Convolutional Neural Networks for Bird Classification: Technical Report

Sepehr Goshayeshi (KSUID: 000322936)

Kennesaw State University

Table of Contents

Motivation

When the general topic of data mining or deep learning is brought up, one useful implementation is supervised learning for visually or acoustically classifying birds. In fact, avian biomass and biodiversity can be used as an indicator of ecological health [1]. Granted, it will shock many that at least a quarter of birds from North America have died out from anthropogenic climate change and environmental degradation [2]. If researchers can utilize deep-learning-based bird detection models as a replacement for banding and human observation, it would allow for huge advancements for conservation efforts and the monitoring avian populations [3]. Human counting methods are either tinged with bias or time-consuming, giving unreliable wildlife surveys, but the use of unmanned aerial vehicles (UAV), utilizing deep-learning-based bird detection models, would provide much more efficiency [4]. Moreover, supervised-learning classification methods have the benefit to learn features from the actual data itself and outperform humans [4]. To begin, let’s understand how convolutional neural networks (CNN), also known as convnets, work. This is important to keep in mind as I implement a deep learning model for the classification of birds.

Literature Review

In supervised deep learning, one has a dataset and already knows what the correct output should look like. The neural network is initially trained on the dataset, so the inputs' weights are tuned to predict the outputs, which means there is a relationship between the input and output [5]. Supervised learning problems are categorized as either "regression" or "classification" problems. In "regression", the program predicts results within a continuous output whereas in "classification", one is trying to map input variables into discrete categories. Identifying birds is an example of supervised learning classification. Basically, the neural network learns to map input data to known targets based on a set of many examples [6]. This is done in order to learn useful representations of the input data that brings one closer to the expected output. It is about learning the relationship between training inputs and training targets.

CNNs are popular for the use of supervised learning classification problems. They are good at classifying objects based on either RGB-formatted images or audio streams. They can let you know the probability of the species of bird based on visual or acoustic input. Within a CNN, there are input, hidden, and output layers. In the hidden layers, you have low-level features such as pixels, edges, dots, and so forth. Within subsequent layers, you have higher-level features such as beaks, legs, and plumage. Eventually, you reach the types of birds. It is a multistage way of learning data representations [6]. This deep learning algorithm was inspired by past research into the mammalian visual cortex, which processes visual stimuli in hierarchical layers of increasing complexity [7].

In order to understand how this is done, I am going to break the explanation into four stages: convolution & normalization, pooling, fully connected layer, and optimization [8-9]. It is important to keep in mind a computer sees both RGB and HSV-formatted images expressed as a matrices of numbers. First, CNNs begin with a convolution stage, which involves learning local patterns and extracting features from the input image based on having learnable kernels [10].

Features are lined up with the input layer whereby each pixel is multiplied with the corresponding feature pixel. Eventually, they are summed and divided by the total number of pixels in the feature. This process is called filtering. Convolutional hidden layers have any number of features in them, and this process of filtering moves a kernel across the image to see if patches match or not and extracts a feature map. Typically, there are multiple feature maps, one for each convolution operation [9]. Normalization immediately occurs after convolution and typically involves the use of a rectified linear unit (ReLU). This procedure changes all of the negative values to 0. Moreover, average or max pooling involves shrinking the filtered image stack. One picks the stride and window size (e.g., 2x2 pixels), and slides this window across the filtered images. From each window, the max value is taken for max pooling or the average for average pooling. Basically, in the pooling layer, the stacks of filtered images turn into a stack of smaller images. The output, or feature map, becomes the input of the next higher-level one via transformation, and there is a process of greater number of stacked, filtered, and shrunken feature maps.

By the time of many iterations, the feature map runs through a final fully connected neural, also known dense network, which is reminiscent of multi-layer perceptron neural network (MLP). It is fully or densely connected to the output of the last layer [11]. Every feature value has a weight predicting the classification. As Kaxiras (2019) explains, "A fully [or densely] connected layer flattens the square feature map into a vector. Then one can use a sigmoid or softmax activation function to output probabilities of classes" [10].

In optimization, the fully connected layer can be trained and stacked. The final loss function (i.e., prediction of the network versus what you wanted it to output) computes a "distance score", which is used as a feedback signal to adjust the weights via a backpropagation algorithm [8]. Backpropagation iteratively computes the partial derivative of the loss function with respect to weight ($\partial J/\partial$), starting from the top layer down to bottom layers, by using the chain rule. The weights of the neurons of various layers are then updated via the algorithm of gradient descent or other variants such as RMSprop or ADAM, which can help minimize the loss function. Many iterations of training epochs are done within a training loop thereby leading to greater optimization. Most of the researchers, including this assignment's coding example, have stacks of different layers trained on datasets whereby they jointly adjusts all of their weights and layers of representations via optimization. Check "Objective" section for more information. It is up to the researchers or programmers to determine the hyperparameters of neurons in the fully connected layer, number and size of features in the convolutional layers, and pooling window size and stride. Also, it is important to note the fully connected layer begins with random weights before being optimized.

