Machine Learning Engineer Nanodegree

Capstone Proposal - **CNN Project: Dog Breed Classifier**

Chukwuemeka Ezumezu

January 7th, 2020.

Domain Background

Artificial Intelligence is becoming part of our everyday life and most of the problems facing our generation are being easily solved with the help of AI. From cancer detection to self-driving cars and so on.

In this project, we will be training an image classifier that will be included in a smartphone and web app where users of the app can use it to easily identity dog breeds of the image of dog taken with their phone or personal computers, it can as well tell us the dog breed camera is looking at without snapping the picture. The app can as well check image of human and predict the closest resembled dog breed, just for fun. We will train the classifier and then deploy it for use in the applications. These applications will be very useful for people shopping for dog and need to know and understand the dog breed they are buying, and so on.

There are already some related academic research on this project, like the research done by Whitney LaRow, Brian Mittl and Vijay Singh of stanford university. The research paper can be found on this link: https://web.stanford.edu/class/cs231a/prev_projects_2016/output%20(1).pdf

Problem Statement

This project make use of Convolutional Neural Network (CNN) to classify dog breeds.

First, will create a CNN from scratch and try to attain a test accuracy of at least 10% and then use transfer learning to create a CNN that can attain greatly improved accuracy.

A pretrained model is obtained, restructure to fit our desired model and then restrained with our dataset to fit our purpose. It is relatively rare to have a dataset of sufficient size, that makes it common to use a CNN that is already trained on a very large dataset (e.g ImageNet, which contains 1.2 million images with 1000 categories). Transfer learning can save us a lot of time and computational resources being that we can not start developing the model structure and training from scratch, It gives a very high accurate prediction of more than 90% if well trained.

The importance of transfer is demonstrated in this project by training a CNN from scratch and comparing the accuracy of its predictions with the accuracy of transfer learning. Building CNN from scratch also help us to understand how CNN works. It helps us understand the convoluted and max-pooling layers, and the learning process of CNN in general.

There are many pretrained models that can be use for this project, like vgg16, Inception-v3, ResNet-50 and so on. A good accuracy level can be gotten from each of them.

Datasets and Inputs

We are going to use image dataset provided by Udacity in the Dog Project Workspace for both the dog and human dataset. There are 8351 total dog images with 133 different breeds and 13233 total human images in the datasets. We will be splitting the 8351 dog images, into training, validation and testing ratio of 80: 10: 10 respectively. We have 6680 dog images for training, 836 dog images for validation and 835 dog images for testing. The images are balanced with an average of 50 images per dog breed (across the 133 dog breed categories). Testing set will help us measure the performance of our model in yet unseen dataset.

The image sizes are different, likewise the orientation . To pre-process the images, we resize them to 224x224 pixels so it can fit our network structure.

We will use OpenCV's implementation of Haar feature-based cascade classifiers to detect human faces in images. First we will convert BGR image to grayscale to be able to easily find faces in the image, print number of faces detected in the image then get bounding box for each detected face. We then convert the image back to color image and display it along with bounding box. Making it easily to detect human faces in a picture with higher accuracy.

Solution Statement

Training a pretrained CNN with our datasets and testing it gives us the ability to detect and classify an image dog breed and human face from an image. We will be writing an algorithm that our app will use to predict the dog breed based on provided image. If the image is a dog our app will return a predicted breed. Our app will also predict the resembled dog breed if the input image is a human but will show an error if the input image is neither that of a dog nor human.

Benchmark Model

One of the earlier works in fine-grained classification was an attempt at identifying plant species by Belhumeur. This approach involved segmenting a leaf and then using shape to determine the species. Along similar lines, a paper by Farrell attempted to identify a bird species by finding key points along the beak, eyes, wings, feet, and tail, and building features around them. More relevantly, however, a 2012 paper by Liu attempted dog breed identification using a similar approach. They first use an SVM regressor using grayscale SIFT descriptors as features to isolate the face of the dog. To handler rotation and scale, the window is also rotated and scaled; by using non-maximum suppression and picking the detection with the highest score, they isolate a single best window. The primary focus of the paper is to find the facial keypoints of the dog. Liu leverages a part localization algorithm, in which a sliding window SVM detector using ear tip. Thus, the goal of this part of the analysis pipeline is defined as follows: given an unseen image from the testing set, predict the key points of the dog face as close as possible to the ground truth points in terms of pixels. Convolutional neural networks have been shown to perform quite well on a variety of image detection and classification tasks, including human face detection, but previous literature on dog face detection had not used neural networks; hence, we decided to tackle a novel approach for dog face keypoint detection.

To solve this problem, we trained a fully connected convolutional neural network on 6680 training images, which was used to predict the key points of 835 testing images. Before training, the images were all scaled to be 224×224 pixels. (Sourced from the research paper: https://web.stanford.edu/class/cs231a/prev_projects_2016/output%20(1).pdf)

Evaluation Metrics

Since we have a balanced dataset with an average of 50 images per class of dog breed and we are trying to solve a classification problem, we are going to use accuracy as our evaluation metric. Otherwise we would have also use precision or F1 Score, if we do not have a balanced dataset.

Project Design

The workflow for the project goes as follows:

Step 0: Import Datasets

Step 1: Detect Humans

Step 2: Detect Dogs

Step 3: Create a CNN to Classify Dog Breeds (from Scratch)

Step 4: Create a CNN to Classify Dog Breeds (using Transfer Learning)

Step 5: Write your Algorithm

Step 6: Test Your Algorithm

We first import the dogs and human datasets, then split them into training, validation and testing sets with ratio of 90: 10: 10 respectively.

We will then use OpenCV's implementation of Haar feature-based cascade classifiers to detect human faces in images. By first converting BGR image to grayscale to be able to easily find faces in the image, print number of faces detected in the image then get bounding box for each detected face. We then convert the image back to color image and display it along with bounding box. Making it easily to detect human faces in a picture with higher accuracy.

We will use a pretrained model from ImageNet either (VGG-16 or ResNet-50), which is customised and retrained with our dataset, to detect dogs in images using PyTorch.

When training our classifier, we first create a CNN dog breed classify from scratch and try to attain a test accuracy of at least 10%. It also helps us to understand the convoluted and max-pooling layers, and the learning process of CNN in general.

We will then Create a CNN classify using transfer learning which is going to serve as the main classifier for the project.

We will write the inference algorithm that does the prediction of dog breeds and then test our model.

References

The dataset and starter codes used in this project is available in Udacity in the Dog Project Workspace as part of Udacity Machine Learning Nanodegree Program.