A Project Report On

Real-time Monitoring and Detection of Distracted Driving using Deep Neural Networks

Submitted in partial fulfillment of the requirement for the 8^{th} semester

Bachelor of Engineering

in

Computer Science and Engineering

DAYANANDA SAGAR COLLEGE OF ENGINEERING

(An Autonomous Institute affiliated to VTU, Belagavi, Approved by AICTE & ISO 9001:2008 Certified)

Accredited by National Assessment & Accreditation Council (NAAC) with 'A' grade

Shavige Malleshwara Hills, Kumaraswamy Layout, Bengaluru-560078

Submitted By

K Sudheevar Reddy 1DS19CS065

Kiran N Deshmukh 1DS19CS070

Lokesh M 1DS19CS081

Prathik Raj RC 1DS19CS117

Under the guidance of

Dr. Kavitha K S

Professor, CSE , DSCE

Mr. Yash Shah

Co-guide - Industry

2022 - 2023

Department of Computer Science and Engineering

DAYANANDA SAGAR COLLEGE OF ENGINEERING

Bangalore - 560078

VISVESVARAYA TECHNOLOGICAL UNIVERSITY

Dayananda Sagar College of Engineering

(An Autonomous Institute affiliated to VTU, Belagavi, Approved by AICTE & ISO 9001:2008 Certified)

Accredited by National Assessment & Accreditation Council (NAAC) with 'A' grade

Shavige Malleshwara Hills, Kumaraswamy Layout, Bengaluru-560078

Department of Computer Science & Engineering

CERTIFICATE

This is to certify that the project entitled **Real-time Monitoring and Detection of Distracted Driving using Deep Neural Networks** is a bonafide work carried out by **K Sudheevar Reddy [1DS19CS065], Kiran N Deshmukh [1DS19CS070], Lokesh M [1DS19CS081]** and **Prathik Raj RC [1DS19CS117]** in partial fulfillment of 8th semester, Bachelor of Engineering in Computer Science and Engineering under Visvesvaraya Technological University, Belgaum during the year 2022-23.

Dr. Kavitha K S	**Dr. Ramesh Babu D R**	**Dr. B G Prasad**
(Guide)	Vice Principal & HOD	Principal
Prof. CSE, DSCE	CSE, DSCE	DSCE
Signature:.....	Signature:.....	Signature:.....

Name of the Examiners:	Signature with date:
1.....	.....
2.....	.....

Acknowledgement

We are pleased to have successfully completed the project **Real-time Monitoring and Detection of Distracted Driving using Deep Neural Networks**. We thoroughly enjoyed the process of working on this project and gained a lot of knowledge doing so.

We would like to take this opportunity to express our gratitude to **Dr. B G Prasad**, Principal of DSCE, for permitting us to utilize all the necessary facilities of the institution.

We also thank our respected Vice Principal, HOD of Computer Science & Engineering, DSCE, Bangalore, **Dr. Ramesh Babu D R**, for his support and encouragement throughout the process.

We are immensely grateful to our respected and learned guide, **Dr. Kavitha K S**, Professor CSE, DSCE and our co-guide **Mr. Yash Shah**, Alumnus, CSE, DSCE for their valuable help and guidance. We are indebted to them for their invaluable guidance throughout the process and their useful inputs at all stages of the process.

We also thank all the faculty and support staff of Department of Computer Science, DSCE. Without their support over the years, this work would not have been possible.

Lastly, we would like to express our deep appreciation towards our classmates and our family for providing us with constant moral support and encouragement. They have stood by us in the most difficult of times.

K Sudheevar Reddy 1DS19CS065
Kiran N Deshmukh 1DS19CS070
Lokesh M 1DS19CS081
Prathik Raj RC 1DS19CS117

