Bird Species Classification Using Convolutional Neural Networks

Brendon Bottle and Sean McManus

Problem Statement

The primary objective of our project is to train a CNN model that can accurately classify bird species based on the visual features present in the images. The model will need to successfully differentiate between 20 species of birds. We want to design and implement a CNN that will bring us as close as possible to the baseline model recommended by the authors of the dataset.

Problem Set Up

The 20 bird species used for this project are a random selection of the 525 species in this Kaggle dataset. They include larger birds such as herons and eagles as well as smaller birds such as barbets and quails.

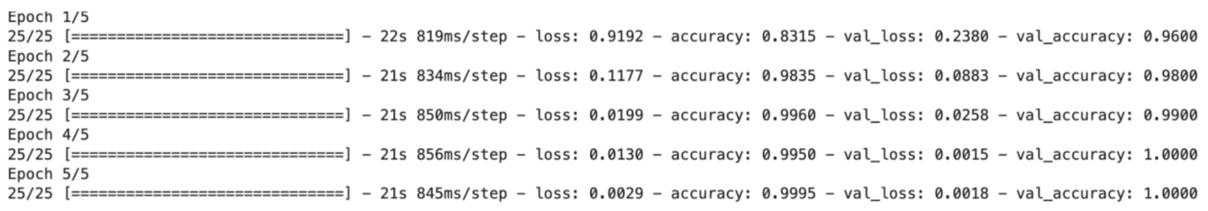
The training data includes 100 images per species. The data author also provided a hand selected validation and test set, each with 5 images per species. The size of each image is 224x224 and contains only a single bird per image that comprises at least 50% of the pixels in the image.

The training dataset contains more male birds than female birds, giving it a bias towards the males, which tend to be more colorful and decorative, however the train and validation contains both sex's.

Model Implementation

Baseline Pretrained Model

As a benchmark for how well this problem could be solved by a neural net, an EfficeintNetB0 pretrained model suggested by the authors of the dataset was used as a baseline. The weights are initialized to ImageNet weights. The model achieved an accuracy of 100% on the validation data after only 4 epochs. This indicated that this problem was well suited to a neural network architecture.

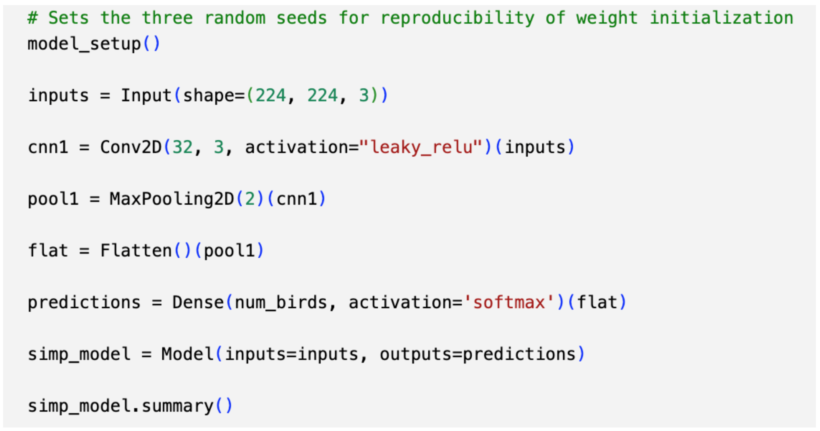


```
Epoch 1/5
25/25 [==============================] - 22s 819ms/step - loss: 0.9192 - accuracy: 0.8315 - val_loss: 0.2380 - val_accuracy: 0.9600
Epoch 2/5
25/25 [==============================] - 21s 834ms/step - loss: 0.1177 - accuracy: 0.9835 - val_loss: 0.0883 - val_accuracy: 0.9800
Epoch 3/5
25/25 [==============================] - 21s 850ms/step - loss: 0.0199 - accuracy: 0.9960 - val_loss: 0.0258 - val_accuracy: 0.9900
Epoch 4/5
25/25 [==============================] - 21s 856ms/step - loss: 0.0130 - accuracy: 0.9950 - val_loss: 0.0015 - val_accuracy: 1.0000
Epoch 5/5
25/25 [==============================] - 21s 845ms/step - loss: 0.0029 - accuracy: 0.9995 - val_loss: 0.0018 - val_accuracy: 1.0000
```

Because the EfficeintNetB0 model performed suspiciously well, it's worth taking a look at some of the model architecture to get an understanding of what techniques are employed. EfficientNet offers a unique solution to try and increase accuracy. As of this paper, it was common to scale networks by increasing one of these three components: depth, width, or image-resolution. EfficientNet increases the three components simultaneously by a constant ratio. This is done with a Mobile Inverted Bottleneck Convolutional layer. While this isn't something we are going to attempt in this project, we unknowingly employed some of the same ideas, attempting a re-scaling of the images, increasing model depth and model width, although ours were not performed simultaneously.