CNNs are popular compared to various other neural networks in classifying objects because they contribute to improving the efficiency of deep neural networks. One of the main advantages of CNNs would be sparse representations. When working on an image classification problem of millions of pixels in size, a traditional neural network will attempt to solve it using matrix multiplication operations, which can result in billions of computation. However, using a CNN, its algorithms would focus on relevant features of the image and, thereby, require fewer parameters, and the results would be a few billion operations smaller and more efficient [12]. Another advantage of CNNs would be that they tend to use the same parameter across different functions of the network, which results in massive savings in memory compared to traditional models [12]. CNNs are also picked more compared to other neural networks for its equivariance. CNNs are equivariant to many data transformation operations which helps us to predict how specific changes in the input will be reflected in the output [12].

Brief Literature Review I:
Bird Image Retrieval and Recognition Using a Deep Learning Platform [17]

In this study, Yo-Ping Huang and Haobijam Basanta (2019) from the National Taipei University of Technology utilized a CNN to localize prominent image features of 27 species of birds endemic to Taiwan [17]. They did this by establishing/maintaining a region of interest bounded by the shapes and colors of object granularities, adding a skip connection method to linearly combined the outputs of both previous and current layers for better feature extraction, and then applying a softmax function to obtain a probability distribution of bird features, eventually ending up with a 99.00% accuracy rating of identification.

This was done primarily to provide information to an app called “Internet of Birds”, which aids people who visit bird sanctuaries in identifying species that they may not be able to get a close enough look at but can snap a photo of. It is a crowd sourced app, which any individual can upload captured images of birds, and instantly retrieve information about the birds in question.

The researchers compiled existing images from the IoB app, along with other Internet sources for increasing the diversity of the dataset. They limited the potential for image contamination (with noise/pixels/blurred parts/et. Al), by only utilizing high-pixel Internet images with clear boundaries. They then standardized the dataset by random flipping, rotation, translation, and zero-phase component analysis whitening, along with Gaussian filtering.

The dataset was comprised of 3563 total images of the 27 bird species in question; 2280 utilized for training, 570 for validation, and 713 for testing. The utilization of a skip connection greatly increased the accuracy of the detection method, and overall assisted the app in correctly identifying the bird species in question from a variety of photographs.

Overall, the study helped to develop a mobile app platform that utilizes cloud-based deep learning to process images in order to identify different bird species, predominantly 27 species endemic to Taiwan. It is ultimately aimed at designing an automatic system for differentiating fine-grained objects among bird images with shared characteristics, but minor variations in appearance.

The system is both expandable and upgradable, with the development for predicting different generations of specific bird species in the works, along with the addition of several more bird species.

Brief Literature Review II:
Application of Deep-Learning Methods to Bird Detection Using Unmanned Aerial Vehicle Imagery[18]

In this project, the researchers Suk-Ju Hong, Yunhyeok Han, Sang-Yeon Kim, Ah-Yeong Lee, and Ghiseok Kim (2019) of Seoul National University were looking to replace the traditional counting methods currently in use by people to determine wildlife/bird populations; e.g. total ground count, line-transect count, dropping count, and aerial count [18]. They instead utilized aerial photographs collected by UAV; the dataset contained images of birds and decoys in various environments, near their habitats, and near lakes as well as farmlands. The decoys were included primarily to achieve various bird patterns and also classify bird information more accurately. Several Convolutional Neural Network types were experimented with during this process, with Faster Region-based Convolutional Neural Network (R-CNN) proving to be the most accurate, and You Only Look Once (YOLO) to be the fastest among all the models tested.

The hardware used to compile the dataset consisted of:

- A commercial UAV (K-mapper, SISTECH Inc., Seoul, Korea)

    - Dimensions: 750 mm × 750 mm × 250 mm
    - Maximum takeoff load: 4.5 kg
- Cameras for bird imaging (NX-500, Samsung) and decoy imaging (Da Jiang Innovation)
    - resolution of 6480 × 4320 pixels (Samsung) and 5472 × 3078 pixels (Da Jiang Innovation)

They collected 393 aerial photographs from which 13,986 images of birds were generated. All photos of actual birds were taken at an elevation of 100m at Shihwa Lake and Yeongjong Island, and were enlarged at places to confirm the existence of wild birds. The photos of the decoys were taken at 50m in 15 different places; on farms, in parks, reservoir areas, etc. This was to provide a more robust learning process by adding the pictures of the decoys to the dataset.

All of the images were flipped vertically and horizontally with a probability of 50%, and the RGB of the images was randomly adjusted as well. These were fed into their 5 different deep learning-based object detection methods (Faster R-CNN, R-FCN, SSD, Retinanet, and YOLO) and the efficiency was evaluated, with AP values ranging from 85.01% to 95.44% during the testing process.

The purpose of all this was to demonstrate that UAVs and deep learning could be used to replace the human element in species counts; this would be invaluable to help predict migration patterns, habitat shifts and diversity, as well as the potential spread of avian borne pathogens.

Brief Literature Review III:
Deep Learning Case Study for Automatic Bird Identification [19]

In this study, researchers Juha Niemi and Juha Tanttu (2018) propose a system for automatic bird identification for wind farms that are being constructed off the Finnish coastline [19]. They chose to integrate a radar, motorized video head and single lens reflex camera with a telephoto lens. To process the image classification, a CNN was trained with a deep learning algorithm on a dataset count of 9312 manually taken original images They chose to combine this with a data augmentation method in which the images were rotated and converted in accordance with desired color temperatures.

