VEHICLE DRIVER CHARACTERIZATION USING TIME DISTRIBUTED DEEP NETWORKS

A Project Report

Submitted by

Ms. ANAGHA K J

REG NO : TKM21MEAI01

SEMESTER : IV

In partial fulfillment for the award of the degree of

MASTER OF TECHNOLOGY
IN
Mechanical Engineering (Artificial Intelligence)

Under the guidance of
Dr. SABEENA BEEVI K

Thangal Kunju Musaliar College of Engineering
Kerala

MAY 2023

DECLARATION

I, the undersigned hereby declare that the project report "VEHICLE DRIVER CHARACTERIZATION USING TIME DISTRIBUTED DEEP NETWORKS", submitted for partial fulfillment of the requirements for the award of the degree of Master of Technology of the APJ Abdul Kalam Technological University, Kerala is a bonafide work done by me, under the supervision of Dr. Sabeena Beevi K. This submission represents my ideas in my own words and where ideas or words of others have been included, I have adequately and accurately cited and referenced the original sources. I also declare that I have adhered to the ethics of academic honesty and integrity and have not misrepresented or fabricated any data or idea or fact or source in my submission. I understand that any violation of the above will be a cause for disciplinary action by the institute and/or the University and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been obtained. This report has not been previously formed as the basis for the award of any degree, diploma or similar title of any other University.

Place: Kollam
Date: ANAGHA K J

Thangal Kunju Musaliar College of Engineering
Centre for Artificial Intelligence

C E R T I F I C A T E

This is to certify that, this report titled ***VEHICLE DRIVER CHARACTERIZATION USING TIME DISTRIBUTED DEEP NETWORKS*** is a bonafide record of the **Project** presented by **ANAGHA K J (TKM21MEAI01)**, under our guidance and supervision, in partial fulfillment of the requirements for the award of the degree, **M.Tech in Mechanical Engineering (Artificial Intelligence)** in **APJ Abdul Kalam Technological University** .

Project Guide	Project Coordinator	Head of the Department
Dr. Sabeena Beevi K Assoc. Professor & HOD Department of Electrical and Electronics engineering	Prof. Chinnu Jacob Assistant Professor Centre for AI	Dr. Imthias Ahamed T P Professor Centre for AI

Internal Examiner

External Examiner

ACKNOWLEDGEMENT

A successful project is a fruitful culmination of efforts by many people, some directly involved and some others indirectly, by providing support and encouragement. Firstly I would like to thank the almighty for giving me the wisdom and grace for making my project a memorable one. I thank him for steering me to the shore of fulfilment under his protective wings.

I express my sincere gratitude to **Dr. T A Shahul Hameed**, Principal of TKM College of Engineering for giving me an opportunity to present my project. I would like to thank **Dr. Imthias Ahamed T P**, Professor and Head of Department, Centre for Artificial Intelligence, TKM College of Engineering, Kollam, for his constant support and encouragement throughout the project work.

With a profound sense of gratitude, I would like to express my heartfelt thanks to my guide, **Dr. Sabeena Beevi K**, Associate Professor and Head of Department, Department of Electrical and Electronics Engineering, and my Project Coordinator, **Prof. Chinnu Jacob**, Assistant Professor, Centre for AI, for their valuable suggestions, guidance and immense encouragement. I am grateful to **Prof. Sumod Sundar**, Assistant Professor, Centre for AI, for his valuable feedback and patience, which helped me to complete this project to the best of my abilities. I would like to thank **Mr. Sreejith Pai P S**, Project Manager and **Mr. Sajin S**, Project Mentor, Tata Elxsi for their expert guidance and cooperation. I also extend my thanks to the entire faculty and staff members of the Centre for AI, TKM College of Engineering, Kollam, who have encouraged me throughout this work.

I also express my thanks to my loving parents and friends, for their support and encouragement in the successful completion of this work.

Anagha K J

Abstract

Every year, thousands of accidents can be attributed to drivers who were either distracted or too tired to pay attention to the road. Observing drivers in their natural environment makes it hard to study how they act. The naturalistic driving study (NDS) has become the most popular method because it let researchers see how drivers act in the real world and help them drive safer by detecting various driver behaviours. In the studies, different types of data are used, such as the driver's physical condition, audio and visual features, and information about the car. However, the main source of data is images of the driver's face, and hands on the steering wheel taken by a camera inside the car. In order to increase the system's capacity for generalisation, it is suggested in this work that sensor data be incorporated into the vision-based distracted driver detection model. In order to achieve this, a new data set was generated that consists of driver face images, road images, and sensor data obtained from the Carla simulator. Later, a model was developed to identify four distracted driving behaviours such as aggressive, distracted by phone, drinking, and normal driving. In the proposed system, CNN extracts the visual characteristics and LSTM processes the temporal data. The collected data is evaluated using a CNN-LSTM model. The performance of the model is improved by combining the sensor data with image data. Driver behaviour detection model using both sensor data and face images achieved a classification accuracy of 97%.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Modern vehicles are nearly autonomous, thanks to Advanced Driver Assistance Systems (ADAS), which rely on ultrasonic sensors, cameras, radars, and lidars. These sensors primarily focus on environmental perception, with limited attention given to the human driver's perspective. While automation could eventually replace the driver, the vehicle must provide performance on par with a human driver to be trusted. Thus, ADAS should concentrate on comprehending, modeling, and predicting human agents, as well as surrounding traffic conditions. Real-world driving situations involve multiple agents interacting with each other and infrastructures, and addressing this complexity will contribute to solving the challenges of fully autonomous driving. This includes evaluating traffic situations, deciphering the intentions of nearby road users, detecting potential hazards, planning the vehicle's trajectory, and executing the driving task.

Gaining a comprehensive understanding of a driver's actions is a complex problem, but it is crucial for determining how vehicles will adapt to different driving conditions and environments. Accidents can occur if drivers focus their attention on something other than the road, such as texting, talking on the phone, changing the radio station, eating, or drinking [1]. Therefore, many collisions may be avoided if these behaviours were detected. Because of the gravity of the issue, researchers have focused extensively on methods to identify distracted drivers. According to [2], driving monitoring and support systems not only serve to minimise the likelihood of car accidents, but they also assist in reducing distracted driving, which directly impacts fuel efficiency. The driver's driving pattern significantly impacts fuel usage, as also suggested in [3,4]. According to the research of Birrell and Young [5], when drivers improve their driving technique, they reduce their fuel consumption and their likelihood of getting into accidents caused by distracted driving.

The World Health Organization (WHO) reported in 2022 that distracted drivers caused road traffic crashes resulting in the deaths of 1.3 million people every year. With the current fast-paced lifestyle, multitasking has become more prevalent, and people may underestimate the risk of inattention while driving. However, failing to pay attention while driving can significantly increase the likelihood of being involved in a motor vehicle accident. Cell phone use, changing radio stations, texting and drinking while driving are a few of the most frequent dangerous driving behaviours [6,7].

Numerous efforts have been made to identify this type of behaviour through phone location using sound or mobile phone detection methods. Researchers have also experimented with eye gaze tracking to detect such actions [8,9]. In many instances, a camera is positioned inside the vehicle, capturing the driver during the drive. These images are then utilized to categorize the driver's behaviours.

Another method involves utilizing sensors built into a vehicle. Some cars come equipped with hundreds of physical sensors that provide information about the car's internal subsystems. Examples of sensor readings include speed, revolutions per minute, throttle position, and fuel capacity. These readings can be accessed through the standard On-Board Diagnostics (OBD-II) port found in all vehicles. In related studies [10,11], researchers employed these sensors to detect distractions.

In some recent studies [12,13], researchers focused on categorizing driving conditions as either normal or fatigued. In one study [12], infrared cameras were used to capture eye regions for 26 participants. By employing a CNN combined with LSTM-RNN units, they were able to achieve more than 95% accuracy in the binary classification task.

In recent times, numerous applications involving driving behaviour analysis have emerged. For example, insurance companies are creating methods to automatically assess their clients' driving behaviors for usage-based insurance (UBI) policies. Likewise, ridesharing companies use data gathered from their smartphone apps to monitor and evaluate their drivers' driving behaviors for risk management and service enhancement. Additionally, in the realm of advanced driver-assistance systems (ADAS), researchers are working on developing smart systems that offer feedback based on the driver's behaviour, enabling them to take corrective measures during their driving to prevent accidents.

This study aims to explore the possibility of developing a system that can identify when a driver is not paying enough attention while driving. The system could use trained models to create a driver-state detection system that effectively monitors a driver's level of distraction while driving. It is common for drivers to engage in distracting activities such as talking on the phone, texting, drinking, or even applying makeup while driving, but they should be aware of the potential hazards of these activities. It is the duty of vehicle manufacturers to provide intelligent systems in their vehicles that can detect driver distractions. By implementing an in-vehicle system that alerts the driver every time they become distracted, the risks associated with these activities could be reduced. To develop an effective vision-based detection system, we train visual data using a CNN model, incorporating transfer learning. Concurrently, we employ LSTM to process sensor data, distinguishing between normal, aggressive and distracted driving patterns. In this study, we utilize a collected dataset containing both visual and sensor data.

1.1 Objective(s)

- To build a driver-assist system that would improve driver safety on the road by alerting them during driving.

  - Gather data (driver face images, road view images, sensor data) that can be used to categorise different driving styles.
  - Detects whether the driver is aggressive, normal, distracted by the phone, or drinking.
  - Combining the sensor data collected within the vehicle with image data while driving is likely to improve the accuracy of the driver behaviour detection task.

- To enhance fleet management by monitoring drivers for inefficient driving habits and routes to optimize fleet performance and reduce fuel costs.
- To improve fuel efficiency by identifying wasteful or inefficient driving habits.
- To automatically evaluate the clients' driving habits in order to implement usage-based insurance policies.

1.2 Organization of the report

The rest of the report is structured as follows: Chapter 2 examines various methods for creating driver behaviour detection models. Chapter 3 provides a comprehensive explanation of the methodology used. Chapter 4 covers the experimental work. The findings from several case analyses are discussed in Chapter 5. Finally, Chapter 6 presents the conclusions drawn from this study.

Chapter 2

Related Works

Research in the field of driver behaviour detection can be categorized into three groups: vision-based, sensor-based, and multimodal driver action behaviour tasks. In this section, we provide a concise summary of relevant studies within these three classifications.

2.1 Vision-based driver behaviour detection

In the domain of vision-based driver action detection, prior research primarily relies on images captured by an in-car camera to monitor the driver's actions and movements. Machine learning algorithms such as CNN, support vector machines (SVM), k-nearest neighbour (k-NN), and AdaBoost are used to classify the frames obtained by the camera.

