Udacity's Machine Learning Engineer Nanodegreee Program
State Farm Distracted Driver Detection
Capstone Proposal

Rodrigo S. Veiga

April 29, 2018

This proposal is based on the Kaggle's homonymous challenge.

1 Domain Background

We have all been there: a light turns green and the car in front of you does not budge. Or, a previously unremarkable vehicle suddenly slows and starts swerving from side-to-side. When you pass the offending driver, what do you expect to see? You certainly are not surprised when you spot a driver who is texting, seemingly enraptured by social media, or in a lively hand-held conversation on their phone. 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.

2 Problem Statement

Given a dataset of 2D dashboard camera images, the idea is develop an machine learning algorithm to automatically classify driver's behavior. Are they driving attentively, wearing their seatbelt, or taking a self with their friends in the backseat? Such an algorithm could be highly important to prevent accidents if it could automatically warn distracted drivers based on their behavior captured by on-board cameras.

3 Datasets and Inputs

The dataset is composed by 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).

There ten 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

The data files can be found here.

File descriptions

- `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`.

4 Solution Statement

Following the previous Dog Breed Prediction Project, a multi-layer convolutional neural network (CNN) will be implemented with Keras. The optimization will be made by the minimization of the multi-class logarithmic loss described below.

5 Benchmark Model

The leading model of the Kaggle's Public Learderboard will be used as a benchmark model. It achieved a multi-class logarithm loss of 0.08689.

6 Evaluation Metrics

Following the challenge by Kaggle the models will be evaluated using the multi-class logarithmic loss. Each image has been labeled with one true class. For each image, a set of predicted probabilities (one for every image) will be presented. The formula is then,

$$\mathcal{L}_{\log} = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{m} y_{ij}\log(p_{ij}) \ ,$$

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

The presented 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 logarithm function, predicted probabilities are replaced with $\max(\min(p, 1-10^{-15}), 10^{-15})$.

7 Project Design

Since this problem is similar to the one presented in the Dog Breed Prediction Project it is reasonable to treat it with a CNN-based deep learning algorithm.

Initially, data exploration and data preprocessing will be carried out before the CNN implementation with Keras. After training the proposed architectures will be evaluated on test data using the evaluation metrics mentioned above.