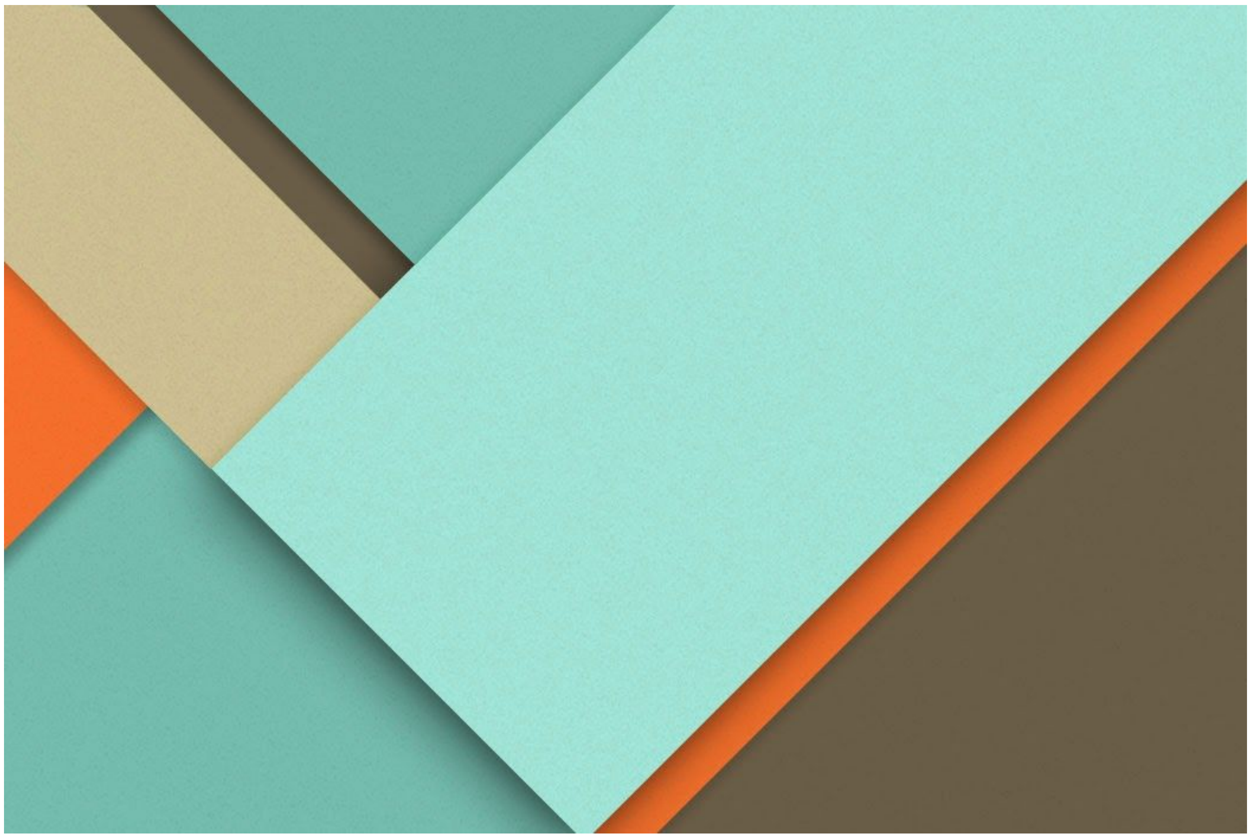

Dog Breed Classifier Capstone Proposal

10/10/2020

Raghad B.Hethnawi
Machine Learning Engineer Nanodegree

Domain Background

Many people have dogs, but don't actually know the breed of theirs. Dog breed classifier can help owners - especially new ones or inexperienced - about their dog's breed. This can be done by image classification, in the area of deep learning - which as a part of machine learning-. Worth noting that image classification is the process in computer vision which tries to classify images due to its visual contents, and Convolutional Neural Networks (CNNs) can be perfectly implemented to that role.