In a study by Ohn-Bar et al. [14], the goal was to categorize input images into one of three actions: hands on the wheel, gear interaction, and instrument cluster (radio, etc.) interaction. They employed two distinct cameras aimed at the hand and face, respectively. Using SVM, they attempted to identify the driver's action and achieved 90% accuracy with only hand-related data, and 94% accuracy when combining hand and face information. In the research of [15–17], authors utilized the same dataset collected by Southeast University, where each image is labelled with an action: hands on the wheel, operating the shift lever, eating, or talking on the phone. Various classifiers were used in these studies, with the highest classification accuracy of 99% attained using CNN [17].

In studies [18,19], StateFarm's distracted driver detection dataset was utilized, consisting of images labelled with one of ten actions: safe driving, texting with the right hand, talking on the phone with the right hand, texting with the left hand, talking on the phone with the left hand, operating the radio, drinking, reaching behind, hair and makeup, and talking to passengers. In [18], to give an efficient solution, the Machine Learning model employs Convolutional Neural Networks to not only identify the distracted behaviour of the driver but also to determine the source of his distraction by analysing images received from the camera module deployed inside the car. Convolutional Neural Networks are notable for learning spatial information from images, which can then be investigated further by fully connected neural networks. The experimental results demonstrate a 99% average performance in distraction identification, indicating that the Convolutional Neural Networks approach can be

utilised to detect distraction among drivers. In [19], they used two distinct CNN models to classify the actions. In one of these models, they incorporated triplet loss to enhance the overall accuracy of the classification model, resulting in an accuracy rate of 98%. It should be noted that they did not apply a leave-driver-out cross-validation strategy.

In studies [20,21], the American University in Cairo (AUC) distracted driver dataset was utilized, which is similar to StateFarm's dataset. In [20], researchers trained five distinct types of CNN using the original image, face image, hand image, face with hand image, and skin-segmented image. The results obtained from CNN were then weighted using a genetic algorithm to classify the action. Baheti et al. [21] implemented a modified CNN with various regularization techniques to prevent overfitting, ultimately achieving 96% accuracy in the distracted driver detection task.

Gesture patterns are less recognisable in vehicles due to physical constraints and driver body occlusions. However, by leveraging modern camera technology, a convolutional neural network (CNN) can be employed for visual processing. In paper [22], the authors develop a hybrid CNN framework (HCF) for detecting distracted driving behaviour by applying deep learning to analyse visual characteristics. To improve the accuracy of the driving activity detection system, they first used a cooperative pre-trained model based on transfer learning that incorporates ResNet50, Inception V3, and Xception to extract driver behaviour variables. To avoid the suggested HCF from overfitting the training data, they use an enhanced dropout technique. During the evaluation, they also used the class activation mapping (CAM) technique to emphasise the feature region containing ten evaluated classes. According to the testing data, the proposed HCF has a classification accuracy of 96.74%.

Then, in paper [23] by investigating the significance of human poses key points and involved items, they demonstrate that the suggested posture and human-things interaction descriptor is basic yet rich and meaningful. The novel contextual scene descriptor consists of high-level semantic information (i.e. spatial arrangements of body parts and objects in an image) and benefits from pre-trained body parts and objects that can obtain these distinctive characteristics that are then passed on to the classification layer in order to determine the classification category.

2.2 Sensor-based driver behaviour detection

Sensor-based driver action detection systems typically rely on data gathered from sensors like gyroscopes and accelerometers, or devices such as GPS and OBD. This collected data is employed to identify driver actions using various classifiers, similar to those used in vision-based driver action detection systems. In one study, Jafarnejad et al. [24] aimed to determine if a driver was distracted using only car sensor data, which included parameters such as percentage gas pedal, steering wheel angle, vehicle speed, engine RPM, pitch, roll, and yaw. They compared the results of five different classifiers—AdaBoost, gradient boosting, random forest, k-NN, and SVM—and found the best model using the gradient boosting classifier.

Wesley et al. [25] collected driver images using a thermal camera and sought to determine

the driver's mental load for various actions using these images. They observed skin temperature differences in the driver's forehead while performing different actions and demonstrated that the mental load of the driver was lower when performing a single action (driving) compared to doing two tasks (driving and talking on the phone) simultaneously. Their work focused on monitoring findings rather than classification.

Bo et al. [26] concentrated on identifying the driver's actions using only smartphone sensors, such as accelerometers, gyroscopes, and magnetometers, without any external sensors. They developed a Hidden Markov Model (HMM) that included the detection of successive or interrelated actions, such as determining vehicle entry, boarding side, whether the driver is seated in the front or back seat, normal texting, and texting while driving. They achieved an accuracy of 87.18% in detecting these actions.

In [11], authors aimed to prevent mobile phone usage while driving in order to avoid distractions. They utilized an Arduino interface with OBD and an accelerometer to collect data, which was then sent to a mobile phone application via Bluetooth. The app restricts incoming and outgoing calls when the vehicle's speed surpasses 10 km/h. Additionally, the incoming data was assessed according to specific threshold values, and certain actions were identified during driving. These actions included normal driving, turning left, turning right, sudden lane changes to the right or left, u-turns (leftward), sudden acceleration, and harsh braking. The detected actions were analyzed using graphs and threshold values obtained from the sensor data.

Amarasinghe et al. [10] developed a mobile application for driver monitoring using real-time OBD sensor data. First, they pre-processed the data collected by the mobile app. Then, employing a cloud-based back-end, they aimed to determine if the driver was reckless and if any driving anomalies were present. They used a simple formula that measured acceleration and deceleration according to specific threshold values for particular time intervals to identify reckless driving. Lastly, measurements exceeding the threshold were classified as reckless driving.

In [27,28], the objective was to identify driver behaviour and driving style using sensor data. In [27], the aim was to determine if the driver was performing normal or aggressive driving using accelerometer data. They experimented with various feature sets derived from accelerometer data for classification using the k-NN algorithm. The best classification results were achieved using the following measurements: mean of longitudinal acceleration, mean of vertical acceleration, a median of vertical acceleration, covariance between longitudinal and lateral acceleration, and three fifth-order polynomial coefficients after inverse cumulative density histogram approximation. In [28], the goal was to identify aggressive driving using data from smartphone sensors, such as accelerometers, gyroscopes, and GPS. To determine the driving style, they first identified the driver's actions, such as aggressive right and left turns, aggressive acceleration and deceleration, and aggressive lane changes. As a result, they found that classification success rates using multiple sensors together were higher than those achieved using individual sensors. They were able to detect aggressive actions with 97% success.

Khodairy et al. [29] have developed an optimized computational model based on LSTM (Long Short-Term Memory) for classifying driving behaviour in two different ways. The first model distinguishes between normal, drowsy, and aggressive driving behaviours using data from an accelerometer, gyroscope, GPS, and preprocessed vehicle detection data obtained from a video-camera sensor. The second model also distinguishes between aggressive and non-aggressive driving behaviours, but without using preprocessed vehicle detection data. In both models, the researchers have attempted to determine the ideal time-series window size and upsampling factor for the proposed models based on various measurement metrics, including accuracy, recall, precision, and F1-score. They achieved an F1 score of 95.26%.

2.3 Multimodal driver behaviour detection

There has been limited research that has combined data from multiple sources for detecting distracted driving. Using multiple sources of data can provide a more comprehensive view of the driver's behaviour, making it easier to detect distractions or other unsafe driving behaviour. Additionally, multimodal approaches can help overcome the limitations of individual sensors or data sources, which may be unreliable or prone to error in certain situations. Overall, multimodal approaches have the potential to improve driver safety by providing more accurate and reliable detection of unsafe driving behaviour.

Streiffer et al. [7] used video recordings from a camera and sensor data from a mobile phone to identify six different driver actions, including normal driving, talking, texting, eating/drinking, hair and makeup, and reaching. They used three deep learning techniques for classification, achieving the highest accuracy with CNN + RNN at 87%. Our work is different in that we aim to detect ten actions and fine-tune our model using a publicly available dataset. We collected image and sensor data from the same source using a mobile app during data collection.

Du et al. [30] used three modalities, including facial expression, speech, and car information, to classify distraction, and their dataset was collected using a driving simulator. The study presents a novel dataset for detecting distracted driving behavior using multiple sources of data and investigates the effectiveness of automatic distraction detection using features from three modalities, namely facial expression, speech, and car signals. The analysis of multimodal features shows that the predictive accuracy of the model improves with the addition of more modalities. The study also proposes a straightforward and efficient multimodal fusion technique that utilizes a polynomial fusion layer and demonstrates superior performance in detecting distracted driving behavior.Similarly, in [31], speech features were used along with facial expressions and car information to detect the type of distraction.

In [32], various data sources were combined for fatigue and distraction detection, and they used SVM and HMM to process the data from different modules. The data was collected using a driving simulator for two tasks: binary classification to distinguish normal driving from distracted driving and five-class classification to detect one of the five actions. In [33], data were collected from unique sources for discriminating visual and cognitive distractions and estimating the level of distraction, rather than detecting specific driver actions.

A recent approach in the area was to blend vision and sensor-based techniques to increase distracted driver detection accuracy [34]. The authors offer a system with two important steps for this goal. In the first phase, they isolate visual and sensor data to create unique detection models. They train image data using a CNN model with transfer learning and fine-tuning approaches to create an efficient vision-based detection model. Simultaneously, analysed sensor data with LSTM to distinguish between normal and aberrant driving. The second stage is to combine the predictions of vision-based CNN and sensor-based LSTM-based models into a final LSTM model and obtain the final predictions on the test set.

Chapter 3

Methodology

3.1 Proposed Methodology

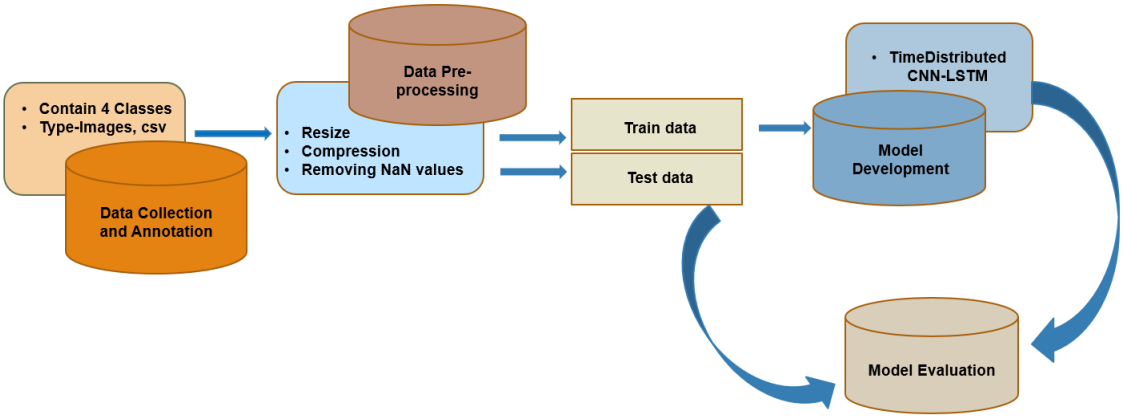

Figure 3.1: Proposed Methodology

Figure 3.1 shows the proposed methodology of the driver behavior detection model. The first step of the process involves collecting data, which includes driver face images, road images, and sensor data obtained from the Carla simulator. The images are in JPG format, while the sensor data is saved as a CSV file. The collected dataset is then annotated based on four types of distracted driving behaviors: aggressive, distracted by phone, drinking, and normal driving. Following this, the data preprocessing stage involves resizing and compressing the input images as needed, as well as removing columns that contain NaN values from the CSV file.

