IMPLEMENTATION OF NEURAL NETWORKS TO PREDICT THE BIRD SPECIES FROM THEIR IMAGES

- RAMGOPAL REDDY PUTTA

ABSTRACT:

This report elucidates the process of prediction and classification of bird species from their images and sounds datasets with the implementation of Convolutional Neural Networks, using an R interface 'Keras', powered by TensorFlow back-end engine. A large data set with 18 species of birds that are commonly found in Western Washington and Seattle areas. Each species with 100 images is used in this analysis. All the images are colored and squared with a length and breadth of 224 pixels and 3 color channels (Red, Blue and Green. The entire data is split into training and testing data in the ratio 4:1 A Convolutional Neural Network model with various layers is built and implemented in this analysis. The model is first trained using the Training data and prediction is performed over the Test data. An accuracy level of nearly 88 percent is achieved in image prediction. Accuracy levels are further raised to 95 percent by fine-tuning the model.

FEATURES OF CNN:

The CNN model is widely considered a 'Gold Standard' for image classification problems. When compared to other classification models with fully connected networks, fewer parameters are required to build the CNN model. Therefore, the processing time is comparatively less. This model is best suitable for image recognition and classification with huge number of images.

HOW CNN WORKS:

The CNN has many hidden layers called convolutional layers. Each of these layers contains neurons that are connected to neurons of the closest layer. These convolutional layers receive the input, transform the input in some way, and output the transformed input to the next layer. The CNN contains filters which are feature detectors which detect edges, corners, circles, squares etc. in an image. The deeper the network goes, the more sophisticated these filters become. In later deep layers, the filters will be detecting specific objects like eyes, feathers, beaks etc. And in even deeper layers, the filters can detect even more sophisticated objects like full images of dogs, cats, birds etc. There are a couple of layers which make CNN unique – 'Convolutional layer', 'pooling layer'. Other important layers are the 'Relu' and 'Fully connected layer', which are common in all neural networks. The *'Convolutional layer'* is the most important layer in the network. It works by placing the filter over an array of image pixels. This then creates a feature map. The *'Pooling layer'* down samples or reduces the sample size of a particular feature map. This also makes processing much faster by reducing the number of parameters. The *'Relu'* layer acts as an activation function ensuring non-linearity as the data moves through each layer in the network. Without this layer, the data which is being fed into each layer would lose the dimensionality that is to be maintained. The *'Fully connected layer'* performs classification on the data set. An additional layer is included in the model,

which is known as *'Dropout layer'*. This layer drops out a random set of activations in a particular layer by setting them to zero. This helps by preventing over-fitting of data.

METHODOLOGY:

Resizing, reshaping and combining:

The images are loaded into the program and create a list/array of images. Images are explored for their respective sizes and shapes. Since the dimensions of all the images are 224*224*3, no further resizing or reshaping is required. Once reshaping and resizing are done, concatenate all the vectors using combine function.

Reordering the dimensions:

By default, the structure of dimensions is 224*224*3*n, where n id the number of images. The dimensions are reordered to n*224*224*3, which is the required format for a CNN model to work.

One hot encoding:

With this process, the numerical values are concerted into categorical values and the categorical values are converted into the form that could be provided to the ML model to perform better at predictions.

Model architecture:

From the input layer (IL), the data is fed into the first convolutional layer (CN1) where new parameters are introduced from the filter for the image processing (64 filters are considered in this model). Then the data is fed into the second convolutional layer (CN2) and new parameters are added from filters. Then the data is fed into the pooling layer (P1) and no new parameters are introduced in this layer. The data then goes to the Dropout layer (D1) and no new parameters are introduced in this layer as well. This layer is used only while training the data and not while testing. Then the data flows to the third and fourth convolutional layer (CN3 and CN4) followed by the pooling layer (P2) and dropout layer (D2). Now a flattening layer (FL) is included for flattening the data sets by converting the data into a one-dimensional array for inputting the data into the next layer. Then Fully connected layer (FC) is added followed by another Dropout layer (D3). Finally, the processed data enters the output layer (OL).

The flow of data through various layers is represented as below:

IL → CN1 → CN2 → P1 → D1 → CN3 → CN4 → P2 → D2 → FL → FC → D3 → OL

Fitting the model:

After modelling the CNN, the data is fit into the model using training data with 60 iterations and batch size 32 and performance of the model is observed. 20 percent of the training data is considered Validation data while fitting the model which is used to determine the performance of the model.

RESULTS AND DISCUSSION:

After fitting the model using training and validation data, the performance of the model is calculated and this performance in terms of accuracy at every epoch/iteration is plotted over a graph.

