SERPENTES

CE 784A Project Report: Semester 2020-21 (I)

Naturalistic Driving Action Recognition

Priyanshu Kumar(190651), Raj Karan Kumar(190674), Yatish Goel(191180), Raj Aryan(190673)

Keywords: Naturalistic driving studies, preattentive vision, Deep CNN, GMM-ResNet

1. Introduction

The key factors that influence driving safety are driver **decisions and behaviour**. According to reports, human driver error is responsible for more than 90 % of light vehicle accidents in the United States, and with a precise driver behaviour monitoring system, the accident rate can be lowered by 10% to 20%. According to China's road traffic statistics(2010), drivers are the most critical component accounting for around 95 percent of all car accidents. As a result, it is vital to recognize anomalous driving to improve traffic safety effectively. Intelligent vehicles can interact with human drivers more effectively using end-to-end methodologies, making smarter decisions and producing human-like driving tactics. **Naturalistic driving studies** are an important method for investigating real-time driver behaviour. Decisions and behaviours of drivers are critical elements that can influence driving safety. **The goal is to categorise the driver's distracted behaviour activities over a particular time frame.** Many tools had been developed to identify the actions of drivers while driving and categorise based on them between 2 broad classes: **distracting** and **non-distracting**. They record all of the driver's actions in the traffic environment, including those concerning tiredness or distracted driving. The common activities observed while driving are **normal driving, eating, drinking, smoking, hand holding a mobile phone, texting, dialling, driving with earbuds or headsets, interaction with passengers, left mirror checking, right mirror checking** and **rear mirror checking**. The recognition models are trained to recognize such frequent driving-related actions as well as whether or not the driver is distracted. Intelligent vehicles that use end-to-end techniques can better interact with human drivers, making better decisions and producing human-like driving tactics.

2. LITERATURE REVIEW

Several studies have been conducted to examine the harmful and non-destructive behaviours of drivers. Using solely **preattentive vision**, N.pugeault and R. Bowden in their study, presented a revolutionary vision-based method to autonomous driving that can predict and even anticipate a driver's action in real-time.
Driver's intention also plays a significant role in safety deciding factors while driving. In view of this, V.A. Butakov and P. Ioannou developed a methodology for learning the features of an individual driver's behaviour vehicle before, during, and after lane changes, as well as in various driving circumstances. These features are collected by a series of models, the parameters of which are modified in real-time to match each vehicle/driver reaction during lane changes. To characterise the **manoeuvre kinematics**, they proposed a two-layer model. As a kinematic model, the lowest layer depicts lane change. The upper layer model determines the kinematic model parameter values for each driver and their dependency on the configuration of the surrounding cars. Detecting driver's behaviour based on physical gestures such as **head pose, eye gaze direction and hand movement** also has been studied by Eshed Ohn-Bar and their team. In their work they proposed a methodology based on **multiview and multichannel vision** for identifying behaviour of drivers based on their hand motion, their head and eye inputs. These are some works which have been done in the past based on vision based feature extraction methods, along with this for real-time driver status monitoring, physiological

factors such as this should also be taken into consideration to see how the driver's brain is functioning for the detection of anomalous behaviour during driving. I.H. Kim, J.-W. Kim, S. Haufe and S.-W. Lee in their study created a **virtual driving environment** to investigate the brain correlates of emergency braking in a variety of scenarios. To predict a driver's braking choice before they show any behavioural reaction these researchers derived some conclusion by looking at how their brain correlates is being utilised. They also preserved data extracted by electroencephalographic (EEG) and electromyographic.

However, as previously stated, the majority of existing driver behaviour studies need the extraction of particular variables in advance, such as head posture angle, gaze direction, EEG, and hand and body joint position. These capabilities aren't always simple to come by, and some may need the purchase of specialised gear, which can add time and money to the process. Nowadays deep learning methodology is being used for various application with the use of large dataset in many fields. Because of this **deep CNN**, we have achieved a much better results than before in object identification, classification, generation, and segmentation tasks. Therefore detecting driver's normal behaviour deep learning techniques have been used. To create a exclusive driver behaviour situational awareness system T. Hoang and his mates developed an algorithm called **Grammer aware parsing algorithm** with the help of some deep features. By this algorithm they trained a deep model in the initial stage which extract highly abnormal or uncommon features of the driver.Following that, a grammatical structure on the deep features is established, which was employed as background experience for the production of semi-supervised proposal choices.To accurately distinct regions of the driver, the Region based Convolutional Neural Networks approach had been applied.

