Machine learning Engineer nanodegree

Capstone Project report

Dog Breed Classification with CNN

1.Definition

Project Overview:

As many people now days are seeking for buying a dog and are new to this business the first thing to know is the breed of the dogs in order to know which one will suit them the most and their environment and know more about their behavior therefor the overview of this project is to build a model which is used to identify the breed of a dog by just giving an image to the model as an input and it will classify it by extracting the features of the dog then detecting the breed of the dog . If the model is supplied with a human image it has to identify the resembling dog breed.

This model will be so useful for dog owners as it will prevent them from getting scammed and it will also save them lots of time and money for identifying the breed of their dog .

Problem Statement

Building a pipeline to process real world, user supplied images that can be used in an application .The model has to perform two tasks.

Human face detection : If the model is given a human image it has to identify the resembling dog breed.

Dog detection : when the model is given an image of a dog it has to detect it's breed

Metrics: As the dataset is imbalanced and the images have different backgrounds .

Therefore accuracy it is better not to rely on accuracy (Accuracy = correct specification /all specifications)

only in evaluating our model and use other metrics as well and in our case It is best to use F1 score for evaluation which can correctly distinguish between positives and negatives

2.Analysis

Data Exploration

The dataset provided by Udacity contains images of dogs and humans

The dog images datasets : (8151 image) of dogs with different sizes and backgrounds which are divided into training , testing and validation.

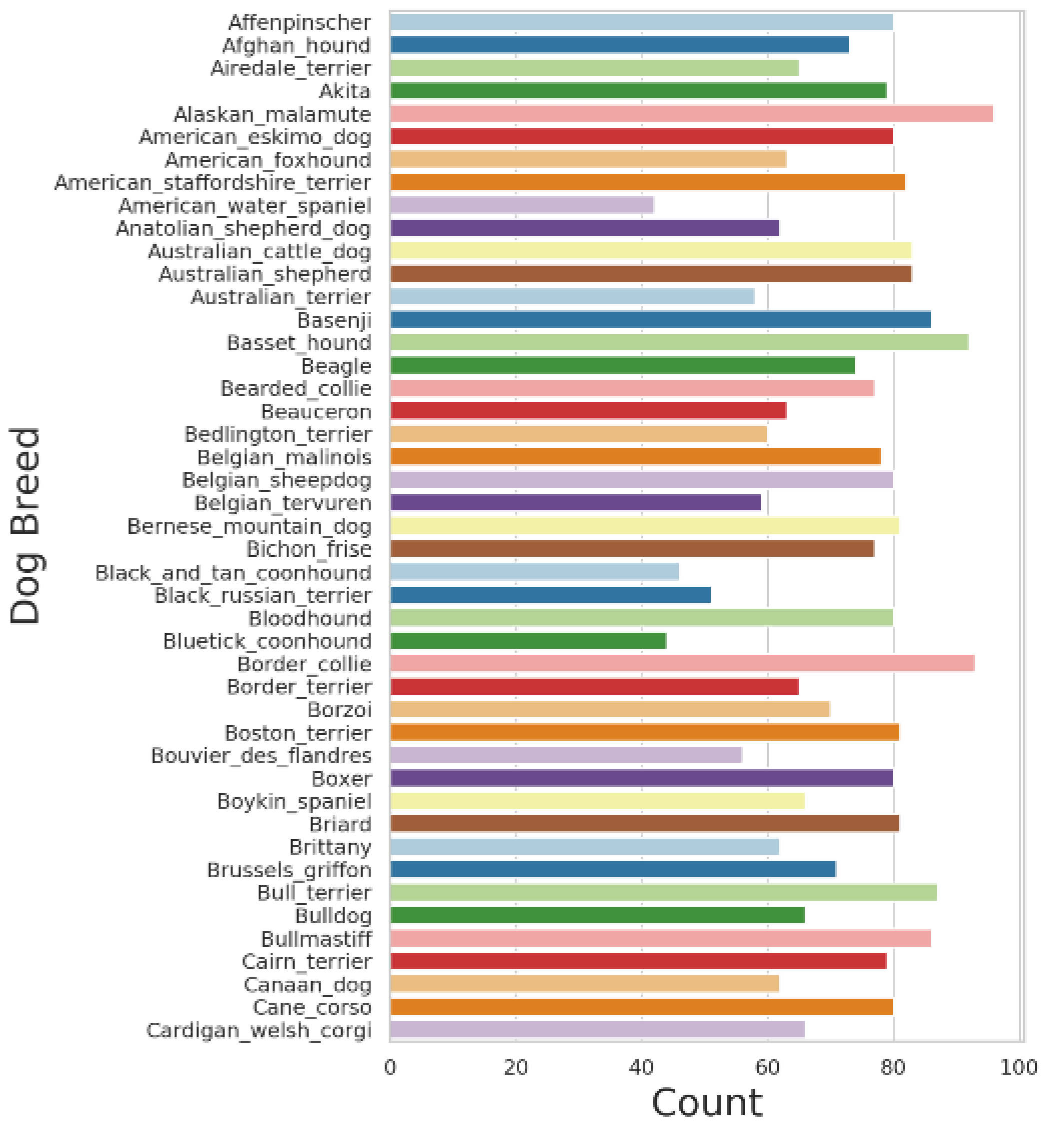
Training (6,680) image , testing(836) image and validation (835) image . The data is also divided to directories with 133 folder each . The number of images of each breed varies widely which makes the data unbalanced .