The next step involves splitting the entire dataset into training and test datasets. A TimeDistribured CNN-LSTM model is developed for the driver behavior detection task, with pre-trained weights obtained from training TimeDistributed CNN models on the ImageNet [35] database used to initialize the model weights. To obtain the best results, hyperparameter tuning is performed on the TimeDistributed CNN-LSTM models. Finally, the TimeDistributrd CNN-LSTM model is trained on the training data for the vehicle driver behavior classification task, and evaluated using the test data.

3.2 Networks used

3.2.1 TimeDistributed Convolutional Neural Network

Convolutional Neural Networks (CNNs) [36] are widely used for image classification tasks due to their ability to effectively capture spatial features from images. They work by using convolutional layers that apply filters to the input image, extracting features and creating a feature map. Pooling layers are then applied to reduce the spatial dimensionality of the feature map while retaining important information. The resulting feature maps are flattened and fed into fully connected layers for classification as shown in Figure 3.2.

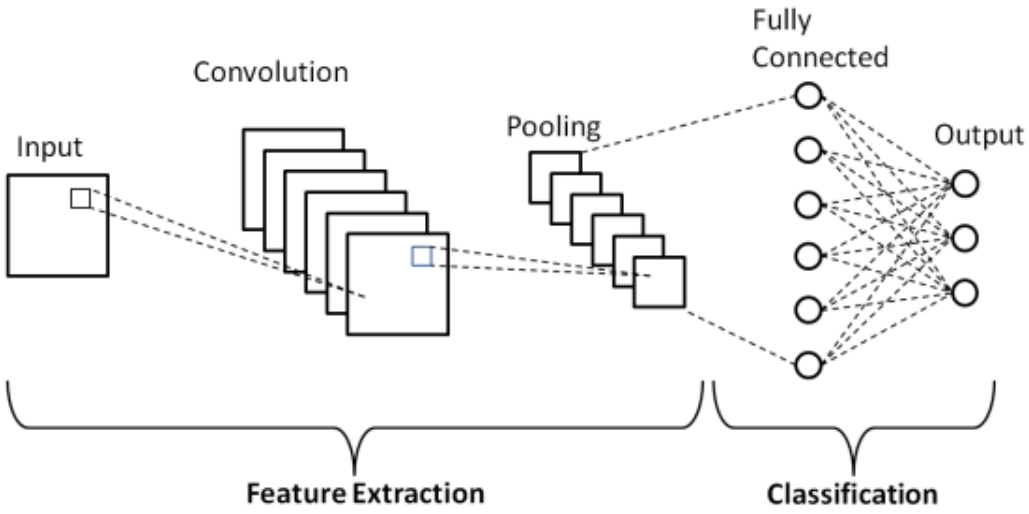

Figure 3.2: Convolutional Neural Network

However, for action recognition in videos, the input data is sequential in nature, meaning that it has both spatial and temporal components. Simple CNNs cannot effectively capture temporal features, since they do not consider the temporal dimension of the data. Therefore, TimeDistributed CNNs are used for action recognition in videos. TimeDistributed CNNs [37] extend the traditional CNN by applying convolutional layers to each temporal slice of the input sequence, allowing for the extraction of both spatial and temporal features. They work by treating each frame of the video as a separate image and applying a traditional CNN to each frame. The resulting feature maps are then concatenated across time and fed into fully connected layers for classification.

In a typical CNN, a single image is processed to identify features that help make a prediction about the object in the image. However, for tasks involving actions or sequences, such as video classification, we need to process multiple images (frames) and capture the temporal relationships between them.

The TimeDistributed layer is designed to handle this need. It can apply the same transformation to a list of input data, such as a sequence of images. This layer works with various

input types, including images, and applies the same layer to each input independently. As a result, it produces one output per input, preserving the temporal structure of the data.

By using the TimeDistributed layer, we can extend the capabilities of a CNN to process sequences of images, apply the same filtering to each image, and then continue the model with other layers to capture the temporal relationships between frames and make predictions based on the entire sequence.

In a standard convolutional neural network, the input is a single image with dimensions (width, height, channel_number), resulting in a 3-dimensional shape. However, when using a Time Distributed layer for processing sequences of images, we need to add an extra dimension to account for the number of images that will be processed simultaneously. Therefore, the input shape for a Time Distributed CNN becomes (timestep, width, height, channel_number), where 'timestep' represents the number of images fed into the model at once. For instance, if we want to process a sequence of 5 images, each having dimensions of (224, 224, 3) (224x224 pixels with 3 RGB channels), the input shape for the Time Distributed CNN would be (5, 224, 224, 3).

The TimeDistributed layer shares the same weights across all input images. When processing a sequence of images, such as 5 images, the weights are updated only once, instead of being tweaked 5 times. This shared weight approach allows for efficient computation while maintaining consistency across all blocks defined within the Time Distributed layer. In general, the TimeDistributed technique is useful when addressing a computer vision challenge that necessitates the use of a complex model.

3.2.2 Long Short Term Memory(LSTM)

LSTMs (Long Short-Term Memory networks) [38] are a type of recurrent neural network (RNN) that can process sequential data such as time series or text data. LSTMs are capable of modeling long-term dependencies in data by using a memory cell to store and control the flow of information through the network. This makes them well-suited for tasks such as behavior classification, where the order of events is important in determining the behavior of the driver.

In the context of driver behavior classification, LSTMs can be used to model the sequence of images and sensor data collected from the driver and vehicle, capturing the temporal dependencies between them. By considering the previous sequence of data, LSTMs can predict the next behavior of the driver. Additionally, LSTMs can handle variable-length sequences and can learn to selectively retain or forget information from previous time steps, making them more effective than traditional RNNs in modeling long-term dependencies. The LSTM memory cell has three key elements: the forget gate, the input gate, and the output gate. The LSTM operation follows a specific set of steps:

- The forget gate takes as input the output value from the previous time step and the input value from the current time step. The forget gate's output is then calculated using equation (1):

$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] + b_f\right) \tag{1}$$

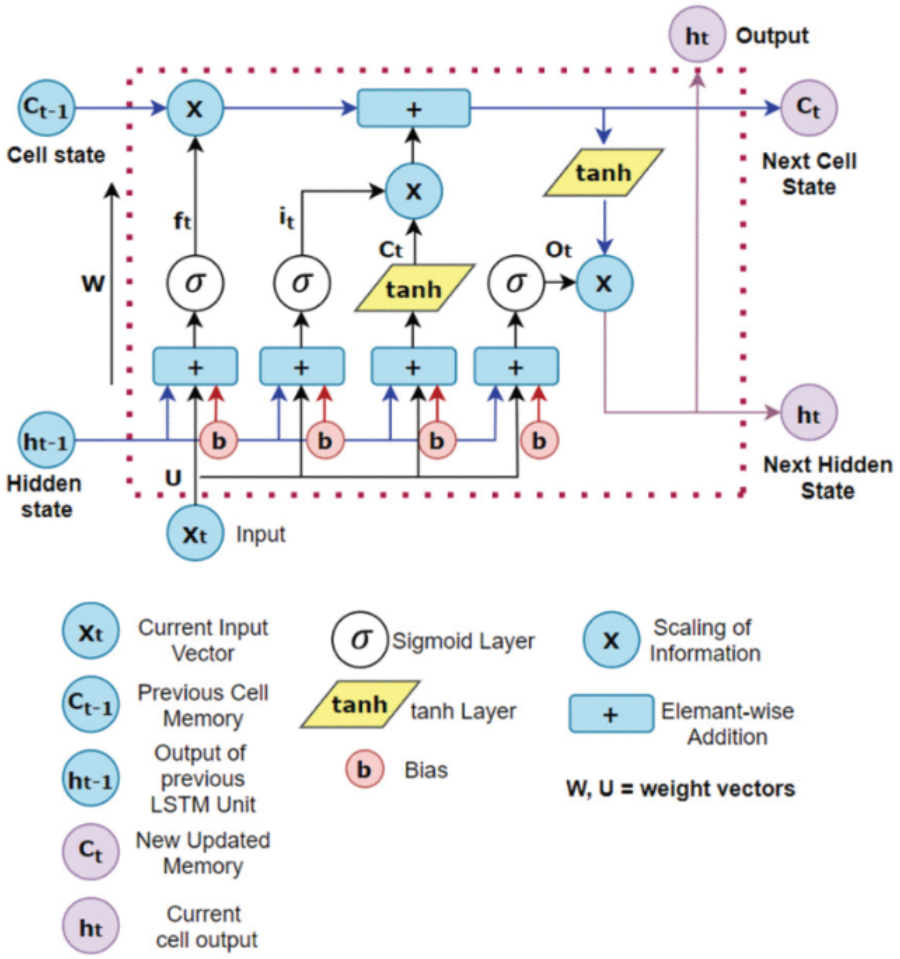

Figure 3.3: LSTM architecture [39]

where the forget gate's output, represented by f_t, has a value between 0 and 1. The calculation of f_t involves the weight of the forget gate (W_f), the bias of the forget gate (b_f), the input value from the current time step (x_t), and the output value from the previous time step (h_{t-1}).

- The input gate takes the output value from the previous time step and the input value from the current time step as input. The input gate then calculates its output value and the candidate cell state using equations (2) and (3):

$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] + b_i\right) \tag{2}$$

$$\tilde{C}_t = \tanh\left(W_c \cdot [h_{t-1}, x_t] + b_c\right) \tag{3}$$

The input gate's output value, represented by i_t, falls between 0 and 1. The calculation of i_t involves the weight of the input gate (W_i), the bias of the input gate (b_i), the weight of the candidate input gate (W_c), and the bias of the candidate input gate (b_c).

- Equation (4) is used to update the current cell state:

$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t \tag{4}$$

where the values of the current cell state C_t range from 0 to 1.

- At time t, the output value (h_{t-1}) and input value (x_t) are fed into the output gate. The output gate then calculates its output value (o_t) using equation (5):

$$o_t = \sigma\left(W_o\left[h_{t-1}, x_t\right] + b_o\right) \tag{5}$$

where the output gate's output value, represented by o_t, falls between 0 and 1. The calculation of o_t involves the weight of the output gate (W_o) and the bias of the output gate (b_o).