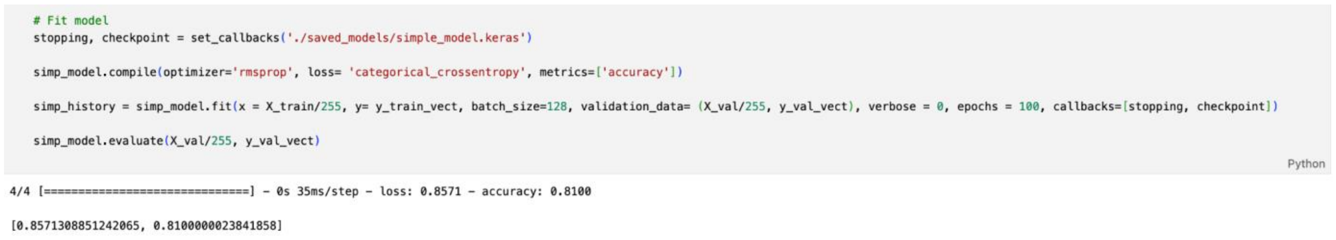
Initial Model

To evaluate a starting point for the model, and assess the classes that were performing well, an initial model was trained with a single convolution layer, followed by a pooling layer, and an output given by a fully connected layer using a softmax activation function.



```
# Sets the three random seeds for reproducibility of weight initialization
model_setup()

inputs = Input(shape=(224, 224, 3))

cnn1 = Conv2D(32, 3, activation="leaky_relu")(inputs)

pool1 = MaxPooling2D(2)(cnn1)

flat = Flatten()(pool1)

predictions = Dense(num_birds, activation='softmax')(flat)

simp_model = Model(inputs=inputs, outputs=predictions)

simp_model.summary()
```

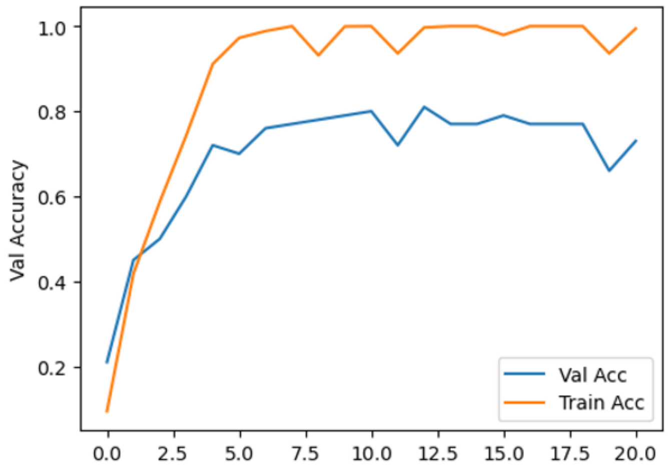
We implemented early stopping as this model isn’t finely tuned and we would not expect accuracy to increase given more epochs. The model achieved a validation accuracy of 81% and early stopping ended the training after 21 epochs.



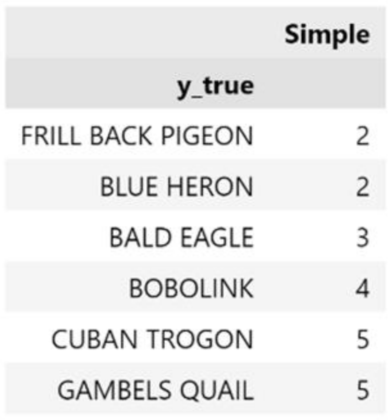
```python
# Fit model
stopping, checkpoint = set_callbacks('./saved_models/simple_model.keras')

simp_model.compile(optimizer='rmsprop', loss= 'categorical_crossentropy', metrics=['accuracy'])

simp_history = simp_model.fit(x = X_train/255, y= y_train_vect, batch_size=128, validation_data= (X_val/255, y_val_vect), verbose = 0, epochs = 100, callbacks=[stopping, checkpoint])

simp_model.evaluate(X_val/255, y_val_vect)
```

```
4/4 [==============================] - 0s 35ms/step - loss: 0.8571 - accuracy: 0.8100
[0.8571308851242065, 0.8100000023841858]
```

