Dog Breed Classifier Using CNN

Capstone Report

Submitted By

Ali Hassan

Machine Learning Engineer Nanodegree

Dec 2020

Definition

1.1. Project Overview

It requires years of experience to recognizing and identify dogs breed accurately, in order to identify dogs' s breed most of people need to consult with an expert. But today we have an alternative, we don't need to consult an expert to identify dog's breed. In computer vision we have lot of emerging technologies for carrying out these classification task.

Here we can use convolutional neural network for the classification of dog's breed. By using computer vision techniques and convolution neural network, we can even defeat a human expert in terms of accuracy (It depends on quality of dataset and model).

There is an interesting paper on Dog Breed Identification with Fine tuning of Pre-trained models in International Journal of Recent Technology and Engineering (IJRTE). This paper describes how the classifier perform VGG-16, Inception V3, Xception in terms of accuracy and loss. In our proposed we are using VGG-16 for dog detection model. And also, this paper describes how we can improve the accuracy of our dog breed prediction model.

1.2. Problem Statement

We are building an ML pipeline to process and classify real-world, user-supplied images. While a user uploading an image (Dog image or human image), the algorithm will identify an estimate of the canine's breed. If supplied an image of a human, the code will identify the resembling dog breed.

1.3. Evaluation Metrics

I'm using accuracy as a metric to evaluate the performance on test data

Accuracy = Number of items classified correctly/ total items classified

Here the dataset is not properly balanced, some categories of breeds has more images compared to others, so It's an imbalanced classification problem. F1 score is appropriate this kind of data, even though here I'm using accuracy as evaluation matrix because my model bench mark is based on accuracy.

Analysis

2.1. Data Exploration

2.1.1. Datasets and Inputs

Here we are using data from Udacity. There are 13233 human images and 8351 dog images are available in our dataset.

Dog images:

- Train set 6680
- Test set 836
- 835 images for validation
- It contains 133 different breeds

