Udacity Machine Learning Nanodegree

Capstone Proposal

Tin Ko Ko Win

April 23, 2020

Dog Breed Classifier

Domain Background

The application of computer vision is very broad can be utilized in many fields like public security, healthcare, automotive, agriculture, industrial etc. And with the help of deep learning, much better performance and cloud computing like SageMaker and Azure, computer vision become much more practical and can even be used in real-time. As I prefer computer vision than other machine learning fields, I decided to choose dog breed classifier.

This project is one of the popular projects by Udacity in machine learning nanodegree program and the aim of this project to be able to create a web app that cat predict dog breed on input image and predict most resemble dog breed if the input image is human.

Problem Statement

This goal of the project is to detect and classify on the user input images, the problem statements in this project are..

1. Detection of human and dog and
2. Classification of dog’s breed

The final output will be imported image and print out that the input image is human or dog and dog’s breed even if the detect image is human.

Dataset and Inputs

Human Dataset

In human dataset, there is 13233 images of 5750 people with 250*250 pixels in all images. Some people got only one image and some got much more images. Because of we only want to detect human and only going to use OpenCV library, I won't split it into train, valid and test.

Dog Breed Dataset

Because the major goal of the project is to classify dog breed, I'll create CNN both from scratch and transfer learning. And that is why, Udacity pre-modified the dataset with train, valid and test splits. In this dataset there is 8351 dog images of 133 breeds that is 6680 images in train, 836 images in valid and 835 images in test. But the image sizes are not identical for each image.

Solution Statement

Firstly, I'll use OpenCV's implementation of Haar feature-based cascade classifiers to detect human in an image. OpenCV provide many haarcascades that is stored in github and I already downloaded haarcascade_frontalface_alt.xml and stored it in harcascade folder to detect images with detectMultiScale method. Then, I'll detect dog using pretrained VGG16 architecture that is trained on imagenet as I know that it has over 100 classes trained for dogs. After that, I'll build a CNN architecture for multiclass classification; this architecture is based on VGG16 architecture. It'll decreased the image size by half in every layer to fully connected layer. I am going to use 64, 128, 256 and 512 filters respectively in layers. I'll set kernel size 3 and padding 1 for convolutions layers and set kernel 2 and stride 2 in maxpooling to reduce image size in half. And for non-linear activation function, I used RELU. I created 3 fully connected layers and the final layer is to yield predicted class. I also added dropout with p=0.5 to avoid overfitting. Finally, I'm going to use transfer learning to get better performance.

Benchmark

Udacity set threshold for minimum 10 percentage accuracy in CNN that created by ourselves as classification of dog's breed is very difficult even for us and we will need bigger and better network like imagenet winners. And for transfer learning, the minimum threshold is 60 percentage of accuracy.

Evaluation Metric

Like any other machine learning approach, the dataset is split up into train, validate, and test dataset. The dataset I got is already split up and so I didn’t need to do it manually. These splits used in both scratch and transfer learning. The evaluation metric is the accuracy. For getting the best trained model during training, I just compared the best loss with the validation loss and assumed as best trained model so far if validation loss is less than best loss.

Best Loss = min (Best loss, Validation Loss)

Accuracy = corrected prediction / dataset size

Project Design

1. Import dog dataset and human dataset and preprocess the image and use augmentation techniques to prevent overfitting
2. Detect human using OpenCV haarcascade classifiers
3. Detect human using pretrained VGG16 model
4. Create CNN model from scratch
5. Create CNN model from transfer learning
6. Write My Algorithm
    a. If detect human print ‘Hey Guy!! You look like a…..’
    b. If detect dog print ‘This dog is a ……’
    c. If don’t detect human or dog print ‘Sorry! We do not detect a human or a dog in this image.’

References.

1. Write an Algorithm for a Dog Identification App from Kaggle
   https://www.kaggle.com/subhagatoadak/dog-breed-classifier-udacity/notebook

2. Liu W. et al. (2016) SSD: Single Shot MultiBox Detector. In: Leibe B., Matas J., Sebe N., Welling M. (eds) Computer Vision – ECCV 2016. ECCV 2016. Lecture Notes in Computer Science, vol 9905. Springer, Cham

3. Resnet101

   https://pytorch.org/docs/stable/torchvision/models.html?highlight=torchvision%20models

4. OpenCV tutorial

   https://docs.opencv.org/master/d9/df8/tutorial_root.html