknown bird calls. Figure 12-14 are examples spectrograms, one from each audio file.

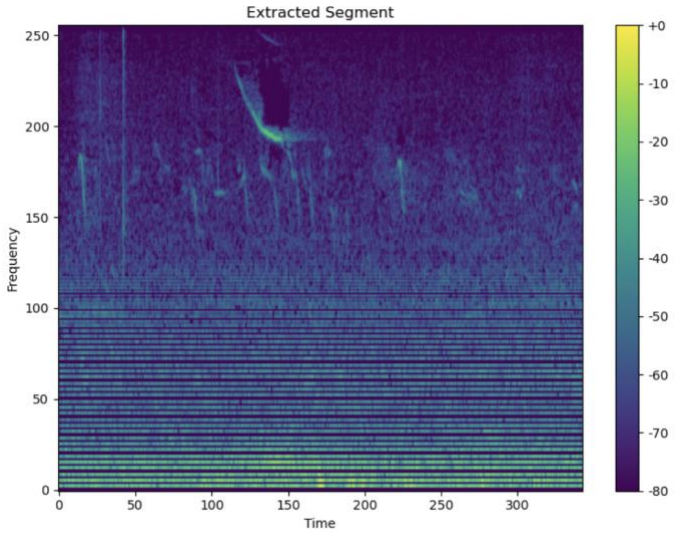

Figure 14 Test 1

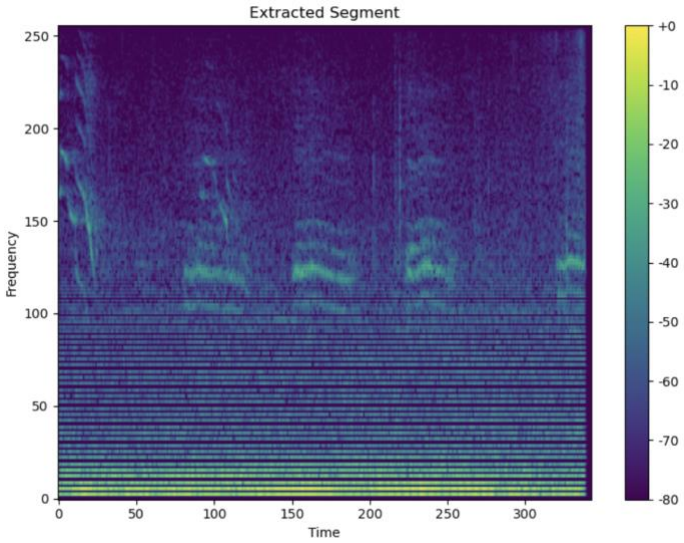

Figure 15 Test 2

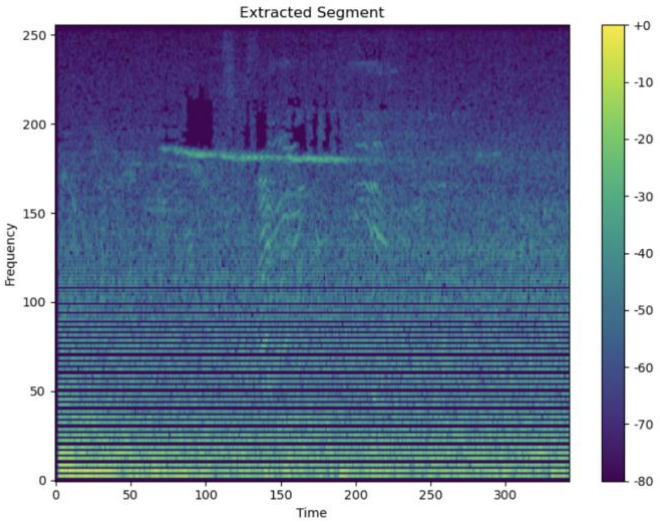

Figure 16 Test 3

Audio File # 1

	1	2	3	4	5	6
0. American crow	0.047	0.023	0.028	0.046	0.016	0.04
1. Barn swallow	0.018	0.002	0.008	0.006	0.004	0.012
2. Black-capped chickadee	0.004	0.003	0.005	0.004	0.002	0.012
3. Blue jay	0.028	0.011	0.015	0.009	0.009	0.026
4. Dark-eyed junco	0.028	0.014	0.018	0.027	0.017	0.034
5. House-finch	0.041	0.015	0.031	0.033	0.039	0.042
6. Mallard	0.233	0.125	0.197	0.18	0.138	0.165
7. Northern flicker	0.006	0.004	0.008	0.006	0.003	0.01
8. Red-winged blackbird	0.042	0.017	0.047	0.061	0.029	0.086
9. Stellar's jay	0.116	0.279	0.197	0.245	0.418	0.125
10. Western meadowlark	0.408	0.499	0.417	0.368	0.303	0.43
11. White-crowned sparrow	0.029	0.009	0.029	0.015	0.023	0.017