Driver Activity Recognition for Intelligent Vehicles: A Deep Learning Approach is one of the research by Yang Xing, Chen Lv, and others. The choices and behaviours of drivers was seen to have a direct impact on driving safety. Deep CNN was used to accomplish this assignment. Normal driving, right mirror checking, rear mirror checking, and left mirror checking are all actions that are deemed normal in this task. Images were taken with a low-cost camera, and the data was collected by ten drivers. The driver body was extracted from the background using **GMM**(Gaussian Mixture Model) **image segmentation**. Pre-trained models such as **AlexNet, ResNet50** and **GoogLeNet** were used to save training costs. To identify driver actions, the driver's head attitude, eye gaze direction, and hand movement were integrated. Ten drivers were asked to do seven tasks: left mirror checking, regular driving, right mirror checking, and rear mirror checking, as well as three activities using in-vehicle radio/video equipment, answering the phone, and texting. A total of 34 thousand photos were taken. For this task, two approaches were used: Image Preprocessing and Segmentation and Model Preparation and Transfer Learning. To speed up the CNN training process, the original image was cropped during image pre-processing and segmentation. After cropping, these photographs were converted into the size of pre-trained CNN models' input needs. The driver body region was then extracted from the backdrop using the GMM algorithm. AlexNet, GoogLeNet, ResNet50, and transfer learning were utilised to prepare the models. AlexNet had an accuracy rate of 81.4 percent on average. The raw-image-based recognition accuracy was only 69.2 percent. With raw images, GoogLeNet scored a 74.7 percent accuracy. The **GMM-ResNet** had a general classification accuracy of 74.9 percent, while the Raw image-based ResNet has a classification accuracy of 61.4 percent. To tackle the binary classification challenge, the CNN models were **fine-tuned**, for the **G-AlexNet-based model**, it attained a 91.4 percent accuracy.

3. METHODOLOGY

3.1. Data

3.1.1. Data Exploration

Our dataset consists of 10 user files and for each user, we have been provided with 6 videos in which the user is performing a different class of activities It also has information about whether the appearance is blocked or not for example if the user is wearing a hat or sunglasses then we consider it as a blocked appearance.can be categorized into distracting or non-distracting. For every user, we have been provided with 6 videos out of which in 3 videos appearance was blocked, and in the rest 3, there is no blockage of appearance. These 6 video files were recorded with three cameras. The location of these cameras was near the dashboard, the right top corner of the right side window of the driver, and the last one was located near the rear view mirror inside the vehicle. Along with these, we had a CSV file for the 5 users which contains information such as user id, and video file name, this file also contains information about the various activities that were performed by different users. There are 18 types of activities that

the user performed. Some of them are activities like drinking, eating, yawing, singing with music, etc which was labeled from 0 to 17 along with this it contains the start and end time of different distracting activities performed by the user. It also has information about whether the appearance is blocked or not for example if the user is wearing a hat or sunglasses then we consider it as a blocked appearance. This CSV file which comprises all this information is being used as the training and validation data and apart from it we were also equipped with another folder similar as the first one which contains the 5 user id having 6 videos for each but this didn't contain any CSV file. The data obtained from this folder after our analysis was used as test data for our trained model.

3.1.2. Data Pre-processing

Firstly we have extracted excel files for each user followed by making a total of 18 directories corresponding to each activity, then we started taking out frames at an interval of 4 seconds from each user and placed them in the directory corresponding to the activity the user indulged in the jpg format. We used pixellib library for this purpose along with pycocotools. Once we went through placing all the pictures we then moved to the segmentation stage. In this stage, we segmented the image and extracted the person from each picture and placed it in the sub folder created inside each activity folder.

3.2. Models

3.2.1. Train-test split

After extracting one frame per 4 second, we got total of 2346 files in which we used 20 % images i.e around 469 images for validation set and 80 % images for training.

3.2.2. CNN Model

We used basic CNN model which consisted of only one convolutional block with 64 filters, each of size 3x3 after batchNormalizing first layer. We employed ReLU activation after this we did max pooling on the obtained convolutional block and employed dropout technique and did batch normalization to get faster convergence. Lastly added a fully connected layer and trained the model using adam optimizer. Detailed model architecture can be seen in image shown above.

```
Model: "sequential_4"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 batch_normalization_18 (Bat  (None, 256, 256, 3)      12
 chNormalization)

 conv2d_6 (Conv2D)           (None, 254, 254, 64)      1792

 max_pooling2d_6 (MaxPooling  (None, 127, 127, 64)     0
 2D)

 dropout_6 (Dropout)         (None, 127, 127, 64)      0

 batch_normalization_19 (Bat  (None, 127, 127, 64)     256
 chNormalization)

 flatten_6 (Flatten)         (None, 1032256)           0

 batch_normalization_20 (Bat  (None, 1032256)          4129024
 chNormalization)

 dense_4 (Dense)             (None, 18)                18580626

=================================================================
Total params: 22,711,710
Trainable params: 20,647,064
Non-trainable params: 2,064,646
_________________________________________________________________
```

