Machine Learning Nanodegree

Capstone Proposal

Mohit Kumar
February 27, 2018

Using Deep Learning to Count Sharks and Rays

Domain Background

One of the main reasons for me to do the Machine Learning Nanodegree originally, was to build automated systems which could detect threats and warn us about them. Although automated threat detection algorithms have been in the industry since the beginning of the technology but they have really began to pick up stream in the last decade with the enhancement of technology. As we know "With great power comes great responsibility" (Spiderman) so with the growth of technology we should keep on building automated systems which can keep on check of the threats. I have seen several people being losing their lives because of wild animals or water animals' attack. And to check such situations I have seen several teams working on building softwares with could help the human kind. There are many great benefits of having such systems one can be early warning of natural disasters.

In parallel to learning I have been doing research on how to build software which can predict the Dangerous Animals and can alert us about them so that we could take precautions while going near to them.

Why this matters

You may say that a fully autonomous system might not be necessary but if have such systems then it can help us in predicting things more effectively and at a better pace. In this case if we have a solution which could predict Sharks near the coast then we can

warn the people about the sharks and we can even track their movement, which can help us avoiding incidence of shark attacks. As a result their will be less number of injuries and deaths due to shark attack. We can say that such solutions can save life and as a result will be quite good for the humankind.

Problem Statement

Human beings, when fully attentive, do quite well at identifying Sharks. Computers can do better at doing the same using drones. However, humans have a disadvantage of not always being attentive (because of several human factors), while a computer is not. As such, if we can train a computer to get as good as a human at detecting Sharks, since it is already significantly better at paying attention full-time, the computer can take over this job from the human. From the computer's perspective, the problem is around detecting a Shark from height say 60ft.

Datasets and Inputs

The datasets I plan to use will be image frames from video I take from some of the drones. I have got few drone videos of Sharks from my friend at South Australian University. I also have googled some images (https://www.google.co.in/search?q=drone+images+of+sharks+from+drones&sa=X&tbm=isch&tbo=u&source=univ&ved=0ahUKEwjB08rZg9jZAhVCK48KHWaIAwsQsAQIJw&biw=1440&bih=803). In order to cut down on the overall processing needed, I plan to scale down the image sizes, likely down to 25% of the original size. There will definitely be some manual pre-processing involved in order to train my model. As such, I will be calculating the polynomial coefficients for a second-order polynomial for each line (the three outputs per line, times two lines, means each image will have six "labels") in each image. I also plan to use the CV techniques of un-distorting and perspective transform to come up with these labels for training images - the hope is that the model will not need these techniques in the fully trained model for new images. Depending on the size of the neural network used, the dataset will likely end up being quite large - at 30 fps, just 5 minutes of video would be 9,000 images, which would be a lot of labelling. I will try a subset of the data first with my labelling technique to ensure the process seems to work before performing the labelling on the full dataset.

I will try to use a mix of both Sharks and various Sea Mammals as well as various conditions (night vs. day, shadows, rain vs. sunshine) in order to help with the model's overall generalization. I believe these will help to cover more of the real conditions that drivers see every day.

Solution Statement

As I mentioned briefly before, I plan to use deep learning toward the final solution. The reason for this is that a deep learning model may be more effective at determining the important features in a given image than a human being using manual gradient and color thresholds in typical computer vision techniques, as well as other manual programming needed within that process. Specifically, I plan to use a convolutional neural network (CNN), which are very effective at finding Shark images by using filters to find specific pixel groupings that are important. My aim is for the end result to be both more effective at detecting the Sharks as well as faster at doing so than common computer vision techniques.

Benchmark Model

I plan to compare the results of the CNN versus the output of some of the images which I have got from google. I will also visually compare the output onto the project video from the test image which I have got, including attempting to use the CNN on the challenge and harder challenge videos in that project.

Evaluation Metrics

My CNN will be being trained similarly to the Udacity CNN problem, in which it will be given the training images of Sharks, and attempt to learn how to detect Sharks. The loss it will minimize is the difference between those actual Sharks and what it predicts to be Sharks. It will then use the algorithm to predict the Sharks from the images taken from the drones. As noted above regarding benchmarking against my computer vision-based result, I will also evaluate it directly against that model in both accuracy and speed, as well as on even more challenging videos.

Project Design

The first step in my project will be data collection. As discussed above, I plan to use my network in a variety of different countries to gain a good drone videos. Once I have collected enough video, I can then process the video to split out each individual video frame. These frames are the images I will use to train (and validate) my CNN eventually. I plan to scale down the images (starting at 25% of the original size) to reduce processing time, although if I am successful at training a good model with my approach I may attempt a full-scale run.

After that I will follow following steps to generate the desired CNN.

1. Rescale the images by dividing every pixel in every image by 255.

2. Create a CNN to classify Sharks using Transfer Learning.

3. Compile the Model

4. Train the Model

5. Load the Model with best Validation Loss

6. Predict the Sharks with the Model