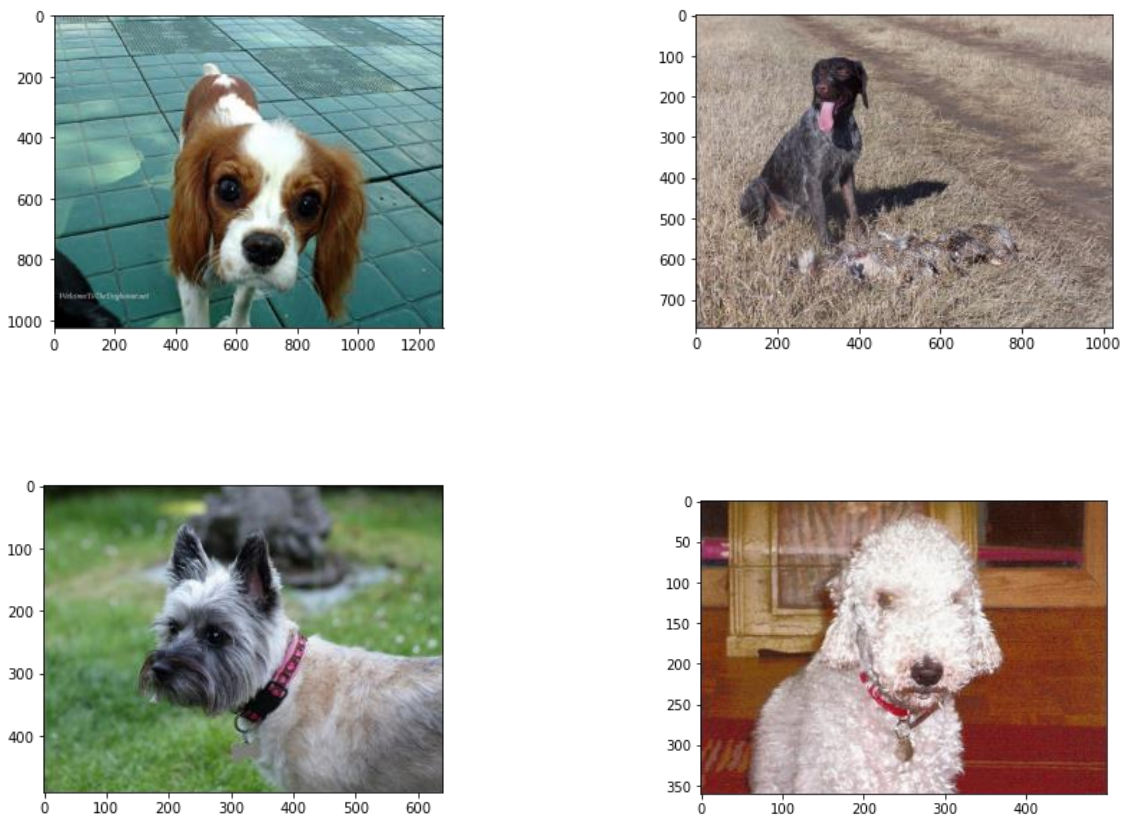


Fig1: Sample images from dog dataset

Human Images:

- 5750 folders, each folder contains different pictures of a person.
- Data is not perfectly balanced.

