Capstone Project

Machine Learning Engineer Nanodegree

Mirza Jhanzaib Iqbal

November 17, 2019

CNN Project: Dog Breed Classifier

Domain Background

Convolutional Neural Networks (CNN) project “Dog Breed Classifier” is a Udacity suggested Project. The aim of the project was to develop an algorithm where the code will accept any user supplied image as an input in order to identify dog breeds from images, If a dog is detected in an image, it will provide an estimate of the dog’s breed. and If supplied an image of a human, the code will identify the resembling dog breed.

Problem Statement

In this project, datasets were provided with images of dogs as well as labels describing the breed of the dog shown in the image. We are looking to create a solution trained on these labeled images of dogs to use on unlabeled dog images and still be able to determine the type of dog shown in the image.

Datasets / Inputs

Our datasets, which I plan to use were provided by Udacity in pre-filled Jupyter notebook (**dog_app.ipynab)** which includes in total **8351** dog & **13233** human images

File descriptions

- **/dog_images**: Dog Dataset Images
- **/lfw:** Human Dataset images

The most important file is **/dog_images** training set images, we have 8351 training examples

Solution Metrics

This project proposed breed classification based on convolutional neural network. The model is extended by applying transfer learning on the given datasets that shifts pre-trained Visual Geometry Group (VGG) model to the next model and consequently increased the resilience and efficiency of the model. Subsequently, the model is tested by the given data sets to validate its accuracy by returning the corresponding dog breed of human or dog images.

Benchmark Model

Conventionally, face recognition and breed classification relied upon hand-crafted features, such as edges and texture descriptors, combined with machine learning techniques, such as principal component analysis, linear discriminant analysis or support vector machines. Conventional face recognition methods have been surpassed by deep learning methods based on convolutional neural networks (CNNs) with high accuracy and robustness acquired by learning from actual deviations appearing in the images.

For the sake of benchmarking, we will restrict the CNN to 10 - 20 Epochs and extend it using Transfer learning that will be used to train and validate the classification of dog breeds**.**

Evaluation Metrics

We are only going to focus on an accuracy score. The goal of what we're looking to do here is: we want to see how well we can do at classifying breeds of dogs and evaluate the quality of the classifier by asking it to predict labels for a new set of images. We will then compare the true labels of these images to the ones predicted by the classifier and Accuracy will be able to tell us in a simple and easy-to-understand way how well our deep learning model is performing in this regard.

Project Design

The official Project design criteria used in **dog_app.ipynab** defined as following:

- Step 0: Import Datasets
- Step 1: Detect Humans
- Step 2: Detect Dogs
- Step 3: Create a CNN to Classify Dog Breeds (from Scratch)
- Step 4: Create a CNN to Classify Dog Breeds (using Transfer Learning)
- Step 5: Write your Algorithm
- Step 6: Test Your Algorithm

Step 0: Import Datasets

Required human and dog datasets will be downloaded.

Step 1: Detect Humans

In this section, we will use OpenCV's implementation of Haar feature-based cascade classifiers to detect human faces in images.

Step 2: Detect Dogs

We will use a pre-trained model of pytorch to detect dogs in images.

Step 3: Create a CNN to Classify Dog Breeds (from Scratch)

In this step, we need to create a CNN from scratch to classify the dog breeds and the objective is to achieve an accuracy of at least 10% by training and validating the models.

Step 4: Create a CNN to Classify Dog Breeds (using Transfer Learning)

At this stage, we will be using transfer learning to create a CNN that can identify dog breed from images and the objective here would be to achieve an accuracy of at least 60%

Step 5: Write your Algorithm

We need to write an algorithm that accepts a file path to an image and first determines whether the image contains a human, dog, or neither. Then,

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

Step 6: Test Your Algorithm

At the final stage, we will test using trained datasets to check either we are having a desired accuracy result or not.

References:

http://ijarcet.org/wp-content/uploads/IJARCET-VOL-5-ISSUE-12-2707-2715.pdf
https://www.rivas.ai/pdfs/mulligan2019dog.pdf
http://pytorch.org/docs/
https://medium.com/@hazutecuhtli/udacity-dog-breed-classifier-96da5e492f96
https://habr.com/en/post/447732/
https://github.com/udacity/deep-learning-v2-pytorch/tree/master/project-dog-classification
https://programqa.com/question/53942331/
https://github.com/pytorch/pytorch/issues/15563
https://s3-us-west-1.amazonaws.com/udacity-aind/dog-project/dogImages.zip
https://s3-us-west-1.amazonaws.com/udacity-aind/dog-project/lfw.zip
https://www.youtube.com/watch?v=umGJ30-15_A
https://www.researchgate.net/publication/325384896_Modified_Deep_Neural_Networks_for_Dog_Breeds_Identification
https://arxiv.org/pdf/1811.00116.pdf