Machine Learning Engineer Nanodegree

Capstone Project

Vivek Kandasamy
May 24th, 2020

I. Definition

Project Overview

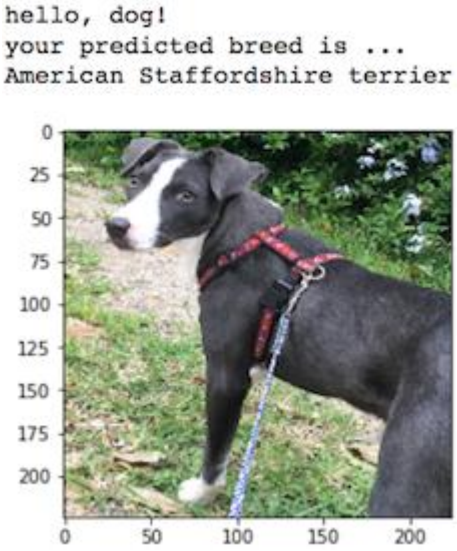
Convolutional neural networks are gaining popularity in computer vision problems because of its ability to balance the accuracy of the results and memory requirements in comparison to conventional deep neural networks. Here in this project we see an application of convolutional neural network to identify dog breeds from its pictures. This is an example of multi-class classification problem using supervised learning. A total of 13233 human images and 8351 dog images are used to train, validate and test the model.

Problem Statement

The project gets an input of a picture and identifies whether the picture contains a dog or a human. The application than prints one among the following results

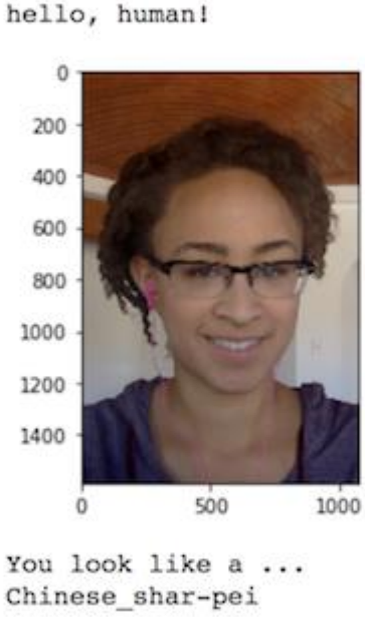
- If a dog is detected the algorithm predicts the canine's breed.
- If a human is detected the algorithm predicts the resembling dog breed.
- If the application identifies that the image does not have a dog or a human, it prints out the image has neither human nor dog.

The image below displays a potential output if **a dog** is detected on the image.



```
hello, dog!
your predicted breed is ...
American Staffordshire terrier
```

The below image is a potential output if **a human** is detected on the image.



```
hello, human!
```

```
You look like a ...
Chinese_shar-pei
```

The notebook is broken into following steps

1. Import Datasets
2. Detect Humans (using CV2 Haarcascade)
3. Detect Dogs (using VGG16)
4. Create a CNN to Classify Dog Breeds (from Scratch)
5. Create a CNN to Classify Dog Breeds (using ResNet-50)
6. Write your Algorithm (Implementation)
7. Testing the Algorithm

Metrics

The input dataset is a group of images with human and dog faces. The dataset is split into test, train and validation datasets. Train and validation datasets are used to train the data and the test dataset is used to measure accuracy of the model. Accuracy of the model is defined as the ratio of total number of true positives and true negatives to the total population.

$$Accuracy = \frac{True\ positives + True\ Negatives}{Total\ population}$$

Also, during training cross entropy loss function is used. Cross entropy loss is used, because of its ability to combine both log SoftMax and negative log likelihood function in a single class. This proves to be advantageous for multi class classification problems.

