Himakiran Kumar Machine Learning Engineer Nanodegree

Feb 29th, 2020

Capstone Project : Dog Breed Classifier

HIMAKIRAN KUMAR

DEFINITON

Project Overview

Image identification is a well known problem in Artificial Intelligence. Image identification projects can be either face detection, detection of objects , text detection and detection of landmarks or symbols and logos. This project aims to build a program that can not just identify a face of a Dog but also be able to detect which breed the dog belongs to. The project uses Convolutional Neural Networks which are specialized kinds of neural networks that have been very successful particularly at computer vision tasks, such as recognizing objects, scenes, and faces, among many other applications for this task. We also implement a program which detects the human face and outputs the nearest dog-breed-face it resembles!!.

Problem Statement

The goal is to implement a program that takes a picture of a dog as an input and outputs the breed of the dog. If the face belongs to a human it is able to detect the human face and output the closest dog breed face it resembles. The following are the tasks involved

1. Download the dog and human faces datasets.
2. Implement a human face detector.
3. Implement a dog face detector and breed predictor using below two methods
   a. Creating a CNN classifier and training it from scratch.
   b. Creating a CNN classifier using Transfer Learning and then training the model.
4. Implement a final program to input an image and output as follows
   a. if a **dog** is detected in the image, return the predicted breed.
   b. if a **human** is detected in the image, return the resembling dog breed.
   c. if **neither** is detected in the image, provide output that indicates an error.

Metrics

We have decided to go ahead with accuracy as the metric because though the data set exhibits mild class imbalances but on a whole, the class imbalance has been observed not be an extreme case to affect judgment of whether a classifier performs properly. As can be ascertained from the exploratory analysis of the data certain classes(breeds) had samples in the range 75-60 whereas some classes had the samples in the range 35 - 40. Moreover there was no single class with an overwhelming majority or minority that would make the use of accuracy as a metric untenable. Accuracy is defined as

$$\text{Accuracy} = \frac{(\text{True positives} + \text{True negatives})}{(\text{dataset size})}$$

ANALYSIS

Data exploration

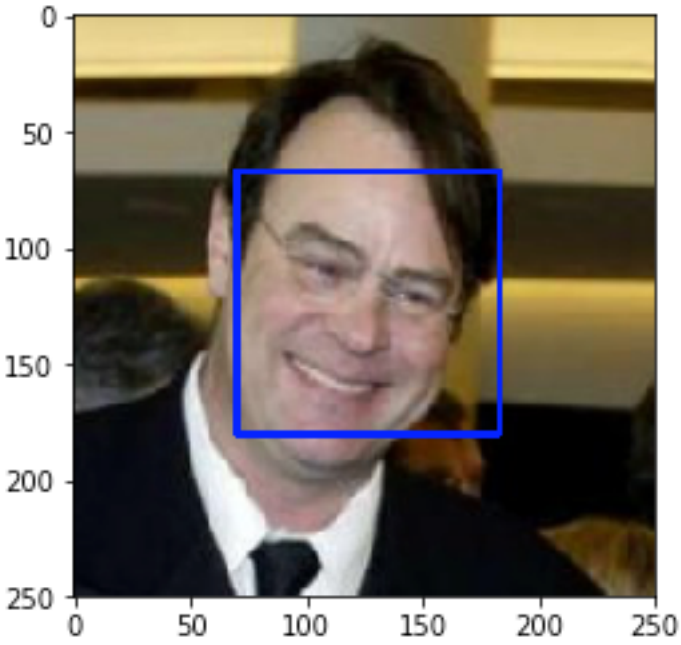
There are two datasets and five pre-trained models available to us.

Dogs dataset : Dog dataset provided by Udacity deep learning v2 PyTorch repository. (https://s3-us-west-1.amazonaws.com/udacity-aind/dog-project/dogImages.zip)

Humans dataset : Human dataset provided by Udacity deep learning v2 PyTorch repository. (http://vis-www.cs.umass.edu/lfw/lfw.tgz)

There are 13233 total human images and 8351 total dog images.
There are 6680 dog images in the train folder and 836 dog images in the test folder and 835 dog images in the valid folder.

