Machine Learning Engineer Nanodegree Nilesh Sonawane

CAPSTONE PROJECT PROPOSAL May 01, 2020

CNN Project: Dog Breed Classifier

Domain Background

Dog Breed Classifier is a famous Image Classification problem in the Computer Vision domain. There are two tasks involved in this project, first to identify if the input image is a dog image or a human face, if it's a dog image then classify it to one breed of dog and if the image is human face then identify the most resembling dog breed associated to face. This is a multi-class classification problem. Main vision is to build a ML pipeline which will take an real world image from the user and output the specific dog breed. We can use Supervised ML techniques to solve this problem. Further, building a web app would be feasible for users to input the image and get predicted output. This project generates many challenging opportunities to try out, deploying this project workflow using AWS SageMaker would be my next plan after building a working model and hence the same becomes the reason I chose this project.

Problem Statement

Aim of the project it to build ML model that can be deployed and used within a web app to process real world, user-supplied images. Two main tasks:-

- *Face detector* - If given a Human face image, model will predict the most resembling dog breed associated to face.
- *Dog breed detector* - If given a dog image, model should predict its breed.

Dataset and Inputs

Inputs - Input type for this project must be an Image!
Dataset - Dataset is provided by Udacity. There are 13233 total human images.

There are 8351 total dog images. (Links to the datasets are hyperlinked). All the images in Human image data are of size 250x250 and are arranged in sorted names of humans in total 5750 folders, and the images in dog dataset are arranged in 133 folders with dog breed as the folder name, images are not consistent in size, images are also not consistent in number associated to each folder. Hence the dataset is not balanced!

Solution Statement

Given problem is a typical multi-class classification in visual recognition. This type of problems can be solved by using *Convolutional Neural Networks(CNN)*. CNN helps in efficient feature extraction from given images based on the filters used in form of weights and biases, these weights are learnable and in the same way CNN learns to classify given set of images in to multiple classes by extracting important spacial information. Solution involves 3 stages:

- Human face detection - We can use existing OpenCV's algorithms to detect face like Haar Cascades classifiers.
- Dog image detection - To detect dog-image we will use transfer learning in which we will make use of a pretrained VGG16 model which was trained on ImageNet challenge.
- Breed classification - After detection of dog/human face, we can create CNN which will output the predicted dog breed.

Benchmark Model

We can benchmark all stages of our project workflow separately as given below:-

- VGG16 model used in transfer learning must have accuracy of 60-90% in initial stages. This will ensure that our dog detector is working properly with desired accuracy.
- The custom CNN must have some accuracy to get an intuition of whether the model is working, if working it must be able to output only one dog breed among 133 total dog breeds. This will ensure that our custom model is woking and can be trained on full dataset. Finally it should be able to predict with high Precison and high Recall after all stages of our workflow are working.
- Another creiteria would be like having our OpenCV's Haar cascades classifier to work with high precision.

Evaluation Metrics

Due to class imbalance and other critical parameters in our dataset, Precision and Recall Numbers would be great choice for evaluating our model. We could also target log loss as evaluating metric.

Project Design

Project would be divided in the following steps:

1. Import necessary python packages and libraries required for various tasks. Import and preprocess the data and split it into train, validation and test sets.
2. Detect human faces using OpenCV's Haar cascade classifiers.
3. Detect dogs using pretrained VGG16 model.
4. Create a custom CNN to classify dog breeds.

5. Algorithms to combine Dog detector and human detector :
    - If dog is detected, return dog breed prediction
    - If human is detected, return resembling dog breed prediction.
    - If neither is detected, return Error as output.

References

1. OpenCV's Haar cascades classifier for face detection:
    - https://docs.opencv.org/trunk/db/d28/tutorial_cascade_classifier.html
2. Pre-trained Pytorch models(Torchvision.model) :
    - https://pytorch.org/docs/master/torchvision/models.html
3. ImageNet: http://www.image-net.org/
4. Precision and Recall: https://towardsdatascience.com/beyond-accuracy-precision-and-recall-3da06bea9f6c
5. CNN: https://cs231n.github.io/convolutional-networks/
6. Original Project Repo: https://github.com/udacity/deep-learning-v2-pytorch/tree/master/project-dog-classification