Machine Learning Engineer Nanodegree
Capstone Final Report

Kiran Vasishta TS
April 12th, 2020

Dog Breed Classifier

Domain Background

Dog breed classifier is one of the popular problems in the Convolutional Neural Networks. The main idea behind this problem is to clearly explore and use the capabilities of the Deep Learning neural networks to get the best classifier to predict the breed of the dog using its images. So, I selected this problem domain since I really want to explore this area of Machine Learning and further use the skills, I obtain here for the purpose of solving other real-world challenges like identification of the crop diseases using the image of the diseased plant.

Problem Statement

The Dog Breed classifier problem is the scenario where the dog-images are fed to the app and we must classify as to which breed the dog in the given image belongs to. Now the real challenge is that, the image may not be a dog image, it can also be a human or can also be any other image like a car, chart etc., Our model must be able to correctly identify this as well. If the image is human, the model must determine to which breed of the dog the human resembles.

Datasets and Inputs

> The datasets are provided by the Udacity. Please refer to the below links for downloading the data. Also, we need the human data to train the model to detect the human faces.
>
> Dog Data:
> https://s3-us-west-1.amazonaws.com/udacity-aind/dog-project/dogImages.zip
>
> Human Data:
> https://s3-us-west-1.amazonaws.com/udacity-aind/dog-project/lfw.zip
>
> Here I will provide the sample data for the same. This is just the sample data from the large dataset of approximately 1GB of data.

Input Size: All the images are of the size 250X250. There is a total of 133 breeds of dogs present in the dataset.

Data Preprocessing:

In this application we must do a fair amount of data pre-processing. The main reason for this is since the dogs in the images can be in any orientation and the image can have other objects along with a dog Ex: Dog going for a walk with a human.

So, to tackle this I have used the following steps:

1. First, as usual the data is split into train, test, validation sets.
2. Resize the image to 224 X 224 models and the images are normalized.
3. For the training data, the **image augmentation** is applied. That is the images are horizontally rotated, horizontally flipped to prevent the overfitting of the model.
4. The images are fed to the model in the form of tensors.

Algorithms and Techniques:

This problem is a multiclass classification problem with 133 different classes. So, we will use the Convolution Neural Network (CNN). The CNN is the algorithm which accepts the image, differentiates between various features (like car in the image where the dog is sitting next to car) and detects the main feature of the image.

In this application I am planning to use a pretrained VG16 model to detect the dog images. Also, for the other part of our dataset, the human images we use the existing algorithms like OpenCV's implementation.
Once this is done, we can use the model thus created to predict the breed of the dog or if human, then detect the breed he/she resembles. Also, it is neither, then output an appropriate message.

Benchmark Model

The benchmark model is to create our own model which must have at least 10% accuracy. This conforms that the model is working as expected since the random guesses will lead to successful prediction in 1 out of 133 runs since there are 133 breed.

Implementation:

For this project I have followed the below steps:

- As with all the data-science and machine learning project this project too begins with the data preprocessing.
- The data is split into the Train, Test and Validate datasets.
- Then comes the main process, the image augmentation. This is done to get the variations of the images. This helps to predict the cases where the dogs in the images are in different angles to the frame. In short to prevent overfitting
- Detect the human faces using the OpenCV implementation of the classifiers.
- VG16 model is used to create the dog detector.
- The CNN model is created from scratch. This includes the training, testing and validating the model. **Here there are 3 convolutional layers with each layer having kernel size of 3 ang stride of 1.**
- A CNN model with Resnet101 architecture is used to create the model using the transfer learning process.
- Finally, the main code of this app is written. Here the dog detector and human detector is combined, and the following cases are handled.
    1. If the dog is detected, then output the breed.
    2. If the human is detected, then output the breed the human resembles, along with the message that the human is detected.
    3. If neither, then output appropriate message indicating the same.

Refinement:

The model that is created from scratch gives the accuracy of 10%. This is meeting the benchmark defined for this project. But the model has been refined since the accuracy of 10% is not enough to accurately predict in most cases. So, I have decided to improve the model using the Transfer Learning technique. For this technique I have chosen the Resnet101 architecture. This architecture has 101 layers and is trained on ImageNet dataset. I performed the training for 20 Epochs and got an accuracy of 74% which is way better than the 60% benchmark we set ourselves.

Model Evaluation and Validation:

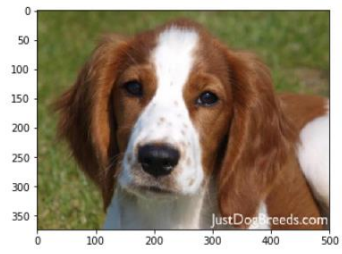
Initially, the human face detector was created using the OpenCV Haar feature based cascade classifiers. I got the accuracy of 98% in the detection of human faces in the 100 human face images. Also 17% human faces were detected in the 1oo dog images dataset

Then the dog detector function using the pre-trained VG-16 model was created. This detected 97% accurately in dog dataset and detected 2% as dogs in the human dataset.

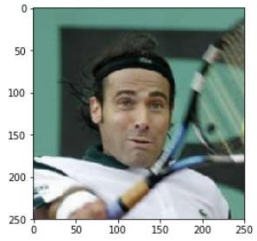
Finally, the transfer learning method gave an accuracy of 74%. I trained the model for 20 epochs. It correctly predicted 6 19 out of a total 836 images.

Test Accuracy: 74% (619/836)

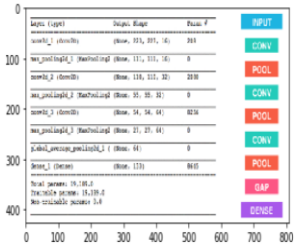
Sample Output:



./images/Welsh_springer_spaniel_08203.jpg
Dogs Detected!
It looks like a Welsh springer spaniel



/data/lfw/Alex_Corretja/Alex_Corretja_0001.jpg
Hello, human!
If you were a dog..You may look like a Papillon



./images/sample_cnn.jpg
Error! Can't detect anything..

Justification:

The model created using the transfer learning is better by a large margin (74% accuracy) when compared to the model created from scratch (10% accuracy). Thus, we have done better by selecting the transfer learning.

Improvement:

There is a lot of room for improvement. We can use architectures other than ResNet101 which may lead to an improvement in the performance. Also, image augmentation can be increased to prevent more overfitting and improve accuracy.

Reference

1. Udacity dog-breed-classifier https://github.com/udacity/deep-learning-v2-pytorch/tree/master/project-dog-classification
2. Resnet101 from Kaggle: https://www.kaggle.com/pytorch/resnet101
3. Pytorch tutorials: https://pytorch.org/
4. More references are given inside the jupyter project file where necessary.