Machine Learning Engineer Nanodegree

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

Sedat Erkal

July 30th, 2020

Domain Background

In recent years, software developers will need to know how to incorporate deep learning models into everyday applications. Anything with a camera will be using image classification, object detection, and face recognition, and so on, all based on deep learning models. Previously working on classifying images with standard neural network architecture, this project is a good opportunity to experiment with convolutional neural network (CNN) architecture as well as transfer learning.

Problem Statement

Given an image of a dog, the proposed algorithm will identify an estimate of the dog's breed. If supplied with an image of a human, the code will identify the resembling dog breed. One alternative solution to the problem is benefiting from the pre-trained models that lead to obtaining higher accuracies.

Datasets and Inputs

The related dataset consists of human and dog images in order to use in the classification task. Human images are used to detect human faces and give resembling dog breed in the prediction phase. Similarly, dog images are used to detect dogs, train the model, and give the predicted breed. Dog images dataset is particularly helpful to detect dogs in images since we use a pre-trained model (VGG-16) that has been trained on **ImageNet**, a very large, very popular dataset used for image classification and other vision tasks.

Solution Statement

Starting with detecting humans and dogs in images, we need a way to predict breed from images. Our vision-based algorithm will have to conquer high intra-class variation to determine how to classify different shades as the same breed. That's why we will use

transfer learning to create a CNN architecture in the classification task that attains greatly improved accuracy.

Benchmark Model

The task of assigning breed to dogs from images is considered exceptionally challenging. To see why consider that even a human would have trouble distinguishing between different breeds. Our proposed approach to this task will be using a CNN from scratch that classifies dog breeds and attains a test accuracy of at least 10%.

Evaluation Metrics

Due to dealing with a classification task, the accuracy score is a good evaluation metric for our specific case both in the benchmark model and the solution model. For the benchmark model, it is supposed to have at least an accuracy score of 10% and for the solution model, it is ensured that the accuracy score is higher than 60%.

Project Design

The initial step of the workflow is importing human and dog datasets to the specified directories. Then, to find human faces in a simple image, we use a pre-trained face detector. The next step is to detect dogs in mages with the help of a pre-trained model. In our task, we will download and use the VGG-16 model along with weights that have been trained on ImageNet that contains images from 1000 categories. Given an image, this pre-trained VGG-16 model returns a prediction for the object that is contained in the image. An important step before making predictions with a pre-trained model is pre-processing the image (scale, crop, rotate, and normalize).

Now that we have functions for detecting humans and dogs in images, the next step is to predict breed from images. For this purpose, it is time to create a benchmark CNN model. Since we are using the PyTorch framework, we need to specify data loaders for the training, validation, and test datasets of dog images. Then, we are creating a CNN architecture with several CNN layers together with defining non-linear activation function and max-pooling. In addition, fully-connected layers are defined with necessary activation function (*ReLU*) and the regularization method (*dropout*). Another step is to specify a loss function and optimizer, following to train and validate our model. We save the model with the highest validation accuracy and then try out on the test dataset of dog images.

As a solution model, we will now use transfer learning to create a CNN that can identify dog breed from images. Initially, we are specifying the pre-trained model architecture. Here, we want to freeze the parameters so we do not backdrop through them. After that,

we define new sequential fully-connected dense layers different from the pre-trained model to make it work in our specific task. We are going to use already defined data loaders, loss function and optimizer, train and test functions in the definition of the solution model as well since they are providing the same functionality as in the benchmark model.

We will conclude the project with an algorithm that accepts a file path to an image and first determines whether the image contains a human, dog, or neither. Then, if a dog is detected in the image, the algorithm returns the predicted breed, if a human is detected in the image, the algorithm returns the resembling dog breed, and lastly, if neither is detected in the image, the algorithm provides an output that indicates an error.