Machine Learning Engineer Nanodegree

Capstone Proposal

Sebastian von Hoesslin
October 16th, 2019

Proposal

Domain Background

In this section I will provide brief details on the background information of the domain from which the project is proposed. As my Machine Learning Engineer Nanodegree Capstone Project I chose the dog breed classifier challenge, which was once a Kaggle challenge (https://www.kaggle.com/c/dog-breed-identification).

In this challenge a strictly canine subset of the famous ImageNet Dataset is provided with the purpose to reach a fine-grained image categorization. This dataset is mixed with the Stanford Dogs Dataset (http://vision.stanford.edu/aditya86/ImageNetDogs/) with the aim to gain a large enough Dataset to make high accuracy predictions possible.

As I am personally very fascinated by deep learning algorithms for computer vision tasks, I chose the project for my capstone challenge

Problem Statement

The problem to solve was as follows. I was given two datasets. One dataset with dogs of different breeds and one dataset containing images of human faces. The underlying problem is to define a model strong enough to recognize whether it is confronted with an image of a dog or with an image of a human. In case an image of a dog is provided, the model should be able to estimate its breed with an acceptable amount of confidence. In case an image of a human is provided, the model should compare it with its breed classes and tell us which breed our human face resembles most. The goal is to achieve a high accuracy for image classification on both above mentioned tasks.

Datasets and Inputs

In this section, I will briefly analyze the data I will use to solve the problem. As I mentioned above, two datasets were provided, one with images of dogs of different breeds and one with images of human faces.

The first step to take, when provided with data, is to explore all given datasets in order to get a first impression on the possibilities one has in order to solve a problem.

In contrast to the human dataset our dog dataset naturally will be provided with features that indicate their breed. These identification labels are provided in the form of class ids and breed labels. A first check on the dataset revealed, that there are 133 classes of different dog breeds in my dataset.

Solution Statement

So, the underlying problem is to determine whether we have a human or a dog in our picture. If it is a human, we want our model to recognize that and then find out which dog breed this human resembles most. If our model is confidant it found a dog, we want to have the probabilities about the different breeds this dog could belong to.

Benchmark Model

A benchmark model was provided by Udacity. For the model we build from scratch an accuracy of correctly classified images with the value of 10% is given. While this may sound small in the beginning, it is in fact reasonable, as we must train parameters from scratch and are being given relatively small datasets. It is therefore very likely that our model will be prone to overfit the training data and will not be too good at generalizing well, when confronted with new, previously unseen data.

For the second task of transfer learning a benchmark of 60% in accuracy must be outperformed. Here, a relatively high benchmark makes sense, as we don´t have to train the parameters of our pre-trained model, but instead can simply modify the output layer according to our needs.

Evaluation Metrics

As I mentioned above, this project will use the accuracy of the models, that is number of correctly classified images by their labels, as the evaluation metrics. In addition, I will observe the loss and cross validation loss during the training process of our models in order to identify whether our models overfit the training data or hyperparameters.

Project Design

The project design has the following outline:

Step 0: Import Datasets

Here, I will load the dog and human datasets and import the necessary libraries as well.

Step 1: Detect Humans

I first created a human detector. A pretrained face detector will be used for this purpose ('haarcascades/haarcascade_frontalface_alt.xml').

Step 2: Detect Dogs

Next, I will use a pre-trained VGG-16 model for the dog detector. It has to be considered, that the input images must be cropped to 244x244 pixel tensors in order to be used be the model.

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

In this step I will split our datasets into train, validation and test sets, extract the possible class names and define a model architecture which is inspired by the VGG 16 Model, developed by Oxford University. The goal is to attain a test accuracy of at least 10%.

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

In this step I will use a pre-trained ResNet50-Model for transfer learning, which has proven to be very powerful for a multiclass classification task like ours before. My CNN must attain at least 60% accuracy on the test set.

Step 5: Write your Algorithm

Next, I will write my own algorithm that accepts a file path to an image and first determines whether the image contains a human, dog, or neither. This is the conditions my algorithm will consider during its task:

• if a dog is detected in the image, return the predicted breed.

• if a human is detected in the image, return the resembling dog breed.

• if neither is detected in the image, provide output that indicates an error.

Step 6: Test Your Algorithm

In a last step I will test my algorithm on sample data which I have downloaded separately. By doing so I will get a direct feedback on my work and can conclude on where possible weaknesses can be identified.