Figure 1: Model Architecture

This model gave us the accuracy of 96.27 % on the validation set and 95.60% on the training set.

```
177/177 [==============================] - 27s 153ms/step - loss: 11.2026 - accuracy: 0.9208 - val_loss: 15.4222 - val_accuracy: 0.9236
Epoch 22/35
177/177 [==============================] - 27s 152ms/step - loss: 8.0943 - accuracy: 0.9375 - val_loss: 17.9312 - val_accuracy: 0.9218
Epoch 23/35
177/177 [==============================] - 27s 154ms/step - loss: 10.0783 - accuracy: 0.9318 - val_loss: 7.9054 - val_accuracy: 0.9414
Epoch 24/35
177/177 [==============================] - 27s 154ms/step - loss: 9.5374 - accuracy: 0.9262 - val_loss: 27.4042 - val_accuracy: 0.8792
Epoch 25/35
177/177 [==============================] - 27s 153ms/step - loss: 11.4750 - accuracy: 0.9208 - val_loss: 21.0522 - val_accuracy: 0.8970
Epoch 26/35
177/177 [==============================] - 27s 153ms/step - loss: 9.5074 - accuracy: 0.9340 - val_loss: 20.0318 - val_accuracy: 0.9112
Epoch 27/35
177/177 [==============================] - 27s 154ms/step - loss: 8.0059 - accuracy: 0.9425 - val_loss: 13.2524 - val_accuracy: 0.9343
Epoch 28/35
177/177 [==============================] - 27s 153ms/step - loss: 5.9313 - accuracy: 0.9581 - val_loss: 10.8644 - val_accuracy: 0.9520
Epoch 29/35
177/177 [==============================] - 27s 153ms/step - loss: 5.0884 - accuracy: 0.9570 - val_loss: 14.2570 - val_accuracy: 0.9290
Epoch 30/35
177/177 [==============================] - 27s 153ms/step - loss: 6.8368 - accuracy: 0.9524 - val_loss: 14.9794 - val_accuracy: 0.9325
Epoch 31/35
177/177 [==============================] - 27s 153ms/step - loss: 5.5929 - accuracy: 0.9638 - val_loss: 5.8971 - val_accuracy: 0.9574
Epoch 32/35
177/177 [==============================] - 27s 153ms/step - loss: 5.6198 - accuracy: 0.9570 - val_loss: 14.6407 - val_accuracy: 0.9254
Epoch 33/35
177/177 [==============================] - 27s 154ms/step - loss: 7.3983 - accuracy: 0.9468 - val_loss: 11.0978 - val_accuracy: 0.9361
Epoch 34/35
177/177 [==============================] - 27s 153ms/step - loss: 6.4940 - accuracy: 0.9560 - val_loss: 5.7573 - val_accuracy: 0.9627
```

Figure 2: Result on Basic CNN

3.2.3. Resnet-50 v2 Model

We imported the Resnet-50 v2 model directly from the tensorflow and excluded the top layers. We faced a problem while using this model and that is we had images in our dataset which was in the grayscale format but resnet 50 can only be trained for RGB types of images. To resolve this problem we looked for two alternate solutions. First of which is expanding the grayscale images into 3 dimensions and then training the model. Another solution was that we add one layer at the start of resent 50 model corresponding to our input. But what we observed that the second alternate solution didn't provide better accuracy and had cumbersome approach. So we trained our resnet model using the first alternate proposed solution. We added one dense layer after flattening the final convolutional layer. The model architecture can be seen below:

```
Model: "model_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_5 (InputLayer)         [(None, 256, 256, 3)]     0

tf.math.truediv_1 (TFOpLamb  (None, 256, 256, 3)       0
da)

tf.math.subtract_1 (TFOpLam  (None, 256, 256, 3)       0
bda)

resnet50v2 (Functional)      (None, 8, 8, 2048)        23564800

flatten (Flatten)            (None, 131072)            0

dense_2 (Dense)              (None, 18)                2359314

=================================================================
Total params: 25,924,114
Trainable params: 2,359,314
Non-trainable params: 23,564,800
```

Figure 3: Model Architecture

Finally we compiled the model and trained it where we got an accuracy of 99.47 % on validation dataset and 98.69% on training dataset.

