Machine Learning Engineer Nanodegree

Capstone Proposal

CNN Project: Dog Breed Classifier

Shikher Srivastava

6th November, 2020

Domain Background

The Dog breed classifier is a well-known problem in Machine Learning. The problem is to identify a breed of dog if dog image is given as input. So if dog owners are unsure about their dog's breed, a dog-breed classifier can help them identify their dog's breed.

Machine Learning/Deep Learning and Computer Vision helps you build machine learning models where you train a model to recognize the dog breed. CNN was a huge invention in deep learning, this gave a boost to lots of applications in visual recognition. I will build a CNN from scratch using Pytorch and use some of the popular CNN architectures for our image classification task.

My motivation of working on this project comes from my interest in the field of computer vision and deep learning and solving problems by deploying solutions through AWS SageMaker using the knowledge learn from this nanodegree course.

Problem Statement

The objective of the project is to build a machine learning model that can be used within web app or any operating system app to process real-world, user-supplied images. I will be using CNNs and pretrained models in solving this problem.

The models used have to perform following tasks:

- **Dog face detector**
  Given an image of a dog, the model will identify an estimate of the canine's breed.

- **Human face detector:**
  If supplied an image of a human, the model will identify the resembling dog breed.

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.

The following datasets will be used (by Udacity) :

- **The dog dataset**
  with 8351 dog 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. Few have 4 images while some have 8 images.

- **The human dataset**
  with 13233 human images which are structured in folders by celebrities' name (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.

Solution Statement

For performing this multiclass classification, I will use Convolutional Neural Network to solve the problem.

The solution involves three steps:

1. Detect human images, we can use existing algorithm like OpenCV's implementation of Haar feature based cascade classifiers.
2. Detect dog-images where we will use a pretrained VGG16 model.
3. After the image is identified as dog/human, we can pass this image to an CNN which will process the image and predict the breed that matches the best out of 133 breeds.

The accuracy of the predicted breed of dogs' images will be measurable, because we have a labelled data that we can compare the predictions against. The accuracy of the predicted breed for humans' images will not be measurable.

Benchmark Model

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

Evaluation Metrics –

- VGG16 model used in transfer learning for dog detection should predict dog in images with high precision. This ensures that our dog-detector is well-trained.
- 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

I would use test accuracy as an evaluation metric to compare my models with the benchmark model. Additionally, the performance will be compared between the following convolutional neural networks: dog breed's classifier trained from scratch and dog breed's classifier trained with transfer learning.

$$\text{Accuracy} = \frac{Number\ of\ items\ correctly\ classified}{All\ classified\ items}$$

Project Design

The following steps will be followed to go through the project:

1. Import the necessary dataset and libraries, Pre-process the data and create train, test and validation dataset. Perform Image augmentation on training data.
2. Detect human faces using OpenCV’s implementation of Haar feature based cascade classifiers.
3. Create dog detector using pretrained VGG16 model.
4. Create a CNN to classify dog breeds from scratch, train, validate and test the model. I will implement this using PyTorch.

5. Create a CNN to Classify Dog Breeds using Transfer Learning with resnet-101 architecture. Train and test the model. I will implement this too using PyTorch.
6. Write an algorithm to combine Dog detector and human detector.
    - If dog is detected in the image, return the predicted breed.
    - If human is detected in the image, return the resembling dog breed.
    - If neither is detected, provide output that indicates the error.
7. Test my algorithm.

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-accuracyprecision-and-recall-3da06bea9f6c
5. CNN: https://cs231n.github.io/convolutional-networks/
6. Original Project Repo: https://github.com/udacity/deep-learning-v2pytorch/tree/master/project-dog-classification
7. Resnet101: https://arxiv.org/abs/1512.03385