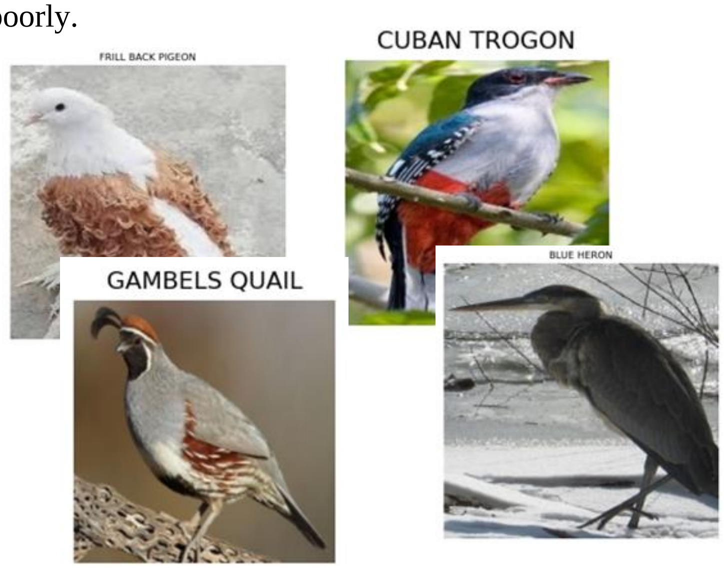
When the training and validation accuracy are plotted alongside one another we notice a few trends that led us to believe our simple model was overfitting. The validation accuracy had plateaued around 6 epochs, and started on a downward trend at the later epochs.



Reviewing the classes that were predicted well compared to the classes that were predicted poorly, a few patterns jumped out. Birds with very distinct patches of color or distinctive feathering were being classified very well. Birds with muted colors or a lot of variation between images were being classified very poorly.



	Simple
y_true	
FRILL BACK PIGEON	2
BLUE HERON	2
BALD EAGLE	3
BOBOLINK	4
CUBAN TROGON	5
GAMBELS QUAIL	5



This indicated that the model was picking up the very prominent features, but wasn't identifying more nuanced features needed for the less distinctive birds.

Model Refinement

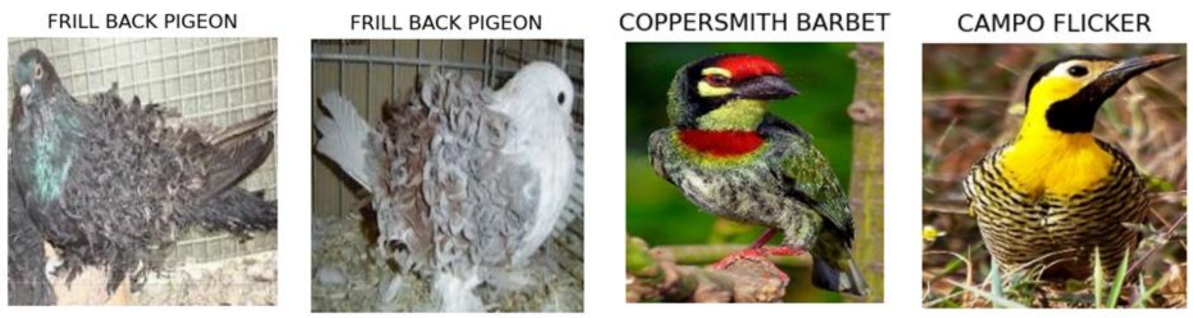
Increased Complexity

To allow the model to better evaluate the features in the images, we increased the complexity of the model. To do this, we introduced two additional convolutional layers to increase feature extraction and differentiation. To support model invariance, we also included MaxPooling layers after each convolutional layer. This also served to help minimize the time to train the model.

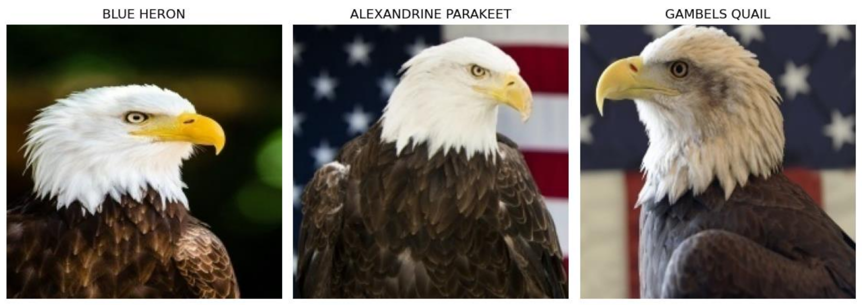
```
comp_model = Sequential()

comp_model.add(Conv2D(32, 3, activation='leaky_relu', input_shape=(224, 224, 3)))

comp_model.add(MaxPooling2D(2))

comp_model.add(Conv2D(32, 3, activation='leaky_relu'))

comp_model.add(MaxPooling2D(2))

comp_model.add(Conv2D(32, 3, activation='leaky_relu'))

comp_model.add(MaxPooling2D(2))

comp_model.add(Flatten())

comp_model.add(Dense(num_birds, activation='softmax'))

comp_model.summary()
```

This more complex model produced mixed results for our class predictions. While some species had higher levels of classification, like the Frill Back Pigeon, other species had significant decreases in their classifications including the Bald Eagle and Bald Ibis.