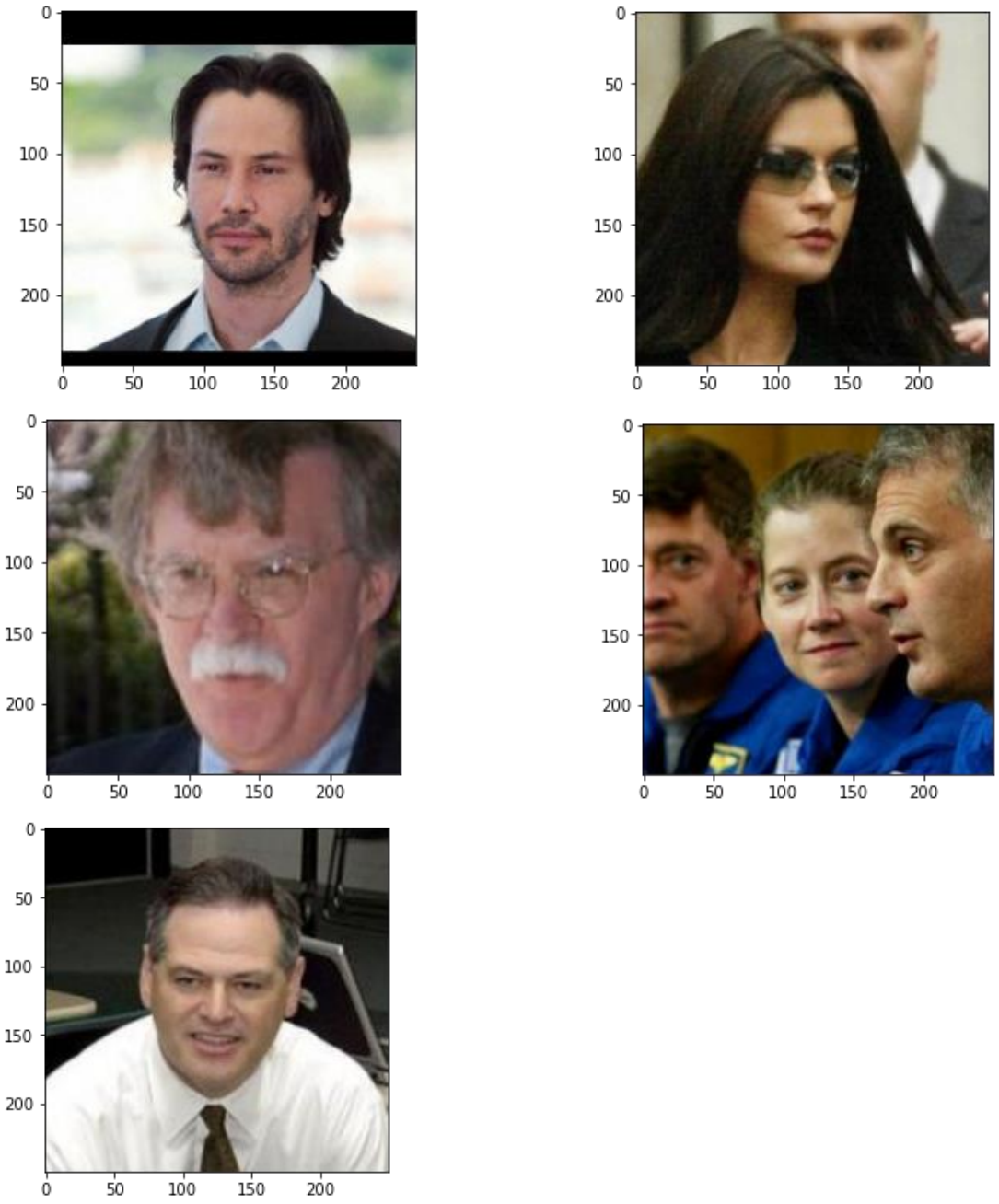


Fig2: Sample images from human dataset

2.1.2. Bar charts

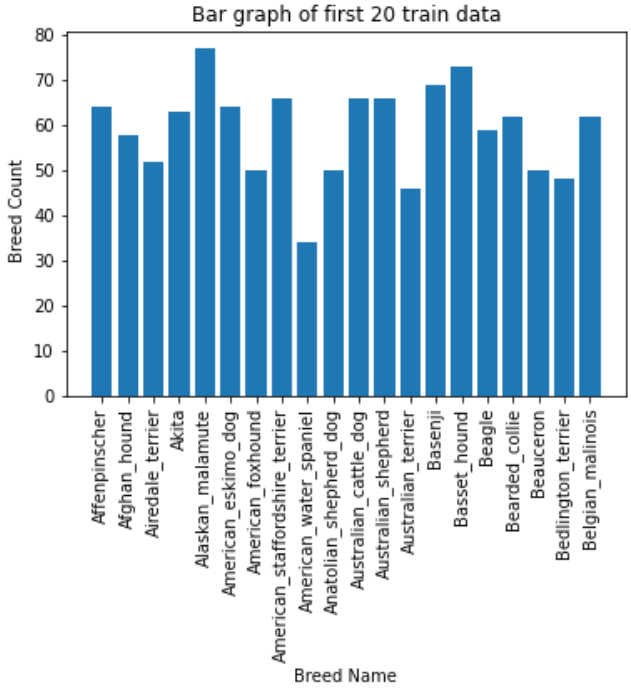


Fig3: Number of images of first 20 dog breeds (Train set)

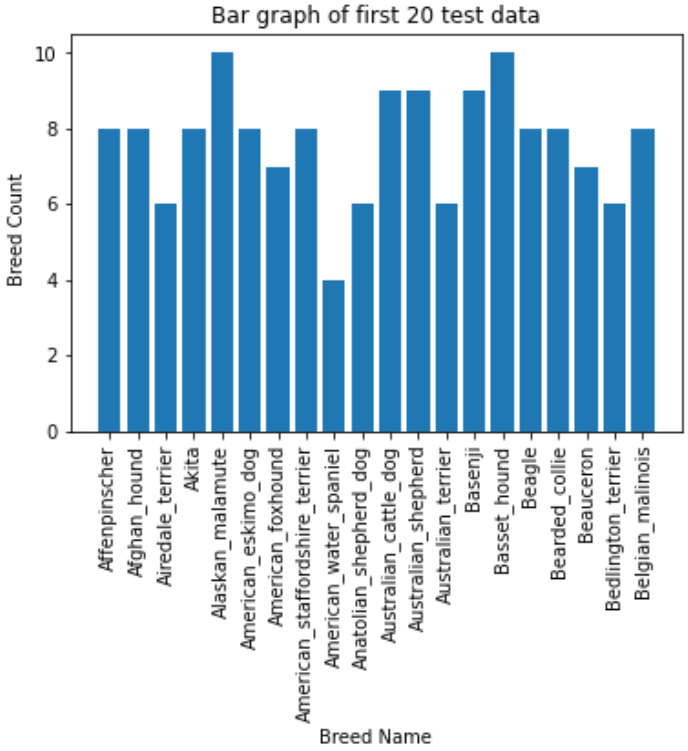


Fig4: Number of images of first 20 dog breeds (Test set)