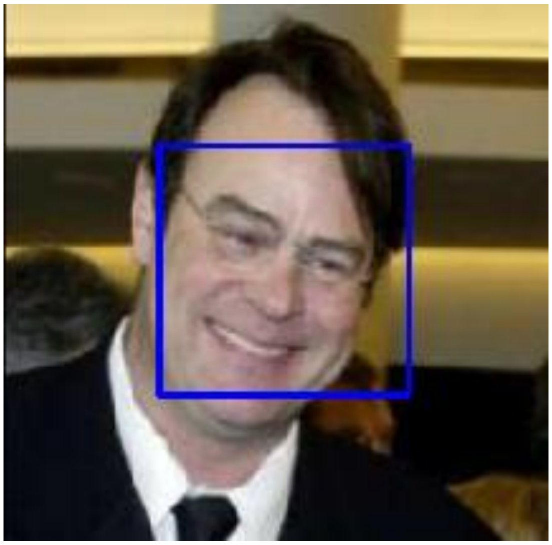
II. Analysis

Data Exploration

A total of 13233 human images and 8351 dog images are used to train, validate and test the model. These images are supplied by Udacity.

Sample dog images from the dataset are given below.

Sample human image from the dataset are given below



Exploratory Visualization

Dog files: In this project, 133 different breeds of dogs are being used. The data is split into train, test and validation dataset with training dataset containing 80 % of the total images, each test and validation dataset containing 10 % of the total images. The pixel size for each image varies and the dataset contains an uneven distribution across different breeds. i.e., one dog breed has 10 images on the training dataset, and another have 5 images on the training dataset. The background environment of each image is mostly different from one another.

Human files: A total of 13233 images of 5750 different humans are supplied. All the images have the same pixel size 250x250 and here also the dataset contains an uneven distribution across different humans. The background environment for each image is mostly different from one another.

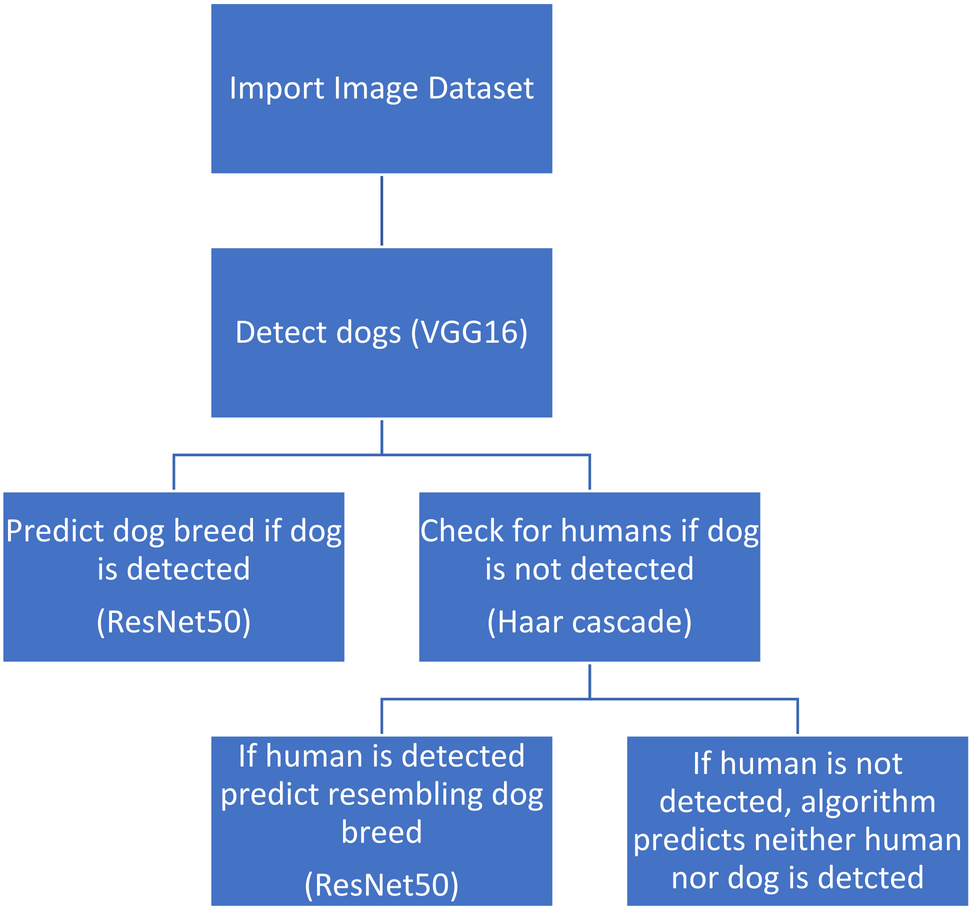
Algorithms and Techniques

The first step is to detect whether the picture is an image of a dog. For this we use an VGG16 pretrained PyTorch model. Later we predict the dog breed using a ResNet50 model trained on the 8351 dog images.