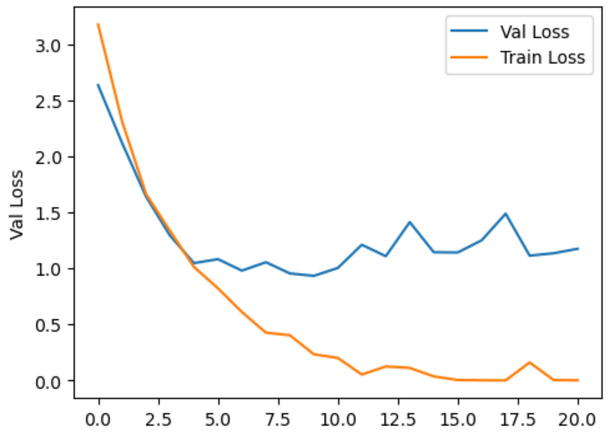
Looking at the birds that improved, they tended to follow the same pattern as seen in the simple model. Birds like the Coppersmith Barbet and Campo Flicker had very bright and distinct color patterns. However, as evidenced by the improvement of the Frill Back Pigeon, it did seem like the additional complexity was allowing the model to identify some additional features that helped it predict birds that had significant variation in coloring.

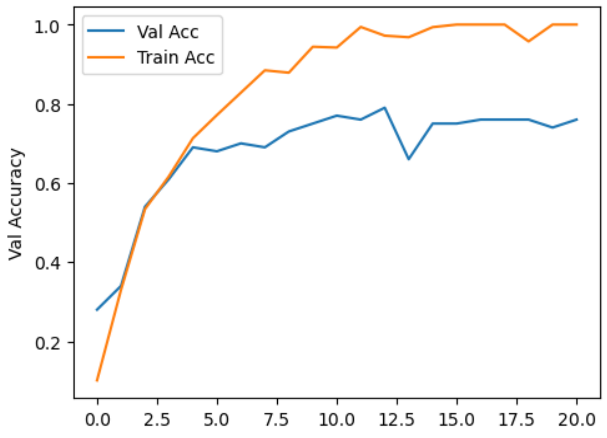


Looking at the Bald Eagle, which had performed much worse on the complex model, it was hard to determine exactly why the model had predicted the classes that it did. While there were some color similarities, there didn't seem to be a clear feature that would have resulted in these predictions. This seemed to indicate that the complex model was overfitting to some noise or some feature that wouldn't be clear to a human observer.



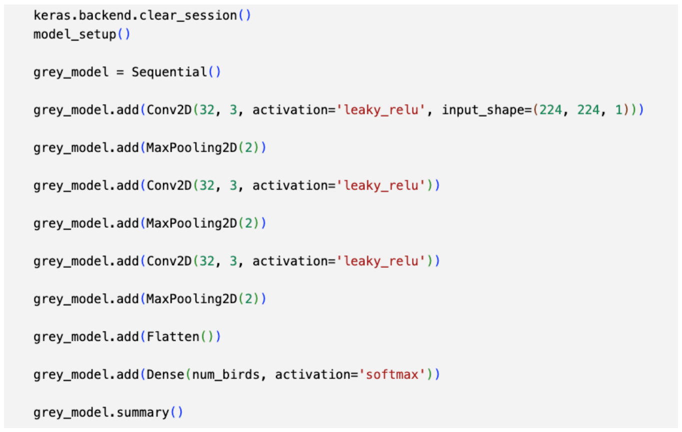
One other problem with this new model was that it appeared to be overfitting more than the simple model. Accuracy was still plateauing early, and the loss appeared to be on an upward trend in later epochs. This was not surprising, since more complex models are more likely to overfit, and we hadn't focused on methods to reduce overfitting.





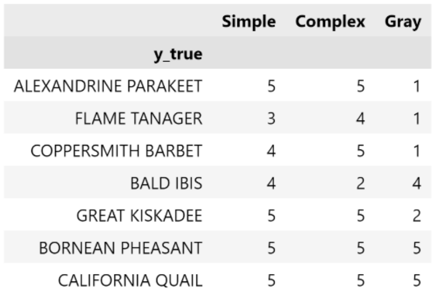
Grayscale Image Evaluation

To validate the assumption that the model was relying heavily on color, we trained a model on gray scale images instead of color. The greyscale model architecture was the same as the complex model, with the exception of the input size, as it now had just a single-color channel.

