Machine Learning Engineer Nanodegree

Capstone Project

Yijun Wei
August 3rd, 2020

I. Definition

Project Overview

Machine learning (ML) is widely used in different areas nowadays such as image recognition, and classification. One insteresting application is to use ML for classifying animal's breed, which is challenging for human beings, because different breeds of the same animal look similar to each other. Dog is the most common animal in our society so dog breed classification becomes really popular these days.

In this study, we not only design a classifier to better classify the dog's breed, and we go one step further, that is, if the given picture is not a dog, but a person, what is the most resembling dog. This application could help people to decide which breed of dog is their best option in terms of appearance.

Problem Statement

The goal of this project is to build a real-time app that given a picture, if the picture is consisted of a dog, the breed is detected, and if the picture contains a human being, a human being is detected. Otherwise, an error will be detected.

Metrics

The entire dataset is split into training, validation, and test dataset at 0.8-0.1-0.1 level. The model is trained on the training dataset and hyperparameters are tuned on the validation dataset. We will evaluate our model on the test dataset using accuracy calculated as wrong predicted instances divided by the total number of instances.

II. Analysis

Data Exploration

For this project, there are totally two types of dataset: the first type is human image dataset, and the second type is the dog breed image. Human images are used to evaluate the face detection algorithm and also look for the resembling dog breed. The second type is the dog breed image dataset, which are used to train the dog breed classifier.

To be specific,the human image dataset contains 13233 total images, and sorted by the name of the human. All the image has the size 250 * 250.

The dog image dataset has 8351 images with dimension 250 * 250, which are split into training, validation, and test dataset at 0.8-0.1-0.1 level. Different images are saved in the directory with the breed's name, and there are 133 breeds. The number of instances are diffrent for the 133 classes.

Algorithms and Techniques

The algorithm is divided into 3 steps:

1. Dog detection: 1.1 A pre-trained VGG on imagenet is used to determine if a dog is detected. 1.2 If a dog is detected, then this problem becomes a typical multi-class image classification problem, and a convolutional neural network (CNN) is used to detect the dog breed. 1.3 If a dog is not detected, then it goes to step 2
2. Human face detection: 2.1 A haar feature based cascade classifiers is used to detect if a human face is included in the image. 2.2 If a human face is detected, a CNN that is used in 1.1 is used to the resembling dog breed. 2.3 If a human face is not detected, continue to step 3
3. Print "Error! Can't detect anything.."

Benchmark

Only model in step 1.2 should be developed, and the CNN model for step 1.2 is created from scratch, which have accuracy of at least 10%. This can confirm that the model is working because a random guess will provide a correct answer roughly 1 in 133 times, corresponding to an accuracy of less than 1%.

III. Methodology

Data Preprocessing

Since only model in step 1.2 should be determined, then only data for the model should be processed. All the dog images are resized to 224 * 244, and normalized to mean 0.485, 0.456, 0.406, and standard deviation 0.229, 0.224, 0.225 for RGB channel in training, validation, and test dataset. Image are randomly flipped, rotated and ColorJittered for the data augmentation in the training dataset.

Implementation

A CNN model from scratch is built using 3 convolution layers with 32/64/128 3 *3 filters, followed by a 2* 2 maximum pooling layer for each convolutional layer. All the pixels are flattened out after the last max pooling layer followed by 0.3 dropout, and the second last layer is a 500 dimensional fully connected layer, followed with the same 0.3 dropout. The last layer converts 500 dimensional layers to 133 dimensional vector, and one dimension represents a dog breed.

Refinement

The CNN created from scratch have accuracy of 16%, which meets the benchmarking. The performance of the model could be significant improved if we are using transfer learning, i.e., retrain a pretrained imagenet model on this specific dog breed classification problem to fully make use of the features the model learnt from the imagenet dataset. The natural choice is the resnet50, which is very popular in classification problem.The last convolutional output of Resnet50 is fed as input to our model. We only need to add a fully connected layer to produce 133-dimensional output (one for each dog category). After training for 10 epochs, the model achieved approxiamtely 81% accuracy.

IV. Results

Model Evaluation and Validation

Human Face detector (step 2.1): The human face detector function was created using OpenCV’s implementation of Haar feature based cascade classifiers. 98% of human faces were detected in first 100 images of human face dataset and 17% of human faces detected in first 100 images of dog dataset.

Dog Face detector (step 1.1): The dog detector function was created using pre-trained VGG16 model. 95% of dog faces were detected in first 100 images of dog dataset and 0% of dog faces detected in first 100 images of human dataset.

CNN using transfer learning (step 1.2): The transfer learning ResNet50 model was trained for 10 epochs, and the final model produced an accuracy of 81% on test data. The model correctly predicted breeds for 682 images out of 836 total images.

Justification

I think the model performance is better than expected. The model created using transfer learning have an accuracy of 81% compared to the CNN model created from scratch which had only 16% accuracy.

Improvement

1. Improve the model training by adding more tricks
2. Increase training dataset size
3. Crop out faces to remove background noise

Before submitting, ask yourself. . .

- Does the project report you've written follow a well-organized structure similar to that of the project template?
- Is each section (particularly **Analysis** and **Methodology**) written in a clear, concise and specific fashion? Are there any ambiguous terms or phrases that need clarification?
- Would the intended audience of your project be able to understand your analysis, methods, and results?
- Have you properly proof-read your project report to assure there are minimal grammatical and spelling mistakes?
- Are all the resources used for this project correctly cited and referenced?
- Is the code that implements your solution easily readable and properly commented?
- Does the code execute without error and produce results similar to those reported?