If the image does not have a dog, it checks whether the image has a human using Haar cascade model. If human is detected, using the trained ResNet50 model the project predicts the resembling dog breed.

The cross-entropy loss function and Stochastic gradient descent optimizer functions are used to train the ResNet50 and VGG16 model.

Below is the schematics of the prediction process.


Import Image Dataset
Detect dogs (VGG16)
Predict dog breed if dog is detected
(ResNet50)
Check for humans if dog is not detected
(Haar cascade)
If human is detected predict resembling dog breed
(ResNet50)
If human is not detected, algorithm predicts neither human nor dog is detcted

Benchmark

Two models are used to compare the predicted dog breeds.

- In Model 1 a convolutional neural network is built from scratch to classify dog breeds. This model has three convolutional layers and two fully connected layers to predict the result. Model 1 is built from scratch and the model is expected to have an exceptionally low accuracy of around 10% on test set.
- In Model 2 a pretrained ResNet-50 model is used. This pretrained model is trained again to predict the dog breed. As the model allows skip connection it avoids vanishing gradients. In Model 2 (ResNet50) a pretrained model with 50 layers is used. Therefore Model 2 will have a minimum accuracy of 60% on the test set.

III. Methodology

Data Preprocessing

The data preprocessing for each image is done based on the prerequisites of the algorithms.

- CV2 Haar cascade algorithm - The input image is converted into a grayscale image. Later this image is fed into CV2 Haar cascade algorithm to detect the number of human faces in the image.

- VGG16 - The image is converted into an image with 224 x 224 pixel, which is the standard size for PyTorch VGG16 using tensor transform.

- CNN to classify dog breeds (from scratch) – Each image is resized to 224*224 pixels. Various data augmentation techniques like cropping and horizontal flipping are applied to the image. The image data is also normalized with mean = [0.485, 0.456, 0.406] and std = [0.229, 0.224, 0.225]. This reduces the problem of overfitting as the network will find difficult to see the same inputs twice.

- CNN to classify dog breeds (using ResNet50) - Each image is converted into a 224x224 pixel. Like previous step various data augmentation techniques are applied and image is normalized with mean = [0.485, 0.456, 0.406] and std = [0.229, 0.224, 0.225]. The number of output features are set to be 133, since we have 133 different canines in the training set.

Implementation

Following steps are involved in the implementation.

- Dog detector – This step is to detect whether the image has a dog. For this a pretrained VGG16 PyTorch model is used. Also, to train this model GPU is used to save computational time.

- Haar cascade classifiers – This is an algorithm used to detect whether an image has a human face. The algorithm is widely used since it runs faster than a machine learning (Neural network) algorithm to detect faces on an image. The method was first presented by Paul Viola and Michael Jones in their paper, "Rapid Object Detection using a Boosted Cascade of Simple Features" in 2001.

The preprocessed image is fed into CV2 Haar cascade algorithm to detect the number of human faces. The code returns true if a human face is detected.

- CNN from scratch - This step is to predict the dog breed from images. Two models were used to classify the dog breed. Here we discuss the first model. The model has three convolutional layers and two fully connected layers. Max pooling of size (2,2) is done after each convolutional layer and dropout layers with a probability of 0.5 is used after each fully connected layer. For convolutional layer, a 3x3 filter along with stride and padding 1 is used. Various preprocessing techniques are implemented on images. Stochastic gradient descent optimizers and cross entropy loss function are used. The batch size is set to be 10. GPU is used to train the model. The model had an accuracy of 13%.

Refinement

The CNN from scratch algorithm had an exceptionally low accuracy of 13%. Here we discuss the second model to classify the dog breed using transfer learning. This is a pretrained ResNet50 model. This method was presented by Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun in their paper, 'Deep Residual Learning for Image Recognition'.

ResNet50 model is chosen because of an acceptable computational cost. It has a top-1 error of 23.85 and a top-5 error of 7.13. As the model allows skip connection it avoids vanishing gradients.