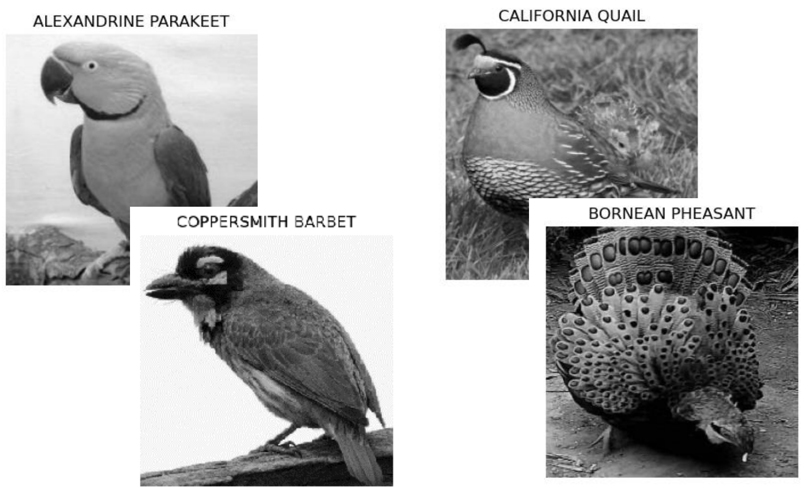


```
keras.backend.clear_session()
model_setup()

grey_model = Sequential()

grey_model.add(Conv2D(32, 3, activation='leaky_relu', input_shape=(224, 224, 1)))

grey_model.add(MaxPooling2D(2))

grey_model.add(Conv2D(32, 3, activation='leaky_relu'))

grey_model.add(MaxPooling2D(2))

grey_model.add(Conv2D(32, 3, activation='leaky_relu'))

grey_model.add(MaxPooling2D(2))

grey_model.add(Flatten())

grey_model.add(Dense(num_birds, activation='softmax'))

grey_model.summary()
```

Evaluating these results showed a clear trend that very colorful birds that had been well classified in the model using color, were now being poorly classified. However, birds with more distinctive feathering that had been classified well previously were still being classified well.



	Simple	Complex	Gray
y_true			
ALEXANDRINE PARAKEET	5	5	1
FLAME TANAGER	3	4	1
COPPERSMITH BARBET	4	5	1
BALD IBIS	4	2	4
GREAT KISKADEE	5	5	2
BORNEAN PHEASANT	5	5	5
CALIFORNIA QUAIL	5	5	5



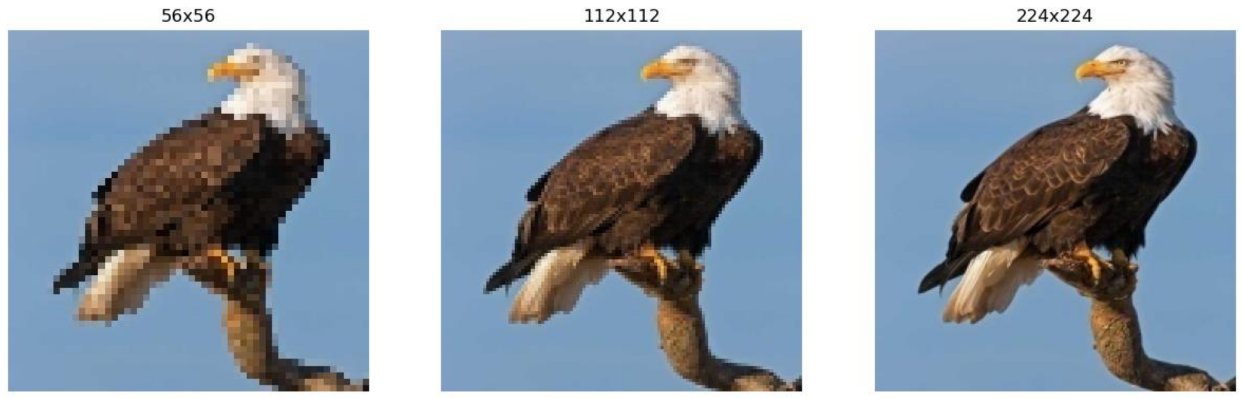
Progressive Resizing

Because our original images were relatively high quality, the model was able to easily identify the highly distinct features for certain species and was focusing on those to make its predictions while not learning enough about birds that were less distinct or had a lot of variation in color and position. To force the model to learn different features, we employed a method called progressive resizing.

By training the model on lower quality images and then adding new layers trained on higher quality images, it can limit the impact of noise learned in the early layers on the weights of the later layers. In our case, we hoped that by downscaling the images, the model would rely less on some of the distinctive features like the sharp-edged red patch of the Cuban Trogon and be forced to find other ways of identifying birds.

