State Farm Distraction Driver Detection Project

Project Report

Prepared by: Manish Kumar
Enrolment No. 18119018
Date: 1st July 2020 to 30th July 2020

Introduction

We've all been there: a light turns green and the car in front of you doesn't 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 aren't 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 ensure their customers, by testing whether dashboard cameras can automatically detect drivers engaging in distracted behaviours. Given a dataset of 2D dashboard camera images, State Farm is challenged to classify each driver's behaviour. Are they driving attentively, wearing their seatbelt, or taking a selfie with their friends in the backseat?

Aim

Given 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). Your goal is to predict the likelihood of what the driver is doing in each picture.

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

Method

Used the dataset which includes images of drivers while performing a number of tasks including drinking, texting etc. The aim is to correctly identify if the driver is distracted from driving. We might also like to check what activity the person is performing.

The notebook will be broken into the following steps:

1. Import the Libraries.
2. Import the Datasets.
3. Create a vanilla CNN model.
4. Create a vanilla CNN model with data augmentation.
5. Train a CNN with Transfer Learning (VGG16)
6. Results

Import the Libraries

Used Keras and Tensorflow libraries to create a **Convolutional Neural Network**. So, imported the necessary libraries to do the same such as import time import tensorflow import Pandas etc.

Import the Datasets

Imported the .csv file to read the labels.

Create a vanilla CNN model

Building the model

Developed the model with a total of 4 Convolutional layers, then a Flatten layer and then 2 Dense layers. Used the optimizer as rmsprop, and loss as categorical_crossentropy.

Create a vanilla CNN model with data augmentation

Here augmented the previous model classifier, used the data on which I want to train the model. The folder train includes the images I need. I generated more images using **ImageDataGenerator** and split the training data into 80% train and 20% validation split.

Train a CNN with Transfer Learning (VGG, MobileNet)

To reduce training time without sacrificing accuracy, I'll train a CNN using **transfer learning**.

Result

Got a good result with accuracy of about 79%.