# Dog Breed Identification

**Capstone Project Proposal**

Grzegorz Lippe

February 23, 2021

## Domain Background

The task of identifying dog breeds is a known topic amongst the community for machine learning. I've chosen the dog breed identifier amongst the proposed projects, because it attracted me the most.

The benefit of the dog breed problem is the high availability of labeled datasets which can be used for training and validation of the model. A quick search on kaggle and github provide lots of sources.

Also I must admit that it is the most supported task on Udacity's site.

## Problem Statement

The goal of this project is to provide an app, that can take an arbitrary picture and return a statement of a most likely dog breed. If a picture of a cat or human is provided, the app should identify the human, but still provide a statement of a most similar looking dog breed.

## Datasets and Inputs

The data consists of two different sets, provided by Udacity and are described as follows:

**dog dataset** The dog data set is already divided into three different datasets (folders) for test, training and validation. Each folder contains sub directories of the specific formating `DDD.breed_name`, where `DDD` represents a 3 digit number, followed by the breed name after a dot. Each subfolder contains a hand full of images of the specific breed.

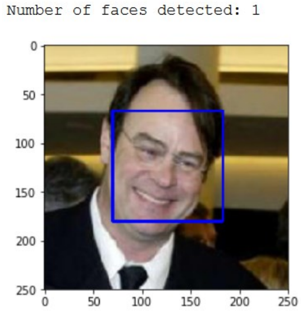
Overall there are 133 different dog breeds, and 835 images provided for validation, 6680 images for training and 836 images for testing.

The dog images contain a singular dog each, mostly of the whole dog, sometimes only of the snout.

**human dataset** The human dataset consists of 13233 images of 5749 persons, which are stored in a separate directory named after each celebrity. The images of the people are already cropped and centered around their face, but sometimes there are additional people in the background.

## Solution Statement

The human face detection will use an OpenCV model with a pre-trained haar cascade dataset of `haarcascade_frontalface_alt.xml` This algorithms provides bounding boxes for each face found in a given image, like shown in the figure below with an image of Dan Ackroyd:



For the dog breed detection a pre trained torch model `resnet50` is used within a transfer learning approach. The pre trained model will be trained on the described dog dataset until it reaches a suitable level of accuracy. Since the app shall provide some fun with human images the accuracy of the prediction may not be that high.

## Benchmark Model

As a benchmark Microsoft's website provides AI recognition for dog breeds and human faces:

- What dog will serve to compare the results of the implemented `resnet50` model to the results of the Microsoft classifier,
- and famous people will serve to benchmark the face detection.

## Evaluation Metrics

For the evaluation the labelled data is used. For example the accuracy of the face detection algorithm can be measured by feeding it with (previously labelled) images of humans:

$$Score = \frac{\text{Humans detected}}{\text{Images presented}}$$

A slightly modified algorithm can be used to evaluate the dog detecting model. The dog detector provides a breed identification on top, so in the first step the upper term can be used to judge the accuracy of the dog detection. Additionally the quality of the breed identification can be measured as follows:

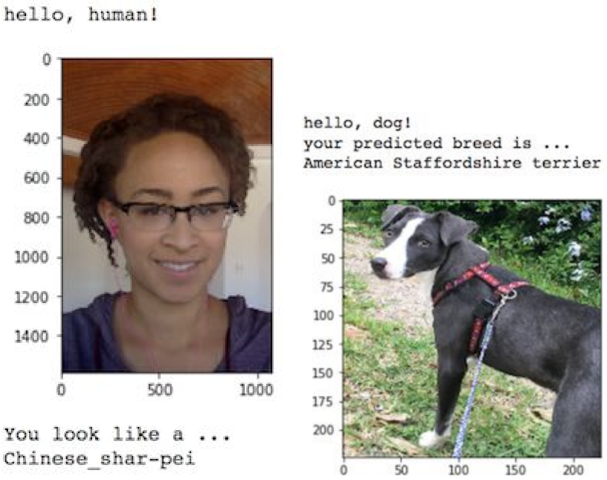
$$Score = \sum_{breeds} \frac{\text{correct breed identified}}{\text{breed provided}}$$

## Project Design

In the initial state the app will consist of a python function which will provide the following informations:

1. Was there any human detected on the image?
2. If so, provide a statement of a similar looking dog breed.
3. Are there any dogs on the Image? If so, provide the most likely breed
4. If neither humans, nor dogs where detected also say so.

The output may look as follows:



```
hello, human!

You look like a ...
Chinese_shar-pei
```

```
hello, dog!
your predicted breed is ...
American Staffordshire terrier
```