An example of a human image and a dog image with their characteristics is given below

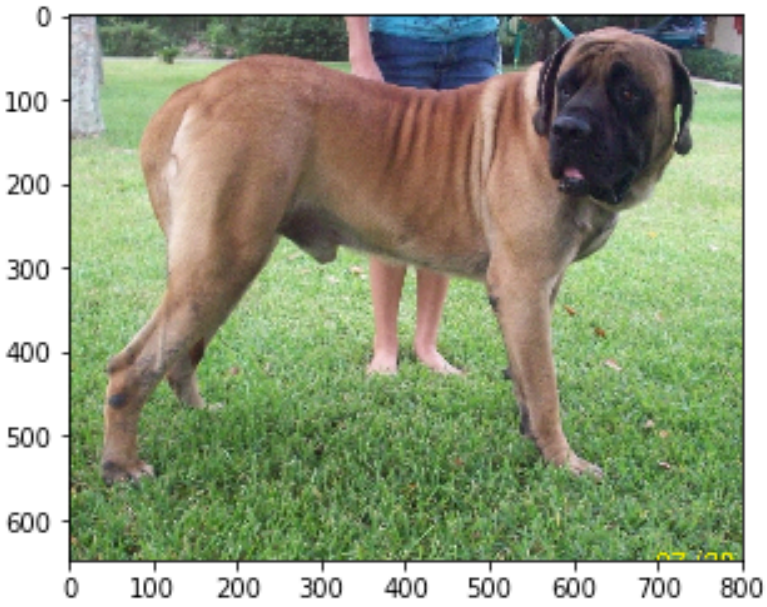



```
In [16]: from PIL import Image
         img = Image.open(human_files[0])
         img.size

Out[16]: (250, 250)

In [19]: img.mode

Out[19]: 'RGB'
```

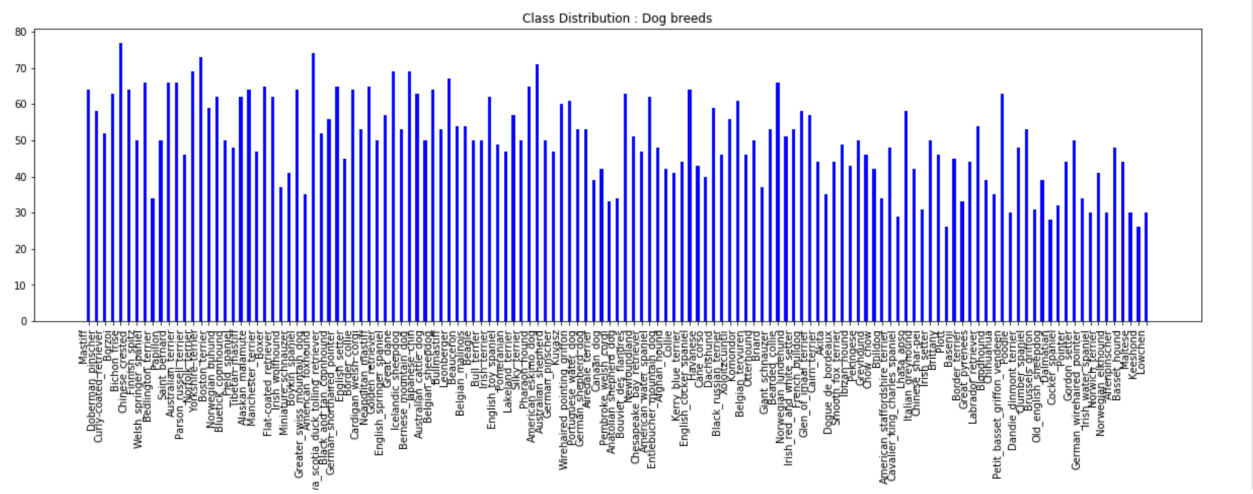



```
In [21]: img.size
Out[21]: (800, 648)

In [22]: img.mode
Out[22]: 'RGB'
```

Exploratory Visualization

There are 133 breeds of dogs and the breed-wise class distribution of the dogs is as shown below



The above class distribution of the training data set shows that the data is not exactly balanced . There is a definite variation in the count of different breeds available and this shall have an effect on our accuracy.

IMPLEMENTATION

Algorithms and Techniques

The classifier is a Convolutional Neural Network, which is the state-of-the-art algorithm for most image processing tasks, including classification.

A neural network consists of layers of interconnected nodes that interact with one another to find ever increasing patterns amongst data to eventually make calculated predictions in the output layer.