Abstract

As the prevalence of sophisticated onboard systems like infotainment platforms and mobile phones continues to rise, there has been a corresponding surge in driver distractions. These distractions pose a significant risk, leading to lane deviations and reduced attentiveness while driving, thereby contributing to road accidents. Consequently, there is a pressing need to explore and create effective approaches to address this issue. In this research, we developed an efficient model for driver distraction detection by restructuring and modifying the dataset, utilizing the YOLOv8 Object Detection model for cell phone usage identification, and leveraging the VGG19 model for detecting other distractions. To optimize the performance of our model, we conducted an extensive hyperparameter tuning process, exploring different learning rates, batch sizes, and optimizer configurations. This iterative refinement allowed us to find the optimal set of hyperparameters that maximized the model's accuracy and convergence speed. Furthermore, we conducted rigorous error analysis to identify common misclassifications and sources of false positives, enabling us to fine-tune the model and mitigate potential limitations. By addressing these challenges head-on, we were able to significantly enhance the model's robustness and overall performance in detecting driver distractions. With data augmentation techniques and training on the augmented dataset, we achieved a notable validation accuracy of approximately 79%, showcasing the model's proficiency in accurately detecting driver distractions. Comparisons with other models demonstrated that VGG19 exhibited superior stability and accuracy, further solidifying the effectiveness of our approach. Our findings highlight the potential of our model to enhance safety and driver assistance systems.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 The Problem

Road traffic accidents present a grave public safety and health peril, constituting one of the primary causes of untimely deaths across various age groups, notably among individuals aged 15-29. Astonishingly, the World Health Organization (WHO) estimates that each year, these accidents claim the lives of approximately 1.2 million drivers and occupants, while permanently incapacitating another 50 million individuals. Alarming as it may be, road accidents have ascended to the tenth position as a leading global fatality contributor, with projections indicating a move up to the fifth rank by 2030. Strikingly, India bears the brunt of this worldwide surge in traffic-related deaths.

Distracted driving stands out as one of the major causes for these incidents. At its core, driver distraction epitomizes the perilous practice of operating a motor vehicle while being immersed in alternative activities, often entailing smartphone usage or engagement with ancillary devices, such as car stereos, and other correlated operations.

Given the alarming statistics and the devastating consequences of road traffic accidents, it is imperative to develop effective solutions to detect and address distracted driving. Detecting distractions in real-time can help identify risky behavior and provide timely interventions to mitigate the potential for accidents. By implementing advanced technologies and innovative approaches, we can enhance road safety, reduce fatalities, and create a safer environment for all road users.

1.2 Real World Applications

Coming up with a solid mechanism to detect distracted driving could prove to be highly advantageous.

1. Real-time detection of driver distraction enables autonomous vehicles to allocate computing resources efficiently, ensuring that critical functions receive prioritized

attention, enhancing overall system performance.

2. By integrating distracted driving detection into Advanced Driver Assistance Systems (ADAS), autonomous vehicles can adaptively adjust their assistance levels based on the driver's level of engagement, promoting safer driving practices.

3. Autonomous vehicles equipped with distracted driving detection capabilities can assist in ensuring compliance with regulations and standards related to driver attentiveness, reinforcing accountability and responsible autonomous driving practices.

4. The development and implementation of distracted driving detection in autonomous vehicles present new avenues for research and innovation, encouraging advancements in sensor technology, artificial intelligence algorithms, and autonomous vehicle systems.

5. Insurance companies can leverage the data and statistics collected through such a system to better assess driver behavior and tailor insurance premiums accordingly.

1.3 Organisation of the Project

The project report is organized as follows:

In Chapter (2) we discuss the problem statement and the proposed solution. We also take a look at the systems that exist today and the drawbacks they face.

Chapter (3) takes a more in-depth look at various hardware and software based solutions that exist, with a survey on existing literature available.

Chapter (4) looks at the architecture of the proposed solution with an overview of the system design, utilizing system block diagrams and data flow diagrams.

Chapter (5) dives into the Implementation of the solution, by describing the hardware and software requirements, along with dataset descriptions and implementation details.

Chapter (6) describes our testing process, while Chapter (7) looks at our experimentation process and the obtained results.

Chapter (8) summarizes our findings and concludes the paper.

Chapter 2

Problem Statement and Proposed Solution

2.1 Problem statement

To develop and implement a robust deep learning model to detect distracted behaviour while driving in real-time.

2.2 Existing Systems

Distinguishing distracted drivers typically incorporates two main methods. The initial method revolves around utilizing wearable sensors to gauge brain signals, heartbeat data, and muscular activity. However, this approach may sometimes prove to be expensive or involve complex hardware, requiring substantial user participation. The alternative method involves employing camera vision systems to identify and categorize distractions. These systems commonly leverage deep learning techniques to extract features and perform classification tasks. While the first method can detect cognitive distractions, the second method is effective in identifying manual and visual distractions. Therefore, combining these two approaches seems logical to ensure effective detection of various types of distractions.