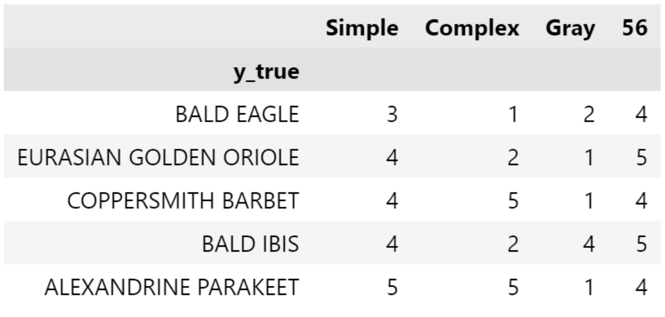
We used the complex model built in the previous step as our base model. Because we were reducing the quality of the images, we increased model complexity by increasing filter sizes to 128, 64, and 32 on three convolutional layers. We also included a second Dense layer before the softmax to include additional differentiation of the images.

To combat overfitting, we added a dropout layer with a rate of 20%, this allowed us to train for more epochs with the hope of combatting any overfitting.



56x56

We trained the new base model on images of size 56x56. This model performed extremely well just on its own. The accuracy jumped to 88.99% and several birds that had been poorly classified on the complex model, such as the Bald Eagle and Bald Ibis were being 100% correctly classified. While there were a few classes that were being misclassified more often than in the initial complex model, overall this model performed much better. We believe this is because of that initial layer's ability to look past the noise/background and pick out features of the bird at a higher rate.



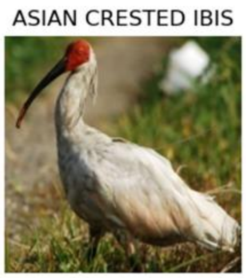
	Simple	Complex	Gray	56
y_true				
BALD EAGLE	3	1	2	4
EURASIAN GOLDEN ORIOLE	4	2	1	5
COPPERSMITH BARBET	4	5	1	4
BALD IBIS	4	2	4	5
ALEXANDRINE PARAKEET	5	5	1	4

112x112

An additional CNN layer with input size of 112x112 was then added to the model. To avoid picking up too much noise from the higher quality images, filter size was kept at 32 for this new layer. A MaxPooling layer was used to reduce the size to 56x56 for the previous model.

Accuracy for this model increased further to 89.99%. With the exception of the Campo Flicker, all classes were either better predicted or remained the same.

	Simple	Complex	Gray	56	112
y_true					
FRILL BACK PIGEON	2	4	3	4	5
BALD EAGLE	3	1	2	4	5
CAMPO FLICKER	4	5	2	5	4
ASIAN CRESTED IBIS	5	4	4	5	5
ALEXANDRINE PARAKEET	5	5	1	4	4

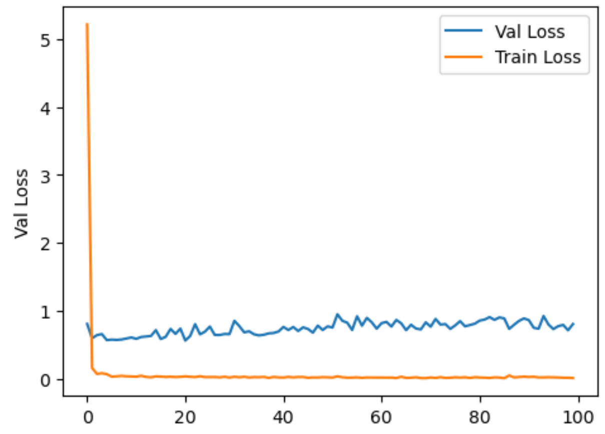


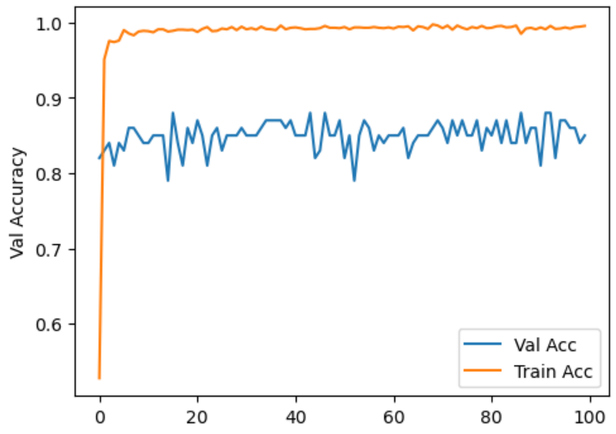
224x224

A final CNN layer was added with a filter size of 16 and an input shape of 224x224. While loss improved on this model, the overall accuracy declined. Several birds had their class accuracy decrease with only one class, Blue Heron, increasing.