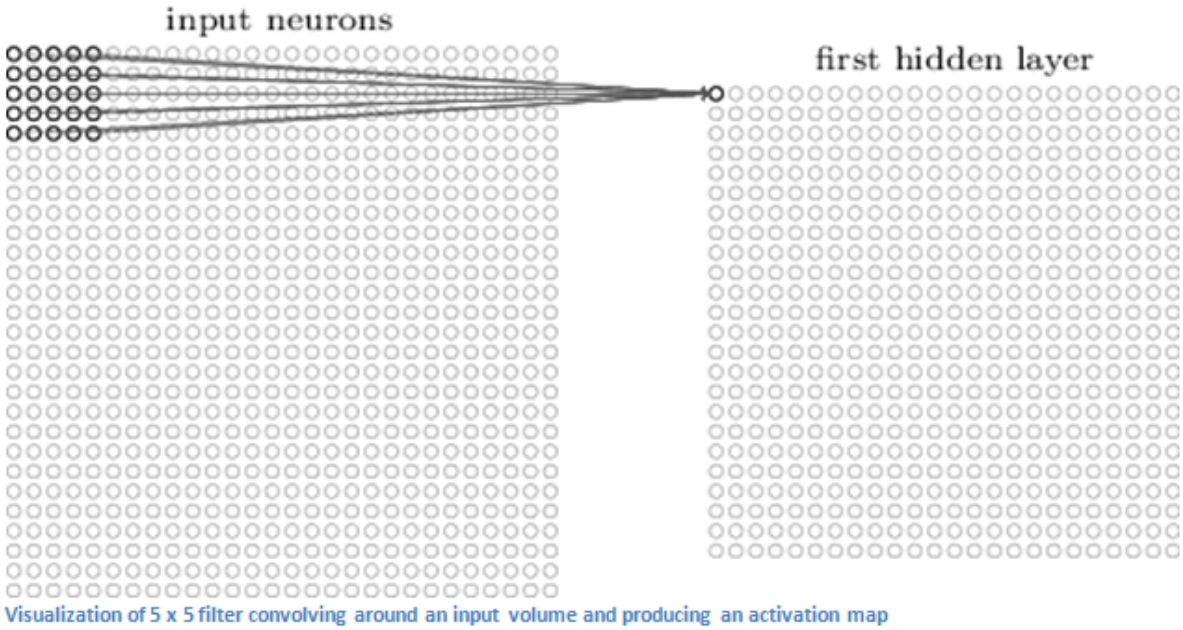
A Convolutional Neural Network is an artificial deep learning neural network.

Convolutional Neural Network architecture consists of four layers:

- Convolutional layer - The convolutional layer is designed to identify the features of an image.
- Rectified Linear Unit layer (ReLu) - This layer is an extension of a convolutional layer. The purpose of ReLu is to increase the non-linearity of the image. It is the process of stripping an image to provide a better feature extraction.
- Pooling layer - It is designed to reduce the number of parameters of the input i.e., perform regression.
- The Connected layer - It is a standard feed-forward neural network. We initially implement a CNN from scratch and then implement a CNN using transfer learning.

In case of image recognition CNN perceives an image as a volume, a three-dimensional object. Digital color images contain Red-Blue-Green ie RGB encoding. CNNs understand images as three distinct channels of color stacked on top of each other. It groups pixels and processes them through a set of filters designed to get certain kinds of results. The filter operation is explained below. The number of filters applied usually depends on the complexity of an image and the purpose of recognition. Pooling layer is designed to reduce the number of parameters of the input, i.e., perform regression to minimise the computational complexity.

To understand the convolution layer consider an input image ie an array say 32X32X3 array of pixel values. To detect features a convolution operation of CNN is nothing but moving a filter which is an array of values with the same depth for eg 5X5X3 moving across the image. As the filter is sliding, or convolving, around the input image, it is multiplying the values in the filter with the original pixel values of the image (computing element wise multiplications). This leaves us with a 28 x 28 x 1 array of numbers, which we call an activation map or feature map.



Visualization of 5 x 5 filter convolving around an input volume and producing an activation map

(Image courtesy https://adeshpande3.github.io/A-Beginner%27s-Guide-To-Understanding-Convolutional-Neural-Networks/

When we train a model with a CNN we have the following hyper parameters which we can tune and re-run the training till we achieve satisfactory results.

- Learning Rate - If the learning rate is too less the model takes a lot of time to reach the ideal state and if it is too large the model might not converge.
- Number of Epochs - This is the number of iterations to train a model. This parameter has to adjusted in such a way that the validation error keeps decreasing.
- Batch Size - has an effect on the resource requirements of the training process, speed and number of iterations in a non-trivial way.

