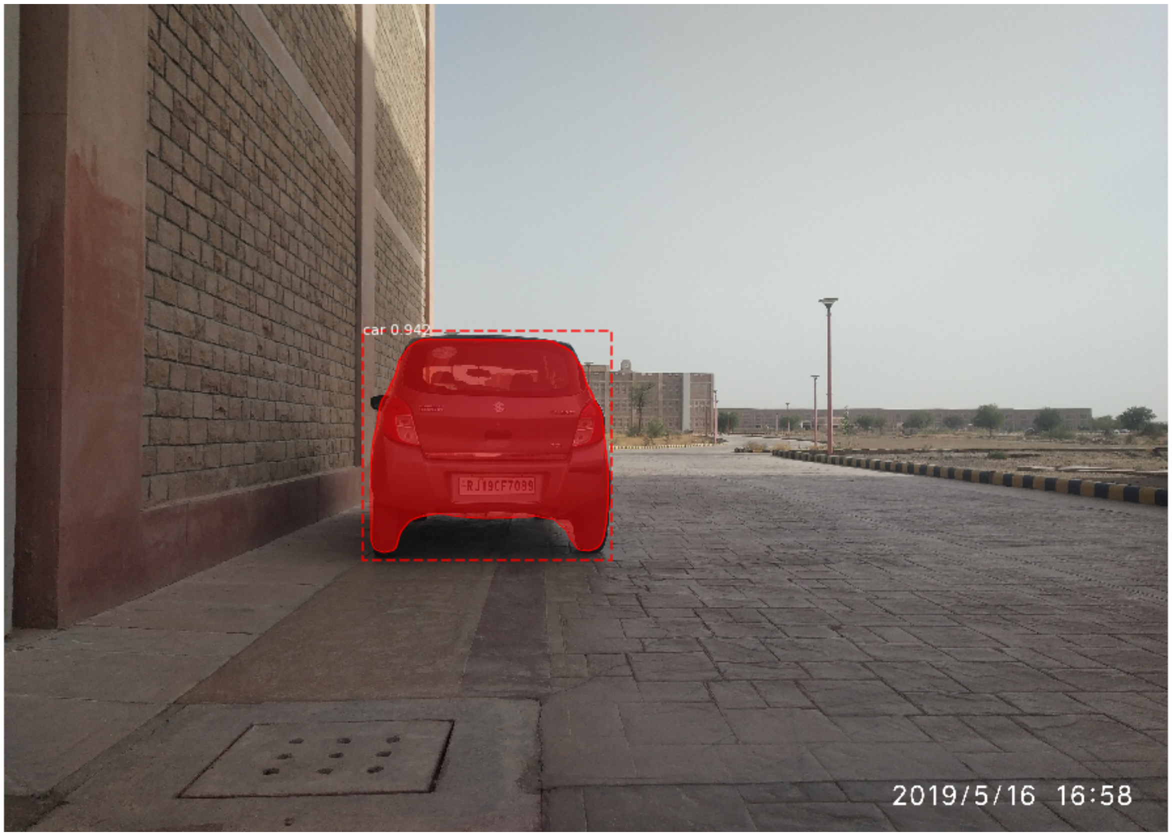

ASStech

Assisted Driving Car

Harsh Anand
Anukul Kumar
Pradumn Kumar
Yashvi Ramanuj
Divya Singh

Overview

The aim of the project was to create an Assisted Driving Car. NVIDIA's "End to End Deep Learning" research paper was implemented to predict the path to be taken by the vehicle. The presence of the driver's eyes on the road determines whether the vehicle would be controlled by the algorithm or not. This, in turn, provides assistance to the driver (at moments when his/her gaze is not on the road). The algorithm involves detecting obstacles and accordingly, controlling the vehicle's movements; hence, fulfilling the motive of preventing road accidents that can occur due to a driver's negligence.

This Github Repository contains:

- all the libraries required to be uploaded on Google Colaboratory in a folder named Files ,
- a few of the videos which the Neural Network was trained on, along with a JSON file containing the data marked with timestamp and a few demo images that can used for detecting eyes in the eye tracking program in a folder named Media ,
- the weights trained with the help of NVIDIA's End to End Deep learning paper, along with the Haar Cascade weights that were used to detect face, in a folder named Weights
- Python scripts to detect eyes, train the Neural Network, predict the output from an image using the self-trained Neural Network, detect and display the masks of the obstacles trained on the MS COCO dataset.

Eye Tracking

The eyes of the driver were detected by initially tracking the face (Using Haar Cascade Dataset) and then using a self-developed method to track the pupils of the person. The position of the pupils is then used to determine whether the person's gaze is fixed on the road or not.

Detecting Faces and Eyes

A highly efficient means of detecting the face using Haar Cascade Weights was used. The eyes were detected in a similar fashion.

Detecting Pupils and Gaze

From the detected eyes, pupils were detected and then the approximate position of the center was used to determine the gaze of the driver.

Application of End to End Deep learning

I. Structure

The structure involves a Sequential model consisting of 5 convolutional layers and a dense Neural Network comprising 4 layers.

II. Training and Testing

The Neural Network was then trained and tested on the Berkeley Deep Drive dataset.

III. Prediction

Prediction pertaining to steering the vehicle was made on the above trained Neural Network.

Rendezvous with Obstacle Detection

Application of Mask-RCNN

Detection of obstacles is done using Mask-RCNN which uses semantic segmentation technique to detect objects by creating bounding boxes and segmentation masks around them. Mask-RCNN is implemented on Python, Keras and Tensorflow and trained on the MS COCO dataset.

References

- Open CV : Haar Cascade Dataset
- NVIDIA's End to End Deep Learning paper
- Mask R-CNN for object detection and instance segmentation on Keras and TensorFlow
- Stack Overflow