Machine Learning Engineer Nanodegree

Capstone Proposal

Ibrahim Makhadmi
December 6, 2020

Dog Breed Classifier

Domain Background

Classifying dog breeds is a challenging task, as there are more than 300 dog breeds recognized by the Federation Cynologique Internationale (FCI)[1], and there are even different variations among the same breed. This shows the need for a more robust solution to classifying dogs than what a typical person could possibly achieve, and the ideal solution would be to implement a machine learning model to aid in the classification problem. Convolutional neural networks (CNNs) achieve state-of-the-art performance in multiclass image classification, so it will be a great method to use for this objective. Aside from implementing my knowledge gained in machine learning, I am excited to complete this capstone project because my wife and I have a dog whose breed is unknown to us, so this will also help give us an idea of what he could possibly be.

Problem Statement

The mission is to build a machine learning pipeline that can be integrated in a web application to process user-supplied images. The algorithm will be required to complete two different tasks. If the supplied image is of a dog, the output will be an estimate of the dog's breed. If the supplied image is of a human, though, the output will be an estimate of the closest resembling dog breed. If there is no dog or human in the image, the output will simply be an error message. Given the required output, the algorithm must also be able to perceive a dog or human in the provided image.

Datasets and Inputs

The input type required for this project is an image file, as this is what the model will require the user to supply for inference. The dataset of image files used in this project is provided by Udacity. The dataset includes a set of dog images and a set of human images.

The dog image dataset contains 8,351 images of 133 different dog breeds. The images are split into three sets; a training set with 6,680 images, a validation set with 835 images, and a test set with 836 images. Each of these subsets include images of the 133 different dog breeds to be classified. The images in the dataset are not uniform in size, shape or background. The number of images for each breed also vary, with some breeds having as low as 33 images in the training set, and as high as 80 images in the training set, so the data is not balanced.

The human image dataset contains 13,233 images of 5,749 different people. The number of images for each person varies from as low as 1 image in the dataset to as high as more than 10 images in the dataset, so the data is not balanced, but the objective is not to identify each person in the images; rather it is to identify a human face, so this is not an issue. The images in this dataset are uniform in size and shape, though, with each image being 250x250 pixels. The images are also centered around each person's face.

Solution Statement

In order to solve this multiclass classification problem, a convolutional neural network (CNN) will be implemented. A CNN keeps track of spatial information and learns to extract features in convolutional layers, in which a series of image filters are applied to the input image, and apply the best weights for each filter as it trains on the image dataset. Therefore, this model will be ideal to extract the best features to identify each of the 133 dog breeds it will be trained on. To begin, OpenCV will be used to detect human faces using the pre-trained Haar feature-based cascade classifier[2]. Secondly, the VGG-16 model that has been pre-trained on the ImageNet[3] dataset will be used to detect dogs in an image. After detecting whether an image contains a dog or human, the next step will be to create a CNN from the ground up to classify dog breeds and implement it. After testing the model to achieve an accuracy of at least 10% on the test set, another CNN will be created using transfer learning, which after training should attain an accuracy of at least 60% on the test set.

Benchmark Model

The benchmark for the two CNN models created, the first without transfer learning and the second with transfer learning, are given by Udacity to achieve an accuracy of 10% and 60%, respectively.

Evaluation Metrics

Accuracy will be the focal metric used to evaluate the solution models to the benchmark model required of them.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

Where TP = True Positives, TN = True Negatives, FP = False Positives, and FN = False Negatives.

Project Design

The workflow for approaching the solution of the given problem in this project are as follows:

Step 0: Import the datasets for the dog images and human images from their respective directories and verify the number of images in each dataset.

Step 1: Import the Haar feature-based cascade classifier to detect human faces in images from OpenCV's pre-trained model library, and verify it using a sample image.

Step 2: Import the VGG-16 model, that is pre-trained on the ImageNet dataset, to detect dogs in images, and verify it predicts the category indexes corresponding to dogs in the ImageNet class index (indexes 151-268).

Step 3: Create a CNN to classify dog breeds in a given image. Train, validate, and then test the model using the corresponding datasets.

Step 4: Create another CNN to classify dog breeds in a given image using transfer learning. Train, validate, and then test the model using the corresponding datasets.

Step 5: Develop an algorithm to accept an image file, and determine if the image contains a dog, human, or neither. After which, the output will return a predicted dog breed for images that detected a dog, return the closest resembling dog breed for images that detected a human, and return a simple error message if neither a dog nor human were detected in the given image.

Step 6: Lastly, test the algorithm with images from the dataset and possibly other dog images the model has not yet seen.

Bibliography

[1] FCI Breeds Nomenclature. http://fci.be/en/nomenclature/. Accessed 4 Dec. 2020.

[2] Face Detection Using Haar Cascades — OpenCV-Python Tutorials 1 Documentation. https://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_objdetect/py_face_detection/py_face_detection.html. Accessed 4 Dec. 2020.

[3] ImageNet. http://www.image-net.org/. Accessed 4 Dec. 2020.

[4] Udacity Dog Images Dataset. https://s3-us-west-1.amazonaws.com/udacity-aind/dog-project/dogImages.zip. Accessed 4 Dec. 2020.

[5] Udacity Human Images Dataset. https://s3-us-west-1.amazonaws.com/udacity-aind/dog-project/lfw.zip. Accessed 4 Dec. 2020.