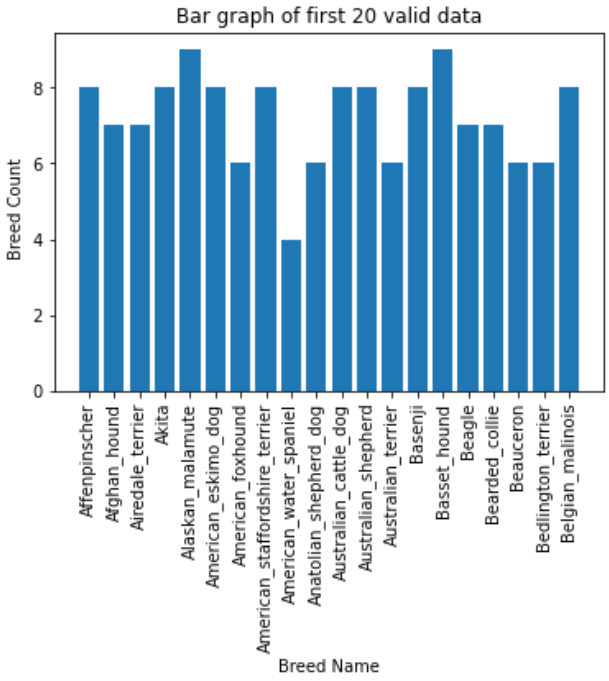


Fig5: Number of images of first 20 dog breeds (Validation set)

2.1.3. Shannon entropy as a measure of balance

Shannon entropy,

$$H = -\sum_{i=1}^{k} \frac{c_i}{n} \log \frac{c_i}{n}.$$

Where k = numbers of classes (here dog breeds),
c_i = number of instances in individual classes (In our case number of images which belongs to a breed)
n = Total number of instances (total images available in dataset)

$$\text{Balance} = \frac{H}{\log k} = \frac{-\sum_{i=1}^{k} \frac{c_i}{n} \log \frac{c_i}{n}.}{\log k}$$

Here we are analysing imbalance in our dataset by using the above balance formula

The range of the balance function is between 0 and 1(0 and 1 are included), 0 implies imbalanced data, 1 implies balanced data

2.1.4. Statistical analysis result over dataset

Dataset	mean	Standard deviation	Balance
Test	6.28	1.71	0.99
Train	50.22	11.82	0.99
Validation	6.28	1.34	0.99

2.2. Algorithms and Techniques

Here we are using four techniques to solve our problem, we are using Haar cascade classifier model for identifying human faces, pretrained VGG16 model for identification of dog, CNN from scratch for identification of dog breed and an improved CNN build using transfer learning (using ResNet50) for classifying dog breed with high accuracy.

2.2.1. Haar Cascade Classifiers for Detecting Human Face

Object Detection using Haar feature-based cascade classifiers is an effective object detection method proposed by Paul Viola and Michael Jones in their paper, “Rapid Object Detection using a Boosted Cascade of Simple Features” in 2001. It is a machine learning based approach where a cascade function is trained from a lot of positive and negative images. It is then used to detect objects in other images. Here I’m using haarcascade_frontalface_alt.xml, That is an XML file containing serialized Haar cascade detector of faces (Viola-Jones algorithm) in the OpenCV library. It is coded list of decision trees in which each vertex test one Haar Feature and each list claims “this is not face” or “this could be face”. It can be used the check that a part of image is face. Combined with the sliding window algorithm, it provides face detector. Particularly this file is for frontal faces only and the “alt” means that it is not the standard OpenCV solution but result of another - alternative - dataset and training.

2.2.2. VGG16 For detecting Dogs

Here we are using a pretrained model VGG16 for our dog detection model. The VGG network architecture was introduced by Simonyan and Zisserman in their 2014 paper, Very Deep Convolutional Networks for Large Scale Image Recognition.

