MLND CAPSTONE PROJECT

DOG BREED CLASSIFIER

6-Jan-2021

Ricardo Castellanos Herreros

DOMAIN BACKGROUND

Recognising dogs according to their respective breed is a challenging task even for humans. There are hundreds of breeds in existence which are grouped into 10 distinct groups according to physicial characteristics. This shows that classification is a complex task as remembering all the breeds and distinguishing between similar breeds cannot be easily done by humans. Therefore, the need for this to be done by a computer is necessary. The task at hand involves image recognition and classification which uses machine learning techniques to aid computer vision. A convolutional neural network (CNN) will be used to accomplish this task.

PROBLEM STATEMENT

The aim of the project is to build a pipeline to process real-world, user-supplied images. The algorithm will identify an estimate of the dog's breed given an image. When the image is of a human, the algorithm will choose an estimate of a dog breed that resembles the human. If neither a dog or a human is detected, then an error message is output. Therefore, the models in place should be capable of detecting a dog or human in an image, classify the dog to its breed and classify a dog breed that the human resembles.

DATASETS AND INPUTS

For this project, the input format must be of image type, because we want to input an image and identify the breed of the dog. All the datasets used to train, test and validate the CNN model are provided by Udacity.

Dog images dataset: The dog image dataset has 8351 total images which are sorted into train (6,680 Images), test (836 Images) and valid (835 Images) directories. Each of this directory (train, test, valid) have 133 folders corresponding to dog breeds. The images are of different sizes and different backgrounds, some

images are not full-sized. The data is not balanced because the number of images provided for each breed varies. Some have four images while others have more.
Human images dataset: The human dataset contains 13233 total human images which are sorted by names of human (5750 folders). All images are of size 250x250. Images have different background and different angles. The data is not balanced because we have 1 image for some people and many images for some. The human dataset will be used to detect human faces in images using OpenCV's implementation of Haar feature-based cascade classifiers. The dog dataset is used to detect dogs in images using a pre-trained VGG-16 model.

SOLUTION STATEMENT

The project will first use OpenCV's implementation of Haar feature-based cascade classifiers to detect human faces from the images supplied. Next, a pretrained VGG-16 model with trained weights on ImageNet, a large popular dataset for image classification, will be used to detect dogs in the user-supplied images. The pre-trained model should be checked that it returns indexes from 151 to 268 inclusive, that is, it must include categories from 'Chihuahua' to 'Mexican Hairless'. A CNN model is then built from scratch to classify dog breeds, that is, transfer learning cannot be used just yet. This model should surpass a test accuracy of 10% set by Udacity because the model is being built from scratch so classifying similar breeds can be a challenge however transfer learning will greatly improve this. Finally, a transfer learning will be used with a ResNet50 model to significantly boost the accuracy of the CNN model. It should surpass the 60% test accuracy set by Udacity.

BENCHMARK MODEL

The CNN model created from scratch must have accuracy of at least 10%. The CNN created from scratch have accuracy of 14%, though it meets the benchmarking, the model can be significantly improved by using transfer learning. This can confirm that the model is working because a random guess will provide a correct answer roughly 1 in 133 times, which corresponds to an accuracy of less than 1%.

The CNN model created using transfer learning must have accuracy of 60% and above. With just 20 epochs, the model got 63% accuracy.

EVALUATION METRICS

Accuracy will be the main metric used to test both the benchmark model and the solution model.

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

True Positive(TP), True Negative(TN), False Positive(FP) & False Negative(FN).

PROJECT DESIGN

The workflow for the project to approach the solution is given below:

Step 0: The datasets for the human and dog images are imported and the total number of dog and human images is determined.

Step 1: OpenCV's implementation of Haar feature-based cascade classifiers is used to detect human faces from the datasets.

Step 2: Pre-trained VGG-16 model is used to detect dog images from the datasets.

Step 3: CNN model is created from scratch to classify the images. Test accuracy should be greater than 10%. Resizing and cropping the image. Augmentation through rotations.

Step 4: CNN model is created using transfer learning to classify the images. Test accuracy should be greater than 60%. Resizing and cropping the image. Augmentation through rotations. CNN layers are utilized for feature extraction and image generalization. The dropout layer ensures that overfitting is dealt with and the linear classifier converts the extracted features to a classified type.

Step 5: An algorithm is developed that returns the predicted breed if a dog is detected. If a human is detected, it should return the resembling dog breed. . If neither dog or human is detected, then error message is output.

Step 6: Testing the solution model with images from the datasets to see the

model in work.

BIBLIOGRAFÍA

[1] Original repo for Project - GitHub: https://github.com/udacity/deep-learning-v2- pytorch/blob/master/project-dog-classification/

[2] Udacity (2019).Udacity. [online] Available at: https://s3-us-west-1.amazonaws.com/udacityaind/dog-project/dogImages.zip [Accessed 5 Dec. 2020].

[3] Udacity (2019).Udacity. [online] Available at: http://vis-www.cs.umass.edu/lfw/lfw.tgz [Accessed Dec. 2020].

[4] Pytorch Documentation: https://pytorch.org/docs/master/

[5] Imagenet training in Pytorch: https://github.com/pytorch/examples/blob/97304e232807082c2e7b54c597615dc0ad8f 6173/imagenet/main.py#L197-L198