CNN is a type of neural networks that is mostly used for processing data in structure of girds. CNNs now form the backbone of many computer vision systems, and were used from hundred years ago until now to solve many problems related to various fields. The origins of CNNs have started from biological experiments in the 1950s. [1]

Problem Statement

Image classification is the process that starts from taking an input, applying some processes on it, and outputting it as a class. For example, taking an image as input, applying CNN to it, and output the result as a class of a dog. It's easy to us as humans to know that a specific picture represents something (a cat or a dog for example), but how can machines do that?

Datasets and Inputs

- Inputs: Input-type for this project is image
- Datasets: Provided by udacity, divided into two main parts, one for dogs, the other for humans.
    - Human Images: With a total of 13233 of human face images, distributed in many folders. Images are not fairly distributed among the folders, and the variation in the number of images in each folder can be noticed.
    - Dogs Images: Distributed into 3 main folders, named as 'train', 'test', and 'valid' for purposes of training, testing and validating the data respectively.

      Each of these folders is distributed into 133 different folders representing the dogs' breeds. Hence, our dogs dataset contains 133 different dog breeds.

        a. Train folder: contains 6680 total images for training. These images are divided among 133 dog breed folders (classes)
        b. Test folder: contains 836 total images for testing. These images are divided among 133 folders of dog breeds classes.

c. Valid folder: contains 835 images for validation, divided among 133 folders (dog breed classes)

So, we have a total of 6680+846+835 = 8351 dog images. Worth noting that the images are not fairly divided among dog breeds folders in the given dataset, and there's a variation in the images sizes.

Solution Statement

The proposed solution for this problem includes a trained model implemented as a Convolutional Neural Network (CNN) in order to detect whether the image is for a human or a dog. If the image is for a dog, it should predict its breed, and if it is for a human, the closest dog breed is identified.

CNN is a deep learning algorithm, which is going to take input images, assign learnable weights and biases to various objects and aspects in the image.

The solution follows the following steps:

1. Detect human images. Here, we can use existing algorithms (e.g. OpenCV's implementation of the Haar feature based cascade classifier)
2. Detect the dog images. For this step, we will use the pretrained VGG16 model.
3. After detecting whether the image is for a human or a dog, we can now pass that image into the CNN model, which will do some processing on it, and then detect the breed of the dog, or predict the closest dog's breed of a human out of 133 dogs breeds.

Benchmark Model

1. The CNN which is created from scratch must have at least accuracy of 10%. This informs that the model is working properly, where a random guess gives a correct answer roughly once in 133 times, giving a very low accuracy (less than 1%).
2. The CNN model which is created by the Transfer Learning should have accuracy of 60% or above.

Benchmark Model

For the proposed multi-class classification, the evaluation model will be based upon multi-class log loss. The log loss is going to take into the account the uncertainty of the prediction based on how much it varies from the actual label. This will help in evaluation of the model.

Accuracy could also be used in the evaluation process, where the accuracy percentage:

Accuracy = (Number of correctly predicted images/ Total Number of images) *100%

Project Design

1. Import the necessary datasets and libraries, provided by Udacity
2. Detecting the human images by OpenCV's implementation of Haar feature-based cascade classifier.
3. Detecting the dog's images using the pre-trained VGG16 model.
4. Implementing CNN to classify the dog's breed from scratch, then train, validate, and test the model
5. Implement CNN to classify the dog's breed using Transfer Learning, then train, and test the model
6. Implement an algorithm to integrate the dog's detector and human's detector.
    - Giving the dog's breed if the predicted output was a dog.
    - Giving the closest dog breed if the predicted output was a human
    - Giving an output indicating that the model can't detect the image, if it was neither a dog, nor a human

References

1. https://glassboxmedicine.com/2019/04/13/a-short-history-of-convolutional-neural-networks/
2. https://pytorch.org/docs/master/
3. https://www.kaggle.com/gpreda/haar-cascades-for-face-detection