The model is trained using the training and validation dataset. GPU with 10 epochs are used to train the model. The loss function is set to be cross entropy loss which combine both log SoftMax and negative log likelihood loss as discussed previously. The optimizer used is stochastic gradient descent. The learning rate is set to be 0.05. The model had an accuracy of 86% on test data.

IV. Results

Model Evaluation and Validation

Dog detector

Based upon the 100 test images with human and dog faces the algorithm predicted the following.

1. Percentage of the first 100 images in human_files have a detected dog face 0%
2. Percentage of the first 100 images in dog_files have a detected dog face 94%

Haar cascade classifiers

The code returns true if a human face is detected on the given image. Based upon the 100 test images with human and dog faces the algorithm prediction are

1. Percentage of the first 100 images in human_files have a detected human face 98%
2. Percentage of the first 100 images in dog_files have a detected human face 17%

CNN from scratch

The model is required to produce a test set accuracy of 10%. The model after 30 epochs produced a training loss of 4.105296 and validation loss of 3.750196. Later the trained model is checked with the test set. The model had a test loss of 3.732507 and an accuracy of 13%.

CNN using transfer learning (ResNet50)

A training loss of 0.981760 and a validation loss of 0.528315 is achieved. An exceptionally low training and validation after 10 epochs in comparison to convolutional neural network built from scratch with 30 epochs can be due to 50 layer pretrained network. Here we achieved a test loss of 0.466678 and a test accuracy of 86%.

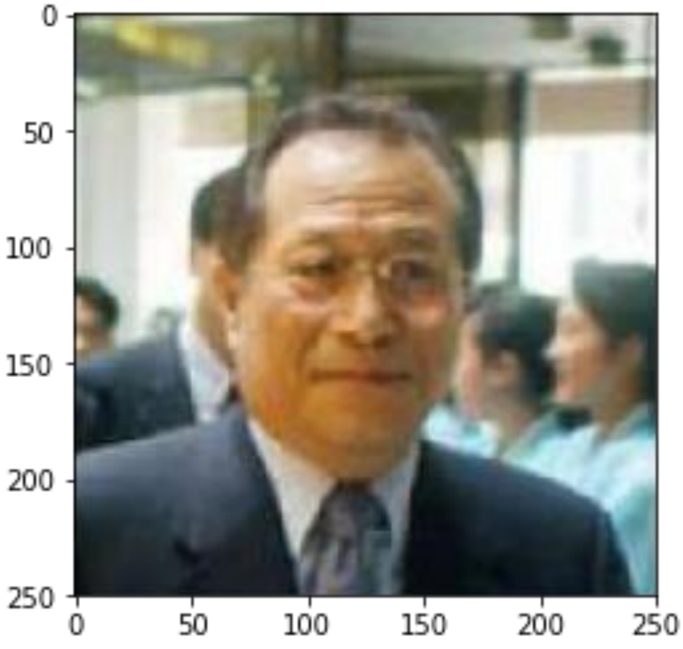
Justification

I think the model performance after refinement is better than expected. The dog face detector had an accuracy of 94% and human face detector had an accuracy of 98%. The final breed classifier model has a test accuracy of 86% which is 26% more than the required 60%. Sample final output of three human and dog images are given below.

```
Hello human!
```

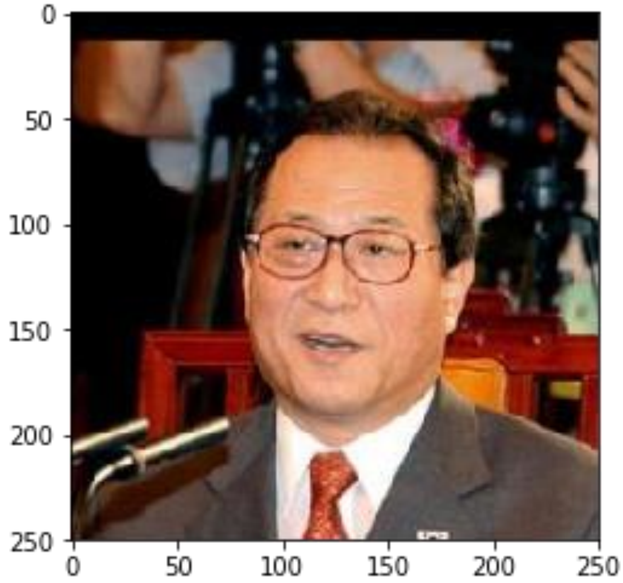
```
You look like a...

 Pharaoh hound
```

```
Hello human!
```