Table 1Probabilities, Audio File 1, Samples 1-6

	7	8	9	10	11
0. American crow	0.028	0.036	0.02	0	0.004
1. Barn swallow	0.011	0.002	0.006	0	0

2. Black-capped chickadee	0.008	0.038	0.003	0.001	0
3. Blue jay	0.02	0.043	0.015	0	0
4. Dark-eyed junco	0.022	0.017	0.016	0	0.003
5. House-finch	0.044	0.01	0.031	0	0.003
6. Mallard	0.151	0.226	0.12	0.999	0.04
7. Northern flicker	0.008	0.018	0.004	0	0
8. Red-winged blackbird	0.062	0.045	0.047	0	0.003
9. Stellar's jay	0.154	0.11	0.193	0	0.793
10. Western meadowlark	0.47	0.443	0.533	0	0.154
11. White-crowned sparrow	0.022	0.011	0.013	0	0.001

Table 2 Probabilities, Audio File 1, Sample 7-11

Audio File # 2

	1	2
0. American crow	0	0
1. Barn swallow	0	0
2. Black-capped chickadee	0	0
3. Blue jay	0	0
4. Dark-eyed junco	0	0
5. House-finch	0	0
6. Mallard	0	0.006
7. Northern flicker	0	0
8. Red-winged blackbird	0	0
9. Stellar's jay	0.965	0.994
10. Western meadowlark	0.035	0
11. White-crowned sparrow	0	0

Table 3 Probabilities, Audio File # 2, Samples 1-2

Audio File # 3

	1	2	3	4	5	6
0. American crow	0.002	0.018	0.015	0.019	0.067	0.058
1. Barn swallow	0	0.11	0.007	0.002	0.029	0.022
2. Black-capped chickadee	0	0.006	0.009	0	0.011	0.017
3. Blue jay	0	0.014	0.007	0.005	0.036	0.041
4. Dark-eyed junco	0.001	0.024	0.011	0.006	0.041	0.038
5. House-finch	0.002	0.058	0.017	0.015	0.056	0.053
6. Mallard	0.018	0.165	0.422	0.175	0.2	0.218
7. Northern flicker	0	0.007	0.004	0.002	0.012	0.19
8. Red-winged blackbird	0.003	0.057	0.049	0.016	0.102	0.081
9. Stellar's jay	0.968	0.318	0.052	0.599	0.128	0.132
10. Western meadowlark	0.004	0.293	0.389	0.14	0.291	0.291
11. White-crowned sparrow	0.003	0.29	0.018	0.02	0.029	0.031

Table 4 Probabilities, Audio File 3, Samples 1-6

Discussion

The binary model produced high accuracy rates but can only be used to differentiate between the two species. Training the model took little time, usually less than a minute. Long training times was the major limitation in training the multi-class model. Depending on the parameters chosen the training time could be 20-30 minutes for one model. This made it challenging to test and compare different values for the many parameters. The number of different combinations of parameters is large. The learning plots were sometimes useful in deciding what parameters to change. If validation accuracy is significantly lower than the training accuracy, that could indicate overfitting. To remedy overfitting using different dropout rates and regularization kernels can be effective. Where the training loss and validation loss

plateaus indicates what epoch is sufficient. Models with lower complexity performed similarly and sometimes better than models with many layers. For one of the tested models, adding 2 more convolution layer to our final multiclass model doubling the number of filters at each layer but keeping all other hyperparameters the same performed at a rate of 58.45%.