For hardware, they utilized:

- a Robin Radar Systems B.V. model 3D Flex;·
- for video head control, they used the PT1020 motorized video head.
    - The control software for it was developed with C in Ubuntu 16.04
- For camera control, they developed software in C# using Visual Studio, because this was the only language/development environment their Canon 7D mark II camera API could support

The area where the input data was captured had to be put under altitude constraints, area constraints, and they applied image rotation and color temperature gradients to simulate different lighting conditions, positions of the birds, and weather conditions as well.

At the time of the research, all images were taken manually, but their goal is to completely automate the process. The proposed system involves the radar being utilized to initially detect the flying birds, as well as their velocity and trajectory. The size of the bird is estimated from this as well. The system also includes the camera with telephoto lens and motorized video head, controlled by the API of the camera manufacturer.

The research performed concerned 2 species in particular; the White-tailed Eagle and the lesser black-backed gull, that are explicitly mentioned in the environment license. The ultimate purpose of bird monitoring at wind farms is collision reduction/avoidance, including determining the best deterrent methods to utilize. This would allow us to continue utilizing wind energy while minimizing the harm it does to the ecosystem around the wind farms.

Objective

For my programming exercise, a multiclassification CNN model will be written in Python 3.6, utilizing the Keras API, within Jupyter Notebook. I will train the convnet to tune its weights via backpropagation [6]. **Samples for the training, validation, and test will respectively be split 94%/3%/3% and taken from 180 Birds dataset [13].** I will train the convnet to visually identify 180 birds in the kaggle test set. The accuracy and loss will be recorded for training, validation, and testing. Both overfitting (i.e., when training loss is low but model does not generalize well to testing or validation sets) and underfitting (i.e., when training loss is high) will be avoided. Loss graphs for training versus testing, learning curves for training, and classification report for testing will all be provided.

The design of my data mining project is thus:
1. A CNN is chosen as my supervised learning model.
2. The loss will be *categorical cross-entropy* and the optimizer *RMSProp.*
3. Weights will be updated via backpropagation in order to optimize the loss and predict classes.

Before optimization, the loss function has to be computed. For our particular multiclassification problem, the 'categorical cross entropy' will be used because it is effective for more than two target labels (i.e., the class associated with a specific sample). It measures the distance of the label of the training set's classifications relative to the predictions, and it then averages these class-wise errors to obtain the loss (J) [14]. Here is the algorithm for the loss (J):

$$J = -\sum_{c=1}^{M} y_{o,c} log(p_{o,c})$$

M refers to the number of classes, y refers to whether class label c is the correct classification for observation o, and p is the predicted probability observation o is of class c.

The standard algorithm for optimization is gradient descent [15]. This is done by modifying the learning rate, denoted as α. The goal is for the "gradient component" to become as close to 0 through training, which is the gradient of the loss function with respect to weight or the parameter (J) represented as a partial derivative, $\frac{\partial J}{\partial \theta_t}$, and this leads to convergence with the local minimum of the loss function via simultaneous update of all weights (θ_{t+1}). Here is the algorithm for updating weights:

$$\theta_{t+1} := \theta_t + \alpha \times \frac{\partial J}{\partial \theta_t}$$

However, I chose Adam for the optimization of our example, which is a variant of gradient descent [15]. It is important to remember there are many algorithms used for calculating loss function and optimization, depending largely on the nature of the problem [15-16]. In short, our loss will be categorical cross entropy and the optimizer will be ADAM.

Let's consider what happens inside of a CNN when one feed it with an image intended for classification. One inputs a RGB image to the network of a specific width and height, and its depth would be 3 because it's an RGB image. When the image is being manipulated by the network, it is known as a 'feature map' throughout the network. A convolution layer takes patches from its input feature map and applies transformations on all of these patches to create an output feature map. The output feature map would also have width and height, and its depth is determined by the parameters of that specific layer. The convolutional layer slides its kernel (size of 3 x 3 in our case) on the entire feature map and extract its 'features'. Layers at the start of the network extract basic features like lines, corners, edges, and etc. whereas the layers at the end extract more complex features. In this way, one has trained the network by passing through it a lot of images of species of birds. By extracting the dominant features of the birds, the network has tuned its weights so that it would correct class them.

The metrics of accuracy will be evaluated for the training and validation sets. Moreover, for the testing set, accuracy, precision, recall, and f1 score will be evaluated.

$$Accuracy \ = \frac{TP + TN}{TP + TN + FP + FN}$$

$$Precision \ = \frac{TP}{TP + FP}$$

$$Recall \ = \frac{TP}{TP + FN}$$

$$f1 \ = \frac{2 \times precision \times recall}{precision \times recall}$$

TP refers to true positive, which is number of correctly predicted birds, which corresponds to the ground truth class or labels. TN is true negative, and number of correctly predicted birds as not belonging to their negative labels. FP refers to false positive, which is number of incorrectly predicted birds, which does not correspond to the ground truth class or labels. FN is true negative, and number of incorrectly predicted birds as not belonging to their negative labels.