2.2.1 Deep Learning and Computer Vision based systems

A new approach to detect manual driver distraction has been introduced by L. Li and colleagues [1]. The researchers divided the method into two parts. Firstly, using RGB images captured by the driver's onboard camera, the first module simultaneously predicts and locates the boundaries of the driver's right hand and right ear. These bounding boxes serve as input for the second module, which determines the type of distraction. The first module employs YOLO, a deep neural network for object detection, to detect the right ear and right hand using the image input. The second module utilizes a multilayer perceptron that takes regions of interest (ROIs) as input to predict the type of distraction. The

proposed method considers the outputs of both modules and their overlap to assess the driver's condition and subsequently predict the type of distraction. The results indicate that the suggested approach effectively identifies common driving scenarios, touch screen usage, and phone calls, achieving F1 scores of 0.84, 0.69, and 0.82, respectively. The overall F1 score for distraction detection is reported as 0.74.

Qin et al. [2] have addressed the need for an optimized framework that balances parameter efficiency, accuracy, and speed. They propose the D-HCNN model, which introduces a decreasing filter size to meet these requirements effectively. In comparison to other models, the D-HCNN model utilizes only 0.76M parameters, significantly reducing the parameter count. To address overfitting, the model incorporates dropout, batch normalization, and L2 weight regularization techniques. Additionally, HOG feature pictures are employed to focus solely on the driver's posture while eliminating background noise, further enhancing the model's performance. The D-HCNN model follows a unique approach, starting with a larger convolution filter size to capture a wider receptive field. As the network deepens, the receptive field narrows, and the convolution filter size reduces, leading to fewer network parameters. This design allows for finer tuning and faster processing times. Experimental evaluations on datasets AUCD2 and SFD3 demonstrate the effectiveness of the D-HCNN model, achieving impressive accuracy ratings of 95.59% and 99.87% respectively. The model successfully addresses the requirements for an optimized framework with minimal parameters, high accuracy, and efficient processing, making it a promising solution in the field.

Chen et al. [3] developed a driver behavior analysis system utilizing a spatial stream ConvNet, similar to the VGG-16 model, to extract spatial features. They also employed a temporal stream ConvNet to capture the driver's motion information, along with a Fusion network. The spatial stream ConvNet was pre-trained using recordings of activities from the UCF-101 dataset, which clearly depicted hand movements. Pretraining was performed on the extensive ImageNet dataset. The fusion network processed a feature map of size 7x7x256 pixels obtained from the final convolution layer of both ConvNets. The accuracy rates achieved by the researchers were 49.21% for the Spatial Stream ConvNet,

65.08% for the Temporal Stream ConvNet, and approximately 68.25% for the fusion result.

2.3 Proposed Solution

The proposed work involves several key steps, including modifying the dataset to align it with real-world driver distraction scenarios, training advanced deep learning architectures using transfer learning and fine-tuning strategies, analyzing the obtained results, and testing the trained models for generalizability and robustness. The modifications made to the dataset were meticulous and aimed to ensure that it closely resembled real-world driver distraction scenarios. We employed advanced deep learning architectures that leveraged transfer learning and fine-tuning strategies to train their models. We then analyzed the obtained results in detail to evaluate the performance and effectiveness of their approach. Finally, they tested their trained models rigorously to assess their generalizability and robustness.

