CMPE258  
Team Project on Deep Learning

PROJECT SUMMARY

TITLE : **DISTRACTED DRIVER PREDICTION**

**Team Members:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **SJSU ID** | **Email** | **Affliation** |
| Pravin Ramasamy Balachandran | 015443272 | Pravin.ramasamybalachandran@sjsu.edu | Software Engineering |
| Kishore Kumaar Natarajan | 016013270 | Kishorekumaar.natarajan@sjsu.edu | Artificial Intelligence |
| Varun Reddy Seelam | 016015948 | Varunreddy.seelam@sjsu.edu | Software Engineering |

**Team Coordinator**: Kishore Kumaar Natarajan

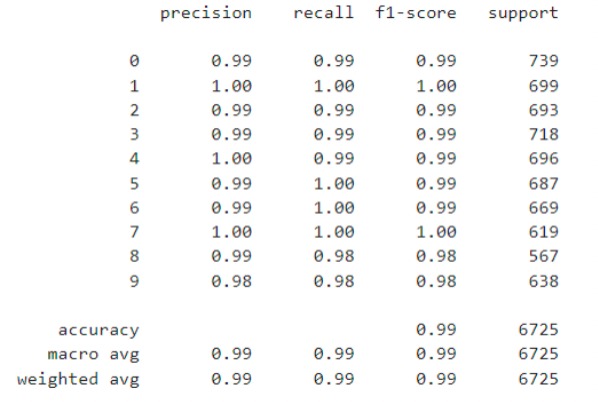
**ABSTRACT**

The primary objective of our project is to detect a driver seated in a vehicle from a media input and identify if the driver is driving safe or if he is inattentive. We believe that computer vision-based solutions can be effectively used to detect inattentive drivers using any sort of dashboard cameras and alert drivers. This report provides various Machine Learning solutions we experimented and compared it that can classify the driver action and predict if driver is distracted which can be further integrated with computer vision solution pipeline.

We made use of Convolutional Neural Networks to achieve this. One of the major technical challenges we faced was to do the literature survey of each of the model we trained as we were faced issues while creating the data model from scratch due to the lack of expertise. Another problem we encountered was the overfitting of models where the models showed better accuracies on the training dataset but not showing better performances on the testing segments. There were also few challenges while designing the model architecture as there were few obstacles in the form of shape errors, module errors. There was also a constraints on computing resources and due to which it took significant amount of time to train the models. We also faced issues training deep layer models on Google colab notebook, hence we decided run on JupyterLab on HPC. On HPC we faced issues downloading imagenet weights for pre-trained Tensorflow models. Once we fixed, it we were able to run the models on HPC, however faced timeout issues for every 1hour or so. We tried few things that were not successful in extending the session but however managed to get the results.

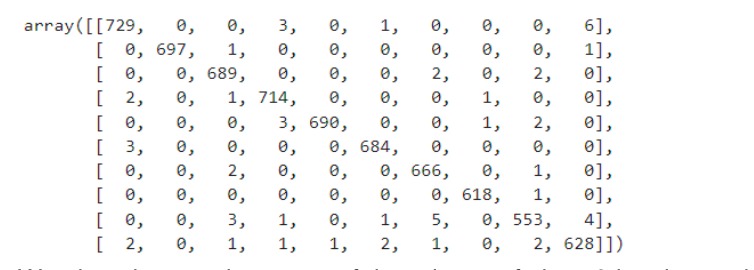
We developed Resnet 50, MobileNet v2, Deep CNN Self designed architecture and VGG 16 models in this project for spotting the best predicting model. The MobileNet model is a network model in which the basic unit is depth wise separable convolution. There are two sorts of blocks in MobileNetV2. One is a one-stride residual block. Another option for shrinking is a block with a 2 stride. Both sorts of blocks have three levels. The first layer is 11 convolutions with ReLU6 this time. The depth wise convolution is the second layer. VGG-16 is a convolutional neural network that is 16 layers deep. All the hidden layers in the VGG network use ReLU. VGG does not usually leverage Local Response Normalization (LRN) as it increases memory consumption and training time. For our dataset, we have compressed the images to 150 x 150 px for our VGG model. Residual Neural Network (ResNet), is a deep layered architecture which range from 18 layers - ResNet-18 to 1202 layers deep - ResNet-1202. We used TensorFlow’s pre-trained Resnet50 with ImageNet weights.

We used few software tools and platforms to develop the projects. We used PyCharm IDE, Jupyter Notebook and Anaconda navigator platform for creating and testing the models. We also used Flask for backend development and HTML, CSS for the frontend GUI. We did use TensorFlow, Keras and opencv for training the models and verifying it.  
  
We achieved the highest accuracy of 98% on test data for the VGG-16 model and close to 99% for the Resnet 50 model. We also calculated the precision, recall, f1 metrics to analyze the performances of the models. Below if the screenshot of the Resnet 50 model



From classification report, we see that classification class 8 and 9 are not classified properly compared to other classes and have lower f1-score.

We also calculated the confusion matrix of Resnet 50 model to evaluate the performance of the classification models when they make predictions on test data.



We see that most of the misses of class 9 has been classified as class 0 and class 9.  Class 2 seem to have perfect score.

We had a great learning curve while developing this project using state of the art CNN vision architectures such as Resnet 50, MobileNet v2, Deep CNN Self designed architecture and VGG 16 to extract features and to transfer the functionality to our problem. We are more confident and better versed to deal with Convolutional Neural Networks and also we came across a lots of good references in the internet which we can make use of, when we get into the professional world.

**Contributions of each member**

|  |  |  |
| --- | --- | --- |
| **Name** | **Responsibility of the work** | **Contribution** |
| Pravin Ramasamy Balachandran | Data Pre-processing and cleaning, model training and testing for DEEP CNN self-designed architecture and VGG 16 | Coding - Trained and tested VGG 16 and Deep CNN self-designed architecture model, Executive summary, PPT |
| Kishore Kumaar Natarajan | Front end using HTML, CSS and model training for transfer learning model mobile net v2 | Coding - Model training and testing for transfer learning model MobileNet v2, front-end development using HTML, CSS, Readme File |
| Varun Reddy Seelam | Backend using flask, model training for transfer learning model | Coding - Model training and testing for transfer learning model  Resnet 50, backend development using Flask, coordinator |