This network is characterized by its simplicity, using only *3×3* convolutional layers stacked on top of each other in increasing depth. Reducing volume size is handled by max pooling. Two fully-connected layers, each with 4,096 nodes are then followed

by a softmax classifier. The smaller networks converged and were then used as *initializations* for the larger, deeper networks - this process is called pre-training**.**

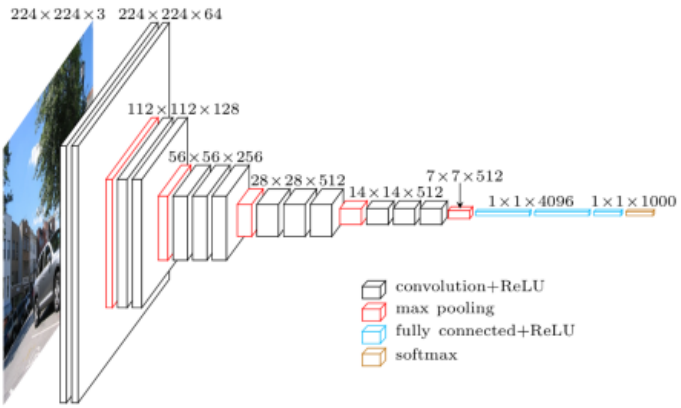


Fig6: A visualization of the VGG architecture (**source**).

2.2.3. Custom CNN for Classification of Dog Breeds

The CNN which developed from scratch has three convolution layers, two fully connected layer and pooling layer of (2,2). The connected layers produce the 133-dimension output. All of the convolution layer has kernel size of 3. Conv 1 and Conv2 have 2 strides, while Conv 3 has only one stride. The Conv3 produces output size 128. We are using relu activation function as activator and for avoiding overfitting we are using a dropout of 0.3. Using cross entropy as loss function and Adam as optimizer with lr = 0.001 while training the model.

2.2.4. ResNet50 for Classification of Dog Breeds

Here I'm using ResNet50 to improve my model accuracy. The accuracy of CNN which I build from scratch was only 12%. So, I initialized weight of neural network by using transfer learning technique and ResNet50. With the help of ResNet50 I improved the accuracy of my model up to 82%.

Unlike traditional *sequential* network architectures such as AlexNet, OverFeat, and VGG, ResNet is instead a form of "exotic architecture" that relies on micro-architecture modules (also called "network-in-network architectures").

The architecture of ResNet50 has 4 stages as shown in the diagram below. The network can take the input image having height, width as multiples of 32 and 3 as channel width. For the sake of explanation, we will consider the input size as 224 x 224 x 3. Every ResNet architecture performs the initial convolution and max-pooling using 7×7 and 3×3 kernel sizes respectively. Afterward, Stage 1 of the network starts

and it has 3 Residual blocks containing 3 layers each. The size of kernels used to perform the convolution operation in all 3 layers of the block of stage 1 are 64, 64 and 128 respectively.