The humans images datasets : (13233) images of humans all with same size of 250X250 but the pictures are taken of different humans and different angles which also makes the data unbalanced.

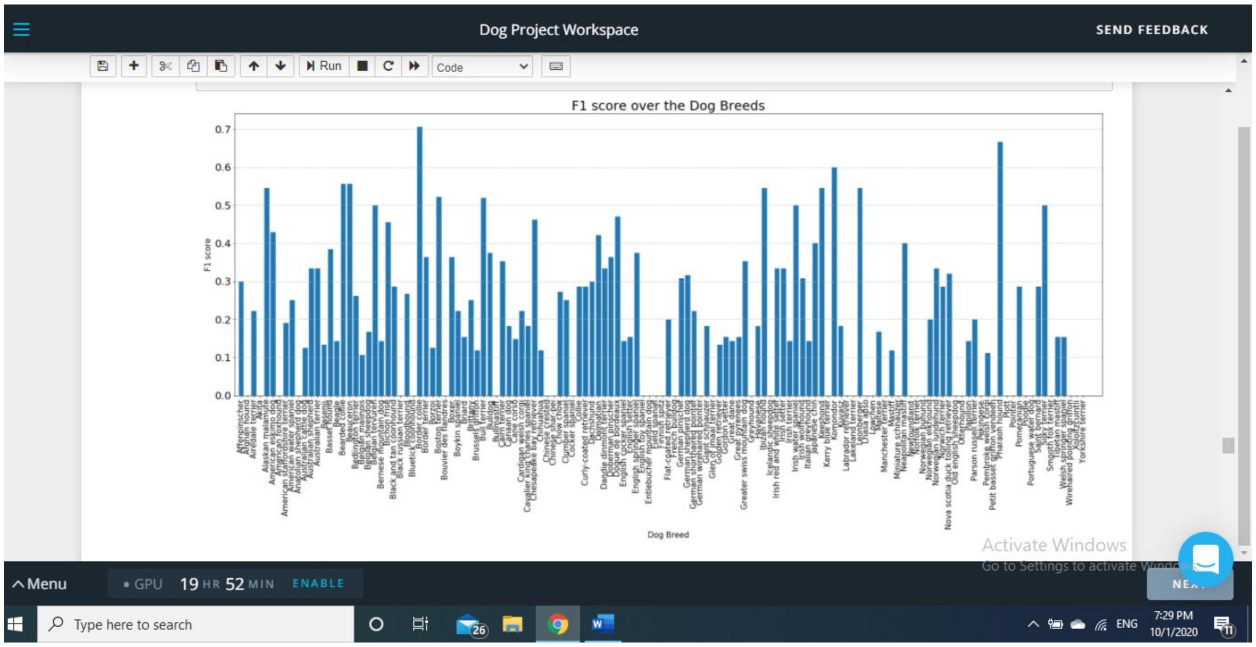
Exploratory Visualization

In this section I will show some plots to visualize the data and make it easier to understand