Problem Formulation

This project was done largely in Google Colab Pro. All of the images are already labeled for supervised learning and in dimensions of 224x224x3 and divided into train, test, and validation sets with no overlapping images. 24497 train images, 900 for validation, 900 for testing images. In the testing set, there are 180 bird species, each constituting a class, to be predicted, and within each class there are 5 test samples (i.e., 180 * 5 = 900). Here are some random images of the birds.

Fig. 1. Examples of bird images in 224x224x3 dimensions. There are 180 different bird species.

Afterwards, I made the train, testing, and validation image generators. I used horizontal flipping for training because images of birds do not assume horizontal symmetry, and I also added shuffling of data which helps training. I chose a batch size of 32 for training and validation, and a batch size of 1 for testing. I used a ResNet-50, which has 24,676,020 trainable parameters and 53,120 non-trainable parameters.

I decided to to train a ResNet-50 model with pretrained weights, based on "ImageNet" for transfer learning (Fig. 5). ResNet-50 is a convolutional neural network, which is 50 layers deep [20]. The RESNet contains many residual blocks, which are skip connections among convolution blocks which help deal with the "vanishing gradient problem", which refers to how even when more layers are added there is a decrease in speed and accuracy (Fig. 6) [20]. Residual blocks help propagate larger gradients in networks [21].

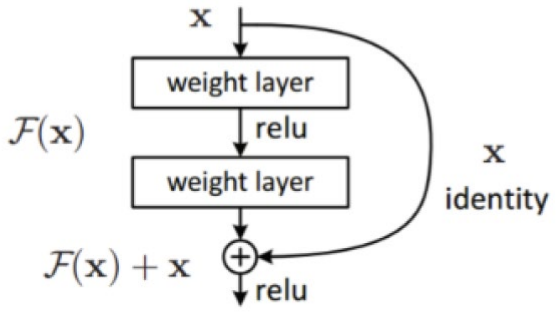
x
weight layer
$\mathcal{F}(\mathbf{x})$
relu
weight layer
x
identity
$\mathcal{F}(\mathbf{x}) + \mathbf{x}$
relu

Fig 2. Example of Residual block. [20]

Therefore, I am training a RESNet-50 with pretrained weights, from ImageNet, for transfer learning, in order to classify 180 birds.

Summary and Results

I trained the model for 20 epochs and saved weights after each epoch. The training and validation learning curves were compared in order to ensure there was no overfitting (Fig. 2). It converged at 14th epoch, and there was no overfitting or underfitting (Fig. 2). Both training and validation converged at the optima near an accuracy of 94%, which is very high. The training accuracy was 94% but the validation accuracy is 92%, which is close enough, but **the validation loss of 33% was lower than training loss of 36%. A lower validation loss means convergence:**

```
Epoch 14/20
765/765 [==============================] - 275s 360ms/step - loss: 0.3585 - accuracy: 0.9398 - val_loss: 0.3290 - val_accuracy: 0.9228
```

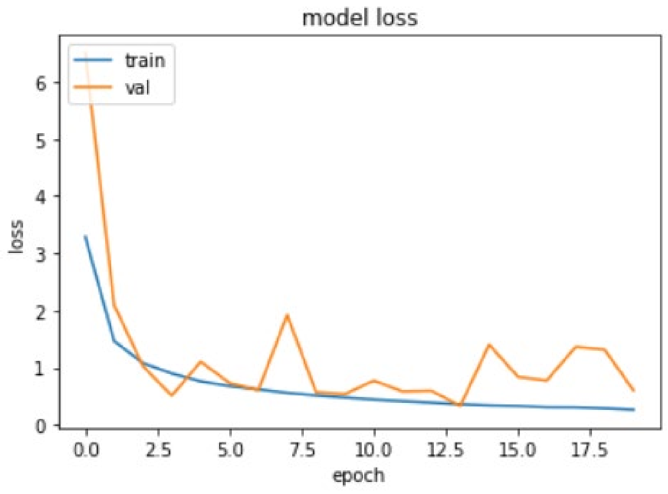

Fig. 3: Loss curve of training and validation.

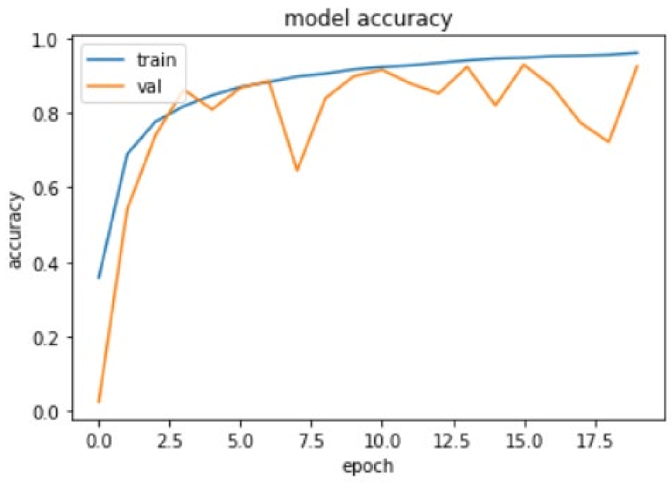

Fig. 4: Learning curve of training and validation.