3.3. Future Work

Apart from these two models we are also looking forward to employing a model which directly works on the video format which has been provided to us. Main objective behind such a model is to reduce the noise present in data. We are also parallelly working on increasing the data by taking the frames at an interval of 1 second in place of 4 seconds. One objective behind increasing the frame rate is more data and the other objective is that it can also cover some instantaneous activities which the user is doing within 4 seconds which if not handled carefully will result in working as an outlier.Moreover we are looking for models which can use temporal information while training like Hierarchical Recurrent Neural Networks. We are also experimenting with an ensemble of other pre-trained models like VGG-19 and Inception Neural Network on the images from three angles.

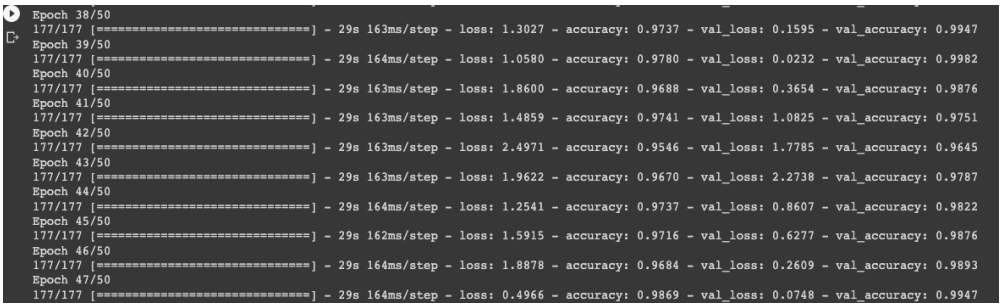

Figure 4: Result on ResNet50-v2

4. METHODOLOGY-2

4.1. Methodology Flowdiagram

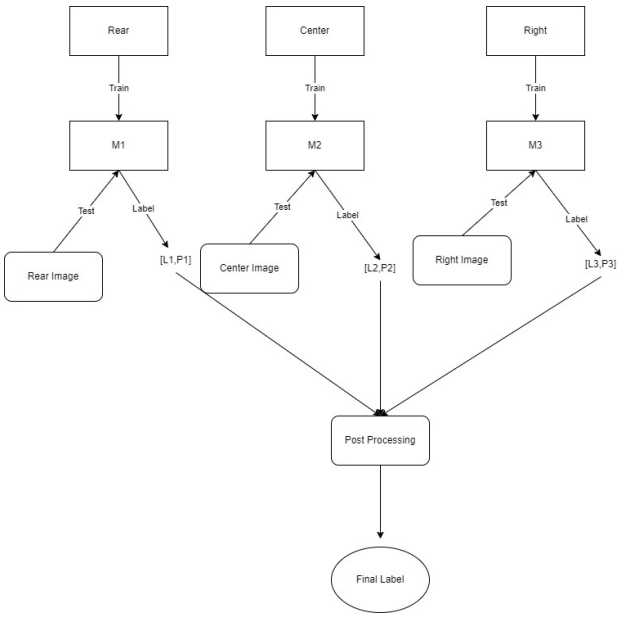

Figure 5: Above diagram sums up the methodology-2

4.2. Data Preprocessing

The image shown below is a collage of some of the segmented images taken from the three cameras located at three different place inside the vehicle, in which we removed the background from the image and extracted the person performing different types of activities.

4.3. Model

4.3.1. Train-test Split

After segmenting images for frame rate of 1 image per 2 seconds, we condensed images into 3 folders : center, rear and right. We trained 3 models for these three folders separately. We have 1770 files in each folder. Out of these we kept 354 files for validation and rest of them for training.

Figure 6: Segmented Images

4.3.2. Model Definition

We have trained three instances of pre-trained model VGG19 namely M1, M2, M3 for segmented images captured from the cameras with center, rear and right views respectively and then saved the model and their weights. The model architecture is shown below:

Layer (type)	Output Shape	Param #
input_2 (InputLayer)	[(None, 256, 256, 3)]	0
block1_conv1 (Conv2D)	(None, 256, 256, 64)	1792
block1_conv2 (Conv2D)	(None, 256, 256, 64)	36928
block1_pool (MaxPooling2D)	(None, 128, 128, 64)	0
block2_conv1 (Conv2D)	(None, 128, 128, 128)	73856
block2_conv2 (Conv2D)	(None, 128, 128, 128)	147584
block2_pool (MaxPooling2D)	(None, 64, 64, 128)	0
block3_conv1 (Conv2D)	(None, 64, 64, 256)	295168
block3_conv2 (Conv2D)	(None, 64, 64, 256)	590080
block3_conv3 (Conv2D)	(None, 64, 64, 256)	590080
block3_conv4 (Conv2D)	(None, 64, 64, 256)	590080
block3_pool (MaxPooling2D)	(None, 32, 32, 256)	0
block4_conv1 (Conv2D)	(None, 32, 32, 512)	1180160
block4_conv2 (Conv2D)	(None, 32, 32, 512)	2359808
block4_conv3 (Conv2D)	(None, 32, 32, 512)	2359808
block4_conv4 (Conv2D)	(None, 32, 32, 512)	2359808
block4_pool (MaxPooling2D)	(None, 16, 16, 512)	0
block5_conv1 (Conv2D)	(None, 16, 16, 512)	2359808
block5_conv2 (Conv2D)	(None, 16, 16, 512)	2359808
block5_conv3 (Conv2D)	(None, 16, 16, 512)	2359808
block5_conv4 (Conv2D)	(None, 16, 16, 512)	2359808
block5_pool (MaxPooling2D)	(None, 8, 8, 512)	0
flatten_1 (Flatten)	(None, 32768)	0
dropout_1 (Dropout)	(None, 32768)	0
dense_1 (Dense)	(None, 18)	589842

Total params: 20,614,226
Trainable params: 589,842
Non-trainable params: 20,024,384

Figure 7: Model Architecture

4.4. Prediction

We did same data pre-processing for the test images as we did for the train images. We did prediction for the test images using M1, M2 and M3 for center, rear, and right view images. For every test image we get the three labels and three probabilities and then concatenated them and stored them in array namely [l1,l2,l3] and [p1,p2,p3] respectively.

4.5. Post-processing

After storing [l1,l2,l3] and [p1,p2,p3] for each image in a row, we get a csv file say target.csv . We ran 2 for loops through entire target.csv to remove the noises in prediction. For the first for loop as described in the figure shown below, we have label of previous index say L_previous and in current row we have [l1,l2,l3] and [p1,p2,p3] . First we checked if any label in the current label array matches the L_previous and if it matches then we keep L_previous for the current index else if it didn't match then we checked for the most occuring label in [l1,,l2,l3] say L_m if it's number of occurance is greater than 1 then we updated L_m as the label for current row else we keep the label with highest probability with respect to [p1,p2,p3] say L_p in the current row.

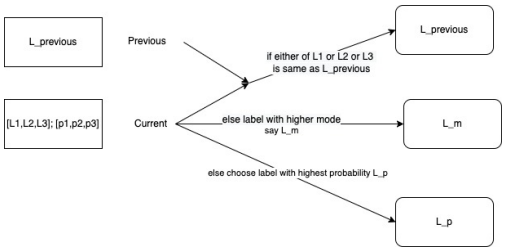

Figure 8: Flow Chart representing algo for first for loop

In the second for loop we did **mode smoothing** with window of size 5 and hope of 1. For the first 2 rows we kept the labels as previously found and from third row we replace current label with the label which has most occured in that particular window.

5. References

- DeepSafeDrive: A grammar-aware driver parsing approach to Driver Behavioral Situational Awareness (DB-SAW) by T. Hoang NganLe etal
- Detection of braking intention in diverse situations during simulated driving based on EEG feature combination by Hwa Kim etal
- How Much of Driving Is Preattentive?
- Personalized Driver/Vehicle Lane Change Models for ADAS
- Driver Activity Recognition for Intelligent Vehicles: A Deep Learning Approach by Yang Xing etal

6. Work Distribution

Priyanshu Kumar: Examined research publications to have a better understanding of current work in the driver activity recognition arena. Experimented with a variety of model architectures, including Basic CNN and several versions of Resnet-5, Inception Net and VGG that were pre-trained.

Yatish Goel: Worked on data pre-processing and model training directly from video. Went through various research papers for better understanding of video based data models. Implemented and fine tuned image segmentation method.

Raj Karan Kumar:Read 5+ academic publications to have a better grasp of the topic of driver activity recognition research. Contributed to the latex report's writing. Contributed to picture segmentation by researching various image segmentation techniques.

Raj Aryan: Worked on Data-preprocessing and went through research papers to contribute to the model team. Implemented various opencv methods for extracting a person from images.

7. Turnitin

11 % similarity found on turnitin

8% similarity found in turnitin