The first plot shows the number of each breed in the dataset



The plot below shows the f1 score to with relation with the dog breeds it shows that the border collie has the highest f1 score which means that it works best with the model



The images have different backgrounds

Algorithms and Techniques

The algorithm used for solving this problem is Convolutional Neural Network (CNN) and we used various techniques each for specific purposes the first is the pretrained VGG-16 is a

CNN model of 16 layers deep it is known to be too great in detecting many object especially animals which made me choose to work with it as it has got a lot of features to classify the images model which I used to recognize the dogs in the images and it is only used for prediction and will not be trained then I used the Cascade Classifier from OpenCV as it got so well training from positive and negative images for example images that includes faces and images that doesn’t so that is why I used it to detect if an image includes a human or not and I used PyTorch to implement ,train and test the model to detect the breed of the dog and lastly I have used transfer learning

Benchmark

I have set the CNN model built from scratch my benchmark model and it must have an accuracy of at least 10% and this proves that the model is working because random guess will roughly provide 1 out of 133 which corresponds to an accuracy of 1% so more than 10% proves the success of my model .

3.Methodology

Data Preprocessing

As the images were having different resolutions which made the data a little bit imbalance I used PyTorch module torchvision.transforms and reduced the resolution to 224X224 then I applied normalization to the train , valid and test datasets and augmented the training data to reduce overfitting and underfitting .The training data was randomly rotated and randomly horizontally flipped then converted the images to tensor.

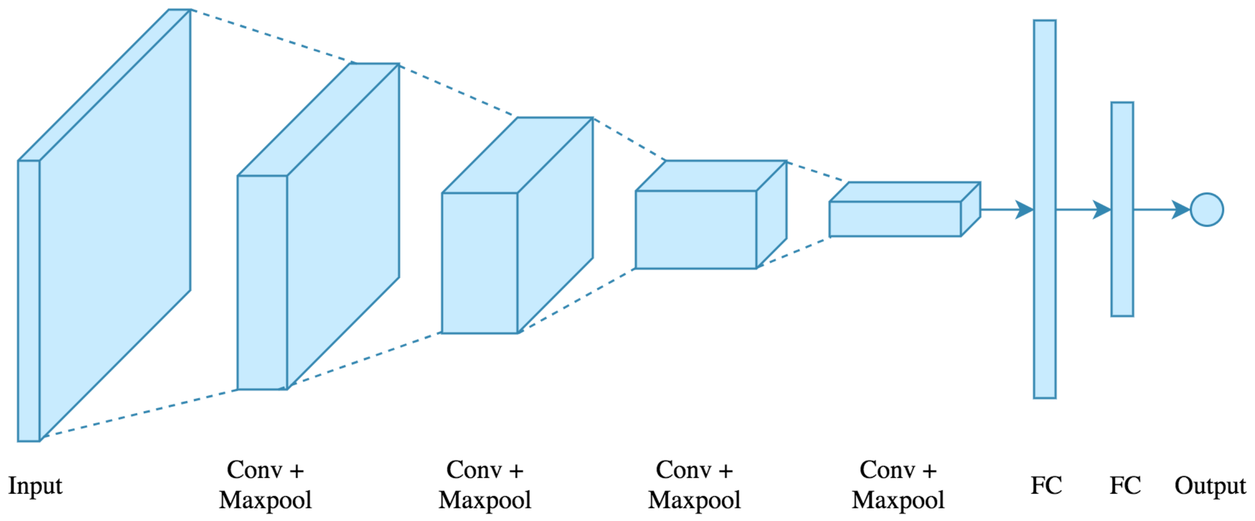
Implementation

Human face detector I have used the Cascade Classifiers to check whether a person's face is included in the image or not ,I have used the cascade classifier of haarcascade_fronalface_alt.xml file and implemented the face_detector function as it is able to recognize 100% of humans as humans**.**

Dog detector I used the pretrained VGG-16 model to be able to recognize dogs in the images as explained in the algorithms section I have implemented the model using PyTorch the the open source machine learning framework . I started by transforming the image of the model to 224X224 pixels which is the input size of the model then converted it to the pytorch tensor then normalized it with the same parameters as those of used when training the VGG-16 model then passed the image to the prediction to get the index value .