- To obtain the output of the LSTM, the output of the output gate and the state of the cell are combined using equation (6):

$$h_t = o_t * \tanh\left(C_t\right) \tag{6}$$

Similar to RNNs, LSTMs have time steps and a memory for each time step. Figure 3.3 illustrates the LSTM cell's mechanism at time step t.

3.3 Different ways of driver behavior classification

Here, Driver behaviour classification is done in three ways:

- **Image-based detection**: This involves using cameras to capture the driver's behavior, such as head movements, eye movements, and facial expressions, and analyzing them to identify patterns of distracted or unsafe driving.
- **Sensor-based detection**: Sensors, such as accelerometers and gyroscopes, can be used to detect sudden changes in speed, direction, or orientation that could indicate reckless driving.
- **Multimodal-based detection**: Refers to the use of multiple sources of data, such as images, sensors to identify patterns of risky driving behavior. This approach combines the strengths of different detection methods to provide a more comprehensive picture of driver behavior. For example, image-based detection can provide information about the driver's head and eye movements, while sensor-based detection can provide information about the vehicle's speed and acceleration.

3.3.1 Image-based detection

Driver behavior classification using only face images with a CNN-LSTM model involves training a model to classify driver behavior based solely on face images captured while driving. The first step is to collect a dataset of face images while the driver is performing different behaviors such as normal driving, distracted by phone, drinking, and aggressive driving. Figure 3.4 illustrates the process of driver behavior classification, which relies solely on the use of facial images.

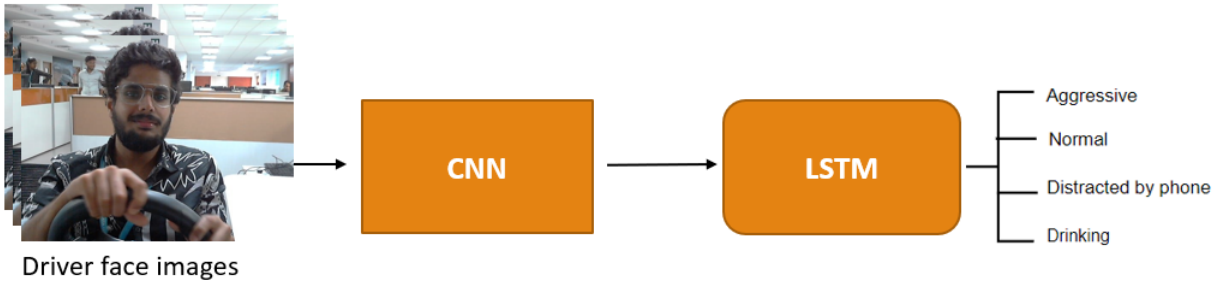

Figure 3.4: Behavior classification using face images only

A CNN-LSTM model is then developed to classify driver behavior based on the face images. The CNN layers of the model extract features from the face images, while the LSTM layer processes the sequence of frames to capture the temporal dependencies. The output layer of the model classifies the driver behavior.

3.3.2 Sensor-based detection

Sensor-based driver behavior detection using LSTM (Long Short-Term Memory) involves using sensor data, such as accelerometer and gyroscope data, to detect patterns of risky or distracted driving. The LSTM is a type of recurrent neural network that is well-suited to modeling sequential data, such as time-series sensor data.

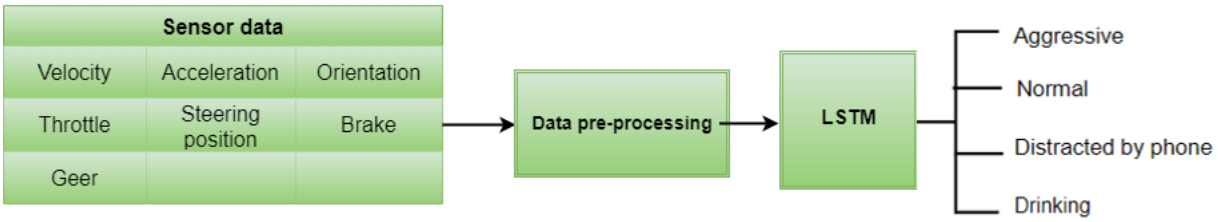

Figure 3.5: Behavior classification using sensor data only

Figure 3.5 illustrates the process of driver behavior classification using sensor data only. The approach involves collecting sensor data from the vehicle and preprocessing it to extract relevant features, such as speed, acceleration, and orientation. These features are then fed into the LSTM model, which learns to recognize patterns in the data that correspond to risky driving behaviors.

3.3.3 Multimodal-based detection

Multimodal driver behavior detection involves the use of multiple sources of data to detect patterns of risky or distracted driving behavior. These sources of data may include video, audio, sensors, and physiological measurements, such as heart rate or skin conductance. Multimodal driver behavior detection provides a more comprehensive and accurate picture of driver behavior compared to single modality detection methods. By combining the strengths of different detection methods, it is possible to improve the accuracy and reliability of driver behavior analysis and provide more effective feedback and intervention to promote safe driving.

3.3.3.1 Using face images as well as sensor data

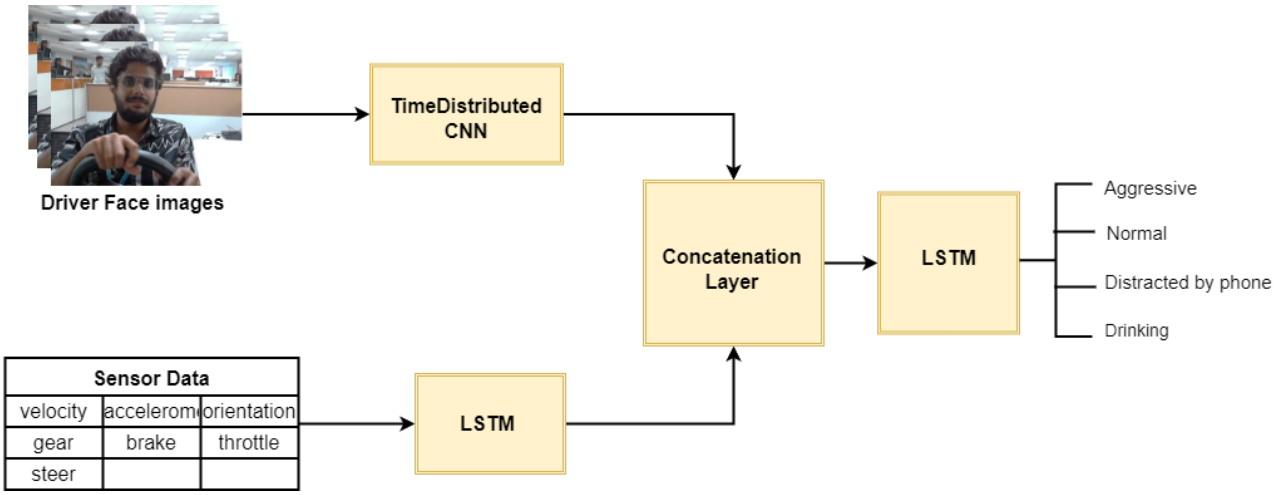

Figure 3.6: Behavior classification using face images as well as sensor data

Driver behavior classification using face images and sensor data can be performed using a combination of TimeDistributed CNN and LSTM models. The TimeDistributed CNN can be used for processing the face images, while LSTM can be used for processing the sensor data. Figure 3.6 depicts the driver behaviour classification model that utilizes both face images and sensor data.

In this approach, the face images and sensor data are first preprocessed, and then combined to create a hybrid dataset. The TimeDistributed CNN can then be trained on the face images portion of the dataset, while the LSTM can be trained on the sensor data portion. The output of both models can then be concatenated, and fed to a LSTM layer to obtain the final classification result.

This approach can provide a more comprehensive understanding of driver behavior, as it takes into account both visual cues and driving patterns. It can also be more robust to noisy or incomplete data, as the sensor data can provide additional information to supplement the face images.

3.3.3.2 Using sensor data along with face and road images

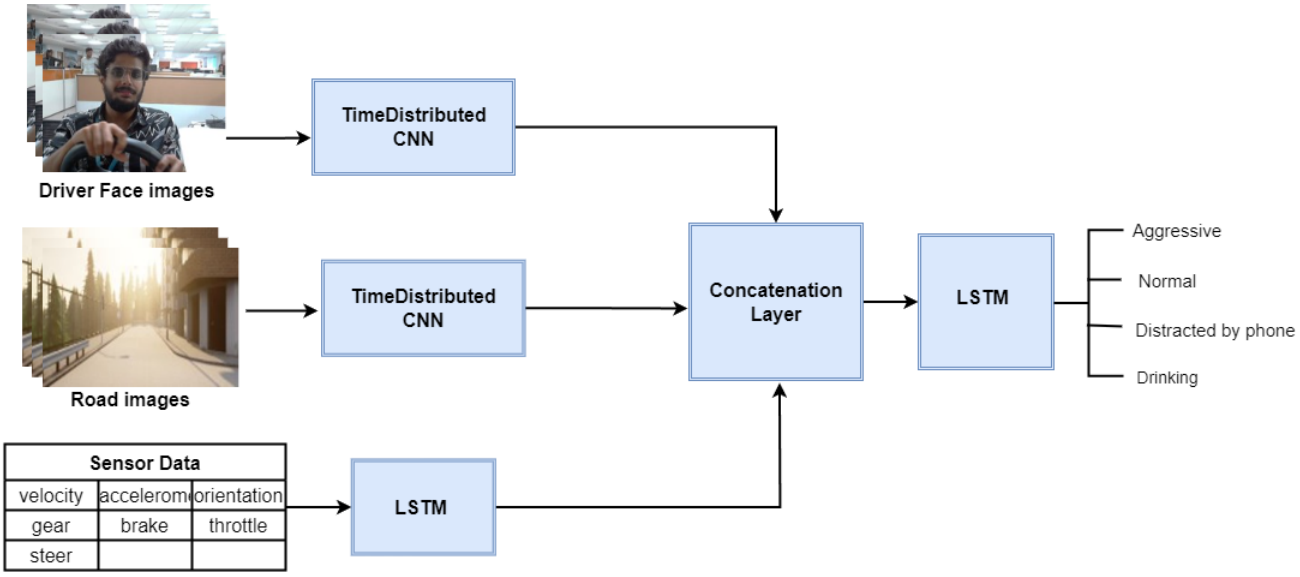

Figure 3.7: Behavior classification using sensor data along with face and road images

Driver behavior classification can be performed using a combination of sensor data, face images, and road images. The sensor data can provide valuable information about the driver's actions such as speed, acceleration, and steering. Face images can be used to detect whether the driver is distracted or not, while road images can provide information about the driving environment such as lane markings, traffic signs, and other vehicles. By combining all these sources of information, it is possible to build a more accurate model for driver behaviour classification.

The driver behavior classification model that incorporates both sensor data along with face and road images is illustrated in Figure 3.7. The TimeDistributed CNN can be used for processing the images, while the LSTM can be used for processing the sensor data. This approach can lead to improved accuracy in identifying and classifying driver behaviour.