The CNN model fine-tuned its weights, and I can use its updated weights to predict the labels of the unseen testing dataset. The classification report for prediction of classes compared to actual classes of test dataset is provided below. **The overall average test accuracy was 94%, which is the same as the training's results at 14 epochs**. The average precision for test set was 95%, the recall 94%, and f1-score 94%.

```
Testing loss:0.14913460612297058
Testing Accuracy:93.55555772781372
                              precision    recall  f1-score   support
                 ALBATROSS       0.56      1.00      0.71         5
      ALEXANDRINE PARAKEET       1.00      1.00      1.00         5
          AMERICAN AVOCET       0.83      1.00      0.91         5
         AMERICAN BITTERN       1.00      1.00      1.00         5
            AMERICAN COOT       1.00      1.00      1.00         5
       AMERICAN GOLDFINCH       1.00      1.00      1.00         5
         AMERICAN KESTREL       1.00      1.00      1.00         5
            AMERICAN PIPIT       1.00      1.00      1.00         5
        AMERICAN REDSTART       0.83      1.00      0.91         5
                  ANHINGA       1.00      1.00      1.00         5
        ANNAS HUMMINGBIRD       1.00      1.00      1.00         5
                  ANTBIRD       1.00      0.80      0.89         5
          ARARIPE MANAKIN       1.00      1.00      1.00         5
               BALD EAGLE       1.00      1.00      1.00         5
         BALTIMORE ORIOLE       1.00      1.00      1.00         5
               BANANAQUIT       1.00      1.00      1.00         5
         BAR-TAILED GODWIT       1.00      1.00      1.00         5
                 BARN OWL       1.00      1.00      1.00         5
             BARN SWALLOW       1.00      1.00      1.00         5
      BAY-BREASTED WARBLER       1.00      1.00      1.00         5
         BELTED KINGFISHER       1.00      1.00      1.00         5
         BIRD OF PARADISE       0.83      1.00      0.91         5
          BLACK FRANCOLIN       1.00      1.00      1.00         5
            BLACK SKIMMER       1.00      0.60      0.75         5
               BLACK SWAN       0.83      1.00      0.91         5
    BLACK THROATED WARBLER       1.00      1.00      1.00         5
            BLACK VULTURE       1.00      1.00      1.00         5
    BLACK-CAPPED CHICKADEE       1.00      0.40      0.57         5
        BLACK-NECKED GREBE       1.00      1.00      1.00         5
      BLACKBURNIAM WARBLER       1.00      1.00      1.00         5
              BLUE GROUSE       1.00      1.00      1.00         5
               BLUE HERON       0.83      1.00      0.91         5
                 BOBOLINK       1.00      0.80      0.89         5
           BROWN THRASHER       1.00      1.00      1.00         5
              CACTUS WREN       1.00      0.80      0.89         5
         CALIFORNIA CONDOR       1.00      0.80      0.89         5
          CALIFORNIA GULL       1.00      0.60      0.75         5
         CALIFORNIA QUAIL       0.83      1.00      0.91         5
                   CANARY       1.00      1.00      1.00         5
          CAPE MAY WARBLER       1.00      1.00      1.00         5
        CARMINE BEE-EATER       1.00      0.40      0.57         5
             CASPIAN TERN       1.00      0.80      0.89         5
                CASSOWARY       1.00      0.80      0.89         5
          CHARA DE COLLAR       1.00      1.00      1.00         5
         CHIPPING SPARROW       1.00      0.80      0.89         5
           CINNAMON TEAL       1.00      1.00      1.00         5
        COCK OF THE  ROCK       1.00      1.00      1.00         5
                 COCKATOO       0.83      1.00      0.91         5
           COMMON GRACKLE       1.00      1.00      1.00         5
      COMMON HOUSE MARTIN       0.71      1.00      0.83         5
              COMMON LOON       0.83      1.00      0.91         5
          COMMON POORWILL       1.00      0.80      0.89         5
          COMMON STARLING       1.00      1.00      1.00         5
          COUCHS KINGBIRD       1.00      1.00      1.00         5
           CRESTED AUKLET       1.00      0.60      0.75         5
         CRESTED CARACARA       1.00      0.80      0.89         5
                     CROW       1.00      1.00      1.00         5
           CROWNED PIGEON       1.00      1.00      1.00         5
               CUBAN TODY       1.00      1.00      1.00         5
```

