Capstone Project Proposal
Machine Learning Engineering Nanodegree
October 2020

Classifying Dog Breeds

Domain background
Computer vision is a field in which machine learning and in particular deep learning has been able to provide significant breakthroughs. Through the use of Convolution Neural Networks, the computer is able to 'learn' about features in the images effectively. They were inspired by biology, where the connectivity pattern between neurons is similar to that of the visual cortex.
Classifying dog breeds is a well known problem and will be used to illustrate the above. This project will allow me to learn about transfer learning, as we will use various pre-trained models. As future work, I would look to focus on deployment and create a web app utilising AWS SageMaker.

Problem statement
The problem is a multi-class classification problem in the field of computer vision. There are two aspects to the problem:
1. Given an image of a dog, we want our model to accurately predict the dog's breed
2. Given an image of a human face, we want our model to identify the closest resembling dog breed.

Datasets and inputs
The data have been provided by Udacity. The input type should be images, as we want to detect and classify dog breeds.

Dog images dataset: The data has been already split into three folders, for train, test and validation purposes. Each folder is further subdivided into 133 folders representing the different dog breeds we are seeking to classify. Note that we are dealing with imbalanced data as there is not the same number of images for each class (breed).

Human images dataset: The data has been split into folders according to names. There are a total of 13233 human face images divided into 5749 folders. Again we note that there is not the same number of images in each folder.

The preprocessing steps are as follows:
- the human images will need to be converted to grayscale before using OpenCV's Haar Cascade classifier and a bounding box for each detected face will need to be established
- The dog images will also need to be converted to grayscale and resized to 224 x 224 pixels. We will need to transform the input to a 4D tensor to use Keras CNNs.

Solution statement

We will use Convolution Neural Networks (CNN) to solve this problem due to their known success in the field of computer vision. We will use existing algorithms and pre-trained models to ensure better results (making use of transfer learning tools.)
In particular, we will use the following:

- To detect human faces, we will use OpenCV's Haar Cascades classifiers
- To detect dog images, we will use a pretrained VGG16 model, which was trained on the ImageNet dataset
- To detect breeds, we will both implement a CNN from scratch and use a pre-trained Resnet-101 model.

Benchmark model

A random guessing model would accurately predict the dog breed once in 133, as there are 133 classes. This would give an accuracy score of less than 1%.
We would expect our CNN model implemented from scratch to achieve at least 10% accuracy on the test set. Our CNN model which uses transfer learning should achieve at least 60% accuracy.
Whilst we appreciate accuracy is not an ideal metric due to the class imbalance, it remains useful for benchmarking purposes.
In terms of AUC, the score is between 0.5 and 1. We note that 0.5 is no discrimination i.e. a random choice. We therefore hope for a score of at least 0.65.

Evaluation metrics

The task is a multi-class classification problem and as such we will use cross-entropy loss and ROC/AUC s our evaluation metrics.
We will also consider precision and recall.
Due to class imbalance, accuracy is not a good choice of metric.

Project Design

To complete the project, we must undertake the following steps:

1. Import the necessary libraries and packages
2. Import the data and complete preprocessing
3. Explore the data and draw insights, in particular investigate class label distribution
4. Detect human faces (using OpenCV's Haar cascade classifiers)
5. Detect dogs (using a pretrained VGG16 model)
6. Create a custom CNN to classify dog breeds from scratch
7. Create a CNN to classify dog breeds using transfer learning
8. Model refinement, optimise hyperparameters
9. Design final algorithm to return result