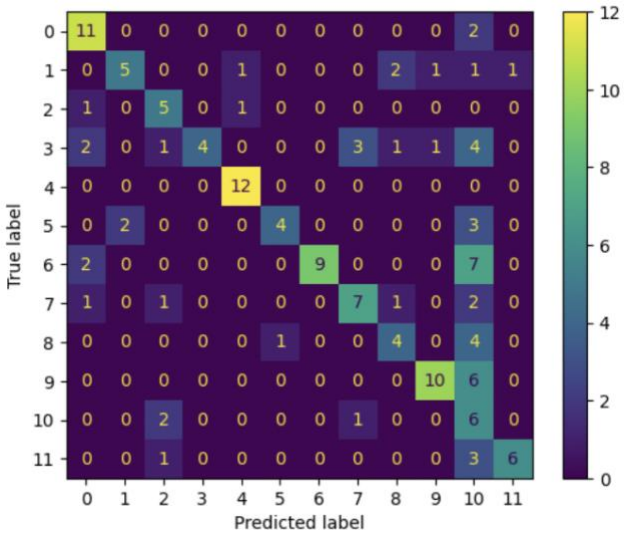

Figure 17 Model with add layers

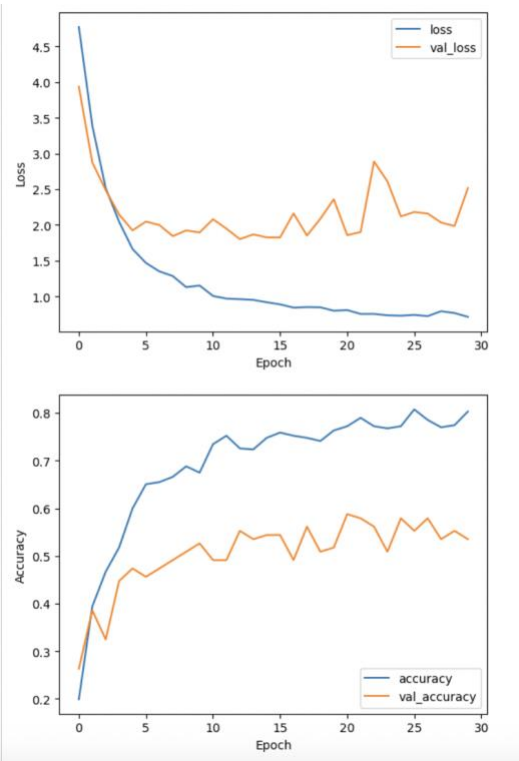

Figure 18 Learning Plots

Across all models the western meadowlark had the lowest accuracy rates, the American crow, the black-capped chickadee, and dark-eyed Junco had the highest accuracy rates.
Figure 16 – 20 is a spectrogram from these species, the colors represent intensity. Looking at the spectrograms the birds with higher accuracy rates had much more consistent and distinct patters. Unlike the binary model the multiclass model frequently confused the Western Meadowlark as a Mallard. The patterns on the spectrograms are distinct from each other, when listening to the audio file it is obvious the difference between the meadowlark and the mallard duck. However, the intensity and frequencies and the times are similar enough that the model could confuse them, especially if the model is not optimally trained.