	Simple	Complex	Gray	56	112	224
y_true						
BLUE HERON	2	3	3	4	4	5
BALD IBIS	4	2	4	5	5	4
BALD EAGLE	3	1	2	4	5	4
GAMBELS QUAIL	5	5	4	4	4	4
BOBOLINK	4	3	3	4	4	4

One benefit of this process was that the model was not overfitting as severely as before, with validation staying relatively steady. By locking the weights of the later layers, the model was less likely to overlearn features presented in the earlier pictures.



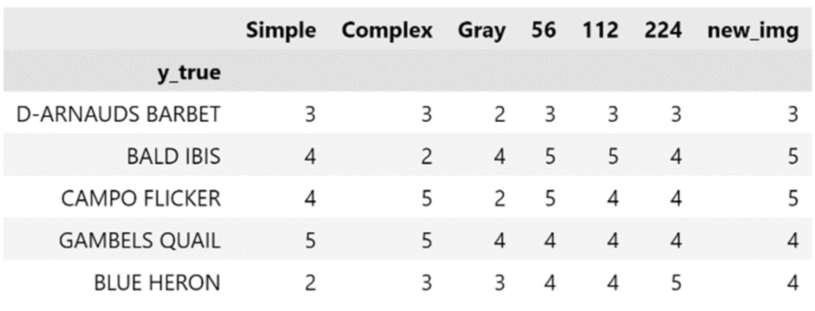


Additional Images

In reviewing the results of the progressive resizing models, we noticed that the Cedar Waxwing and the D-Arnaud's Barbet were consistently poorly classified across all three steps of the model. We decided to add more images of these two birds to the training dataset and see if providing the model with more training data would help boost these two classes and improve overall accuracy.

We reinitialized the 224x224 model with new weights on the 224x224 layer to retrain it on the additional information. We also unlocked the pre-trained weights of the 112x112 layer. The goal was to allow the 224x224 layer to learn different features for these birds while allowing the 112x112 layer to retain the benefit of its previous predictions but gain more insight on them.

This adjustment improved our model by boosting accuracy to 91% and reducing loss to .3924. While it was still doing a poor job on D-Arnaud's Barbet, it did improve classifications for Cedar Waxwing, and it also was now predicting Bald Ibis and Campo Flicker better.



	Simple	Complex	Gray	56	112	224	new_img
y_true							
D-ARNAUDS BARBET	3	3	2	3	3	3	3
BALD IBIS	4	2	4	5	5	4	5
CAMPO FLICKER	4	5	2	5	4	4	5
GAMBELS QUAIL	5	5	4	4	4	4	4
BLUE HERON	2	3	3	4	4	5	4