2.3.1 Object Detection using YOLOv8

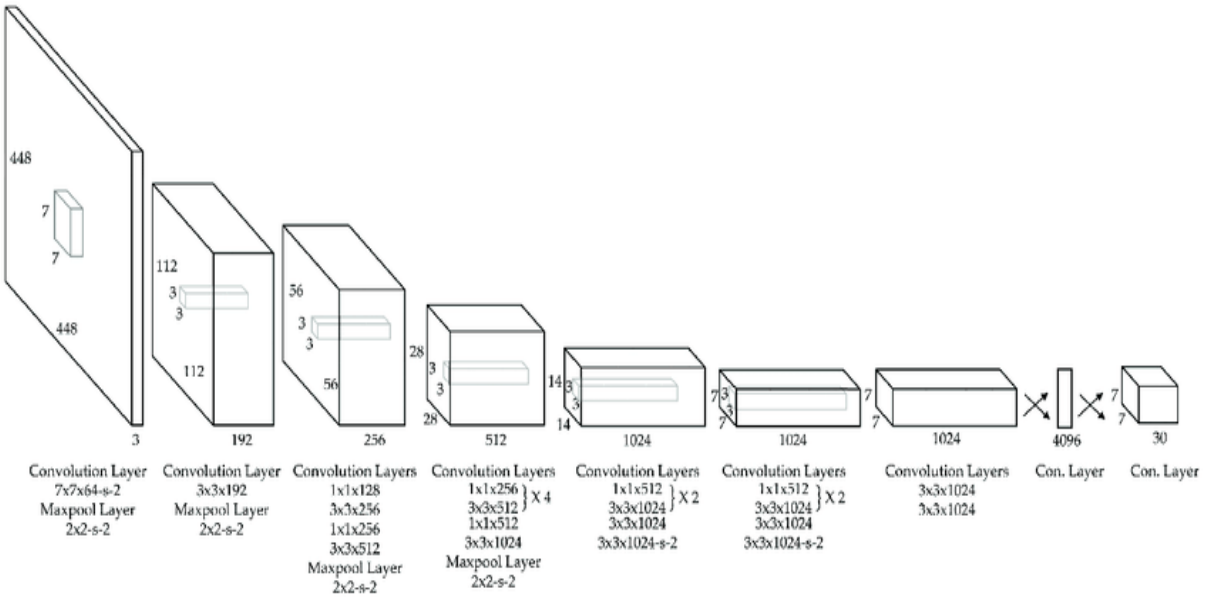

Figure 2.1: YOLOv8 Architecture

The YOLOv8 model, an advanced deep learning architecture (Fig. 2.1) widely acclaimed for its object detection capabilities, played a pivotal role in our project. YOLOv8, which stands for "You Only Look Once," is renowned for its exceptional speed and accuracy in swiftly detecting objects within images and videos. In our research, we harnessed the power of YOLOv8 to specifically identify instances of cell phone usage in our dataset.

By utilizing YOLOv8, we aimed to address two significant challenges commonly encountered in object detection tasks: false positives and false negatives. False positives occur when the model mistakenly detects an object that is not actually present in the image, while false negatives arise when the model fails to detect an object that is present. These errors can significantly impact the reliability and trustworthiness of the results.

Through meticulous tuning and optimization, we minimized both false positives and false negatives, thereby enhancing the accuracy and precision of our cell phone detection methodology. By doing so, we ensured that our findings were robust and trustworthy. YOLOv8's state-of-the-art capabilities (Fig 2.2) enabled us to achieve superior performance and effectively mitigate the challenges associated with object detection, providing us with reliable and dependable results.

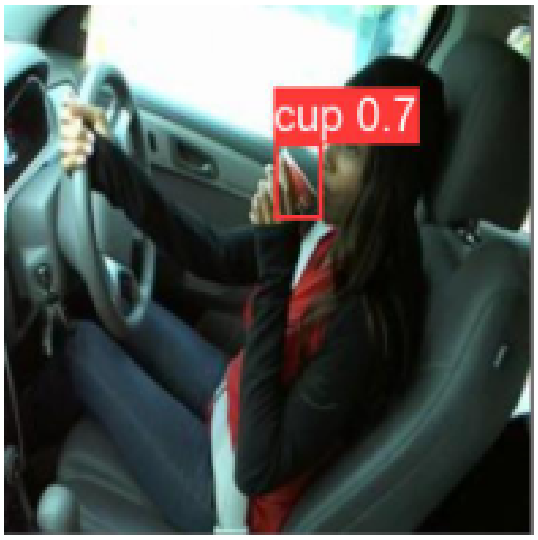

Figure 2.2: Working of YOLOv8

2.3.2 Feature Extraction and Classification using VGG19

One of the key components we employed was the VGG19 deep learning model. VGG19, renowned for its convolutional neural network architecture, has been extensively trained on a diverse range of images to effectively identify various objects and extract intricate visual features. When considering different models such as VGG16 and MobileNet, we specifically chose VGG19 due to its deeper architecture, which enables it to capture more complex visual patterns and intricate details crucial for accurate detection of distracted driving.

Within the VGG19 model, the convolutional neural network (CNN) assumes a crucial role in extracting pertinent features from input images. Comprising a sequence of convolutional layers, the CNN conducts successive convolutions on the image input, capturing diverse spatial information levels. These convolutional layers are followed by pooling layers, which reduce the spatial dimensions of the feature maps while preserving the essential features.

