Machine Learning Engineer Nanodegree

Capstone Proposal

Akash Suresh January 26th, 2019

Proposal

Domain Background

India is one of the world's largest producer of incense and is a healthy exporter to other countries. Burning of incense in India dates back to thousands of years, and India exported the idea to China, Japan and other Asian countries. The primary form of burning incense in India is the incense stick or agarbathi. The basic ingredients of an incense stick are bamboo sticks, paste (generally made of charcoal dust or sawdust and joss/jiggit/gum/tabu powder – an adhesive made from the bark of litsea glutinosa and other trees), and the perfume ingredients.

Incense sticks were traditionally hand-rolled. Of late, through breakthroughs in technology, incense sticks are manufactured using machinery. These machines are

fed with the paste and bamboo sticks as input and mass produce incense sticks. Only downside being, the machine is not always efficient, and might produce subpar incense sticks (around 30-40% of the time). Currently, they use a worker to segregate the good sticks from the bad ones. These workers are paid around INR 500-1000/day. Automating this process could bring down the cost of production by a large margin for the industry.

My dad owns one such industry and this project is to integrate an image processing module for every machine. I propose to build a CNN model which will classify the good sticks from the bad by using the expertise gained in this course in order to help my dad cut down costs. I will be helping him reduce his workforce by 1 worker/machine. (We have around 50 machines). Thus saving him about INR 1.5 million per year.

Problem Statement

The incense stick manufacturing machine, takes the sticks and paste as input, and rolls out the formed incense sticks on a conveyor belt. A Raspberry Pi will be integrated with the machine to automate the segregation process. The problem at hand is to design a CNN that classifies camera procured images of the formed incense sticks on the conveyor belt as good or bad with atleast 85% accuracy and at the same time is simple enough to run on a Raspberry Pi.

Datasets and Inputs

The dataset for this project will be generated manually, by running a script to shoot (around 5000) photos of the sticks on the conveyor belt and manually classifying them as good/bad. This dataset will be used as training images for my CNN.

Each image will be a square image, shot on the RaspberryPi camera. The manufactured incense stick will be captured in the frame.

We will have to then use this CNN to classify the incense sticks realtime, where the input to the network will be photos shot in a similar frame.

Running a python script to shoot photos at specific intervals, I collected around 6000 photos. After cleaning up (discarding the empty frames), and manually tagging them as Good or Bad. Here is how the distribution looks like

Type	Number of images	Percentage
Good	3264	71.9%
Bad	1236	28.1%

Bad sticks comprise of 28.1% of the dataset. This might be a skewed dataset, and we will have to accommodate this skewness in the evaluation metric, depending on how the model performs.

Solution Statement

Using the dataset, I will be trying out different Convolutional Neural Network architectures, to find the optimal architecture that gives me the best accuracy. The key aspect of the project is that the model will be running on a Raspberry Pi and it should function realtime (implying that the prediction should not take more than 0.5 seconds). To ensure this, I will have to design the CNN to be very lean by not using too many layers, but also not compromising on the accuracy. This will be quite challenging.

Despite this being a straightforward problem to solve, in hindsight, a lot of complications could arise while running the network alongside the machine. These machines are very fast and produce incense sticks at a rate of ----- and timing the model to work with the machine is one such complication. Also deploying it on the RaspberryPi with all dependencies resolved could be another issue I may face.

The metric to be measured will be the accuracy of the model and how well it can be integrated into the machine. The metric I will be evaluating this project on will be the cost saved by my dad's industry.

Benchmark Model

The benchmark model could be any one of the VGG_19 models, or the Xception Model. But these might be too big to be deployed on the Raspberry Pi. My model will try to reach the performance of these models while trying to be light weight.

Evaluation Metrics

The evaluation metric will be a comparison between the accuracy of the model, and also the size of the model. Maybe a combination of both will be the best metric to attain optimal performance since the performance of the model in this kind of setting depends on accuracy directly and size inversely. This also gives a way to normalize the results to compare it with the benchmark.
Also, considering the skewness of the dataset, we might have to use F1-Score instead of accuracy. We can decide this once the dataset is generated.

metric_used = [accuracy or f1-score]

evaluation_metric = metric_used/size(model)

At this point in time, I am keeping both accuracy and f1-score as open options. This is so that I have some flexibility in building my model.

Project Design

Part 1 - Setting up the RaspberryPi

For this project, we will be integrating a RaspberryPi directly with the machine. The RaspberryPi will have a camera module which will be used to take photos to generate dataset, as well as predict the quality of the sticks in realtime. We will need to setup Keras, TensorFlow, OpenCV, and the supporting packages on the RaspberryPi in the best way to optimize performance.

Part 2 - Generating Dataset

Using the RaspberryPi, we will be running a script to snap photos of the sticks on the conveyor belt at regular intervals. Around 5000 images will be shot initially. And these will then need to be classified manually.

Part 3 - Building a CNN

These images, will be used to build different CNN models, to evaluate and find the most optimal model as per our evaluation_metric (Defined above.)

Part 4 - Realtime evaluation

Models generated in Part 3 will be run on the RaspberryPi, and will be used to classify and segregate the sticks realtime. If the model is not satisfactory, we will need to look at having a bigger dataset, or using a different CNN architecture.