Capstone Proposal
Machine Learning Engineer
Nanodegree Program

Anup Joseph Sebastian

17th March, 2020

Dog Breed Classification using CNNs

Domain Background

Over the past few years, development in Artificial Intelligence and Machine Learning has improved our lives in many different ways. Its impact has been far and wide, in both industry and research, and has allowed us to solve problems that were considered impossible just a few years ago. One of the biggest winners of this new AI revolution has been the field of Computer Vision.

Computer Vision is used in various different applications, from Self-Driving Cars and Robots, to Production floors to the smartphone you have in your hand. Early neural networks, showed promise in classifying simple datasets such as the MNIST Dataset, but the real breakthrough happened when we started to use Convolutional Neural Networks for Computer Vision Applications.

In this project, I hope to build an application using Convolutional Neural Networks, to distinguish between various dog breeds and humans. I look forward to this project as a stepping stone for my aspiration to one day work on the cutting edge of the next generation of Computer Vision.

Problem Statement

The goal of the project is to build an image classifier, which can perform a series of tasks as follows.

- Given an input image of either a dog or a person, the application should classify whether it is a dog or a human. In other words, it should calculate the probability that it is a dog.

- If the image is of a dog, the application should classify, which dog breed it belongs to from a number of possible classes. This may be extended to

include mixed breed dogs. In a mathematical sense, given an image the CNN should output the probability that the image belongs to a particular breed of dog.

- If the input image was of a human, the CNN should output the dog breed that the human most resembles.

Datasets and Inputs

This CNN accepts images as inputs and it will be trained using two datasets, which were provided by Udacity. They datasets used are as follows:

Dog image dataset: This dataset consists of a total of 8351 images, split into three sets – train (6,680 images) , test (836 images) and valid (835 images). Each of these directories are further split into 133 folders each, which contain images of various dog breeds. The dataset can be found here.

Human image dataset: The dataset contains 18982 images of people, with directory names corresponding to the persons name. Most of the directories consist of only a single image, but there are some which have multiple images. The data consists of images of 13233 unique people. The dataset can be found here.

Both these datasets are going to be used to train neural networks for classification. One to distinguish between humans and dogs, and the other to distinguish between dog breeds.

Solution Statement

This problem is particularly suited to be solved using a Convolutional Neural Network. These neural networks are extremely effective in capturing spatial information present in images, and perform much better with fewer parameters when compared to traditional fully connected networks.

For the first part of the project, there is an option to use a Haas cascade classifier from the OpenCV library, or a neural network to classify between human images and dog images. A decision on which to use will be made based on the performance (accuracy and inference time) of both methods.

For the next part to classify the breed of the dog, or to classify the breed of dog that a human most resembles, a neural network will have to trained using the dog images of various classification. The network could be trained from scratch, but it may be preferable to use transfer learning using one of the available pretrained networks such as AlexNet, ResNet or VGG.

In the end some logic could be written in Python to tie all of these networks together, so that the application can make the right decision on what output to produce based on the input provided.

Benchmark Model

To make sure that the model performs well, there are a number of target benchmarks that the classifiers models should achieve.

- It is expected that a CNN trained from scratch, should achieve a minimum accuracy of 10%. Considering that the dataset has 133 classes, randomly guessing should give us an expected accuracy of 0.0075%. Therefore this is considered acceptable for first model.

- However, when leveraging transfer learning, it is expected that the model should obtain an accuracy above 60% for it to be considered performing fairly well.

Evaluation Metrics

In order to evaluate the performance of our model, we will be using a number of metrics to make sure the model achieves acceptable performance.

Since this is a classification problem and that there is a some imbalance in the data, we will be making use of metrics beyond accuracy such as precision and recall. However, the main one we will use is the F1 score, which is a combination of the precision and recall.

$$Precision = \frac{True\ Positive}{True\ Positive + False\ Positive} \qquad Recall = \frac{True\ Positive}{True\ Positive + False\ Negative}$$

$$F1\ Score = 2 \times \left(\frac{Precison \times Recall}{Precision + Recall}\right)$$

Project Design

The main workflow for the design of this project is as follows

1. Import and preprocess the data. The steps for preprocessing could include converting to tensors, resizing, augmentation and normalization. The data should also be split into train, validation and test sets.

2. Evaluate the performance of the OpenCV face detector and compare it to a pretrained neural network for performance (accuracy and inference time). Choose the most suitable method.

3. Train a CNN from scratch to classify dog breeds and evaluate its performance.

4. Leverage transfer learning to create a similar dog classifier using one of the suitable options in Pytorch. A balance of accuracy and inference time should be considered, to ensure a good user experience.

5. Write an algorithm to make the logical decisions on what model to invoke the logic detailed in the Problem Statement.

6. Deploy the model and access it through a simple web page, where the user can use their own images.

References

1. Udacity Project Repository: https://github.com/udacity/deep-learning-v2-pytorch/tree/master/project-dog-classification

2. MNIST Dataset: http://yann.lecun.com/exdb/mnist/

3. Precision, Recall and F1 Score: https://towardsdatascience.com/whats-the-deal-with-accuracy-precision-recall-and-f1-f5d8b4db1021

4. Pytorch Documentation: https://pytorch.org/docs/stable/index.html

5. Torchvision Model Library: https://pytorch.org/docs/stable/torchvision/models.html