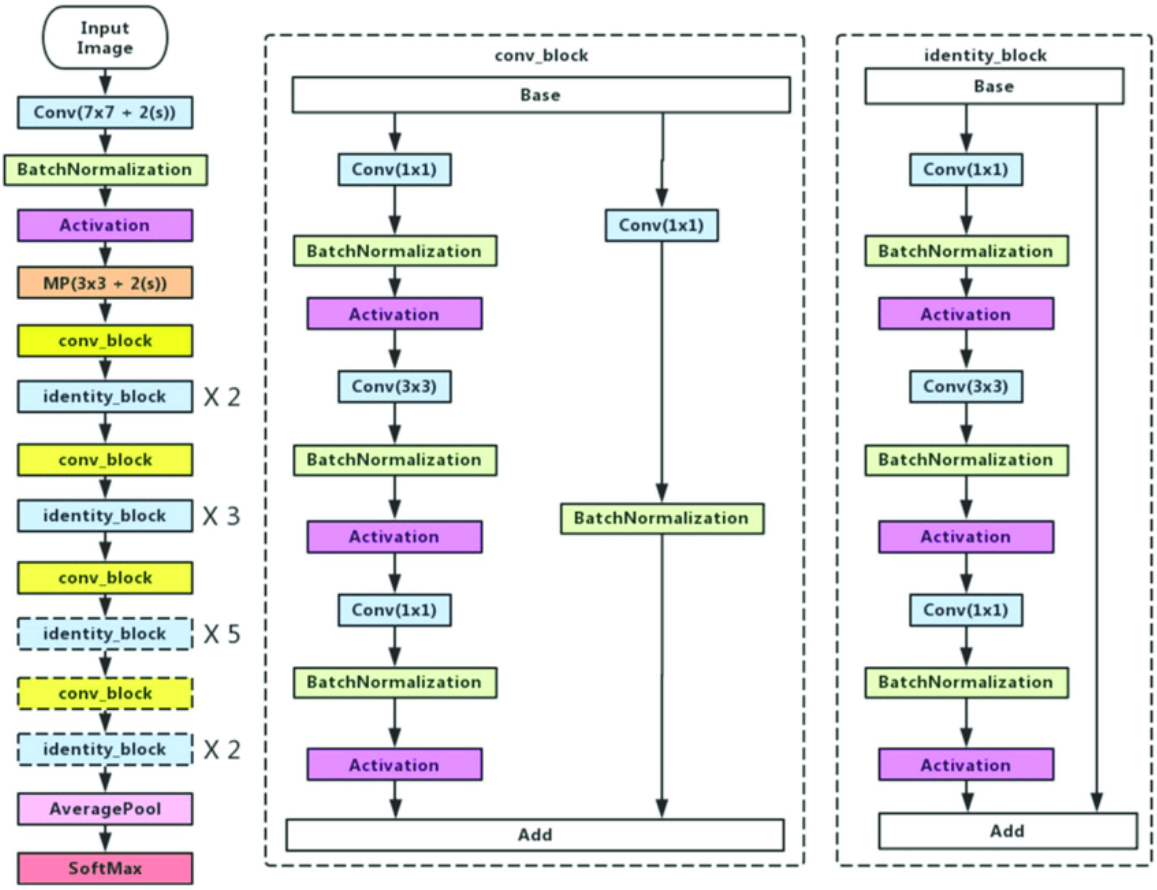


Fig7: ResNet50 architecture

2.3. Benchmark Model

- The convolution neural network must keep accuracy of at least 10%. In our case probability of getting correct answer by a random prediction is 1/133 which is less than 1%.
- The CNN model created by using transfer learning (using ResNet50) must keep 60% above accuracy.

Methodology

3.1. Data Pre-processing

Mainly I applying six type of transformation on dog breed dataset.

1. Resizing: resizing the image to 258x258
2. Random horizontal flip: Applying random horizontal flipping
3. Random rotation: Applying random rotation
4. Centre crop: Cropping images to size 244x244(Required input format for ResNet50)
5. Converting to tenser
6. Normalizing the data

3.2. Implementation

3.2.1. Human Face Detection

I'm using OpenCV Haar cascade classifier for detecting human faces. I tested the performance of this model with first 100 images from the human face dataset, it predicts 96% of images correctly. I converted image to grey scale for haar cascade classifier compatibility. Then passed the grey scale image to face cascade classifier which returned all faces in the input image as list of coordinates.

3.2.2. Dog Detection

For dog detection I used the pretrained model VGG16. I changed the input image to RGB format, then applied transformation like resizing to 256x256, centre crop to 244x244, normalization and converted to tenser. My algorithm predicts the Index corresponding to VGG-16 model's prediction, then I applied a condition to ensure the index is belongs to dog breeds (index ranges from [151,268]). The accuracy over first 100 images on dog images was 93%. It didn't misclassify any human face as dog (first 100 images from human face dataset).

3.2.3. Dog Breed Classification using Customs CNN

For classifying dog breed I created a CNN model from scratch using PyTorch. I already specified the implementation details on CNN in section Algorithms and Techniques. The accuracy of the model was only 12%, which was not a reliable dog breed classifier.