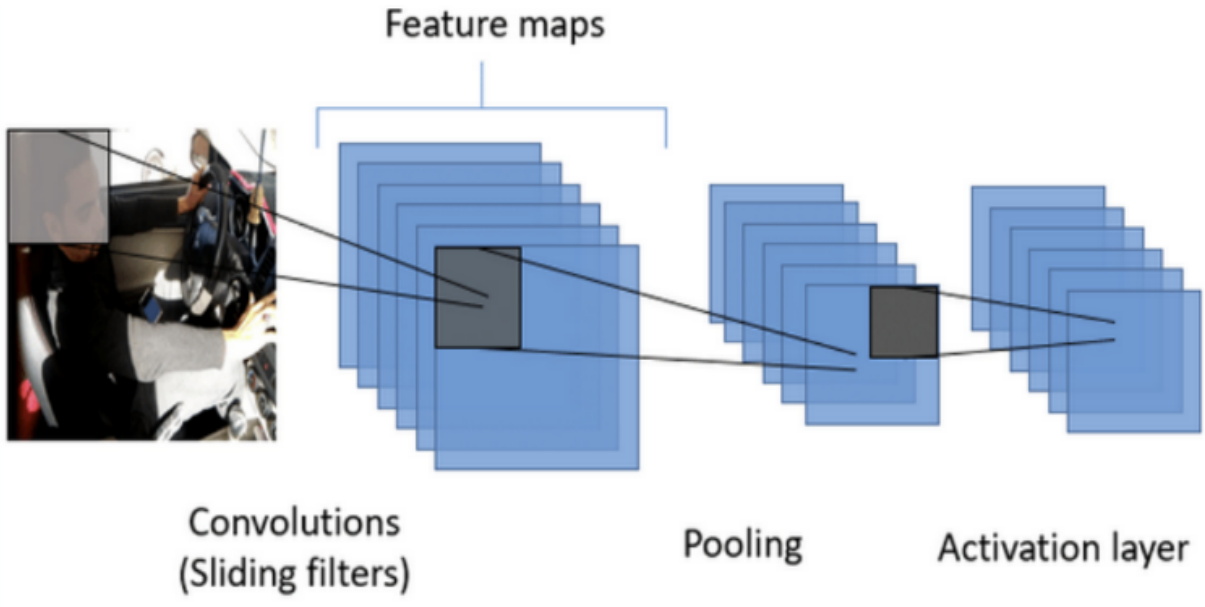

Figure 2.3: Working of CNN

VGG19 adopts a deep architecture with 19 layers, encompassing convolutional layers, pooling layers, and fully connected layers. The convolutional layers undertake the task

of acquiring distinct image features, including edges, textures, and shapes, through the application of convolutional filters. Subsequently, the pooling layers downsample the feature maps, focusing on the most salient information.

By stacking multiple convolutional and pooling layers (Fig 2.3), VGG19 progressively learns increasingly abstract and higher-level features. This hierarchical feature representation empowers the model to discern intricate patterns and intricate relationships embedded within the input images. These acquired features are subsequently fed into the fully connected layers, which perform classification tasks based on the extracted features.

By harnessing the potency of a deep CNN architecture, such as VGG19 (Fig 2.4), we can exploit the benefits of hierarchical feature learning. Through this intricate process, the model becomes adept at identifying consequential visual patterns linked to distracted driving, thereby bolstering the precision and dependability of our system.

