UDACITY Machine Learning Engineer Nanodegree

Capstone Proposal: Dog Breed Classifier using CNN

Chee Chung Lee

September 5th, 2020

Domain Background

Machine learning and AI have been expanding increasingly across the world and influencing every decision we make from digital marketing, targeted advertising, data analytics, robotics, diagnosis in various fields and so on. In today's world alot of the data are digitalised but the implementation of good ML models to generate insights, diagnosis, recommendation, warnings, predictions opportunity is still very large.

The dog breed classifier problem is well known and it has been published numerously online. This is a multiclass classification problem which can be solved with supervised learning. This aim of this project is to build a model which can predict the dog breed if a dog exists in the image.

I chose this dog breed classification project so that I can learn deep learning neural network application for classification in image processing. Through this project I learn all the basics of developing a machine learning neural network model from the ground up. With this skill I hope to implement and improve on different models for similar applications in other industries which I have experience and interest in such as tracking unsafe social practises in the COVID-19 hit world.

Problem Statement

The problem to be solved in this project is to process and classify the images from real world web app users with a high accuracy of above 70%. The algorithm has 3 main tasks to be performed:

1. Determine if the images supplied to the application has a human or a dog?

2. Given the identified image is a human, what dog breed does the person resembles?

3. Given the identified image is a dog, what dog breed is the canine?

Datasets and Inputs

For this project, the input format is of jpg image type because people who uses the web app to determine the dog breed type will most probably use their phone or camera to snap a photo which will be in the same format. The image taken will be uploaded to the server through the app for image processing.

All the photos for this project training, testing and validation are provided by Udacity which consists of dog images and humans.

Dog images dataset have 8351 images which are separated in 3 file directories which are train (6680 images), test (836 images) and valid (835 images). Each of these 3 file directories have 133 folders which corresponds to the 133 different classes of dog breeds.

Human images dataset have 5,749 folders sorted by the name of the human and contain 13,234 total images. Our data is not balanced because we have one image of some people and several for others. The same is for dog images. (the difference is from 1 to 9 images in most cases)

Dog images have different image sizes, different colour, different backgrounds, different angle and posture and orientation of the dog, some dogs are in full sizes and some just a head. The variation is good so that we can train the model to identify a breed regardless of these factors and minimise overfitting.

Human images are all of the same size 250×250. Images are with different backgrounds, light, from different angles, sometimes with few faces on the image. Human classification of the 5,749 folders was not performed, thus not a huge concern as long as a human can be identified in this project.

Solution Statement

To solve this problem, we would deploy a multiclass classification utilising convolution neural network. A convolutional neural network (CNN) is a Deep Learning algorithm which can take in an input image, assign importance through learnable weights and biases to various patterns/objects in the image. The distinct patterns in the images are able to be differentiated from each other using the algorithm.

The solution for this problem involves three steps of the following:

1. Detect human image using existing algorithm like OpenCV of Haar feature-based cascade classifiers which identifies faces.

2. Detect dog images with a pretrained VGG16 model.

3. Once the image has been determined if its a human or dog, the best match out of 133 breed is predicted. This is performed by passing the image through a CNN built either from scratch or from pretrained models.

Benchmark Model

Benchmark model will be the CNN built from scratch which had approximately 10% accuracy rate. 10% is accuracy signifies that the model built is better than a random guess because if it was a guess it would have been 1 out of 133 (breeds).

The CNN model created using transfer learning must have an accuracy level of 60% or above to show that the model is working well and trained well for the problem at hand.

Evaluation Metrics

The problem we try to solve is a classification problem. The data set provided is unbalanced with uneven number of samples in different classes. Two things we will do here. They are as follow

Step Step 1: Training and validation.

