Capstone Project **Purvansh Singh**

Machine Learning Engineer Nanodegree July 3th, 2020

Dog Breed Classifier using CNN

Domain Background

Breed classifier is a well-known Machine Learning Problem, The problem consist of identifying the breed of dogs if dog image is given as input, if we supply image of a human then we have to identify the resembling dog breed. We have to build a model that can process real world user supplied images and identify an estimate of the canine's breed. This is a multi-class classification problem where we can use supervised machine learning to solve this problem. This project gives me an opportunity to build and deploy ML models, so I have chosen this as my capstone project. And major idea behind this is I am very curious to work on Neural Networks, so I choose this as my Project.

Problem Statement

The goal is to build a ML model that can Predict dog breed based on passed real-world, user-supplied images. The algorithm has to perform two tasks:

Dog face detector and ***Human face detector***

Datasets and Inputs

Here We Have used the dataset provided by Udacity, *Dog images dataset:* The dog image dataset has 8351 total images which are sorted into train (6,680 Images), test (836 Images) and valid (835 Images) directories. The images are of different sizes and different backgrounds, some images are not full-sized. The data is not balanced because the number of images provided for each breed varies.

Human images dataset: The human dataset contains 13233 total human images which are sorted by names of human (5750 folders). All images are of size 250x250. Images have different background and different angles. The data is not balanced because we have 1 image for some people and

many images for some.

Solution Statement

We are going performing this multiclass classification, we can use Convolutional Neural Network to solve the problem, and to detect dog-images we will use a retrained VGG16 model. Finally, after the image is identified as dog/human, we can pass this image to a CNN which will process the image and predict the breed that matches the best out of 133 breeds.

Benchmark Model

- ✓ The CNN model created from scratch must have accuracy of at least 10%. This can confirm that the model is working because a random guess will provide a correct answer roughly 1 in 133 times, which corresponds to an accuracy of less than 1%.
- ✓ The CNN model created using transfer learning must have accuracy of 60% and above.

Evaluation Metrics

We are going to use Multi class log loss to evaluate the model. Because of the imbalance in the dataset, accuracy is a not a good indicator here to measure the performance. Log loss takes into the account of uncertainty of Prediction based on how much it varies from actual label and this will help in evaluating the model.
Other than this I will use accuracy as the measurement of choice to improve our model after each iteration. Accuracy is simply how many times our model predicts the dog breed correctly.

Project Design

Step 1: Import the necessary dataset and libraries, Pre-process the data and create train, test and validation dataset. Perform Image augmentation on training data.

Step 2: Detect human faces using OpenCV's implementation of Haar feature based
cascade classifiers.

Step 3: Create dog detector using pretrained VGG16 model.

Step 4: Create a CNN to classify dog breeds from scratch, train, validate and test the model.

Step 5: Create a CNN to Classify Dog Breeds using Transfer Learning with resnet101 architecture. Train and test the model.

Step 6: Write an algorithm to combine Dog detector and human detector.

- ✓ If human is detected in the image, return the resembling dog breed.
- ✓ If dog is detected in the image, return the predicted breed.
- ✓ If neither is detected, provide output that indicates the error.

References

1. *Original repo for Project - GitHub: https://github.com/udacity/deep-learning-v2- pytorch/blob/master/project-dog-classification/*
2. *Resnet101: https://pytorch.org/docs/stable/_modules/torchvision/models/resnet.html#resnet101*
3. *Pytorch Documentation: https://pytorch.org/docs/master/*
4. *https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural- networks-the-eli5-way-3bd2b1164a53*
5. *http://wiki.fast.ai/index.php/Log_Loss*