Capstone Project
AWS Machine Learning Engineer Nanodegree

Puneet Bindal
201651039@iiitvadodara.ac.in

Dog Breed Classification

Definition

1. Project Overview

   Computer vision is a problem that every individual is facing difficulties all around the world. But on the other hand, research and development is taking a boom in this area and making machine learning and computer vision problems way easier than ever before.

   In this project, I created a web application capable of seeing computer dog breeds and also detect if it is a dog or not. I tried face detector using opencv in notebook instance but unable to use that in the endpoint as i failed to install opencv in it.

   Overall I first download the data, explore the data, then test the data on two models, manually created model and pretrained model VGG19. Then we tested the data on both models and chose the best model with better test accuracy. And deployed it.

2. Problem Statement

   The Goal of the project is to classify the dog breed when a image is given through a web application.

   a. Download the data.
   b. Explore the data.
   c. Preprocess, split the data and load the data.
   d. Design the model
   e. Train and test the model
   f. Choose pretrained model, train the model using sagemaker.
   g. test the model

h. Compare the accuracies, and choose the best model among both models.
i. Create detectors to detect dogs and face.
j. Create a server folder, put necessary files in it and deploy the model for real time predictions.
k. Create lambda functions in aws.
l. Create aws api gateway link and attach it with the lambda function.
m. Create index.html for user as web application
n. Test the application.

3. Metrics

For metric evaluation, I used an F1 score.
The F1 score is a number between 0 and 1 and is the harmonic mean of precision and recall.

$$F_1 = 2 * \frac{precision * recall}{precision + recall}$$

We compared F1 scores because in our data case false negative and false positives are crucial. And also since our dataset has imbalance classes, so f1 score is a more appropriate metric to be used here as it is a harmonic means of precision and recall and doesn't depend on weightage of data in particular classes.

Analysis

1. Data Exploration

The dataset is given by udacity itself through this links:
https://s3-us-west-1.amazonaws.com/udacity-aind/dog-project/dogImages.zip
It consists of 133 dog breeds with train, valid and test data splitted in different folders.

Fig.1 Images from the dog imageset.

ClassId	
001.Affenpinscher	64
002.Afghan_hound	58
003.Airedale_terrier	52
004.Akita	63
005.Alaskan_malamute	77
..	..
129.Tibetan_mastiff	48
130.Welsh_springer_spaniel	44
131.Wirehaired_pointing_griffon	30
132.Xoloitzcuintli	26
133.Yorkshire_terrier	30

Name: Filename, Length: 133, dtype: int64

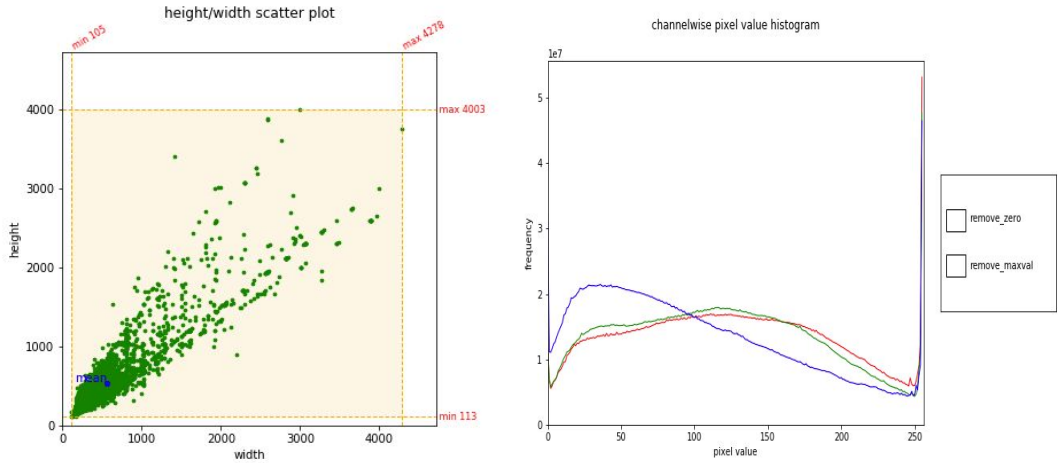
count	133.000000
mean	50.225564
std	11.863885
min	26.000000
25%	42.000000
50%	50.000000
75%	61.000000
max	77.000000

Name: Filename, dtype: float64

Fig No. of images in each class of train data

Here we can see that classes are imbalanced with deviation of 11.

