Machine Learning Engineer Nanodegree

Detecting if there's a car in an image or not (Binary image classification)

Laith Al Rachwani

December 23rd, 2017

Proposal

Domain Background

Image processing has been studied a lot and has caught so many people attention, including mine, mainly because we can extract so many important data from pictures that could be useful, perhaps one of the most famous examples on image processing is facial recognition which has been talked and researched quite a lot, and it's widely used now, it's now even used in the new iPhone X where you can use the facial detection feature in order to access your phone, this article talks about Apple using machine learning algorithms in their lastest iPhone X: https://www.wired.com/story/apples-neural-engine-infuses-the-iphone-with-ai-smarts/

Another famous example on image processing is the "Search by image" feature google provides, where you can search using a picture you choose, and google will find similar pictures for you, all this is done using computer vision and machine learning techniques.

This just proves how important such a field is, which is image processing where we can extract features out of images, and i have always wondered before getting into machine learning how such a thing happens, how can we extract information from pictures, now that i know i thought working on a problem like this would be a great start for me to start working more on image processing since it's a growing field in computer science and especially in machine learning.

Problem Statement

The problem i will be solving here is a binary image classification problem, where i will be classifying pictures into two classes, either a picture that has a car in it or a picture that does not.

For this problem i will be using a Convolutional Neural Network, CNNs have proven to be quite efficient when solving such problems, when we are trying to classify images, so for this problem i will train a CNN how a car looks like (and how a car does not look like) so that it can learn how a car looks and predict the possibility of having a car in a certain picture.

Datasets and Inputs

For this problem i will be using two different datasets, the first dataset will contain pictures of cars, these pictures are taken from different angles and from different views, the dataset i will using that will provide these examples is taken from stanford.edu and can be found through this link: http://ai.stanford.edu/~jkrause/cars/car_dataset.html

Another dataset i will be using is a dataset called “The LabelMe-12-50k dataset” this picture contains from many classes, of course i am going to exclude any picture that has a car in it since i am going to use the pictures in this dataset as examples of what a non-car looks like, the dataset can be found through this link: http://www.ais.uni-bonn.de/download/datasets.html

For my first dataset that contains pictures of cars, the size of each picture is different, as for my second dataset, the size for each picture is 224x224, however i will be converting the size of each picture in both datasets to a size of 150x150

Pictures in both datasets have colored layers, which means each picture is represented as a 3D array.

As for the number of pictures i will be using, i will be using around 4500 pictures from each class, so that will result in around 9000 pictures from both classes, and i will be using 20% of these 9000 pictures to validate my model.

Solution Statement

The approach i will be using for this problem is a deep learning approach, where i will be going to train CNN how a car looks like and how a car does not look like, by providing examples to the CNN of pictures that have a car in them that will be labeled with 1 and pictures that don't have a car in them that will be labeled with 0 , the input of my CNN is going to 150x150x1, since i am going to convert each picture to grayscale and resize each picture down to a size of 150x150.

Benchmark Model

There are many powerful models out there for image classification, perhaps the one that caught my attention the most is GoogLeNet model which was always one of the top models and has been tested on a several multi class classification problems. So i think this model would be a good benchmark.

Evaluation Metrics

I Will be using the accuracy metric for this problem, where our CNN will use our validation data to calculate the accuracy, the accuracy in general means, out of the predictions our model made, how many of these prediction were actually correct, which also mean how many images were actually classified to the correct class out of all the pictures we tried classifying.

Project Design

As mentioned above, i will be using a Convolutional Neural Network for this problem, where the input is going to have a size of 150x150x1, and the output of the CNN will be a single neuron and the value it outputs will be between 0 and 1, where the closer the output was to 0 means the closer the probability that this picture does not have a car in it and the closer the out was to 1 means the closer the probability that this picture does indeed have a car in it.

Before feeding my CNN with the pictures, i am going to do some kind of preprocessing which will involve converting each picture to a grayscale image, which means now our

pictures are represented as a 2D array instead of 3D. And of course we will be resizing each picture to a size of 150x150. Another important normalization step is converting each value in our picture to a value between 0 and 1, it’s always a good practice to work with normalized data.

After training our CNN we can then use it on never seen before data to predict whether the input has a car in it or not.

As for the type of the model, I have seen somewhere before while studying the deep learning course in my nanodegree and doing some research about CNNs that some simple CNNs can yield great results, so based on that I will try to make my architecture simple, which will probably use 3 to 4 convolutional layers, will also include a max pooling layer between e each convolutional layer. this will then be followed by a dense layer and an output layer of one neuron, what i will also be using is a dropout layer at which will help our model generalize better for unseen examples.

In case our above model didn’t yield good results, we could use transfer learning or maybe one of the famous architectures such as VGG16 alexNet, but since the computational power i have is quite limited, i will first try with a simple architecture such as the one mentioned above.