DEEP LEARNING

BINARY BIRD CLASSIFICATION

*TECHNICAL DOCUMENT*



By:

Aakash Mahajan

**Table of Contents**

1. TOPIC, PROBLEM STATEMENT, DATA**…………............................................................................**3

2. CREATING THE MODEL**………………….........................................................................................**4

3. DEPLOYMENT**…………..………………………………………….................................................................**5

**TOPIC:**

Image classification.

**PROBLEM STATEMENT:**

Classification of birds using the Visual Geometry Group (VGG) architecture.

This will be a binary classification wherein an input image will be given and using VGG weights, the image will be recognised and classified whether it is bird A or bird B.

**TOOLS USED:**

Python, Microsoft Azure, Git Bash.

**DATA:**

For this project, data is sourced from [www.Kaggle.com](http://www.Kaggle.com)

It consists of images of 2 birds namely Bald Eagle and Snowy Owl, each having 150 training images and 5 validation Images.

**DATA FOLDER STRUCTURE:**

Main directory – Training – Snowy Owl – Images (150)

Main directory – Training – Bald Eagle – Images (150)

Main directory – Validation – Snowy Owl – Images (5)

Main directory – Validation – Bald Eagle – Images (5)

**STEPS TO RECREATE THE PROJECT:**

***CREATING THE MODEL***

First, create the custom Neural Network model imparting weights of the VGG16 architecture (here we do not use the output layer from VGG16) and train the model on the Bald Eagle and Snowy Owl images from the “Training” folder. Once the model is trained, save the model as an “**.h5**” file.

***CREATING THE APPLICATION***

Here, the application itself consists of certain folders as follows-

* **app.py –** This is the Flask application wherein we import the “.h5” saved model. Here is where we define the controls of “what happens when”. In this we also define the labels of predicted class and control the rendering of predicted class image once submitted for prediction.
* **Static –** This folder is used to save the images which are imported for prediction. The label prediction uses the images stored in this folder to display the predicted image class.
* **Template –** This folder contains the **index.html** file which is the HTML file where it is defined how the website should be rendered (look like).
* **requirements.txt –** This file contains all the libraries that have been used in the app.py file along with their versions. These are the libraries that get installed on the cloud system when the app gets deployed.
* **Model.h5 –** This is the saved model as explained earlier which is used for prediction.

**CLOUD DEPLOYMENT**

Here **Microsoft Azure** is used to deploy the application on internet. First create an account on Microsoft Azure (Azure for Student account was used for deployment).

Following are the steps for deployment process:

* First, create a resource wherein we create a **Web App.**
* Here, create a new Resource Group, provide a name for the app, select the python version and select appropriate Service Plan (For this deployment, the Basic B1 was used).
* Now, go to the resource – Deployment – Deployment centre.
* Now, there are multiple ways by which we can proceed with the deployment. Local Git was used for this deployment process. So, under Settings, select “**Local Git**” and hit Save. Here, we receive our “Git Clone URL”, and also create new credentials with user name and password under “**User Scope**”.
* Now, in your Local machine, go to the directory where all the files mentioned in “*Creating the application*” are stored, right click and select “Git Bash Here”.
* In this, enter the following commands sequentially:

1. git init .
2. git add –all
3. git commit –m “Commit message”
4. git remote add origin (Git Clone URL)
5. git push origin master – At this stage enter the credentials created under “*User Scope*”.

That’s it, if all goes well, the application is deployed on cloud ready to share. In the “**Overview**” section of Azure Dashboard, we get the URL ending with “[.azurewebsites.net](https://trial-1.azurewebsites.net/)” which can be shared with the audiance for them to check the application on their system.

**Dataset Link:** [**https://www.kaggle.com/gpiosenka/100-bird-species?select=birds\_rev2**](https://www.kaggle.com/gpiosenka/100-bird-species?select=birds_rev2)

**Web URL:** [**https://aakashmana4.azurewebsites.net/**](https://aakashmana4.azurewebsites.net/)

(Web URL Link expired as it was an Azure Student subscription)