Western Meadowlark

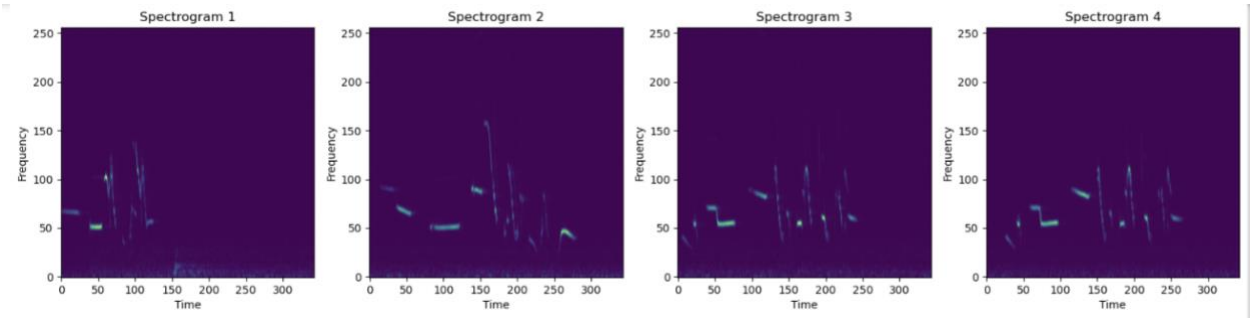

Figure 19 Western Meadowlark

Mallard

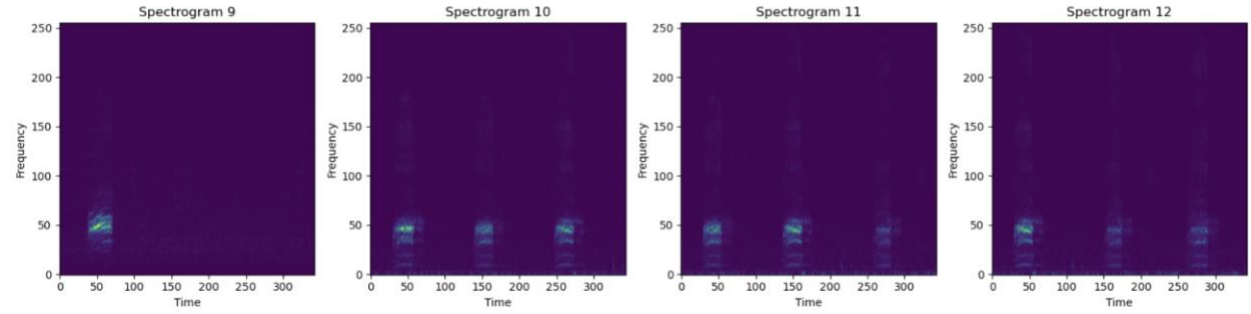

Figure 20 Mallard

American crow

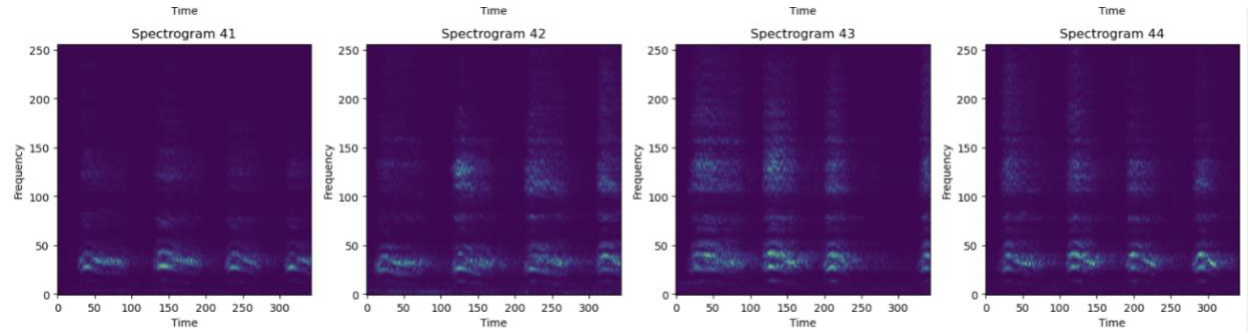

Figure 21 American Crow

Black-capped Chickadee

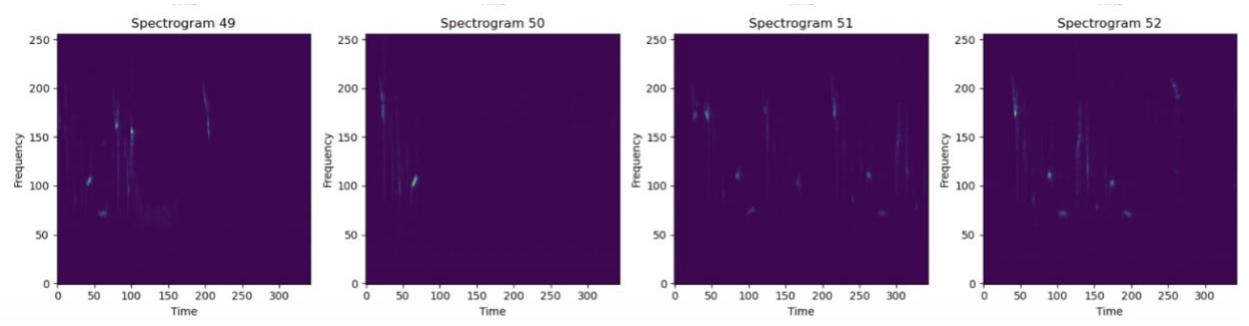

Figure 22 Black-capped Chickadee

Dark-eye Junco

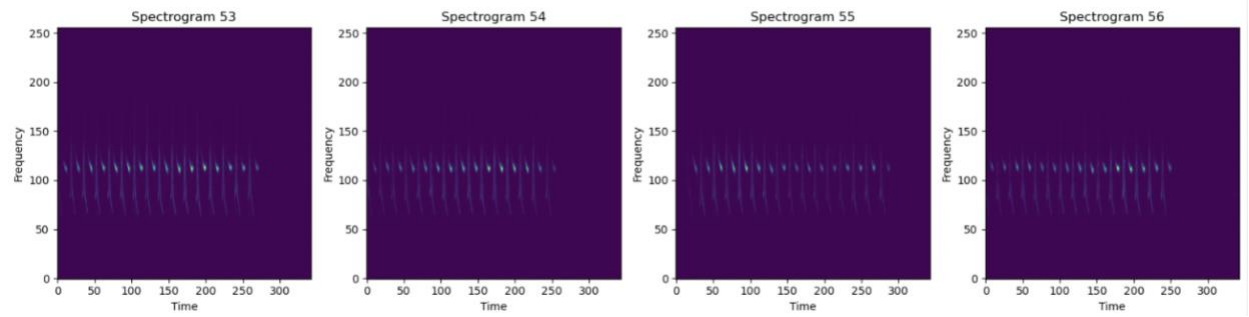

Figure 23 Dark-eye Junco

Examining the probabilities produced when using the model on the 3 unknown bird calls, it is possible that it detected more than one bird call. The probabilities for multiple birds were similar. Another major limitation was inability of further process the audio files. When listening to the audio files, one can hear multiple bird call simultaneously and there were also other sounds present in the audio files that made it difficult to isolate bird calls. In one audio file there is talking that is much louder than the birdcalls in the background. Comparing the spectrogram of

the unknown bird calls to the known bird calls the spectrograms does resemble some of the known birds. Moving ahead it would be useful to have greater computing power to test more model. Ideally, we could cross validate the different parameters to find the ideal model, but the number of combinations can get large so greater computing power would be necessary to do this. Other pre-trained models could also be used for this task. One team of researchers used the popular image classifier ResNet-50 for bird call recognition [10]. Their model was able to achieve 60%-72% accuracy rates.

Conclusion

The objective of this project was to build a model that could distinguish bird calls of 12 different species of birds. The convolution neural network was trained using spectrograms provided by Xeno-Canto was able to perform with an accuracy rate of 62.67%. A binary model was also trained to distinguish between two species the Western Meadowlark and the Mallard duck, it performed at an accuracy rate of 93.33%. The model also predicted 3 unknown bird calls. This model is a good start, but due to the limitations of inadequate computing power it was not possible to cross validate efficiently for the right parameters. Other models such as ResNet-50 and more complex models with more layers have produced good results for bird recognition. Creating a good bird call recognition would not only be an enjoyable tool for the hobby bird watcher it can also serve many other purposes as well. It can give researchers a tool to study bird population, track migration patterns, and evaluate the overall health of an ecosystem.