Chapter 4

Experimentation

4.1 CARLA Simulated Dataset

CARLA, which stands for Car Learning to Act, is an open-source simulator that is specifically designed for conducting research on autonomous driving. It offers a realistic simulation platform for testing and evaluating autonomous driving algorithms that involve perception, planning, and control.

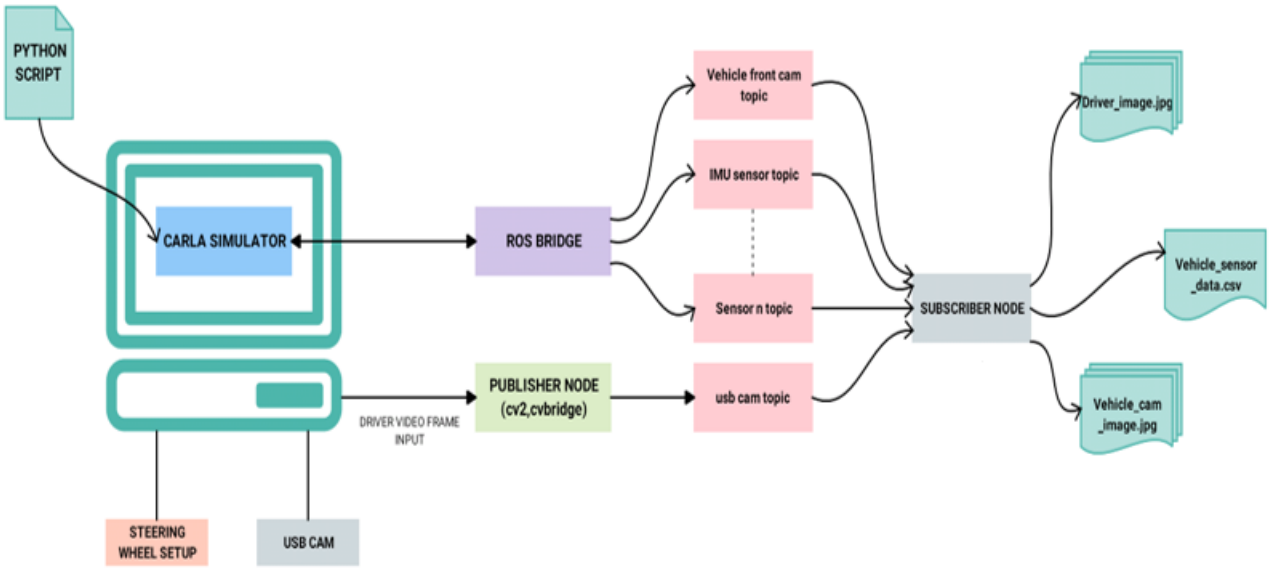

Figure 4.1: Data acquisition from carla simulator

CARLA is capable of simulating different types of sensors such as cameras, lidars, and radars which can be mounted on virtual vehicles. The simulator offers a wide range of pre-built environments that represent different driving scenarios like urban and highway settings, making it suitable for testing and evaluating autonomous driving systems.

One of the key benefits of CARLA is its flexibility and adaptability. It offers a Python API that enables users to interact with the simulator, manage traffic, generate vehicles, and collect data from sensors. This feature allows researchers and developers to create and evaluate various autonomous driving algorithms, from basic tasks such as lane keeping to more complex decision-making and planning.

One of the primary goals of our study is to combine image data with car sensor data to improve the generalization ability of the distracted driver detection model. Therefore, a new dataset consisting of driver face images, road images and sensor data has been collected. For this purpose, Carla simulator was used.

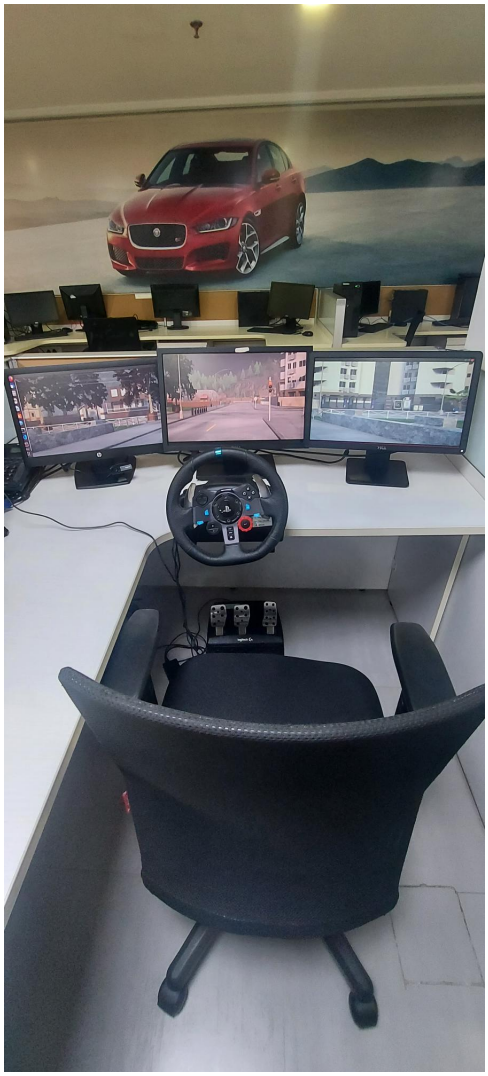

Figure 4.2: Carla Simulator hardware setup

To begin the process, we set up the hardware by configuring a triple monitor system

Table 4.1: Data Distribution

Class Name	Frame Count
Aggressive	24,016
Distracted by phone	37,129
Drinking	28,227
Normal	37,063

for a better driving experience and mounting a USB camera to capture the driver's face images. Carla Simulator hardware setup will look as shown in Figure 4.2. Additionally, we calibrated a steering wheel to control the vehicle in the Carla simulator. We then installed the appropriate versions of Carla and ROS and wrote a Python script to spawn a vehicle in the Carla simulator equipped with necessary sensors, and to control it using the steering wheel. We integrated Carla and ROS through the Ros bridge, which allowed us to make sensor data and concurrent road images from Carla accessible through ROS topics. We created a custom Python ROS publisher node that takes input from the USB camera using OpenCV and converts the images into ROS image messages, which are published to a USB camera topic. To effectively capture data from all the sensors in real-time, we wrote another custom Python ROS subscriber node that subscribes to all the sensor topics. This node saves the sensor data in CSV format and images as JPG files.

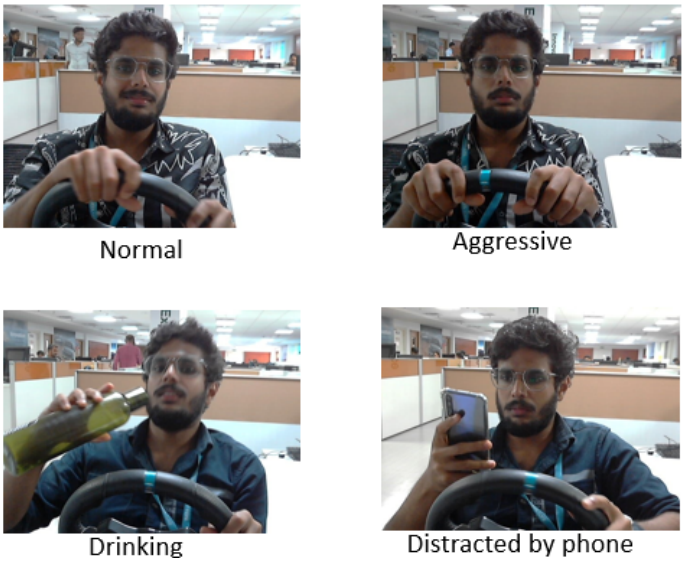

Figure 4.3: Sample dataset of driver face images

Sensor data gathered from simulator are velocity, accelerometer data, orientation, throttle position, steering position, brake, and gear information. With the drivers carried out with this setting, about 126,435 data points with timestamp information were collected. The

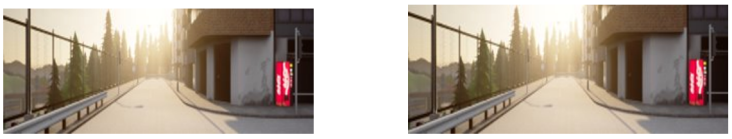

Figure 4.4: Sample dataset of road images

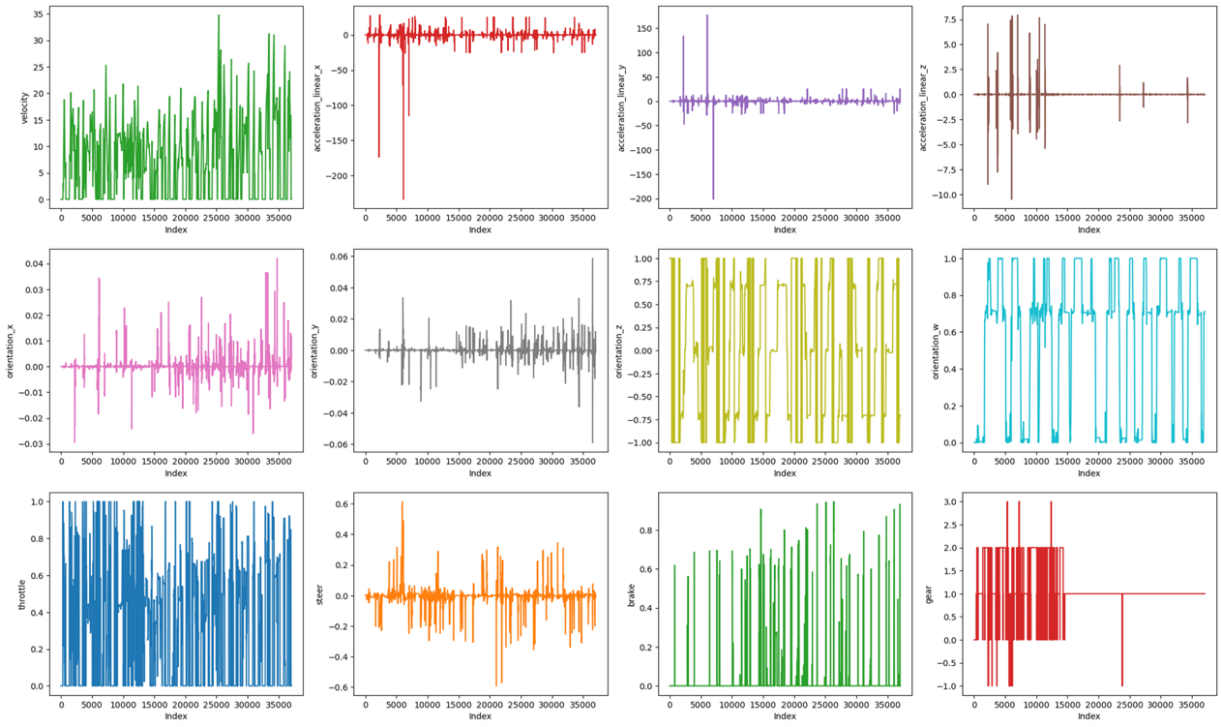

Figure 4.5: 12 Sensor signals of sample dataset

distribution of the data collected during these trips is shown in Table 4.1. Some sample frames that belong to different driver actions from the collected dataset are shown in Figure 4.3. Simultaneously Figure 4.4 shows a few road images obtained through Carla simulator. And the 12 sensor information clogged from the Carla simulator are:

- Velocity
- Acceleration_linear_x
- Acceleration_linear_y
- Acceleration_linear_z
- Orientation_x
- Orientation_y
- Orientation_z

- Orientation_w
- Throttle
- Steer
- Brake
- Gear

Figure 4.5 depicts the collected 12 sensor signals of a sample dataset.

