CAPSTONE PROJECT
Machine Learning Nanodegree

Ameya Shete
25th June 2021

CNN Project: Dog Breed Classifier

Domain Background

The Dog Breed Classifier using Convolutional Neural Networks is a Udacity suggested project in which we were tasked with first classifying if the species in an image was a dog or human and then subsequently classifying the breed of the dog or which breed the human most resembled.

The main challenge of this project was that most dogs have basic features such as four legs, two eyes, tail etc. so we as data scientists needed to come up with an extremely granular method to differentiate between breeds of dogs. Therefore, the main differences we had to use in differentiating between different breeds of dogs was size, color shape.

This project also has many real-world applications. Not only could a project of this nature be applied to classify dogs, we can only apply this to many other species in which there are different breeds such as cats, horses, birds etc. The only requirement for such a project would be that there should be many breeds for each animal and that there are enough images of each breed in the training set so that the model can effectively classify the breed of the animal. Furthermore, this ML classifier would be helpful in the zoology field as it would help in finding out how many animals of a certain breed currently live in the wildlife based purely from satellite images. There is no doubt that such a classifier would help in saving time and resources in zoology and biodiversity fields.

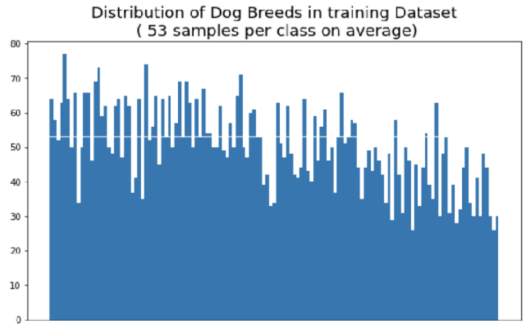
Problem Statement

The problem statement is that given an image of a dog, the algorithm we develop should correctly classify the breed of the dog and thus comes under the realm of fine grained classification. The main challenges of his problem are that at a high-level, most dogs look the same except for their size, shape and color as there are many different dog breeds and there also may be a lot of noise in the images of humans, trees or objects in general in the background.

Convolutional neural networks is the method implemented in this project which develops features to use for classification from the pixels in the image. Furthermore, I also implement a transfer learning approach to converge at a higher performance level, enabling more accurate output since the starting point of a transfer learning approach is better and the learning rate is also faster. An advantage of the transfer learning approach is that the same model can be used for different datasets so this algorithm used to classify dogs can also be used to classify cats or horses. This is due to the fact that the desired performance of the model is reached faster as we use a pre-trained model.

Datasets and Inputs

The datasets I have used in this project are from the Jupyter notebook that Udacity provided, dog_app.ipynb. The datasets have 8351 dog images and 13233 human images. The number of images in the dataset is also interesting because there seems to be a bit of class imbalance within the dog breeds. Class imbalance negatively affects the sampling of the data and also will affect the metric as there are better metrics that can be employed when we have an imbalance class. In this case, we find that there are, on average, 53 samples per class with slight variations within the classes.



The 8351 dog images provided were the images we used for training and within the folders of the dog and human images, the images were already divided up into training, testing and validation sets.

Solution metrics

The project proposed breed classification using convolutional neural networks. The model implements transfer learning on the datasets and shifts a previously trained Visual Geometry Group (VGG) model onto the next model which increases the accuracy of the model. Finally, the VGG models are tested on the test datasets to measure the accuracy of the model in classifying the correct dog breed from the given image.

Benchmark Model

Typically in the past, image classification problems have relied on extracting features by hand such as edges and texture descriptors combined with ML methods such as PCA or linear discriminant analysis. This has now been replaced by CNNs and is far more accurate due to the fact that the model now learns from actual differences in the images.

Therefore, one benchmark model that we can use would be a CNN model developed from scratch in which the model guesses randomly which breed the dog belongs to and so will have a test accuracy of about 10%. We will use transfer learning later in the model we actually want to implement and thus can compare the accuracy of this new model with the benchmark model of a CNN model built from scratch.

Evaluation metrics

The only evaluation metric that we use is accuracy score. The accuracy score is determined by looking at the predicted label for each image and comparing it with the true label. Finally, the accuracy score will be

a measure of how many of the images were correctly labeled as a percent of the total number of images. Accuracy is an easy to understand metric and is quite applicable as well given the simple nature of what we are trying to achieve in this project.

Project Design

Step 0: Import Datasets

The datasets are downloaded.

Step 1: Detect Humans

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

Step 2: Detect Dogs

In this section, we use a pre-trained model to detect dogs in images.

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

In this section, we develop our CNN model from scratch to predict breed from the images and aim to have at least a 10% accuracy by training and validating models.

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

In this section, we use transfer learning to create a CNN that can identify dog breeds from images. Our CNN model must attain at least 60% accuracy on the test set.

Step 5: Write your Algorithm

In this section, we 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.

Step 6: Test Your Algorithm

In this section, we test our algorithm!