The dog breed classifier from scratch I have designed ,trained and tested the model using pytorch from scratch and I made it with only three convolutional neural networks to avoid complexity each with kernel size 3 ,padding 1 and stride 1 and then activated the relu function which outputs zero unless the input is positive it will return it directly and a pooling layer of kernel size 2 which reduces the amount of parameters and reduces the computational time of the network and a dropout with probability of 0.25 .

The image below shows the architecture of the CNN model from scratch



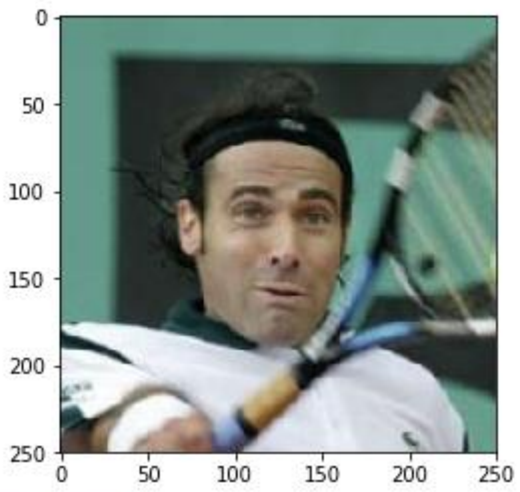
Dog breed classifier with transfer learning I used transfer learning technique to implement the pretrained model

ResNet-50 using pytorch and I have explained the ResNet-50 model in the algorithm section.

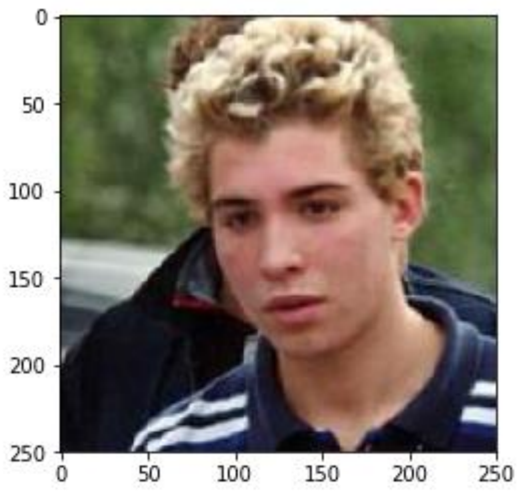
Refinement

In the dog breed classifier with transfer learning I have used ResNet-50 with accuracy of 71% .The CNN model created form scratch accuracy was 13%

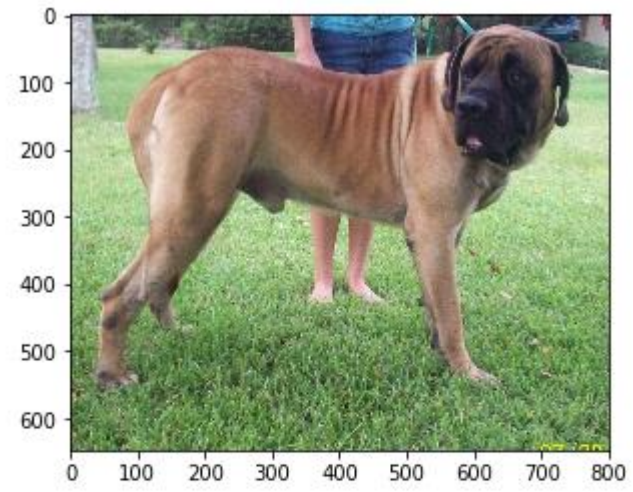
```
Detected Human!
Predicted breed:  English toy spaniel
```