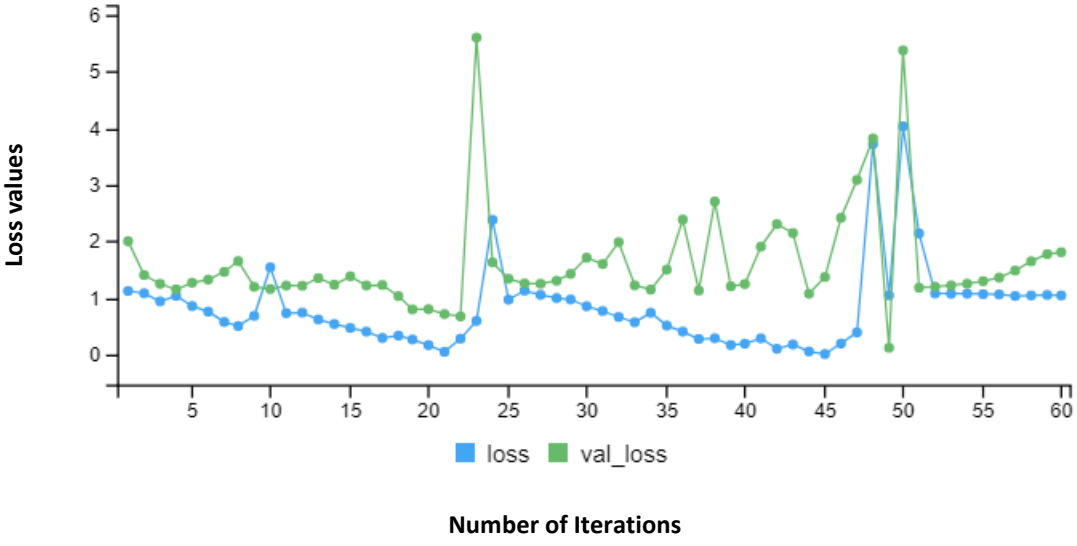

Figure 1: Graph representing loss values at their respective iteration.

The performance graph is obtained for 60 iterations. This graph represents the losses incurred in performance. Blue line represents the loss with respect to training data and green line represents validation data. Lower the loss values, better is the performance of the model. It is observed that the loss values are lower, for almost all the iterations.

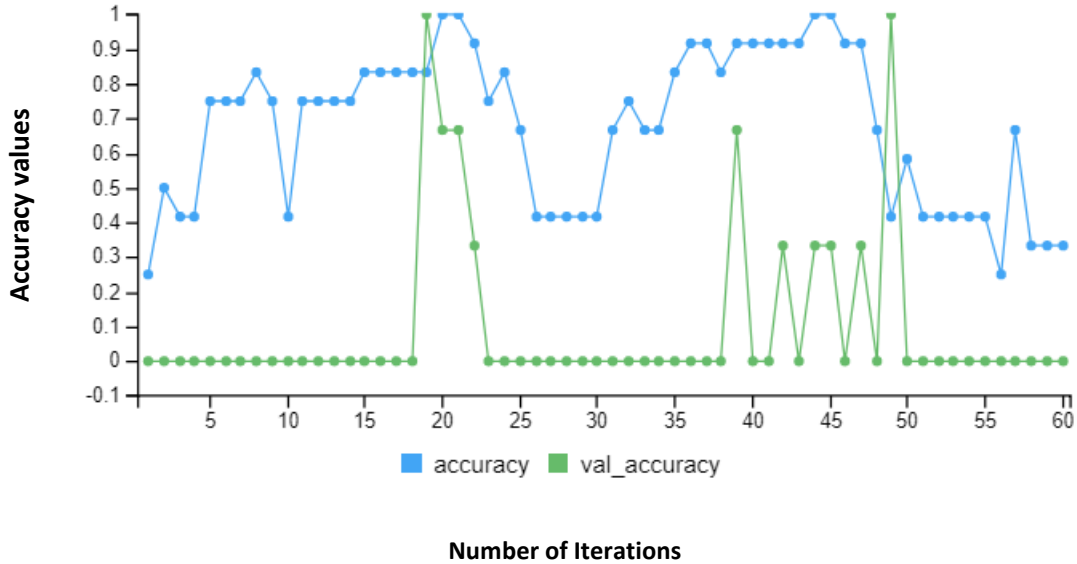

Figure 2: Graph representing accuracy values at their respective iteration.

The above graph is obtained on running the model for 60 epochs. This graph represents the accuracy of the model. Blue line represents accuracy with respect to training data and green line represents validation data. It is observed that the average accuracy value obtained is nearly 55 percent. The models must be tuned to obtain better accuracy values with various learning rate and dropout regularizations to reduce the overfitting and improve the CNN model.

After data augmentation, hyperparameter tuning and a series of runs, accuracy values are increased significantly, and losses are decreased. The following graphs are obtained after tuning the model.