Mainly I applying six type of transformation on dog breed dataset.

1. Resizing: resizing the image to 258x258
2. Random horizontal flip: Applying random horizontal flipping
3. Random rotation: Applying random rotation
4. Centre crop: Cropping images to size 244x244(Required input format for ResNet50)
5. Converting to tenser
6. Normalizing the data

3.2.4. Dog Breed Classification using Transfer learning and ResNet50

The model, which I build from scratch was not a reliable solution for our dog breed classification problem, so I used transfer learning technique to improve the performance of my model, surprisingly it gives 82% of accuracy on testing data on 10 epochs. I'm using this model as dog breed classifier; it is reliable compared to the CNN which I build from scratch.

3.4. Refinement

The accuracy of CNN model which I created from scratch was only 12%. It was not a reliable model for dog breed classification. Then I used ResNet50 to improve the accuracy of classifier model, the result was amazing. It improved the accuracy from 12% to 82%.

Results

4.1. Model Evaluation and Validation

In this project I created 4 classification models, the accuracy of CNN which I created from scratch was really bad. It was not a reliable solution for the problem. So, I used the ResNet50 to improve my model accuracy, which significantly improves the accuracy from 12% to 82%.

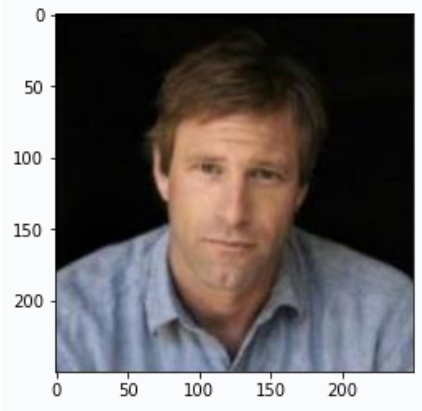
Haar Cascade Classifiers for Detecting Human Face: The accuracy of the system over first 100 images on dog dataset and human dataset really amazing. It predicts, 96 images as human faces in human image dataset and 18 images in dog dataset detected as human face.

VGG16 For detecting Dogs: The accuracy of the system over first 100 images on dog dataset and human dataset really amazing. It predicts, 93 images as dog in dog image dataset and zero images in human image dataset detected as dog face.

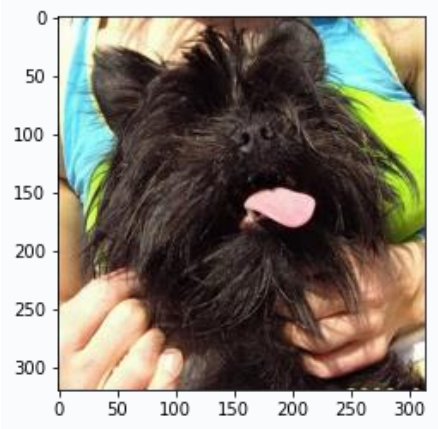
Custom CNN for Classification of Dog Breeds: Accuracy was very less; it was only 12%. And it was not a reliable solution.

ResNet50 for Classification of Dog Breeds: While applied the ResNet50 pretrained model it given accuracy of 82%, which was a significant increase from 12% accuracy of CNN (Created from scratch).