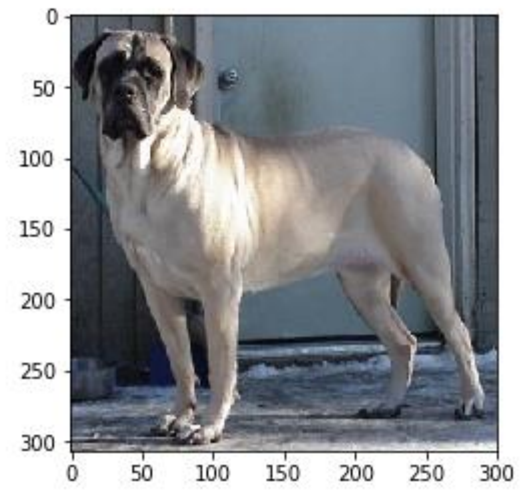
```
Detected Human!
Predicted breed:  Petit basset griffon vendeen
```



```
Detected Dog!
Predicted breed:  Cane corso
```



```
Detected Dog!
Predicted breed:  Bullmastiff
```



```
Detected Dog!
Predicted breed:  Bullmastiff
```

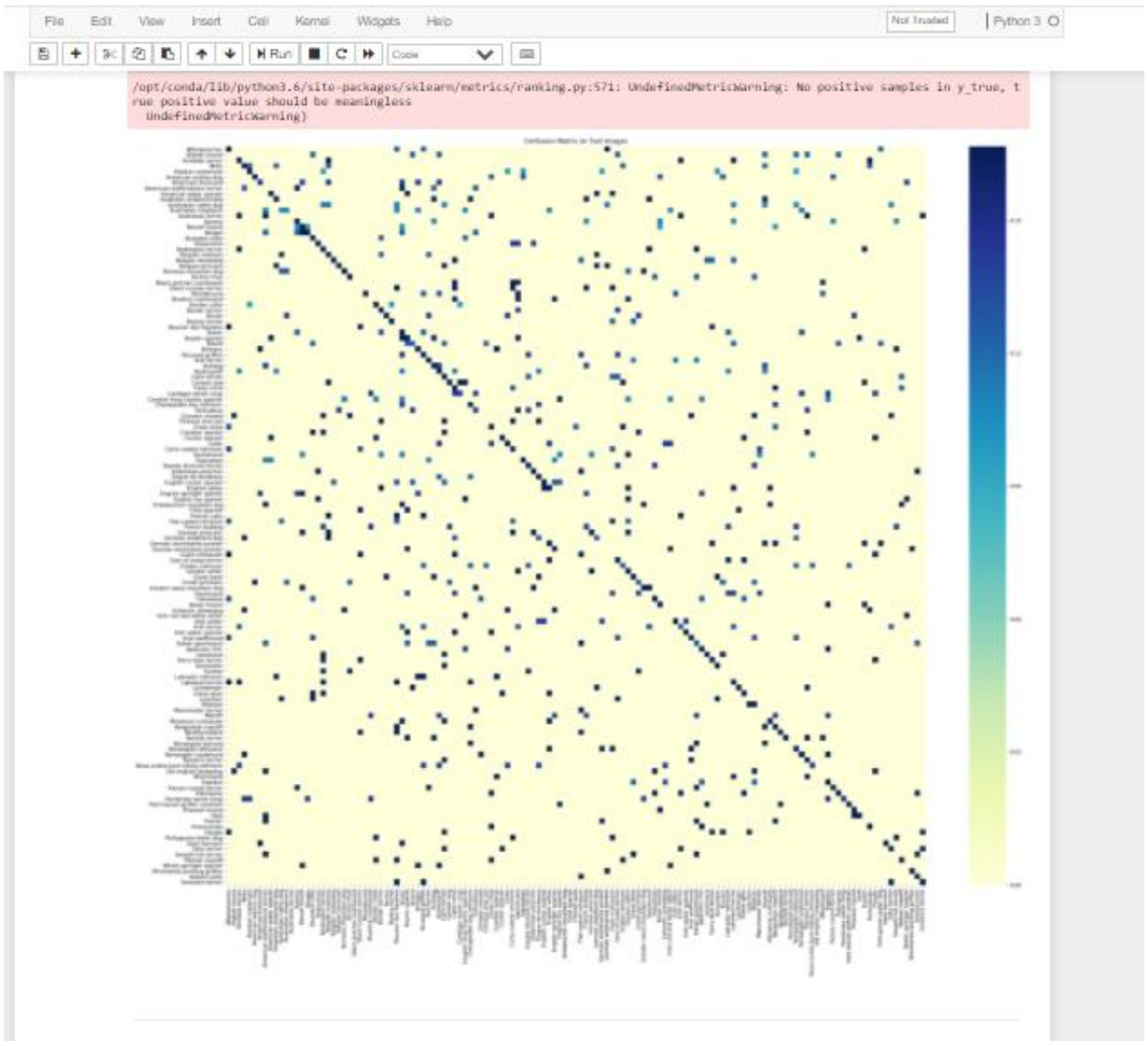
4.Results

Model Evaluation and Validation

Dog face detector : The dog face detector function was created using VGG-16 model 100% of dog faces were detected in the first 100 images of dog dataset and 1% of dog faces detected in first 100 image of human dataset.

Human face detector :For implementing the human face detector function I used the Haar feature based cascade classifier form python OpenCv 98% of human faces were detected in the first 100 images of human face dataset and 17% of human faces detected in first 100 images of dog dataset.

CNN using transfer learning : the model is created using trans fer learning with ResNet-50 which is a CNN model which consi sts of 50 layers deep and It is so good in detecting objects as i t can detect 1000 object in a single image but it needs images of input size 224 X224 and the final model accuracy was 71% on test data 597/836 and the Test Loss: 0.958761 and the con fusion matrix for the ResNet-50 transfer learning model show s with results that it can classify dogs and humans into certain dog breeds and the F1 score plot on the tested data shows th at a lot of dog breeds have high F1 score which indicates that the transfer learning model is more robust for different dog b reeds .



```
/opt/conda/lib/python3.6/site-packages/sklearn/metrics/ranking.py:571: UndefinedMetricWarning: No positive samples in y_true, t
rue positive value should be meaningless