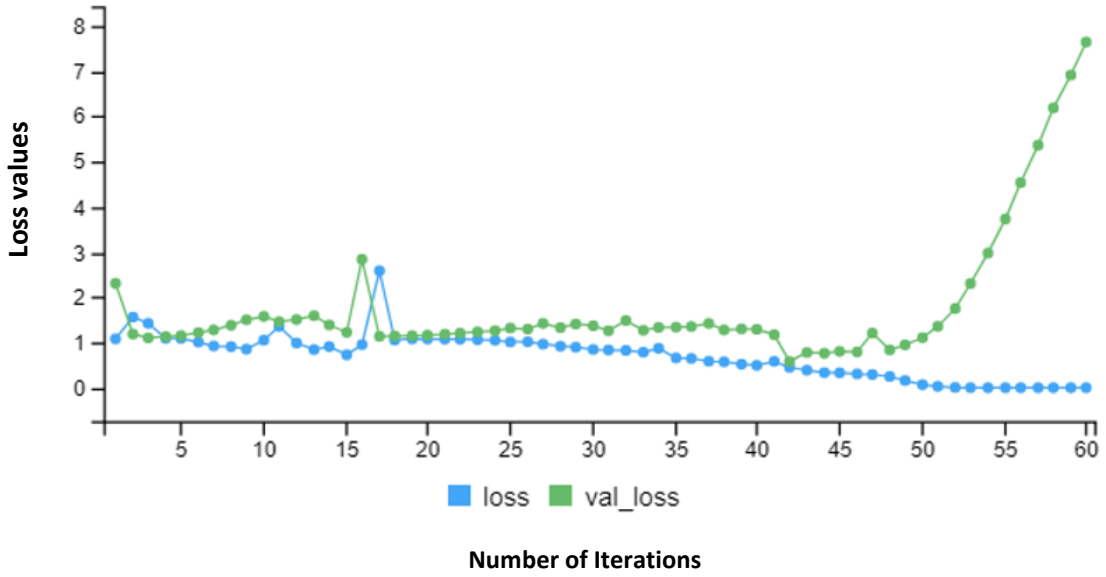

Figure 3: Graph representing loss values at their respective iteration.

When compared to initial recordings, there is a significant decrease in loss values. Since lower the loss values indicated better performance of the model, the model is performing well.

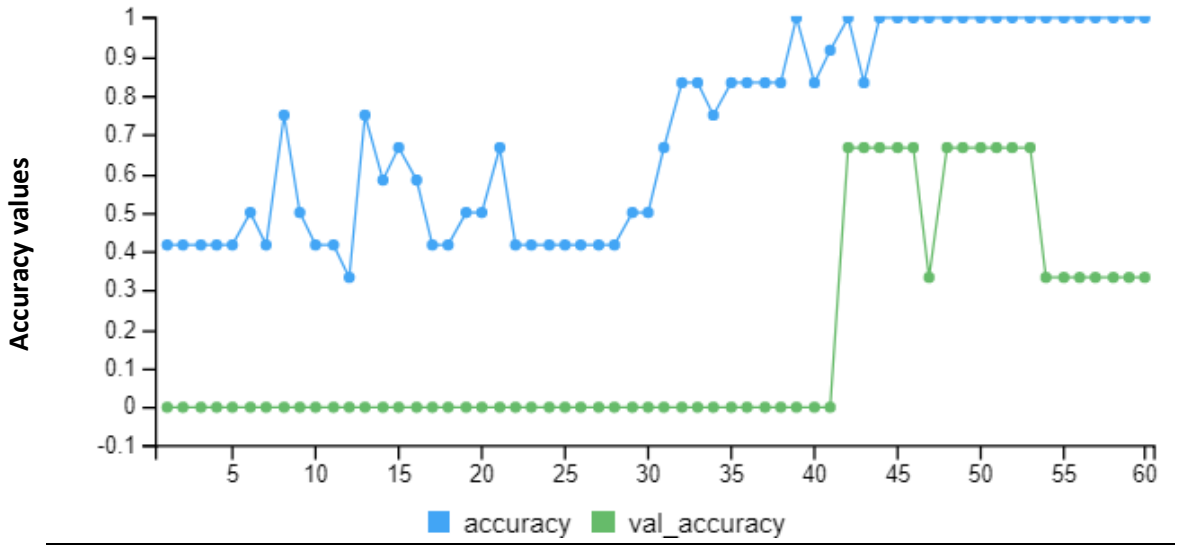

Figure 4: Graph representing accuracy values at their respective iteration.

Also, accuracy values increased when compared to the initial model. The average accuracy value obtained is nearly 90 percent. A significant increase in the performance of the model was observed. It is also observed that the accuracy value is hitting 1 consistently as the number of epochs increases.

CONCLUSION:

The CNN model implemented with various layers seems to be performing well in recognizing the images of the birds. It is observed that the model can classify previously unseen images of birds with an accuracy of nearly 60 percent on its initial run. The efficiency is further raised to 90 percent after a series of runs, tuning the model with various learning rates and optimizing the data overfitting.

References:

https://youtu.be/YRhxdVk_sIs

https://youtu.be/K_BHmztRTpA

How to build your own image recognition app with R! [Part 1] | R-bloggers

https://hastie.su.domains/ISLR2/ISLRv2_website.pdf