CURL CRESTED ARACURI	1.00	1.00	1.00	5
D-ARNAUDS BARBET	0.83	1.00	0.91	5
DARK EYED JUNCO	1.00	1.00	1.00	5
DOVEKIE	1.00	0.60	0.75	5
DOWNY WOODPECKER	1.00	1.00	1.00	5
EASTERN BLUEBIRD	1.00	1.00	1.00	5
EASTERN MEADOWLARK	0.56	1.00	0.71	5
EASTERN ROSELLA	1.00	1.00	1.00	5
EASTERN TOWEE	1.00	0.80	0.89	5
ELEGANT TROGON	1.00	1.00	1.00	5
ELLIOTS PHEASANT	0.80	0.80	0.80	5
EMPEROR PENGUIN	0.83	1.00	0.91	5
EMU	1.00	1.00	1.00	5
EURASIAN MAGPIE	1.00	1.00	1.00	5
EVENING GROSBEAK	1.00	1.00	1.00	5
FLAME TANAGER	0.83	1.00	0.91	5
FLAMINGO	0.83	1.00	0.91	5
FRIGATE	1.00	1.00	1.00	5
GILA WOODPECKER	1.00	1.00	1.00	5
GLOSSY IBIS	1.00	1.00	1.00	5
GOLD WING WARBLER	1.00	1.00	1.00	5
GOLDEN CHLOROPHONIA	1.00	1.00	1.00	5
GOLDEN EAGLE	1.00	0.60	0.75	5
GOLDEN PHEASANT	1.00	1.00	1.00	5
GOULDIAN FINCH	0.83	1.00	0.91	5
GRAY CATBIRD	1.00	1.00	1.00	5
GRAY PARTRIDGE	1.00	1.00	1.00	5
GREEN JAY	1.00	1.00	1.00	5
GREY PLOVER	1.00	0.80	0.89	5
GUINEAFOWL	1.00	1.00	1.00	5
HAWAIIAN GOOSE	1.00	0.80	0.89	5
HOODED MERGANSER	1.00	1.00	1.00	5
HOOPOES	1.00	1.00	1.00	5
HORNBILL	1.00	1.00	1.00	5
HOUSE FINCH	1.00	1.00	1.00	5
HOUSE SPARROW	1.00	0.80	0.89	5
HYACINTH MACAW	1.00	1.00	1.00	5
INCA TERN	1.00	1.00	1.00	5
INDIGO BUNTING	1.00	1.00	1.00	5
JABIRU	1.00	1.00	1.00	5
JAVAN MAGPIE	1.00	1.00	1.00	5
KILLDEAR	1.00	1.00	1.00	5
KING VULTURE	0.71	1.00	0.83	5
LARK BUNTING	1.00	1.00	1.00	5
LILAC ROLLER	1.00	0.80	0.89	5
LONG-EARED OWL	1.00	1.00	1.00	5
MALEO	1.00	1.00	1.00	5
MALLARD DUCK	1.00	1.00	1.00	5
MANDRIN DUCK	1.00	1.00	1.00	5
MARABOU STORK	0.71	1.00	0.83	5
MASKED BOOBY	1.00	1.00	1.00	5
MIKADO PHEASANT	1.00	0.80	0.89	5
MOURNING DOVE	1.00	1.00	1.00	5
MYNA	0.50	1.00	0.67	5
NICOBAR PIGEON	1.00	1.00	1.00	5
NORTHERN CARDINAL	1.00	0.80	0.89	5
NORTHERN FLICKER	1.00	1.00	1.00	5
NORTHERN GANNET	0.71	1.00	0.83	5
NORTHERN GOSHAWK	0.60	0.60	0.60	5
NORTHERN JACANA	1.00	0.80	0.89	5
NORTHERN MOCKINGBIRD	1.00	1.00	1.00	5
NORTHERN RED BISHOP	0.62	1.00	0.77	5

OCELLATED TURKEY	1.00	0.80	0.89	5
OSPREY	0.71	1.00	0.83	5
OSTRICH	1.00	1.00	1.00	5
PAINTED BUNTIG	1.00	1.00	1.00	5
PARADISE TANAGER	1.00	1.00	1.00	5
PARUS MAJOR	0.83	1.00	0.91	5
PEACOCK	1.00	1.00	1.00	5
PELICAN	0.40	0.80	0.53	5
PEREGRINE FALCON	1.00	0.80	0.89	5
PINK ROBIN	1.00	1.00	1.00	5
PUFFIN	1.00	1.00	1.00	5
PURPLE FINCH	0.83	1.00	0.91	5
PURPLE GALLINULE	1.00	1.00	1.00	5
PURPLE MARTIN	0.83	1.00	0.91	5
PURPLE SWAMPHEN	0.80	0.80	0.80	5
QUETZAL	1.00	1.00	1.00	5
RAINBOW LORIKEET	1.00	1.00	1.00	5
RED FACED CORMORANT	1.00	1.00	1.00	5
RED HEADED WOODPECKER	1.00	1.00	1.00	5
RED THROATED BEE EATER	1.00	1.00	1.00	5
RED WINGED BLACKBIRD	1.00	1.00	1.00	5
RED WISKERED BULBUL	1.00	1.00	1.00	5
RING-NECKED PHEASANT	0.83	1.00	0.91	5
ROADRUNNER	0.83	1.00	0.91	5
ROBIN	1.00	1.00	1.00	5
ROCK DOVE	1.00	1.00	1.00	5
ROSY FACED LOVEBIRD	1.00	1.00	1.00	5
ROUGH LEG BUZZARD	1.00	0.80	0.89	5
RUBY THROATED HUMMINGBIRD	1.00	1.00	1.00	5
RUFOUS KINGFISHER	1.00	1.00	1.00	5
RUFUOS MOTMOT	1.00	1.00	1.00	5
SAND MARTIN	1.00	1.00	1.00	5
SCARLET IBIS	1.00	0.60	0.75	5
SCARLET MACAW	1.00	1.00	1.00	5
SHOEBILL	0.75	0.60	0.67	5
SNOWY EGRET	1.00	1.00	1.00	5
SORA	1.00	1.00	1.00	5
SPLENDID WREN	1.00	1.00	1.00	5
SPOONBILL	0.75	0.60	0.67	5
STORK BILLED KINGFISHER	1.00	1.00	1.00	5
STRAWBERRY FINCH	1.00	1.00	1.00	5
TAIWAN MAGPIE	1.00	1.00	1.00	5
TEAL DUCK	0.83	1.00	0.91	5
TIT MOUSE	1.00	1.00	1.00	5
TOUCHAN	1.00	1.00	1.00	5
TRUMPTER SWAN	1.00	0.60	0.75	5
TURKEY VULTURE	1.00	0.80	0.89	5
TURQUOISE MOTMOT	1.00	1.00	1.00	5
VARIED THRUSH	1.00	1.00	1.00	5
VENEZUELIAN TROUPIAL	0.83	1.00	0.91	5
VERMILION FLYCATHER	1.00	0.80	0.89	5
VIOLET GREEN SWALLOW	1.00	1.00	1.00	5
WESTERN MEADOWLARK	1.00	0.40	0.57	5
WHITE CHEEKED TURACO	1.00	0.80	0.89	5
WHITE TAILED TROPIC	1.00	0.60	0.75	5
WILD TURKEY	1.00	1.00	1.00	5
WILSONS BIRD OF PARADISE	1.00	1.00	1.00	5
WOOD DUCK	1.00	1.00	1.00	5
YELLOW HEADED BLACKBIRD	1.00	0.80	0.89	5
accuracy			**0.94**	**900**
macro avg	**0.95**	**0.94**	**0.94**	**900**
weighted avg	**0.95**	**0.94**	**0.94**	**900**