Below are some plots based on the data dimensions, histograms of pixel values of images.



https://s3-us-west-1.amazonaws.com/udacity-aind/dog-project/lfw.zip
It consists of human images which may use in face detection testing of images.

Fig.2 Images from human face dataset.

We used human faces to test our haar algorithm opencv. These will be used in creating our face detector for human images, so that we could detect if a given image is a dog or human face.

2. Exploratory Visualization

We explore the data using an image from dog dataset and here are the results.

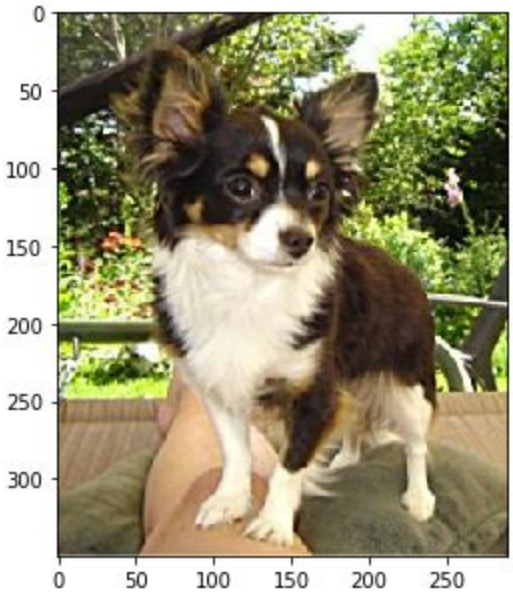


Fig.3 image plot from image taken from dog dataset

```
Type of the image :  <class 'imageio.core.util.Array'>
Shape of the image : (350, 290, 3)
Image Hight 350
Image Width 290
Dimension of Image 3
Image size 304500
Maximum RGB value in this image 255
Minimum RGB value in this image 0
Value of only R channel 117
Value of only G channel 112
Value of only B channel 72
```

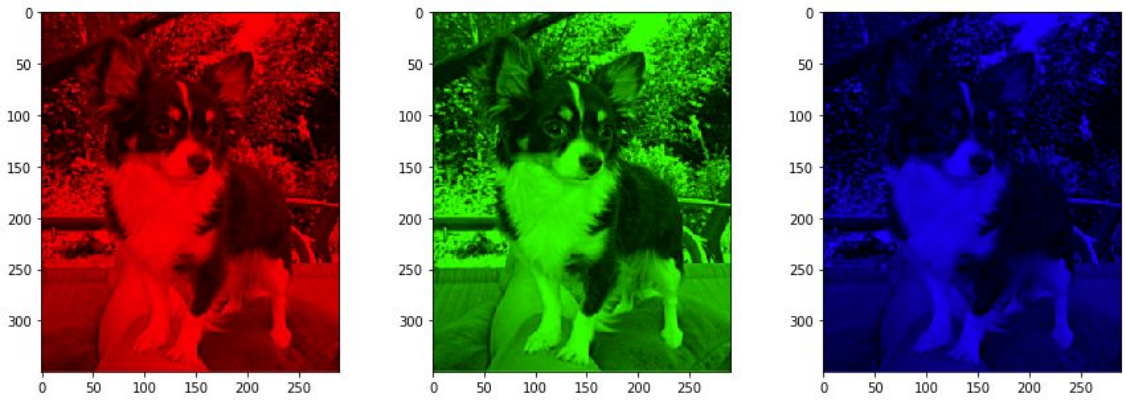


Fig.4 Images with different rgb values each component at a time.

3. Algorithms and Techniques

I used the Convolutional Neural Networks to classify the data, which is state-of-the-art for Image Classification, Segmentation, Object Detection and many other image processing tasks. They are considered as because of their ability to modify images into useful image features and used to train weights in the most efficient way possible. Some of the state of art models are AlexNet, VGG19, resnet, etc.

References:

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In NIPS 2012, 2012.

Karen Simonyan and Andrew Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition. arXiv e-prints, page arXiv:1409.1556, Sep 2014.

Matthew D. Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. CoRR, abs/1311.2901, 2013.

