Capstone Project
Muhammad Faraz Malik

Machine Learning Engineer Nanodegree
April 11th, 2020

Dog Breed Classifier using CNN

Project Overview

The Dog breed classifier is a notable issue in ML. The issue is to distinguish the type of canine, if a dog picture is given as information, whenever provided a picture of a human, we need to recognize the looking like canine breed. The thought is to manufacture a pipeline that can procedure real world user provided pictures and distinguish a gauge of the canine's breed. This is a multi-class classification problem where we can use supervised machine learning to take care of this issue.

Problem Statement

The objective of the task is to assemble an machine learning model that can be utilized inside a web application to process a genuine world, the user provided pictures. The calculation needs to perform two errands:

Dog face detactor: Given a picture of a dog, the calculation will distinguish a gauge of the canine's breed.

Human face detactor: Whenever provided a picture of a human, the code will distinguish the looking like the canine breed.

Metrics

The information is part of the train, test and valid dataset. The model is trained using train dataset. We utilize the testing data to predict the performance of the model on unseen data. We will use accuracy as a measurement to assess our model on test data. Accuracy=Number of things effectively characterized/Every single ordered thing. Likewise, during model training, we compare the test data prediction with the validation dataset and calculate Multi-class log loss to calculate the best performing model. Log loss considers the uncertainity of prediction dependent on the amount it fluctuates from the actual label and this will help in evaluating the model.

Data Exploration

For this task, the input format must be of picture type, since we need to enter a picture and distinguish the type of the dog. The dataset has pictures of

dogs and people.

Dog pictures dataset: The canine picture dataset has 8351 complete pictures that are arranged into a train (6,680 Pictures), test (836 Pictures) and valid (835 Pictures) directories. Every one of these indexes (train, test, valid) has 133 folders corresponding to dog breeds. The pictures are of various sizes and various backgrounds, a few pictures are not full-sized. The data isn't balanced because the number of pictures provided for each breed varies. Few have 4 pictures while some have 8 pictures.

Human pictures dataset: The human dataset contains 13233 total human pictures which are arranged by names of human (5750 folders). All pictures are of size 250x250. Pictures have various backgrounds and various angles. The data isn't balanced because we have 1 picture for certain individuals and numerous pictures for a few.

Algorithms and techniques:

For playing out this multiclass order, we can utilize the Convolutional Neural System to tackle the issue. A Convolutional Neural System (CNN) is a Deep Learning algorithm that can take in an input image, allot significance (learnable weights and biases) to different aspects/objects in the image and be able to separate one from the other. The solution involves three stages. To begin with, to identify human images, we can utilize existing calculations like OpenCV's usage of Haar feature based cascade classifiers. Second, to detect dog images we will utilize a pre-trained VGG16 model. At last, after the picture is distinguished as dog/human, we can pass this picture to a CNN model which will process the image and anticipate the breed that coordinates the best out of 133 breeds.

Benchmark

The CNN model made from scratch must have an accuracy of at least 10%. This can confirm that the model is working because a random guess will provide an answer about 1 in 133 time, which corresponds to an accuracy of under 1%.

Data Preprocessing

All the images are resized to 224*224, then normalization is applied to all pictures (train, legitimate and test datasets). For the training data, image augmentation is done to diminish overfitting. The train data images are arbitrarily rotated and random horizontal flip is applied. At last, all the images are converted into tensor before passing into the model.

Implementation

I have assembled a CNN model from scatch to take care of the issue. The model has 3 convolutional layers. All convolutional layers have a kernel size of 3 and stride 1. The first conv layer (conv1) takes the 224*224 input image and the last conv layer (conv3) produces an output size of 128. The ReLU activation function is utilized here. The pooling layer of (2,2) is utilized which will diminish the input size by 2. We have two completely connected layers that at long last produce 133-dimensional output. A dropout of 0.25 is added to abstain from overfitting.

Refinement

The CNN made from scratch has an accuracy of 11%, though, it meets the benchmarking, the model can be essentially improved by using transfer learning. To create CNN with transfer learning, I have chosen the Resnet101 architecture which is pre-trained on the ImageNet dataset, the architecture is 101 layers profound. The last convolutional model of Resnet101 is fed as input to our model. We just need to add a completely connected layer to deliver a 133-dimensional output (one for each dog class). The model performed incredibly well when compared with CNN from scatch. With only 5 epochs, the model got 74% precision.

Model Evaluation and Validation

Human Face detector: The human face detector function was made utilizing OpenCV's usage of Haar feature based cascade classifiers. 98% of human faces were detected in the initial 100 pictures of the human face dataset and 17% of human appearances recognized in the initial 100 pictures of the dog dataset.

Dog Face detactor: The dog detector function was made utilizing a pre-trained VGG16 model. 100% of dog were detected in the initial 100 pictures of the dog dataset and 1% of dog faces detected in the initial 100 pictures of the human dataset.

CNN using transfer learning: The CNN model made using transfer learning with ResNet101 architecture was trained for 5 epochs, and the final model produced an accuracy of 74% on test data. The model correctly predicted breeds for 621 pictures out of 836 complete pictures.

Accuracy on test data: 74% (621/836)

Justification

I think the model performance is superior to anticipated. The model made utilizing transfer learning has an accuracy of 74% compared with the CNN model made from scratch which had just 11% accuracy.

Improvement

The model can be improved by adding more training and test data, presently, the model is made created using just 133 breeds of the dog. Additionally, by performing more picture expansion, we can abstain from overfitting and improve accuracy. I have attempted uniquely with ResNet 101 architecture for feature extraction, Possibly the model can be improved utilizing the distinctive architcture.

References

1. Original repo for Project - GitHub: https://github.com/udacity/deep-learning-v2pytorch/blob/master/project-dog-classification/

2. Resnet101: https://pytorch.org/docs/stable/_modules/torchvision/models/resnet.html#resnet101

3. Imagenet training in Pytorch: <https://github.com/pytorch/examples/blob/97304e232807082c2e7b54c597615dc0ad8f 6173/imagenet/main.py#L197-L198>

4. Pytorch Documentation: https://pytorch.org/docs/master/