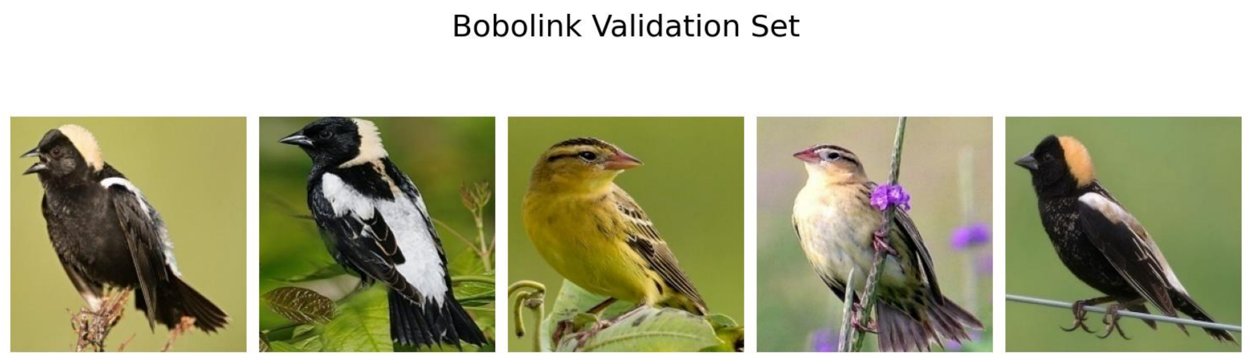
Test Results

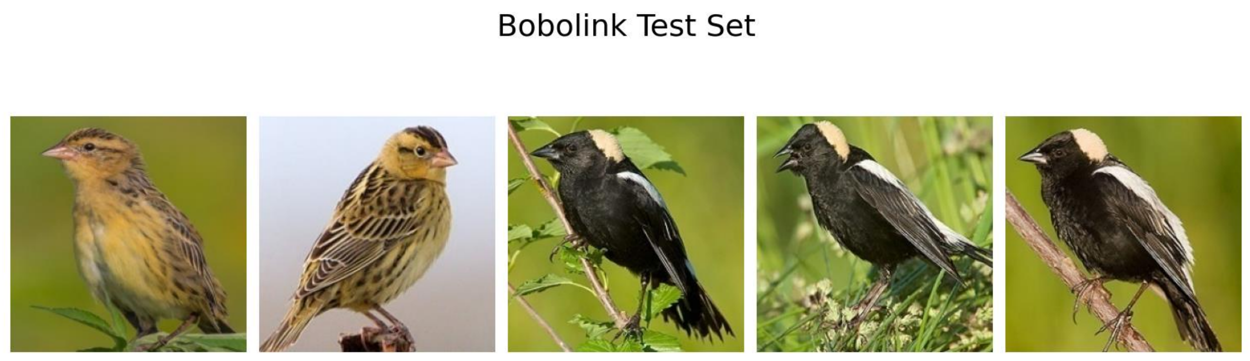
With a 10% point improvement in accuracy, we determined our model was ready for the final evaluation. Unfortunately the gains made through our model tuning did not translate to the test data as well as we'd hoped. Our final accuracy result was 82.99% with a .5899 loss.

Reviewing the individual classes showed that many of the very distinctive and colorful species were still doing well, but other species that had done well on the validation set, were no longer performing well including the Blue Heron and the Bobolink.

Comparing the images in the validation set for these classes against those in the test set, it appeared that there were some distinct differences between the two.

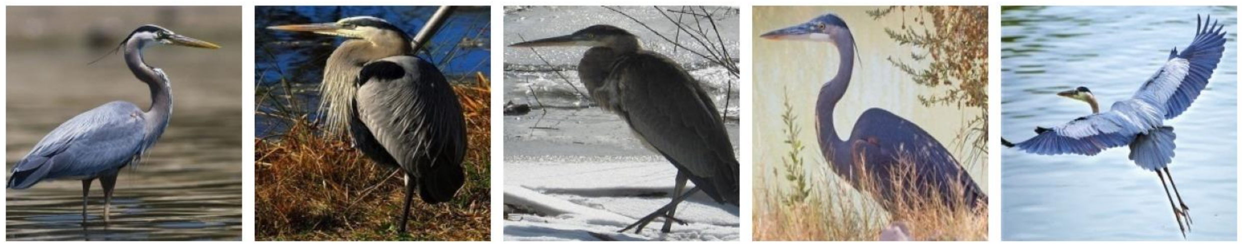
The positions of the bobolink bird in the two sets were very different. In particular, the female of the species was facing forward in the validation set, showing a yellow belly. In the test set the images showed much more of the back of the bird, which had a very distinct black and yellow pattern. It's probable that our model had never learned to recognize this bird from the back and was unable to connect it to the same species when presented with the new pattern.



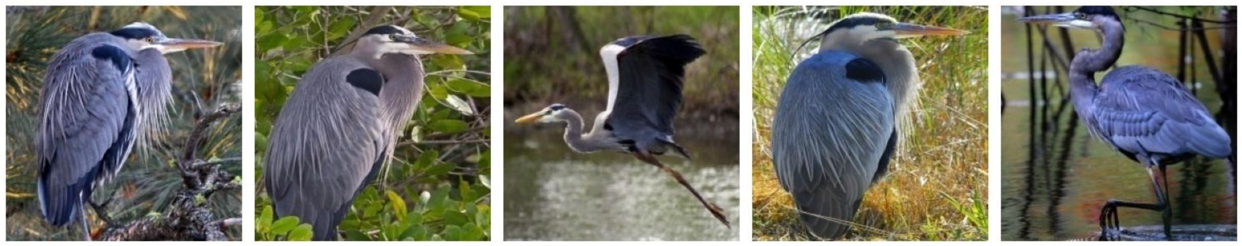


The Blue Heron appeared to be in very different positions in the two sets and appeared to have more color variation in the validation set. It’s possible our model overlearned particular positions of the heron when training against the validation set and wasn’t able to generalize to the new images in the test set.

Blue Heron Validation Set



Blue Heron Test Set



Next Steps

In reviewing the final results, we identified a few potential next steps we could have taken.

First, we identified that our validation set, with only 5 images per bird, might have been too small for us to accurately assess the model’s ability to generalize. With a more robust validation set, we believe the loss function would have been smoother and led to an increase in our models ability to identify unique patterns amongst a species as opposed to general differences between species.

Additionally, we considered that image augmentation could be a useful addition to our model. The potential position bias of some species, such as the blue heron, could indicate that our model became too focused on certain poses or positions, and that the use of flipping or shearing might have increased flexibility of the model to adapt to new images.

Finally, we recognized that our model was relatively shallow, particularly compared to something like EfficientNet. Additional complexity, balanced with the appropriate regularization, could have potentially allowed the model to identify more features for the species allowing it to recognize more variations of that bird overall.