Integrated Engineering Capstone Distracted Driving

Matthew Clark, Joerg Oxborough, Eric Cumiskey, Mark Rabinovich, Pavlo Grinchenko

ES4499 – University of Western Ontario
Advisors: Dr. Darren Meister and Dr. Jeff Wood
Demo Video: youtu.be/4tze6cgGxw8
Source Code: github.com/Joerg-ffs/Capstone-Project

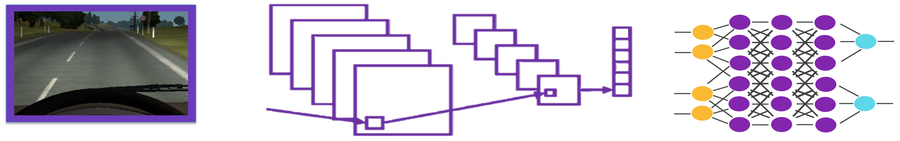
The Problem

Distracted driving is a factor in 80% of collisions in Canada.
There are 4 million driving related crashes in North America per year.
It is illegal in most provinces and states however is hard to enforce.

Our Solution

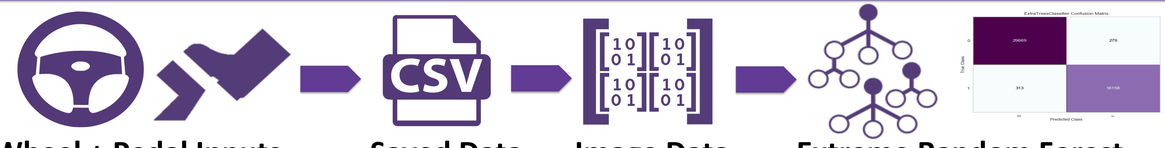
Using machine learning techniques in conjunction with a driving simulator to generate data, a model can be trained to detect distraction. Then the model can alert the driver, allowing them to avoid dangerous situations. This tool can also help improve the user's driving by plotting their distraction levels.

Tier One



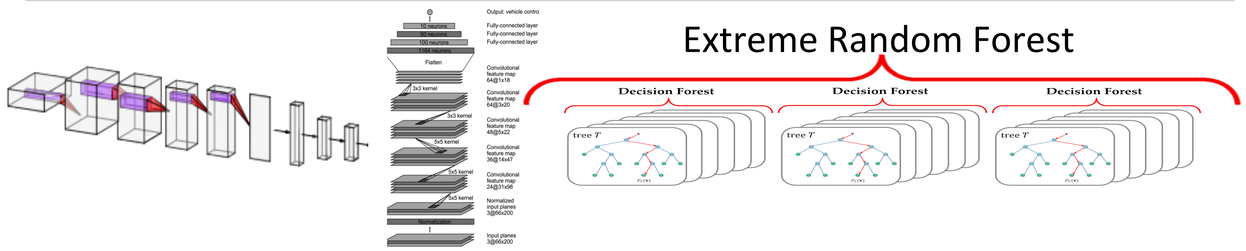
The first tier of the system functions exactly like an autonomous vehicle except the output is not used for driving but will be fed into another model. To achieve this result, the group utilized a Convoluted Neural Network (CNN). The architecture for this CNN was published and open-sourced by NVIDIA to speed up the development of autonomous vehicles.

Tier Two



The second model of the prototype system is an Extremely Random Trees classifier. The input data from the racing wheel is fed into the model as well as the predicted angle from Tier 1 of the system. This model was chosen as it had the best performance in the initial prototype phase as well as for this dataset. The output of this model is a confidence value between 0 and 1 of how distracted the driver is.

Model Technical Details

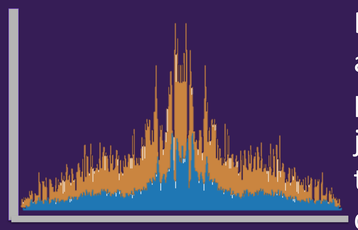


Data Acquisition

Data was collected using the Logitech g920 racing wheel and pedals, and a truck simulator run on Linux Mint. Image data was gathered using Python 3.6 and the libraries evdev for reading events and mss for screen capture.

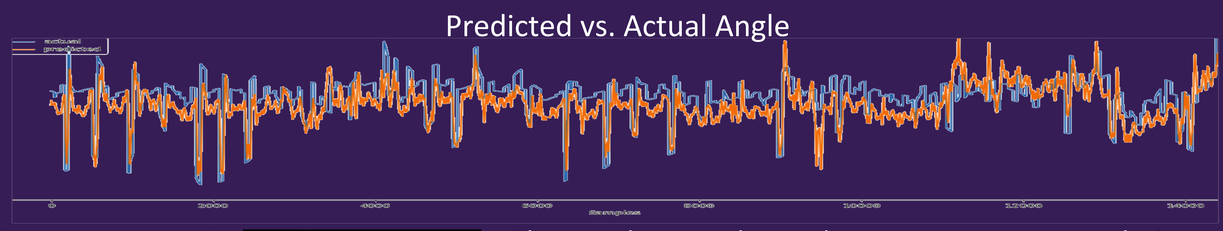
Preprocessing

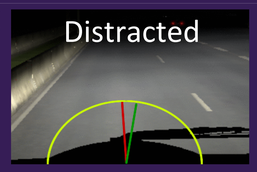
In order to achieve the best training results, the input image is manipulated. The first step is transforming the image into the YUV colour plane, then the image is cropped and resized to both improve results as well as reduce computational loads, finally the image is converted into a numpy array and scaled so that all values are between 0 and 1.



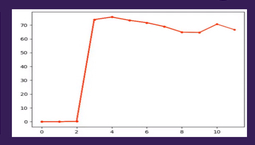
Due to the stability of normal driving, the steering wheel angles are mostly very close to center which is numerically 0. This is problematic when training the model as the CNN could learn to just predict 0 every time and get good accuracy. The distribution of the data needs to be augmented so it is closer to a uniform distribution. Orange represents rescaled data, blue is scaled data.

Results





Actual angle Predicted angle



The graph seen above showcases strong correlation between the actual and predicted values that are then fed into the second model. The two images on the left showcase that when there is a large difference between the predicted and actual values there is also distraction. The bottom leftmost graph details the coefficient of determination versus the number of epochs, while the graph on the right is the user's distraction level plotted in respect to time.

Tools Used