4.2 Data Pre-processing

In the process of driver behavior classification, data preprocessing is an important step to ensure the quality and consistency of the data.

The first step performed in data preprocessing was image resizing and compression. Since it was a triple monitor setup, the obtained road images are the result of front and side cameras as well. Thus, need to be resized because we only require the front view of the vehicle. The original and resized road images are shown in Figure 4.6 and Figure 4.7 respectively.

Figure 4.6: Original road image

Figure 4.7: Resized road image

The input images were resized to a fixed size of 224 x 224 and compressed as required to reduce the data size without losing significant information. This resizing and compression

helped to standardize the input image size, which is important for efficient model training.

Another important step in data preprocessing is handling missing data. In the CSV file containing sensor data, there was some missing values, which can adversely affect the performance of the model. To overcome this issue, columns containing NaN (Not a Number) values were removed from the CSV file. This process helped to clean the data and ensure that the model is trained with only valid and useful data.

4.3 Transfer Learning

Transfer learning is a machine learning technique where a model trained on one task is repurposed on a second related task. It involves using the pre-trained weights of a model trained on a large dataset as the starting point for a new task, instead of training the new model from scratch. This approach allows the new model to benefit from the knowledge learned by the pre-trained model, resulting in faster training and better performance.

Here, transfer learning is used because by initializing the model with pre-trained weights, the model starts with a better set of initial weights, allowing it to converge faster and achieve better results. The model is trained with pre-trained weights that have been trained on ImageNet. By using pre-trained weights, the CNN model can leverage the learned feature representations of the pre-trained model and apply them to new datasets or tasks, allowing for faster and more accurate training.

4.4 Hyperparameter Tuning

Hyperparameter tuning is a process of finding the optimal set of hyperparameters for a given machine learning model. These hyperparameters are not learned from the data, but rather set by the user before training the model.

Optuna is a Python library for hyperparameter optimization, which automates the process of tuning the hyperparameters of a machine learning model. It uses Bayesian optimization to search the hyperparameter space efficiently, and it also provides visualization tools to analyze the results of the search.

Here, Optuna is also used for selecting the best architecture among different types of CNN models. During the optimization process, Optuna will evaluate the performance of the model by checking the accuracy and select the model with the best performance. Therefore, we can choose the model that gives the best result for our specific task.

Thus after Optuna hyperparameter tuning, we got Inception-v3 as the best CNN model and the time step as 20.

Chapter 5

Results

The proposed TimeDistributed CNN-LSTM architecture is developed in Tensorflow 2.0. The network is trained on an image size of 224x224, with a batch size of 2. Training is carried out using the Adam optimizer and sparse categorical crossentropy as loss function. The total dataset is split into 1,00,000 training data and 20,000 validation data.

5.1 Image-based detection results

The given image-based driver behaviour detection system was trained on a dataset of 100,000 images and validated on a dataset of 20,000 images. The model achieved a training accuracy of 89% and a validation accuracy of 86% in classifying driver behaviours into normal, aggressive, drinking, and distracted by phone.

Table 5.1: Quantitative analysis of vision-based model

Training accuracy	89%
Validation accuracy	86%
Training loss	0.15
Validation loss	0.24

Table 5.1 shows the quantitative analysis of the vision-based model. It got a training accuracy of 89%, validation accuracy of 86%, training loss of 0.15 and validation loss of 0.24. To visualize the performance of the model during training, we can plot the training and validation accuracy and loss curves. Figure 5.1 show the accuracy and loss curves during the training process.

From the accuracy curve, we can see that the training accuracy steadily increases with each epoch and eventually level off at around 89%. The validation accuracy follows a similar trend but with slightly less accuracy than the training accuracy. This indicates that the model is not overfitting to the training data, as the validation accuracy is not significantly

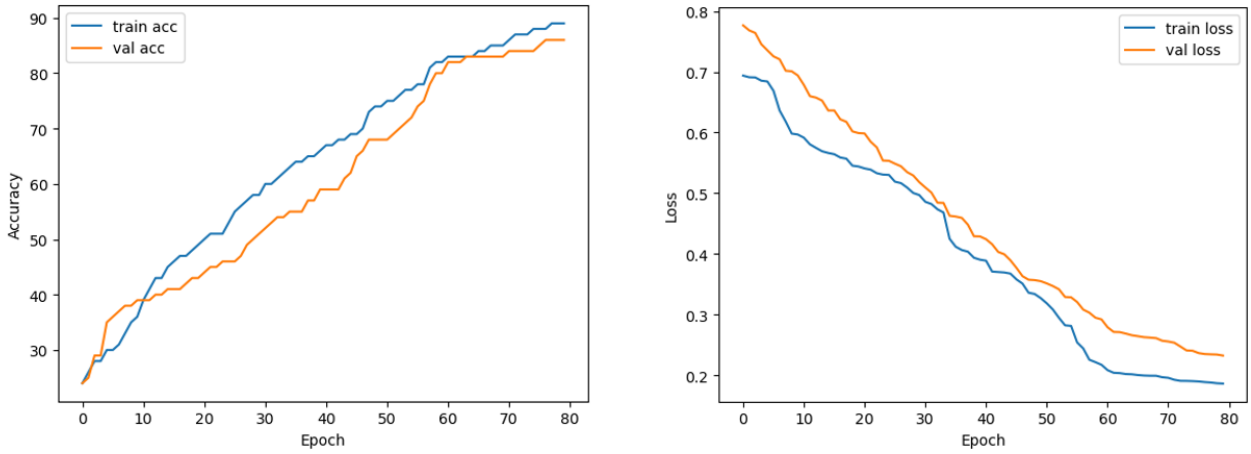

Figure 5.1: Accuracy and loss graph of model using face images only

lower than the training accuracy.

From the loss curve, we can see that both the training and validation loss decrease with each epoch, indicating that the model is improving during training. Overall, the given image-based driver behaviour detection system performs reasonably well with a training accuracy of 89% and a validation accuracy of 86%.

5.2 Sensor-based detection results

In a sensor-based driver behavior detection system, the input data is obtained from various sensors such as accelerometers, gyroscopes in the Carla simulator. The collected data is in CSV format.

Pre-processing has been done by cleaning the data by removing columns with NaN values. The system achieved a training accuracy of 85% and a validation accuracy of 82%.

Table 5.2: Quantitative analysis of sensor-based model

Training accuracy	85%
Validation accuracy	82%
Training loss	0.37
Validation loss	0.43

Table 5.2 shows the quantitative analysis of the sensor-based model. It got a training accuracy of 85%, validation accuracy of 82%, training loss of 0.37 and validation loss of 0.43. To understand the performance of the model, we can plot the accuracy and loss curves during

the training process are shown in Figure 5.2.

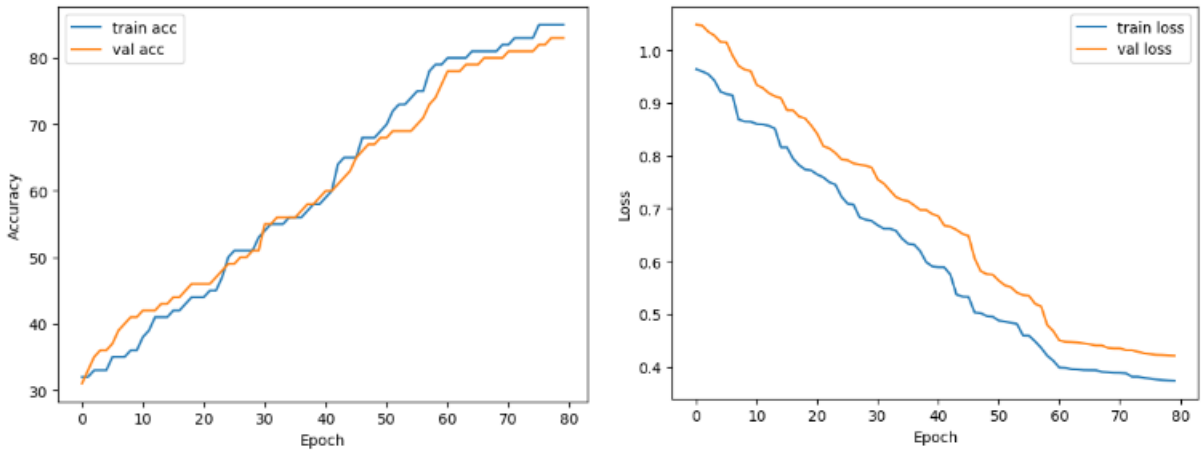

Figure 5.2: Accuracy and loss graph of model using sensor data only

The blue line represents the training accuracy, while the orange line represents the validation accuracy. As we can see from the graph, the model starts with a low accuracy but gradually improves during the training process. Concurrently, the blue line represents the training loss, while the orange line represents the validation loss. As we can see from the graph, the training and validation loss decreases steadily during the training process, indicating that the model is learning from the data. Overall, the sensor-based driver behavior detection system achieved decent training and validation accuracy.

5.3 Multimodal detection results

5.3.1 Using face images as well as sensor data

The driver behaviour detection system that utilizes both face images and sensor data achieved high training accuracy of 99% and validation accuracy of 95%. This suggests that the model is able to accurately classify the driver behaviour into normal, aggressive, drinking, and distracted by phone.

Table 5.3: Quantitative analysis of model using both face images and sensor data

Training accuracy	99%
Validation accuracy	95%
Training loss	0.014
Validation loss	0.026

Table 5.3 shows the quantitative analysis of the model using both face images as well as sensor data. It got a training accuracy of 99%, validation accuracy of 95%, training loss of 0.014 and validation loss of 0.026.

