Machine Learning Engineer Nanodegree
Capstone Proposal

Bas Donker
Date: 2018/01/10

Domain Background:

In the age of social media and smartphones that have better camera's than professional camera's less than two decades ago, we take a lot of pictures. We update our social media feeds with pictures of food, we send selfies over instant messages, and we even create separate social media accounts so we can justify uploading hundreds of photos of our dog.

Just two generations ago, taking a picture of something was an event in and of itself. Film didn't come cheap, so more care was taken into taking the perfect shot. Once the camera roll was filled up, it would be taken to a shop to develop the photos and these photos would then carefully be put into a photo album during a rainy Sunday afternoon.

In the far future, we can show these photo albums to the grandkids to show them what life was like before they were around.

Problem Statement:

With the amount of pictures we take today, an individual photo is significantly less important than a photo used to be. We take a few dozens of takes for a group photo, just in case someone blinked in any of them.

We certainly never take the time to organize our photos anymore. There's too many of them and we're too busy taking new ones to be able to spend time organizing. This also means it becomes very difficult to find a picture again after it's been burried under the thousands of new pictures we've taken since then.

Solution Statement:

We can use a convolutional neural network to look at a selection of pictures and determine what is in the picture. It can then add what objects it found into the metadata of the file, so they can be searched for more easily.

Datasets:

As the input data, I will be using pictures I have taken over the years. I will organize a few hundred of them manually, putting them in directories that correspond to the name of the label I want to give them.

I will use categories of things I frequently take pictures of, but other users can easily arrange the pictures in any way they please, as long is enough material for the CNN to train on.

These images will be split up in separate training, cross validation and test sets.

Benchmark Model:

To test the effectiveness of our Convolutional Neural Network, we will test its results agains

a traditional Fully Connected Neural Network and another algorithm that just makes random guesses as to which category a picture belongs to.

Evaluation Metrics:

If the model is working well, it's able to classify pictures it hasn't seen before and put them into its respective categories. Pictures of landscapes will be tagged with 'landscape', pictures of dogs will be tagged with 'dog', and selfies of my own face will be tagged with 'handsome bastard'.

The model will be evaluated based on the percentage of images it hasn't seen before it classifies correctly in the test set.

Project Design:

In a directory on a computer, there are several subdirectories wich are named according to the labels I want to classify my pictures as. For example, the file structure might look something like:

- images
    - landscapes
    - group photos
    - selfies
    - food

In the sub-directories will be pictures of landscapes, group photos, selfies and food respectively.

During the pre-processing these pictures will be augmented. Copies will be made of them in which they are rotated a random amount, or they may be flipped horizontally or vertically. This is done to generate more data for the model to train on. We then resize all the pictures to about 200 by 200 pixels.

After the data has been processed, we split them up randomly into training, cross validation and testing set.

We train the convolutional neural network to recognize the pictures and the categories to which they belong. We also train a conventional fully connected neural network to do the same, and we'll write an additional model that just randomly guesses the category.

We then compare the results of all three of these models to see which ones perform best on data they haven't seen before (the testing set).

Once we're happy with the model's performance we can implement it in the real world. I can create a new directory where I will upload all the pictures I take. The CNN will see that there are unclassified pictures in there and will attempt to put them in their rightful subdirectories.

This way I can more easily manage my pictures and I should be able to find specific ones without much effort at all.