Benchmark and Accuracy

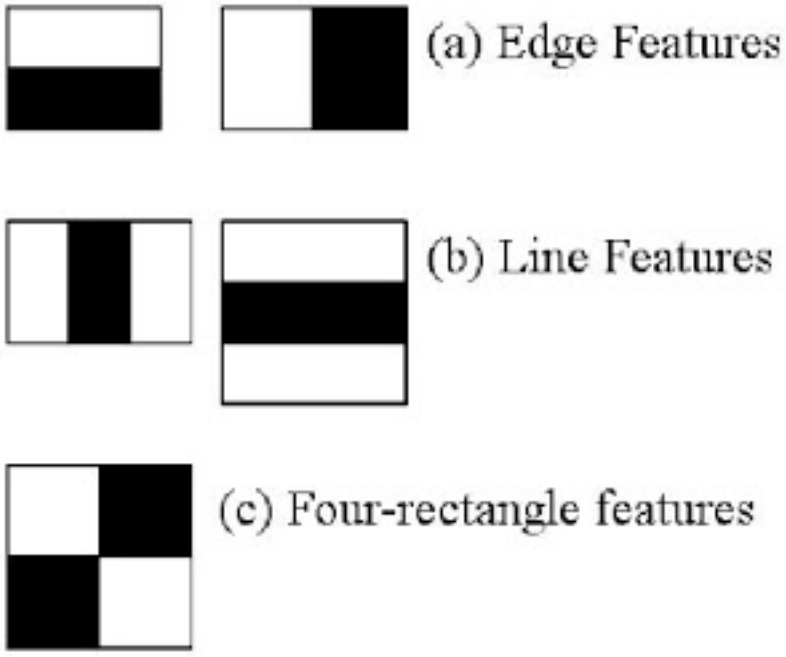
We use the provided accuracy benchmarks int he notebook aiming for a higher than 10% accuracy for the CNN built from scratch and higher than 60% accuracy for the CNN built using transfer learning.

Methodology

The following is the methodology used to realize our solution.

1. Import Datasets. We shall import the given dog and human datasets.
2. Detect Humans. We shall use OpenCV's implementation of Haar feature-based cascade classifiers to detect humans.
    a. Haar feature-based cascade classifiers is an effective object detection method.
    b. It is a machine learning based approach where a cascade function is trained from a lot of positive and negative images.
    c. The the algorithm is trained on a lot of positive images (images of faces) and negative images (images without faces) and then features are extracted from it.

d. These features which are extracted are called Haar features and are as shown below



e. The algorithm applies each and every feature on all the training images. For each feature, it finds the best threshold which will classify the faces to positive and negative.

f. To minimise the errors or misclassifications, we select the features with minimum error rate, which means they are the features that most accurately classify the face and non-face images.

g. The concept of Cascade of Classifiers makes the process efficient by discarding any non face region in the first shot prior to applying the features.

3. Detect Dogs. We shall detect the Dog faces using VGG-16 model.

a. The VGG-16 model is a pre-trained model that is available as a part of the PyTorch package's pre-trained models.

b. This model has been trained on ImageNet which is a very large, very popular dataset used for image classification and other vision tasks.

c. Pre-Processing :

In the preprocessing step we convert the image into 'RGB' mode and then resize the image to (244,244) from the original image size. We also convert the image array into a tensor and normalise the same as required by the documentation of the VGG-16 model.

d. Finally we make us of the PyTorch VGG-16 predict function to get the prediction if the provided image contains a dog face.

4. Breed Classifier : For classifying we the detected dog face according to its breed we have experimented with first creating a CNN from scratch and then using the concept of transfer learning to use a pre-trained model (VGG-16) in this case.

In both the cases we use the preprocessing steps outlined in the above para to transform the image prior to training the model.

We had to adapt the VGG-16 model to output the breed prediction among 133 breeds by changing the final layer of the VGG-16 model. Post this change the next section describes the effort in getting a satisfying accuracy for the model.

5. Tuning HyperParameters and experimentation . We have found that the key to getting desirable accuracy using either approach is the experimentation philosophy in which we tune the hyper parameters and then train and test the model.

We found that experimenting with the learning rate and the number of epochs for training gave us the best possible results for the model.

6. Algorithm. No we shall write an algorithm that accepts a file path to an image and first determines whether the image contains a human, dog, or neither. Then,

- if a **dog** is detected in the image, return the predicted breed.
- if a **human** is detected in the image, return the resembling dog breed.
- if **neither** is detected in the image, provide output that indicates an error.

7. Testing. Finally we shall test this algorithm on at least six sample images.

CONCLUSION

We were able to implement a program that could accept an image as an input and detect whether the image contained a dog face or a human face. If the image contained a dog face it would proceed to identify the breed of the dog using the classifier we built using transfer learning and trained to get a 84% accuracy. If the image contained a human face it would output the closest matching dog-breed. We found that tuning the hyper parameters and selecting the best pre-trained model (VGG-19) in our case and the length of training plays a great part in getting better results.