  UndefinedMetricWarning)
```

Justification

For me the performance of my model was great as the accuracy of the transfer learning model is 71% as the benchmark model (CNN model) was 13%.

5.Conclusion

Reflection

1.Identifying the problem and looking for releavant datasets

2.Downloading the dataset

3.Implementing the human face detector

4.Implementing the dog face detector

5.Implementing the CNN model from scratch

6.Implementing the transfer learning model and training and testing it

7.Documenting the project

Improvement

The model could be improved by increasing the convolutional layers but I have avoided complexity and the algorithm could also be improved by increasing the dataset and hyperparameter tunning and one of the methods also that could have improved the algorithm was implementing ResNet-101 which is very deep neural network.

References

https://pytorch.org/docs/stable/index.html

https://github.com/Abdelrahman13-coder/Capstone-project-Dog-Breed/blob/master/dog_app.ipynb

https://pytorch.org/tutorials/beginner/transfer_learning_tutorial.html

https://www.youtube.com/channel/UCxxljM6JkSvJVSD_T90ZnMw

https://towardsdatascience.com/understanding-confusion-matrix-a9ad42dcfd62

https://docs.opencv.org/3.4/db/d28/tutorial_cascade_classifier.html#:~:text=It%20is%20a%20machine%20learning,detect%20objects%20in%20other%20images.&text=Initially%2C%20the%20algorithm

https://www.mathworks.com/help/deeplearning/ref/vgg16.html#:~:text=Description,%2C%20pencil%2C%20and%20many%20animals.%20needs%20a,faces)%20to%20train%20the%20classifier.

https://www.mathworks.com/help/deeplearning/ref/resnet50.html#:~:text=ResNet%2D50%20is%20

a%20convolutional,%2C%20pencil%2C%20and%20 many%20animals.