Implementation

The libraries used were Numpy, Pandas, Matplotlib, Sklearn, and Keras. The most important one is Keras which allows for manipulation of tensors, which are multidimensional arrays. The manipulation of tensors allows for the convolution, max pooling, and other layers previously described. Sklearn was used to generate classification report.

From Keras, I imported the image generator which helps in generating tensors of image data with real time data augmentation. Then, image generator method 'flow_from_directory' is used which is a directory iterator and yields batches of augmented data (x,y) where x belongs to multidimensional numpy array of images and y belongs to numpy array of corresponding labels. Feeding images to network in batches facilitates training due to the addition of steps, which controls the number of passes through the dataset. The class-mode used is categorical and not binary because this involves the use of 180 classes. Categorical mode returns 2 dimensional one hot enoded labels while binary returns 1 dimensional binary labels.

I also imported Keras' ResNet-50 model with pretrained weights of ImageNet. The model has already been trained on thoughts and hundreds of images, but I am fine-tuning it to classify 180 bird species. For the purpose of transfer learning, there are many classifiers that can be used on the top of convolutional layers i.e. fully connected layers, global average pooling layer and linear support vector machines. In this work, I added a global average pooling layer, followed by a dense layer, and fed its output to a softmax function. The global average pooling layer reduces the spatial dimensions to yield a vector with a single value corresponding to the average of feature maps, and the softmax is applied to get the predicted probabilities [22]. This is recommended for ResNet-50 [22].

As this is not binary classification problem, softmax activation function as shown below is used instead of sigmoid. It uses a linear transformation function and outputs a vector in range (0,1) when applied to output score z for each class of bird species.

$$z = W^T h + b$$

$$Softmax(z_i) = \frac{exp(z_i)}{\sum_j exp(z_j)}$$

As far as loss function is concerned, categorical cross-entropy loss is used as explained which is basically a softmax function with cross-entropy function as shown in the Objective section. It is used for multi class classification problem and I am able to train my network to output probability distribution for 180 classes of bird species using this loss function.

ADAM optimizer is used along with regularizer l2 with a lambda of 0.01 to help minimize overfitting, which is called "ridge regression" and does a very good job to avoid overfitting [23]. It uses a penalty term for the loss function [23].

I also ran model.summary(), a keras function, in order to see the number of training parameters. In order to make my notebook clearer and more concise, I cleared the output for model.summary() since it's not important to overall project's goal.

For training, I saved the model after each epoch, and I loaded the saved model at 14th epoch because it had the lowest validation loss and both the validation and training curves seemed to converge at the optimum, which is lowest loss. Keras' model.fit() trains the model.

For evaluating the model, I defined the ground truth classes as true labels and the predicted classes as y_pred and uses Sklearn's classification report. Due to the number of classes, Sklearn cannot display the full of confusion matrix, so I believe a classification report is good enough. The testing loss and testing accuracy were also printed for the test set.

Final Reflection

This project has taught me several things: the value of transfer learning and the importance of finding a dataset with a lot of samples. Initially, I had found a dataset with too few samples which led to overfitting. I was able to find a better set on Kaggle. Moreover, I learned that Keras has in-built library of pretrained weights for premade models, which one can further train on his or her dataset. This makes training take less time overall.

Creating convolutional neural network models to classify bird can help monitor threatened bird populations. For example, if there are x number of threatened parrot species in the Amazonian, a research can train a convolutional neural network to fine tune its weights in order to classify the birds. Then he can have drones programmed with this model in order to fly around and classify these parrots. This can be better than counting with the eye or banding, which are unreliable as explained in the introduction. It can help monitor the bird populations of threatened species if enough data is at hand.