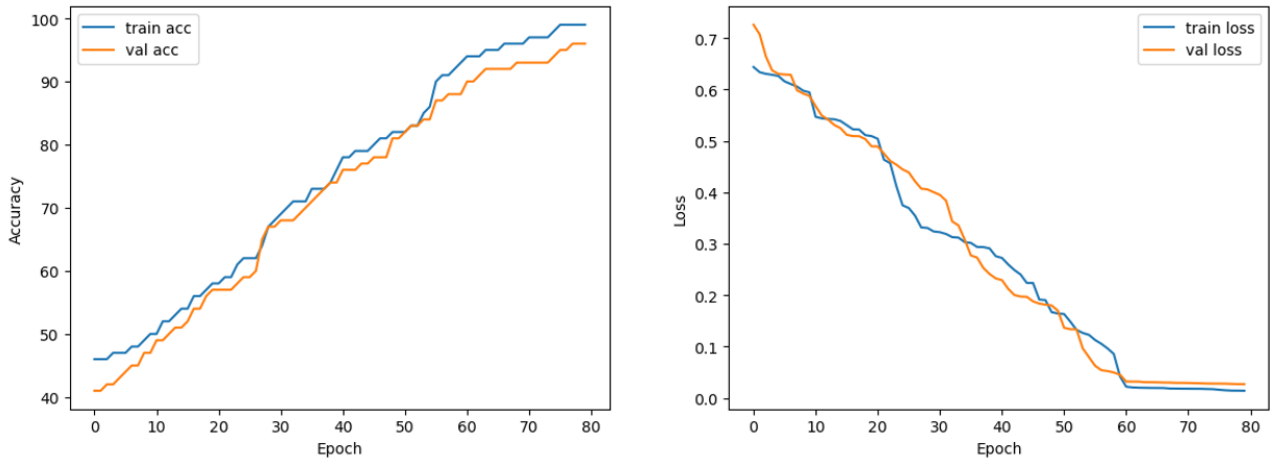

Figure 5.3: Accuracy and loss graph of model using face images as well as sensor data

Figure 5.3 show the accuracy and loss curves during the training process. The accuracy graph shows a steady increase in accuracy during the training process, with occasional dips, indicating that the model is learning from the data. The validation accuracy, on the other hand, follows the same trend but with slightly lower accuracy. The loss graph shows a steady decrease in loss during the training process, indicating that the model is improving in its ability to make accurate predictions.

Overall, the results of this driver behaviour detection system are very promising and suggest that a combination of face images and sensor data can greatly improve the accuracy of driver behaviour detection. And, this model have outperformed the previous two models in performance.

5.3.2 Using sensor data along with face and road images

The driver behavior detection system that uses sensor data along with face and road images to classify into normal, aggressive, drinking, and distracted by phone has shown promising results with a training accuracy of 97% and a validation accuracy of 93%. Figure 5.4 show the accuracy and loss curves during the training process. The accuracy graph shows that the training accuracy steadily increases with epochs, while the validation accuracy also increases initially, but then level off. This indicates that the model is learning the training data well. The loss graph shows that the training loss steadily decreases with epochs, indicating that the model is improving in its ability to classify driver behavior. The validation loss initially decreases along with training loss. However, validation loss is greater than training loss.

Table 5.4: Quantitative analysis of model using sensor data along with face and road images

Training accuracy	97%
Validation accuracy	93%
Training loss	0.048
Validation loss	0.059

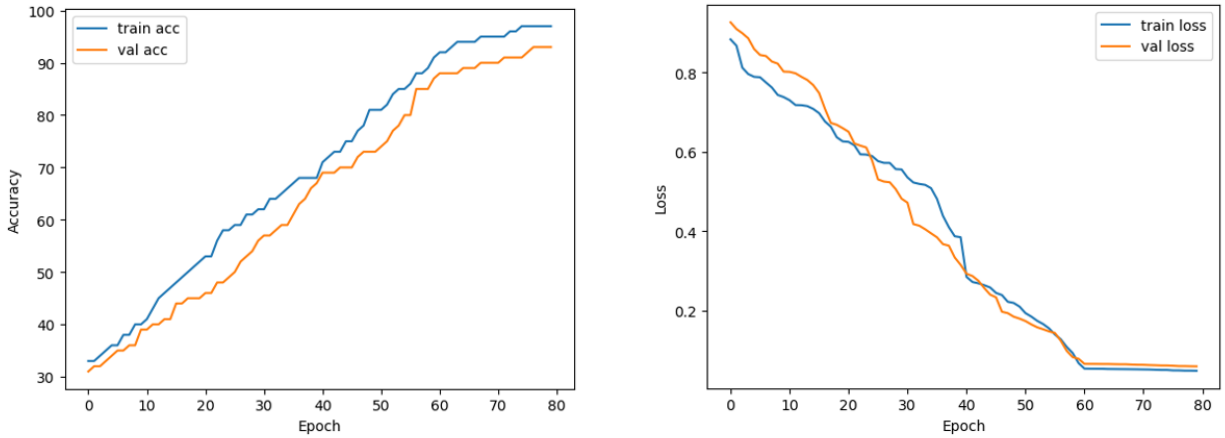

Figure 5.4: Accuracy and loss graph of model using sensor data along with face and road images

Table 5.4 shows the quantitative analysis of the model using both face images as well as sensor data. It got a training accuracy of 97%, validation accuracy of 93%, training loss of 0.048 and validation loss of 0.059.

Overall, driver behavior detection system using sensor data along with face and road images achieved better training and validation accuracy not best. Since the model using both face images and sensor data performed better than this model.

5.4 Comparison of models

Table 5.5 shows the comparative quantitative analysis of the said models. From the table, it is clear that the driver classification model using face images as well as sensor data perform better than other three models. The model got an overall training accuracy of 99% and validation accuracy of 95%. From the results, we can also say that it generalises well to new data.

Face images can provide important visual cues such as eye gaze and head orientation, which can be indicative of distracted driving or drowsiness. However, for certain behaviors such as normal driving or aggressive driving, the visual cues may not be as distinctive. This is where the use of sensor data can be beneficial. Sensor data can provide information on the driver's steering angle, acceleration, and braking, which can be indicative of aggressive driving or normal driving.

Table 5.5: Quantitave Analysis of the models

Proposed model	Training Accuracy	Validation Accuracy	Training Loss	Validation Loss
Using face images only	89(%)	86(%)	0.15	0.24
Using sensor data only only	85(%)	82(%)	0.37	0.43
Using face images as well as sensor data	99(%)	95(%)	0.014	0.026
Using sensor data along with face and road images	97(%)	93(%)	0.048	0.059

By combining the two sources of data, the model can potentially leverage the strengths of each to provide a more accurate classification of driver behavior. In fact, the results of the driver behavior detection system using both face images and sensor data outperformed the other models in terms of accuracy, which supports this idea. This suggests that using both face images and sensor data provides more comprehensive and accurate information about driver behavior compared to using either vision-based or sensor-based data alone.

Furthermore, the addition of road images to the model that uses face and sensor data did not significantly improve its accuracy, as the training and validation accuracies were only slightly lesser than the model that uses face and sensor data alone. Including additional data sources like road images can make the model more complex and increase its capacity to learn patterns from the data. However, adding more complexity to the model does not always result in improved performance, especially if the additional data sources do not provide significant information to the classification task.

In this case, it seems that for the specific classes, the combination of face images and sensor data was already sufficient to achieve good accuracy. Including the road images might have made the model more complex without providing significant additional information for the classification task, resulting in a lower accuracy compared to the other model.

Overall, the accuracy results of this model (training accuracy of 99% and validation accuracy of 95%) are significantly higher than the other models. However, it is important to note that the effectiveness of using both face images and sensor data may depend on the specific context and type of behavior being classified.

5.5 Confusion Matrix for the Best Model for Driver Behaviour Detection

A confusion matrix is a table that is used to evaluate the performance of a machine learning algorithm. The confusion matrix displays the actual and predicted classifications of a dataset, showing the number of true positives, true negatives, false positives, and false negatives.

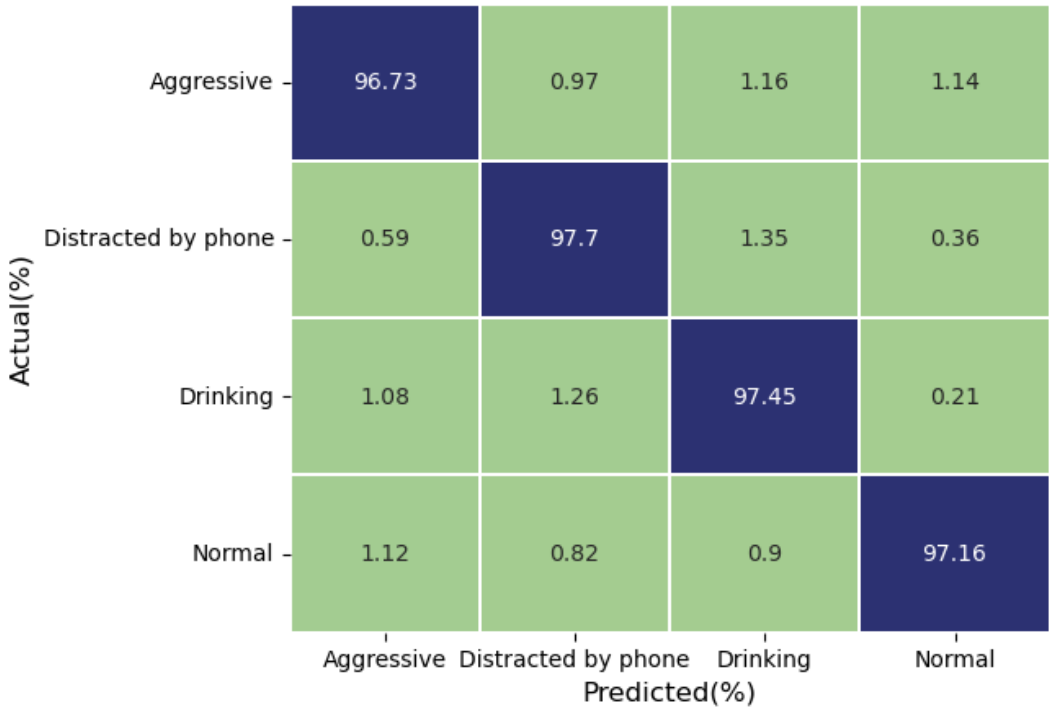

Figure 5.5: Confusion matrix

Since evaluating the performance of the three driver behaviour classification models, using face images, face and sensor data, and sensor data with face and road images, the model that incorporated face and road images outperformed the other two. To further assess the model's performance, we tested it on a separate testing dataset of 6,435 data points, resulting in a confusion matrix shown in Figure 5.5. For the testing dataset we got an overall accuracy of 97%.

The confusion matrix shows that the model is able to accurately classify the majority of the data into the correct categories, with few misclassifications. The highest accuracy is achieved for distracted by phone, drinking driving, followed by normal driving, and aggressive driving, respectively. This suggests that the model is more accurate in detecting distracted driving behaviour and less accurate in detecting aggressive driving behaviour. However, the overall high accuracy of the model indicates that it is effective in detecting multiple types of driver behaviour.