Algorithm for most of the image processing tasks and computer vision problems. It usually requires the big data, and because of the udacity dataset its needs can be fully filled.

Parameters can be use to tune the model:

Training Parameters:

a. Number of epochs
b. Learning rate
c. Batch size
d. Workers

Neural Network Architecture:

a. Number of layers
b. Layer types
c. Layer parameters

Data preprocessing:

a. Normalization
b. Rotation
c. Scaling
d. Random resize and crop

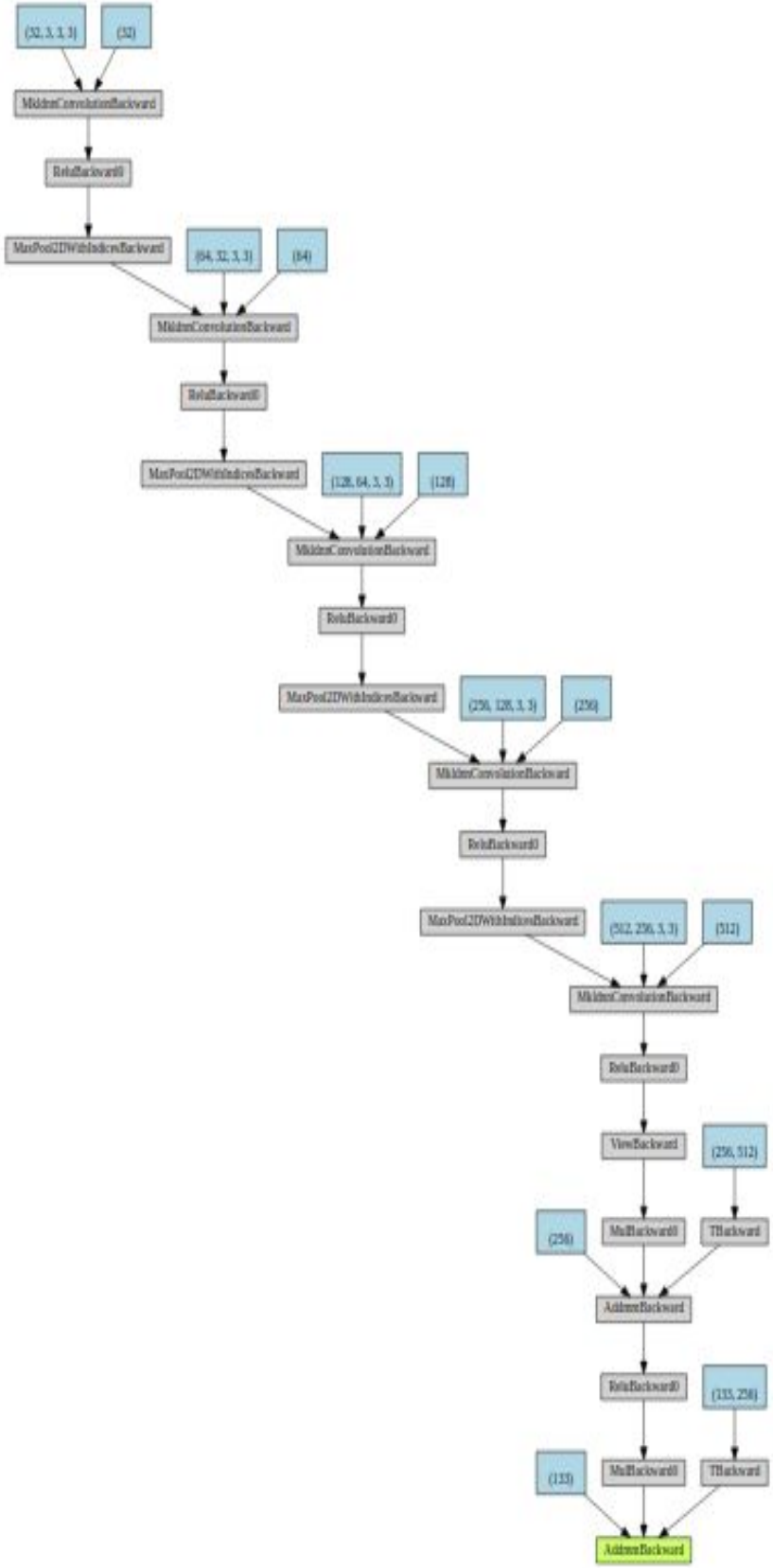
During training, training data and validation data is loaded into the ram, We shuffled trained data, and loaded into the GPU memory. We loaded the model into the GPU memory for processing and training.