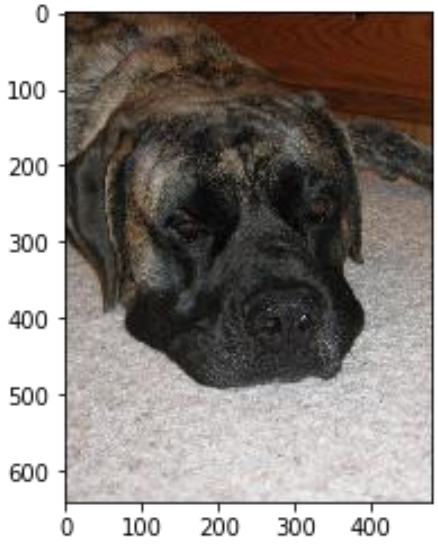
```
You look like a...

 Bearded collie
```

```
Hello human!
```

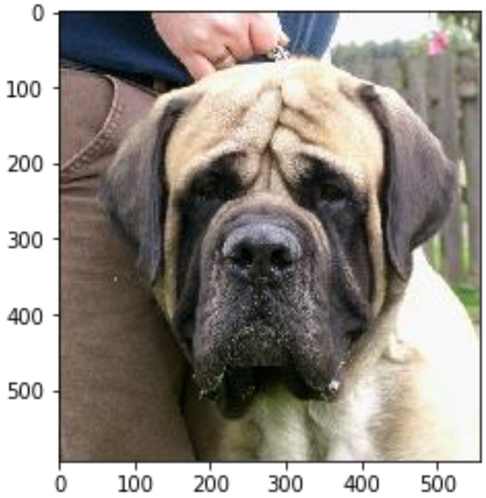


```
You look like a...

 Chihuahua
```

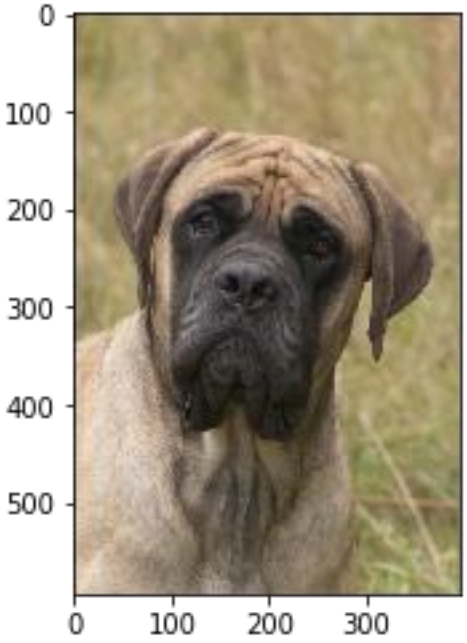
```
Hello dog!

 Your predicted breed is...

 Bluetick coonhound
```



```
Hello dog!

 Your predicted breed is...

 Mastiff
```



```
Hello dog!

 Your predicted breed is...

 Cane corso
```



V. Conclusion

Free-Form Visualization

The model predicts the dog breed classifier with a test accuracy of 84%. This can be mainly attributed to the fact that even most humans will find it exceedingly difficult to clearly predict the various breeds of dog upon visualization. Suppose an image of a cross or mixed breed dog is fed into the algorithm; it may find it difficult to predict the result or predict an unexpected result.

Reflection

In this project report, I presented an application of convolutional neural network to predict the dog breed on an image. The image was loaded and checked whether it is an image of a dog or a human. If the image has a dog, the dog breed is predicted. If it contains a human, the resembling dog breed is predicted. To achieve this Haar cascade, VGG16 and ResNet50 algorithms are used.

Improvement

The following steps can be taken to improve the efficiency.

- The number of epochs to train data can be increased.
- Use a deeper algorithm like ResNeXt-101-32x8d at the cost of computational complexity.
- Increase the size of the training dataset.
- Tune model parameters.
- Increase the number of breeds and training images.