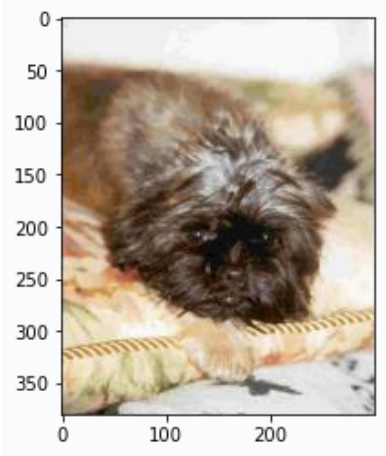
4.2. Final Result



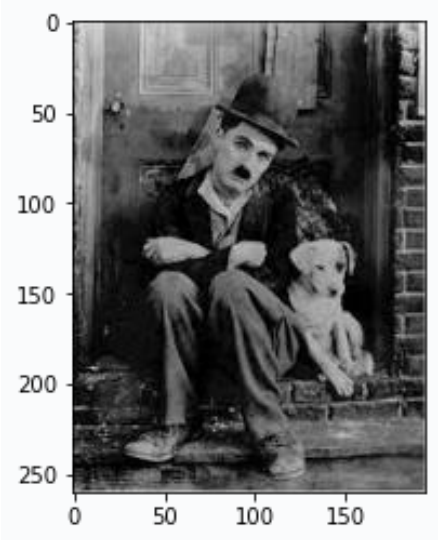
Hello Human, You Look Like a Clumber spaniel



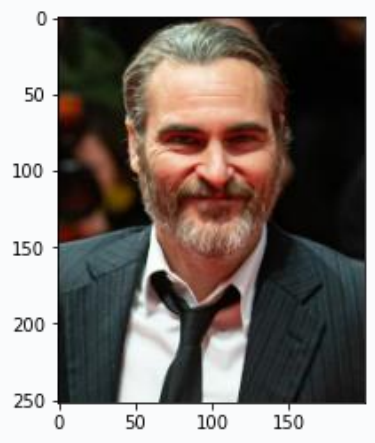
Hello Dog, You Look Like a Affenpinscher



Hello Dog, You Look Like a Affenpinscher

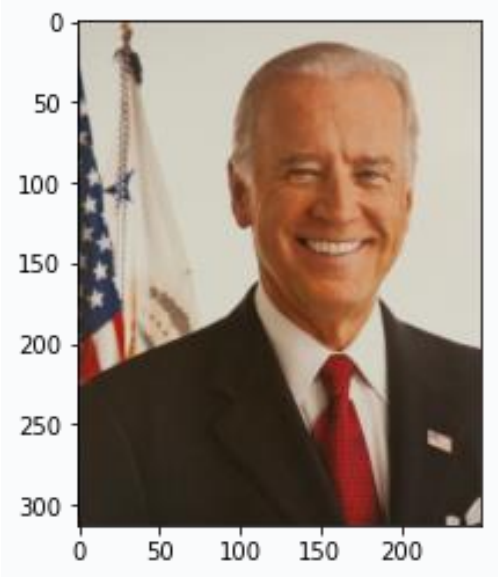


Hey, your image is not valid. please try with a valid image.



Hello Human, You Look Like a Boston terrier

Hello Dog, You Look Like a Bulldog



Hello Human, You Look Like a Norwich terrier

4.2. Justification

The final model accuracy was 82%, which is above of our bench mark accuracy 60% value. So, we can consider it as reliable solution for our dog breed classification. We can use this model to predict 133 breeds of dog. Also, we can improve the accuracy of our model by adding more images on training data, using other pretrained model like ResNet101 and by applying more image transformation techniques on dataset.

Reference

5.1. https://machinelearningmastery.com/tour-of-evaluation-metrics-for-imbalanced-classification/
5.2. https://machinelearningmastery.com/how-to-develop-a-cnn-from-scratch-for-cifar-10-photo-classification/
5.3. https://machinelearningmastery.com/how-to-develop-a-cnn-from-scratch-for-cifar-10-photo-classification/
5.4. https://datawow.io/blogs/face-detection-haar-cascade-vs-mtcnn
5.5. https://www.pyimagesearch.com/2017/03/20/imagenet-vggnet-resnet-inception-xception-keras/
5.6. https://www.ijrte.org/wp-content/uploads/papers/v8i2S11/B14640982S1119.pdf
5.7. https://www.itl.nist.gov/div898/software/dataplot/refman2/auxillar/shannon.htm
5.8. https://medium.com/analytics-vidhya/accuracy-vs-f1-score-6258237beca2
5.9. https://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_objdetect/py_face_detection/py_face_detection.html
5.10. https://arxiv.org/abs/1409.1556
5.11. https://www.researchgate.net/figure/Left-ResNet50-architecture-Blocks-with-dotted-line-represents-modules-that-might-be_fig3_331364877