Machine Learning Engineer Nanodegree

Capstone Proposal

Raghu Dharmavaram

March 14th, 2018

Proposal

Domain Background

According to the CDC motor vehicle safety division, one in five car accidents is caused by a distracted driver. Sadly, this translates to 425,000 people injured and 3,000 people killed by distracted driving every year.

State Farm hopes to improve these alarming statistics, and better insure their customers, by testing whether dashboard cameras can automatically detect drivers engaging in distracted behaviors.

Problem Statement

Given a dataset of 2D dashboard camera images, an algorithm needs to be developed to classify each driver's behavior and determine if they are driving attentively, wearing their seatbelt, or taking a selfie with their friends in the backseat etc..? This can then be used to automatically detect drivers engaging in distracted behaviors from dashboard cameras.

Datasets and Inputs

Driver images, each taken in a car with a driver doing something in the car (texting, eating, talking on the phone, makeup, reaching behind, etc) were provided.

The 10 classes to predict are:

- c0: safe driving
- c1: texting - right
- c2: talking on the phone - right
- c3: texting - left
- c4: talking on the phone - left

- c5: operating the radio
- c6: drinking
- c7: reaching behind
- c8: hair and makeup
- c9: talking to passenger

Following are the file descriptions and URL's from which the data can be obtained:

- imgs.zip - zipped folder of all (train/test) images
- sample_submission.csv - a sample submission file in the correct format
- driver_imgs_list.csv - a list of training images, their subject (driver) id, and class id

https://www.kaggle.com/c/state-farm-distracted-driver-detection/download/driver_imgs_list.csv.zip
https://www.kaggle.com/c/state-farm-distracted-driver-detection/download/imgs.zip
https://www.kaggle.com/c/state-farm-distracted-driver-detection/download/sample_submission.csv.zip

We have total of 102150 colored images with 640 X 480 pixels each. Out of these 17939 are training images, 4485 are validation images and 79726 are test images. Training, validation images belong to the categories shown above.

Solution Statement

A deep learning algorithm will be developed using Tensorflow/Keras and will be trained with training data. Specifically a CNN will be implemented in Tensorflow/Keras and will be optimized to minimize multi-class logarithmic loss as defined in the Evaluation Metrics section. Predictions will be made on the test data set and will be evaluated.

Benchmark Model

The model with the Public Leaderboard score (multi-class logarithmic loss) of 0.08690 will be used as a benchmark model. Attempt will be made so the score (multi-class logarithmic loss) obtained will be among the top 50% of the Public Leaderboard submissions.

Evaluation Metrics

Submissions are evaluated using the multi-class logarithmic loss. Each image has been labeled with one true class. For each image, you must submit a set of predicted probabilities (one for every image). The formula is then,

$$\text{Logloss} = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{M} y_{ij}\log(p_{ij}),$$

where N is the number of images in the test set, M is the number of image class labels, *log* is the natural logarithm, *ij* is 1 if observation *i* belongs to class *j* and 0 otherwise, and *ij is the predicted probability that* observation *i* belongs to class *j.*

The submitted probabilities for a given image are not required to sum to one because they are rescaled prior to being scored (each row is divided by the row sum). In order to avoid the extremes of the log function, predicted probabilities are replaced with $max(min(p,1-10^{-15}),10^{-15})$

Project Design

From the description and problem statement it can be inferred that computer vision can be used to arrive at a solution. CNN class of deep learning algorithm can be employed for this problem.

Initially data exploration will be carried out to understand possible labels, range of values for the image data and order of labels. This will help preprocess the data and can end up with better predictions.

After this necessary preprocess functions will be implemented,

1. Data will be randomized, images are divided into training and validation sets.
2. Images are resized to square images i.e. 224 X 224 pixels.

3. All three channels were used during training process for the colored images.
4. The images are normalized by diving every pixel in every image by 255 a value of 0.5 is subtracted to ensure mean of zero and CNN will be implemented in Tensorflow/Keras.

Finally necessary predictions on the test data will be carried out and these will be evaluated.