Machine Learning Engineer Nanodegree

Capstone Proposal: Dog-Breed Classifier

Yash Naik
August 2, 2020

Domain Background

Face detection has been an upcoming field in computer vision for the last decade. It has gained a lot of popularity recently as more and more technologies are adoption facial recognition as a tool to ensure security. The task of face detection is completely different in humans and machines. For machines, faces are just numbers representing pixels, arranged mathematically in, what are called neural networks.
This project is based on a real-world application. It is part of Udacity's Machine Learning Engineer nanodegree program. This Machine Learning program can detect a human face and also identify the breed of a dog, given a picture.

Problem Statement

This is Convolutional Neural Networks project, which when given an image of a dog should provide an estimate of the dog's breed. On the other hand, if given an image of a human, the program should identify the resembling dog breed. The goal of this project is to understand the challenges involved in "piecing together various Machine Learning models" [1] in order to perform several tasks in a Data process Pipeline" [2].

Datasets and Inputs

In this Project, we have been supplied with two datasets by Udacity-

a) Dog Dataset: This dataset contains (.jpeg) images of 133 different breeds of dogs, around the world.
b) Human Dataset: This dataset contains images with many different human faces, with a view to train our model to identify a dog breed that closely resembles that human face.

While exploring these two datasets, I have found that,

- There is a total of 8,351 dog images.
- Of the total dog images, 80% (i.e. 6,680) images are for training the model
- 10% of the remaining dog images i.e. (10% of 1,671) around 835 images for validation
- Remaining 10% of the total dog images (i.e. 836 images) for testing our model
- There is a total of 13,233 human images

Solution Statement

This project is divided into different tasks – Face detector, Dog Detector, creating a CNN from Scratch, improving CNN using transfer learning, writing and testing the dog breed prediction algorithm.
First, the face detector works on “Haar-Cascade classifiers” [3] using pretrained OpenCV model to detect human faces. Second, a dog detector algorithm is created which detects whether a dog is present in an image. This dog identifier works on a pretrained VGG-16 model. Third, I will be creating my own CNN (Convolution Neural Network) from scratch that will be trained on thousands of dog images which contain 133 different breeds. This custom CNN will need to achieve more than 10% test accuracy to meet Udacity’s minimum criteria. Fourth, I will be using Transfer Learning to increase the overall test accuracy of my CNN to at least 60% to meet Udacity’s requirement. Finally, I will write an algorithm that takes in an image and predicts whether a human face is detected, or a dog is detected and further classifies the breed of that dog.

Benchmark model

In order to effectively tackle the given problem, I will be utilizing several different Machine Learning models and techniques at hand, for different phases of the project.
For the dog detector, I will be training a VGG-16 model with pretrained weights on ImageNet dataset. For the scratch CNN model, I propose to use ResNet-152 model to train the dog images because unlike VGG, this does not hugely increase the training time when the number of layers for the neural network are increased. Resnet works on the principle of "Identity shortcut connection" [4] and skips 1 or more layers, to prevent overfitting and thereby, decreasing training time.

Evaluation Metrics

For this project, we will be only concerned with the “accuracy metric” on the test data for both human and dog images.

$$\textbf{Accuracy} = (\mathrm{TP} + \mathrm{TN})/(\mathrm{TP} + \mathrm{TN} + \mathrm{FP} + \mathrm{FN})$$

TP: True Positives
TN: True Negatives
FP: False Positives
FN: False Negatives

Project Design

- **Step 0:** Importing the two separate datasets containing human images and Dog images.
- **Step 1:** In this step, we can observe a sneak-peak of the data and divide the data into separate sub-datasets for training, Validation and testing.
- **Step 2:** Detecting human faces in the images with Haar-Cascade Classifiers using OpenCV's implementation.
- **Step 3:** Identifying if a dog is present in an image using a pretrained Vgg-16 model.
- **Step 4:** Creating my custom CNN from Scratch to predict the breed of dogs in the images.
- **Step 5:** Increasing the test accuracy of my custom CNN by using transfer learning.
- **Step 6:** Writing my custom algorithm that takes in an image as a parameter and detects whether the object present in that image is a human or a dog or neither.
- **Step 7:** Testing the algorithm by passing in different images containing outliers and corner cases, etc.

References

[1] [2] Dog Breed Classifier; Machine Learning Engineer, Udacity; May 2020; https://classroom.udacity.com/nanodegrees/nd009t/parts/2f120d8a-e90a-4bc0-9f4e-43c71c504879/modules/2c37ba18-d9dc-4a94-abb9-066216ccace1/lessons/4f0118c0-20fc-482a-81d6-b27507355985/concepts/65160313-7054-4ffb-8263-793e2a166d69

[3] Face Detection Using Haar Cascades; Mordvintsev, Alexander & K. Abid; 2013; Revision 43532856; https://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_objdetect/py_face_detection/py_face_detection.html

[4] Residual Blocks- Building Blocks of ResNet; Sahoo, Sabyasachi; November, 2018; https://towardsdatascience.com/residual-blocks-building-blocks-of-resnet-fd90ca15d6ec