The model I created can be used for this same purpose. All it requires is changing the dataset, splitting into testing, validation, and test sets, and retraining.

References

[1] Gregory, R. (2006). Birds as biodiversity indicators for Europe. *Significance*, *3*(3), 106–110. doi: 10.1111/j.1740-9713.2006.00178.x

[2] Rosenberg, K., Dokter, A., Blancher, P., Sauer, J., Smith, A., Smith, P., … Marra, P. (2019). Decline of the North American avifauna. *Science*, *366*(6461), 120-124, doi: 10.1126/science.aaw1313

[3] Duhart, C., Dublon, G., Mayton, B., Davenport, G., & Paradiso, J. (2019). Proceedings from 36th International Conference on Machine Learning: *Deep Learning for Wildlife Conservation and Restoration Efforts.* Longbeach, CA. https://www.researchgate.net/publication/333902509_Deep_Learning_for_Wildlife_Conservation_and_Restoration_Efforts

[4] Hong, S., Han, Y., Kim, S., Lee, A., & Kim, G. (2019). Application of Deep-Learning Methods to Bird Detection Using Unmanned Aerial Vehicle Imagery. *Sensors (Basel), 19*(7): 1651, doi: 10.3390/s19071651

[5] Ng, A. (2019). Machine Learning Course Offered By Stanford. *Coursera*. Retrieved from https://www.coursera.org/learn/machine-learning

[6] Chollet, F. (2018). *Deep Learning with Python*. Shelter Island: New York, NY: Manning Publications Co.

[7] Altenberger, F., & Lenz, C. (2018). A Non-Technical Survey on Deep Convolutional Neural Network Architectures. *arXiv* preprint arXiv:1803.02129.

[8] Brandon, R. (2018, Oct 17). *How convolutional neural networks work, in depth* [Video file]. Retrieved from https://youtu.be/JB8T_zN7ZC0

[9] Hung, C. C.. *Lecture 10: Deep Machine Learning* [Powerpoint slides]. Retrieved from https://d2l.kennesaw.edu/

[10] Kaxiras, E. (2019). Lab 5: Convolutional Neural Networks. *Harvard University*. Retrieved from https://harvard-iacs.github.io/2019-CS109B/labs/lab5/solutions/

[11] Singhal, H. (2017). Convolutional Neural Network with TensorFlow implementation. *Medium*. Retrieved from https://medium.com/data-science-group-iitr/building-a-convolutional-neural-network-in-python-with-tensorflow-d251c3ca8117

[12] Rodriguez, J. (2017). Convolutional Neural Networks for the Rest of Us Part III: Benefits and Motivation. *Medium*. Retrieved from https://medium.com/@jrodthoughts/convolutional-neural-networks-for-the-rest-of-us-part-iii-benefits-and-motivation-d84cd72f5670

[13] Gerry. (2020). 180 Bird Species, Version 21. [Data file] Retrieved from kaggle.com/gpiosenka/100-bird-species

[14] ML Glossary. (2017). *Loss Functions*. Retrieved from https://ml-cheatsheet.readthedocs.io/en/latest/loss_functions.html

[15] Karim, R. (2019). 10 Gradient Descent Optimisation Algorithms + Cheat Sheet. *Towards Data Science.* Retrieved from https://towardsdatascience.com/10-gradient-descent-optimisation-algorithms-86989510b5e9

[16] Bushaev, V. (2018). Understanding RMSprop — faster neural network learning. *Towards Data Science.* Retrieved from https://towardsdatascience.com/understanding-rmsprop-faster-neural-network-learning-62e116fcf29a

[17] Huang, Y. & Basanta, H. (2019). Bird Image Retrieval and Recognition Using a Deep Learning Platform. *IEEE Access*, *7*, 66980-66989. doi: 10.1109/ACCESS.2019.2918274

[18] Hong, S.., Han, Y., Kim, S.-Y., Lee, A.-Y., & Kim, G. (2019). Application of Deep-Learning Methods to Bird Detection Using Unmanned Aerial Vehicle Imagery. *Sensors*, *19*(7), 1651. doi: 10.3390/s19071651

[19] Niemi, J., & Tanttu, J. (2018). Deep Learning Case Study for Automatic Bird Identification. *Applied Science, 8*(11), 2089 doi: 10.3390/app8112089

[20] P. Marcelino, "Transfer learning from pre-trained models," Towards Data Science, 23-Oct-2018. [Online]. Available: https://towardsdatascience.com/transfer-learning-from-pre-trained-models-f2393f124751

[21] S. Sahoo, "Residual blocks — Building blocks of ResNet," Medium-Towards a Science, 27-Nov-2018.

[Online]. Available: https://towardsdatascience.com/residual-blocks-building-blocks-of-resnetfd90ca15d6ec.

[22] A. Cook, “Global Average Pooling Layers for Object Localization,” alexisbcook.github, 9-Apr-2017, [Online]. Available: https://alexisbcook.github.io/2017/global-average-pooling-layers-for-object-localization/

[23] A. Nagpal, “L1 and L2 Regularization Methods,” Medium-Towards a Science, 13-Oct-2017. [Online]. Available: https://towardsdatascience.com/l1-and-l2-regularization-methods-ce25e7fc831c