References

[1] Dhar, Ayan Kumar. “Understanding Activation Functions and Hidden Layers in Neural Networks.” Medium. Feb 6, 2020. https://medium.com/analytics-vidhya/understanding-activation-functions-and-hidden-layers-in-neural-networks-4fca2b980917

[2] Bownlee, Jason. “A Gentle Introduction to the Rectified Linear Unit (ReLu).” Machine Learning Mastery. Aug 20, 2020. https://machinelearningmastery.com/rectified-linear-activation-function-for-deep-learning-neural-networks/

[3] Sanderson, Grant. “Neural Network.” 3blue1brown. Oct 5, 2017. https://www.3blue1brown.com/lessons/neural-networks

[4] Ormesher, Ian. “Convolution Filters.” May 9, 2020. https://medium.com/@ianormy/convolution-filters-4971820e851f

[5] Sahoo, Sabyasachi. “Deciding optimal kernel size for CNN.” August 19, 2018. https://towardsdatascience.com/deciding-optimal-filter-size-for-cnns-d6f7b56f9363

[6] James, Gareth. Witten, Daniela. Hastie, Trevor. Tibshirani, Robert. An Introduction to Statistical Learning with application in R. New York, Springer Scienc+Business Media, 2013.

[7] Permana, Silvester Dian Handy. Rahim, Robbi. Saputra, Gusti. “Classification of Bird Sounds as an Early Warning Method of Forest Fires using Convolutional Neural Network (CNN) Algorithm.” May 2021. https://www.researchgate.net/publication/351564871_Classification_of_Bird_Sounds_as_an_Early_Warning_Method_of_Forest_Fires_using_Convolutional_Neural_Network_CNN_Algorithm

[8] Xeno-Canto. Bird Call Spectrograms. https://xeno-canto.org/explore/taxonomy?grp=birds

[9] James, Gareth. Witten, Daniela. Hastie, Trevor. Tibshirani, Robert. An Introduction to Statistical Learning with application in R. New York, Springer Scienc+Business Media, 2013.

[10] Sankupellay, Mangalam. Konovalov, Dmitry. Bird Call Recognition using Deep Convolution Neural Network and ResNet-50. October 2018.

https://www.researchgate.net/publication/328418948_Bird_Call_Recognition_using_Deep_Convolutional_Neural_Network_ResNet-50