Simple classification accuracy score is not reliable, giving false sense of improvement and calibration on the model. This log loss takes into account the uncertainty of the prediction based on how much it varies from actual label and the prediction. When working with log loss, the sample image would have probabilities to match with all the different classes. The loss is the difference between the predicted probability and the label. On Kaggle they use multi-class log loss metrics to evaluate models which I would use as well. Log Loss of the training and validation set can be evaluated after each epoch cycles to check if the model is improving and not overfitting.

<u>Step 2</u>: Testing

Once the model has been trained and validate and we see the log loss has reduced to a minimum, we can test the model. We test how good the model is by testing its accuracy for prediction. For example if an image contains a dog. Its accuracy would be how many times it predicts the dog breed class correctly out of the total number of predictions.

Project Design

In this section, the theoretical workflow for approaching a solution given the problem is as follow.

<u>Step 1</u>: Import Dataset.

Import all the necessary dataset and libraries from Udacity which contains all the human and dog images required for the project. The human files and dog files are stored separately.

<u>Step 2</u>: Detect Humans

Explore different face detectors from OpenCV to be used. By creating a face detector we would be able to tell if a human is present in the image. To do this we will implement Haar feature-based cascade classifiers. The workflow required to detect faces are:

-initialize pre-trained face detector

-load image of humans imported

-convert image to grayscale

-test the human detector by loading human and dog images

-verify that the human detector can detect human faces accurately and cant detect dog in the image.

<u>Step 3</u>: Detect Dogs

We will use the pre-trained model VGG16 to detect dogs in images. This model has been pretrained for 1000 different classes through ImageNet.

– Define VGG16 model

– Use GPU for better performance if available

– load and pre-process the image

– send an image to the VGG16 model

– verify that the dog detector can detect dog in images accurantely and cant detect dog in the human images.

- Model return true if the index is 151 to 268 as the classes between 151 and 268 is for dogs.

<u>Step 4:</u> Create a benchmark CNN model to classify dog breeds (from scratch)

The dog data is already divided into training, validation and test data. The data in images needs to be converted to batch data in the form of tensors for the model. Training and validation is used concurrently to verify that the validation loss of the model created is actually reducing and improving and not just training loss getting reduced. Once the model is sufficiently trained through a few epochs, it's accuracy is being evaluated using the test data. If it doesnt hit 10% accuracy, the model architecture and training parameters can be fine tuned. Once this is achieved, it can considered as the benchmark model.

Step 5: Create a CNN model using transfer learning using Resnet or VGG to clasify dog breeds. With transfer learning, we can maintain the model architect and just integrate it with the output layer classification we require. We train, validate and test the model just as in step 4. This time we expect the accuracy to be over 60%.

Step 6: Create and test the algorithm.

Create the algorithm by combining the human detector, dog detector and the transfer learning dog breed classifier model. The response the algorithm should produce are the folowing:

- If the image input contain a human, confirm that human is present and what dog breed the human resembles.

- If the image input contains a dog, confirm that a dog is present and what dog breed is the dog.

- If the input image contains neither, provide output which indicates error.

Reference

https://en.wikipedia.org/wiki/Convolutional_neural_network

https://www.kaggle.com/c/dog-breed-identification/overview/description

https://github.com/udacity/dog-project

https://www.kdnuggets.com/2018/04/right-metric-evaluating-machine-learning-models-1.html

https://towardsdatascience.com/metrics-to-evaluate-your-machine-learning-algorithm-f10ba6e38234

https://pytorch.org/tutorials/beginner/blitz/neural_networks_tutorial.html

https://pytorch.org/tutorials/beginner/blitz/cifar10_tutorial.html

https://pytorch.org/tutorials/beginner/transfer_learning_tutorial.html?highlight=transfer%20learning

https://medium.com/@josh_2774/deep-learning-with-pytorch-9574e74d17ad

https://towardsdatascience.com/a-guide-to-an-efficient-way-to-build-neural-network-architectures-part-ii-hyper-parameter-42efca01e5d7

https://towardsdatascience.com/metrics-to-evaluate-your-machine-learning-algorithm-f10ba6e38234