Chapter 6

Conclusion

The goal of the project is to detect different driver behaviour by analyzing various data inputs, including face images, sensor data, and road images. A dataset consisting of these inputs was collected using the CARLA simulator, and four models were trained: one using face images only, one using sensor data only, one using both face images and sensor data, and one using sensor data along with face and road images.

After analyzing the results, it was found that the model using face images and sensor data performed the best and achieved a classification accuracy of 97%. By incorporating both sensor data and face images, the model generalizes better compared to other models. The model was able to identify patterns in facial expressions and sensor information to accurately predict driver behaviour, which has important implications for improving road safety.

The study highlights the potential for using multi-modal data inputs to improve driver behaviour detection and emphasizes the importance of using diverse datasets for training. Furthermore, the study provides insights into the benefits of combining different sources of data to improve the accuracy of driver behaviour detection, which has implications for the development of more sophisticated and effective driver assistance systems.

In conclusion, this project contributes to the understanding of driver behaviour detection using multiple sources of data and provides a foundation for further research in this field. However, it is important to note that the performance of the model heavily relies on the quality and diversity of the data used for training. Therefore, efforts must be made to collect a diverse range of data for driver behaviour detection to ensure that the model can generalize well to new scenarios. In future, we can collect data that can classify more distracted classes such as drowsiness, talking to passengers, and changing radio stations.

Overall, driver behaviour detection using deep learning techniques has the potential to greatly improve road safety and reduce accidents caused by reckless driving. It can also be used in various other applications such as driver monitoring systems and autonomous vehicles.

References

[1] P.F. Ehrlich, B. Costello, A. Randall, Preventing distracted driving: A program from initiation through to evaluation, Am. J. Surg. 219 (6) (2020) 1045–1049.

[2] M.Q. Khan, S. Lee, A comprehensive survey of driving monitoring and assistance systems, Sensors 19 (11) (2019) 2574.

[3] V. Manzoni, A. Corti, P. De Luca, S.M. Savaresi, Driving style estimation via inertial measurements, in: 13th International IEEE Conference on Intelligent Transportation Systems, IEEE, 2010, pp. 777–782.

[4] A. Corti, C. Ongini, M. Tanelli, S.M. Savaresi, Quantitative driving style estimation for energy-oriented applications in road vehicles, in: IEEE International Conference on Systems, Man, and Cybernetics, IEEE, 2013, pp. 3710–3715.

[5] S.A. Birrell, M.S. Young, The impact of smart driving aids on driving performance and driver distraction, Transp. Res. F: Traffic Psychol. Behav. 14 (6) (2011) 484–493.

[6] G.M. Fitch, S.A. Soccolich, F. Guo, J. McClafferty, Y. Fang, R.L. Olson, M.A. Perez, R.J. Hanowski, J.M. Hankey, T.A. Dingus, The Impact of Hand-Held and Hands-Free Cell Phone Use on Driving Performance and Safety-Critical Event Risk, Technical Report, U.S. Department of Transportation, National Highway Traffic Safety Administration, 2013.

[7] C. Streiffer, R. Raghavendra, T. Benson, M. Srivatsa, Darnet: A deep learning solution for distracted driving detection, in: Proceedings of the 18th ACM/IFIP/USENIX Middleware Conference: Industrial Track, ACM, 2017, pp. 22–28.

[8] Vicente, F., Huang, Z., Xiong, X., De la Torre, F., Zhang, W., Levi, D. (2015). Driver gaze tracking and eyes off the road detection system. IEEE Transactions on Intelligent Transportation Systems, 16(4), 2014-2027.

[9] Ji, Q., Yang, X. (2002). Real-time eye, gaze, and face pose tracking for monitoring driver vigilance. Real-time imaging, 8(5), 357-377.

[10] Amarasinghe, M., Kottegoda, S., Arachchi, A. L., Muramudalige, S., Bandara, H. D., Azeez, A. (2015, August). Cloud-based driver monitoring and vehicle diagnostic with OBD2 telematics. In 2015 Fifteenth International Conference on Advances in ICT for Emerging Regions (ICTer) (pp. 243-249). IEEE.

[11] Khandakar, A., Chowdhury, M. E., Ahmed, R., Dhib, A., Mohammed, M., Al-Emadi, N. A. M., Michelson, D. (2019). Portable system for monitoring and controlling driver behavior and the use of a mobile phone while driving. Sensors, 19(7), 1563.

[12] Xiao, Z., Hu, Z., Geng, L., Zhang, F., Wu, J., Li, Y. (2019). Fatigue driving recognition network: fatigue driving recognition via convolutional neural network and long short-term memory units. IET Intelligent Transport Systems, 13(9), 1410-1416.

[13] Li, X., Hong, L., Wang, J. C., Liu, X. (2019). Fatigue driving detection model based on multi-feature fusion and semi-supervised active learning. IET Intelligent Transport Systems, 13(9), 1401-1409.

[14] Ohn-Bar, E., Martin, S., Tawari, A., Trivedi, M. M. (2014, August). Head, eye, and hand patterns for driver activity recognition. In 2014 22nd international conference on pattern recognition (pp. 660-665). IEEE.

[15] Zhao, C., Gao, Y., He, J., Lian, J. (2012). Recognition of driving postures by multiwavelet transform and multilayer perceptron classifier. Engineering Applications of Artificial Intelligence, 25(8), 1677-1686.

[16] Zhao, C., Zhang, B., Lian, J., He, J., Lin, T., Zhang, X. (2011, August). Classification of driving postures by support vector machines. In 2011 sixth international conference on image and graphics (pp. 926-930). IEEE.

[17] Yan, C., Coenen, F., Zhang, B. (2016). Driving posture recognition by convolutional neural networks. IET Computer Vision, 10(2), 103-114.

[18] Masood, S., Rai, A., Aggarwal, A., Doja, M. N., Ahmad, M. (2020). Detecting distraction of drivers using convolutional neural network. Pattern Recognition Letters, 139, 79-85.

[19] Okon, O. D., Meng, L. (2017). Detecting distracted driving with deep learning. In Interactive Collaborative Robotics: Second International Conference, ICR 2017, Hatfield, UK, September 12-16, 2017, Proceedings 2 (pp. 170-179). Springer International Publishing.

[20] Eraqi, H. M., Abouelnaga, Y., Saad, M. H., Moustafa, M. N. (2019). Driver distraction identification with an ensemble of convolutional neural networks. Journal of Advanced Transportation, 2019.

[21] Baheti, B., Gajre, S., Talbar, S. (2018). Detection of distracted driver using convolutional neural network. In Proceedings of the IEEE conference on computer vision and pattern recognition workshops (pp. 1032-1038).

[22] Huang, C., Wang, X., Cao, J., Wang, S., Zhang, Y. (2020). HCF: a hybrid CNN framework for behavior detection of distracted drivers. IEEE access, 8, 109335-109349.

[23] Behera, A., Wharton, Z., Keidel, A., Debnath, B. (2020). Deep cnn, body pose and body-object interaction features for drivers' activity monitoring. IEEE Transactions on Intelligent Transportation Systems.

[24] Jafarnejad, S., Castignani, G., Engel, T. (2018). Non-intrusive Distracted Driving Detection based on Driving Sensing Data. In VEHITS (pp. 178-186).

[25] Wesley, A., Shastri, D., Pavlidis, I. (2010). A novel method to monitor driver's distractions. In CHI'10 Extended Abstracts on Human Factors in Computing Systems (pp. 4273-4278).

[26] Bo, C., Jian, X., Li, X. Y., Mao, X., Wang, Y., Li, F. (2013, September). You're driving and texting: detecting drivers using personal smart phones by leveraging inertial sensors. In Proceedings of the 19th annual international conference on Mobile computing networking (pp. 199-202).

[27] Vaitkus, V., Lengvenis, P., Žylius, G. (2014, September). Driving style classification using long-term accelerometer information. In 2014 19th International Conference on Methods and Models in Automation and Robotics (MMAR) (pp. 641-644). IEEE.

[28] Johnson, D. A., Trivedi, M. M. (2011, October). Driving style recognition using a smartphone as a sensor platform. In 2011 14th International IEEE Conference on Intelligent Transportation Systems (ITSC) (pp. 1609-1615). Ieee.

[29] Khodairy, M. A., Abosamra, G. (2021). Driving behavior classification based on oversampled signals of smartphone embedded sensors using an optimized stacked-LSTM neural networks. IEEE Access, 9, 4957-4972.

[30] Du, Y., Raman, C., Black, A. W., Morency, L. P., Eskenazi, M. (2018). Multimodal polynomial fusion for detecting driver distraction. arXiv preprint arXiv:1810.10565.

[31] Angkititrakul, P., Kwak, D., Choi, S., Kim, J., PhucPhan, A., Sathyanarayana, A., Hansen, J. H. (2007). Getting start with UTDrive: Driver-behavior modeling and assessment of distraction for in-vehicle speech systems. In Eighth Annual Conference of the International Speech Communication Association.

[32] Craye, C., Rashwan, A., Kamel, M. S., Karray, F. (2016). A multi-modal driver fatigue and distraction assessment system. International Journal of Intelligent Transportation Systems Research, 14, 173-194.

[33] Li, N., Busso, C. (2014). Predicting perceived visual and cognitive distractions of drivers with multimodal features. IEEE Transactions on Intelligent Transportation Systems, 16(1), 51-65.

[34] Omerustaoglu, F., Sakar, C. O., Kar, G. (2020). Distracted driver detection by combining in-vehicle and image data using deep learning. Applied Soft Computing, 96, 106657.

[35] Deng, J., Dong, W., Socher, R., Li, L. J., Li, K., Fei-Fei, L. (2009, June). Imagenet: A large-scale hierarchical image database. In 2009 IEEE conference on computer vision and pattern recognition (pp. 248-255). Ieee.

[36] LeCun, Y., Bottou, L., Bengio, Y., Haffner, P. (1998). Gradient-based learning applied to document recognition. Proceedings of the IEEE, 86(11), 2278-2324.

[37] Chowanda, A. (2021, October). Spatiotemporal Features Learning from Song for Emotions Recognition with Time Distributed CNN. In 2021 1st International Conference on Computer Science and Artificial Intelligence (ICCSAI) (Vol. 1, pp. 407-412). IEEE..

[38] Xu, J., Huang, F., Zhang, X., Wang, S., Li, C., Li, Z., He, Y. (2019). Sentiment analysis of social images via hierarchical deep fusion of content and links. Applied Soft Computing, 80, 387-39

[39] Montaha, S., Azam, S., Rafid, A. R. H., Hasan, M. Z., Karim, A., Islam, A. (2022). Timedistributed-cnn-lstm: A hybrid approach combining cnn and lstm to classify brain tumor on 3d mri scans performing ablation study. IEEE Access, 10, 60039-60059.