4. Benchmark

To create the benchmark, we select two models:

1. Design by my own

The diagram below shows the model designed by me, and a detailed explanation is discussed in the methodology section.


(32, 3, 3, 3)
(32)
MkldnnConvolutionBackward
ReluBackward0
MaxPool2DWithIndicesBackward
(64, 32, 3, 3)
(64)
MkldnnConvolutionBackward
ReluBackward0
MaxPool2DWithIndicesBackward
(128, 64, 3, 3)
(128)
MkldnnConvolutionBackward
ReluBackward0
MaxPool2DWithIndicesBackward
(256, 128, 3, 3)
(256)
MkldnnConvolutionBackward
ReluBackward0
MaxPool2DWithIndicesBackward
(512, 256, 3, 3)
(512)
MkldnnConvolutionBackward
ReluBackward0
ViewBackward
(256, 512)
(256)
MulBackward0
TBackward
AddmmBackward
ReluBackward0
(135, 256)
(135)
MulBackward0
TBackward
AddmmBackward

2. Pretrained model VGG19

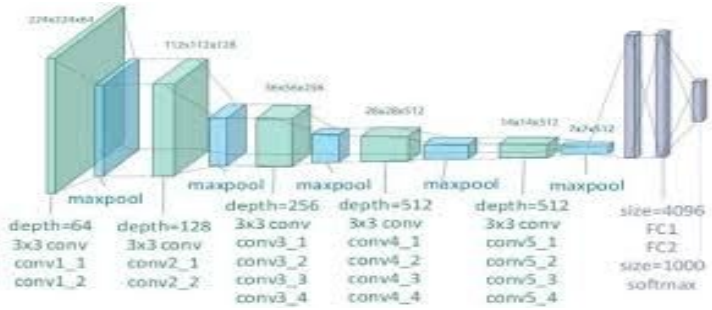


Fig shows Vgg19 model

reference : image source google

The best model is chosen to deploy the model for real time classifications. The test accuracy is used to test the models.

Methodology

1. Data Preprocessing
   The preprocessing the data includes:
   1. Shuffling of images
   2. Creating training and valid, test trainloaders
   3. Using transformations in loaders as discussed

   Then all types of loaders(train, valid and test) are combined in one all_loaders.

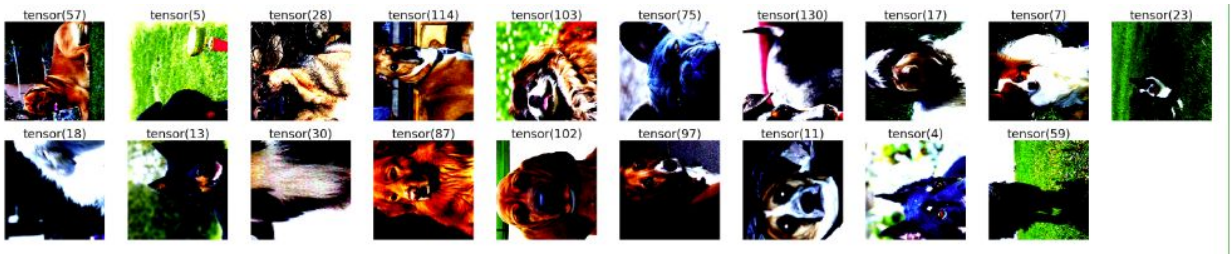


Fig. 5 loader data after transformations.

2. Implementation
   The Implementation process can be split into few stages:
   1. Create the best model to classify breed.
   2. Dog or human face detector.
   3. Wiring the web application

   During the first stage, I Designed the model to train on preprocessed training data. All steps in step 1 is done in aws sagemaker notebook instances. Each

step can be further divided into the following steps:

a. Load training and validation data into memory, preprocessed them.
b. Implemented train function:
    i. We loop over no. of epochs given as parameter
    ii. For each loop, we pass data taken from loaders to the model and get the output data.
    iii. Output data is compared with target value and loss is calculated through criterion which is taken as cross entropy loss.
    iv. We used SGD optimizer to update weights.
    v. Then we evaluated the model with a validation loader and calculated validation loss.
    vi. After all epochs we return the model with updated weights.
c. Implemented test function:
    i. For each test data, batch wise we pass data to the model, get loss value between target value and predicted value and finds the correct value.
    ii. Then we print the test accuracy.
d. I designed the model named as BreedClassifier:
    i. For the model, I put 5 convolutional layers with relu activation function.
    ii. Max Pooling layer after each conv layer
    iii. Then one dropout layer
    iv. Then the fully connected layer with output layer contains 133 classes to classify.
    v. Then I created one method for transfer learning, where I used a Vgg19 pretrained model and in which I changed the last layer to classify all neurons to 133 classes.
e. I trained training data with train function, and got the model with updated weights.
f. I tested the model, and got very poor performance for the data which took me at least an hour to train.
g. Training using sagemaker:
    i. Upload the training data to S3 bucket.
    ii. Define sagemaker estimator with pytorch model
    iii. Put the model.py file and train.py file in src directory
    iv. Fit the data using the train data.
    v. Then Deploy the estimator.
    vi. Then I tested the estimator and got very impressive results on the test data.
h. So, I decided to move forward with the transfer learning approach.

i. Now I tried to improve the estimator with hyperparameter tuning, but later dropped it, since it was taking too much time around 7hrs.
j. Now extract the class names for 133 classes from training classes using regex.
k. Test the model on a single image and it passed the case.

And We get the best model.

For the second phase, we used pretrained Vgg19 to detect if the given image is a dog or not as VGG19 already trained on dog images and is able to detect the dog image. We also choose to detect if the given image is human or not using opencv, but later on the idea dropped, since I was unable to install opencv on the sagemaker endpoint. But created the detector to detect human face using HAAR method on notebook instance and provides the good results.

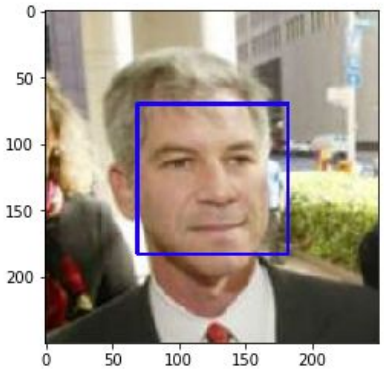


Fig contains image which detect human face using opencv.

For Third Stage,

This is one of the most important stage among all of them.

a. Created Index.html used to send image base64 url to api.
b. Created api gateway which accepts post method in ‘application/x-image’ content type.
c. Connect Api gateway to lambda function
d. In lambda function, create a runtime variable which invokes predictor endpoint and sends the decoded image to endpoint.
e. Created predictor endpoint which accepts image as byte array and converts the data to pil image to use it as an input to model.
f. Model returns the classified class and string data is sent to the lambda function after detecting if the image is dog or not.

3. Refinement

   The model can be improved using sagemaker hypertuning method, by changing training parameters learning rate, batch size, no. of epochs. Test accuracy is calculated in instances of 87%. And for detectors it is 97-98%.

Results

1. Model Evaluation and Validation

   Accuracy calculation for the models:

Model	Accuracy Test
Designed Model	56%
Pretrained Model	87%

detectors	Accuracy Test
Dog detector	98%
Face detector	94%

Find which breed your dog got or u got ?

Upload your image below and click submit to find out...

Upload image

Choose file Bluetick_coo…d_01981.jpg

Submit

It's look like Dogs Detected! It looks like a Bluetick coonhound

Find which breed your dog got or u got ?

Upload your image below and click submit to find out...

Upload image

Choose file 69f3184f1668…94c2c9f.jpg

Submit

It's look like Dogs Detected! It looks like a Belgian malinois

Find which breed your dog got or u got ?

Upload your image below and click submit to find out...

Upload image

Choose file Belgian_mali...is_01407.jpg

Submit

It's look like Dogs Detected! It looks like a Belgian malinois

Results when sending image data from web application

2. Justification

   Prediction takes around a few seconds, to upload image, create base64 image url, then post it to api gateway link, prediction through predictor endpoint, which is done by lambda function, and lastly sends the predicted json string to client end.

   It has become a power application to classify dog breeds.

Thank You