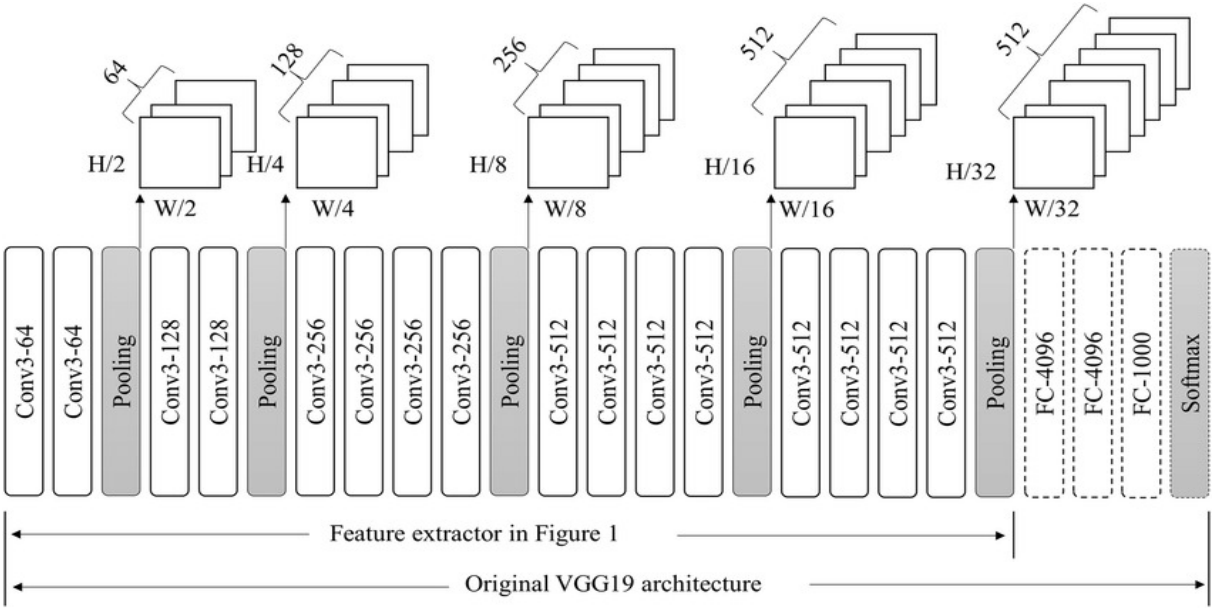

Figure 2.4: VGG19 Architecture

One of the distinguishing features of VGG19 is its superior generalization capabilities, allowing it to perform well on unseen data. This aspect is particularly advantageous when dealing with real-world scenarios where new and diverse instances of distracted driving

may arise. By harnessing the power of VGG19 within our system, we were able to achieve enhanced accuracy and stability in our predictions. This is of utmost importance for real-time monitoring and timely detection of distracted driving incidents, ultimately contributing to improved driver safety.

The successful integration of VGG19 into our project serves as a testament to the effectiveness of deep learning techniques in tackling critical real-world challenges. By leveraging the capabilities of advanced neural network architectures like VGG19, we can pave the way for more robust and reliable systems for promoting safer driving practices.

2.4 Neural Network following the Feature Extraction

The utilization of the VGG19 model serves as a fundamental framework for constructing a profound neural network designed to identify occurrences of driver distraction. We have customized the architecture of VGG19 to align with the specific requirements of our task.

The adapted VGG19 model encompasses a total of 16 convolutional layers and 3 fully connected layers. Initially, the input images traverse the convolutional layers, which are responsible for extracting distinctive features from the images. Subsequently, the output produced by the convolutional layers is flattened and directed through the fully connected layers, which facilitate the classification process based on the extracted features.

To refine the learning capabilities of the model and optimize its adaptability to our dataset, we have introduced modifications to the structure of the fully connected layers. Precisely, two additional fully connected layers have been incorporated, each consisting of 512 neurons, and a dropout layer with a rate of 0.5 has been inserted between them (Fig 2.5). These alterations effectively mitigate the risk of overfitting and enhance the model's capacity for generalization.

Layer	Output Shape	Param #
VGG19	(None, 7, 7, 512)	20,024,384
Flatten	(None, 25088)	0
Dense	(None, 512)	12,845,568
Dropout	(None, 6)	3,078
Total params	(None, 6)	33,873,030
Trainable params		33,873,030
Non-trainable params		0

Figure 2.5: Model Summary of VGG19 with modified Fully Connected Layers

2.5 System Requirements

- Processor : 8 Core, 16 Thread Processor
- Memory : Minimum of 16 GB RAM for computation
- Video Memory : Minimum of 3 GB of VRAM for Object Detection and an extra 2 GB for feature extraction and the ANN.

Chapter 3

Literature Survey

Various approaches exist for detecting driver distractions, which can be broadly categorized into two groups:

- Hardware solutions
- Software solutions

Hardware solutions involve the use of sensors either attached to vehicles or installed at junctions. These sensors capture data related to driver behavior, such as brain signals, heartbeat, or muscular activities, to identify instances of distraction. However, hardware solutions can be costly and may require complex hardware setups, which can be burdensome for users.

On the other hand, software solutions analyze video data to detect and classify different types of driver distractions. These solutions often make use of deep learning techniques, such as convolutional neural networks (CNNs), to extract features from the video frames and classify the distractions. By leveraging the power of deep learning, software solutions can detect cognitive, manual, and visual distractions effectively.

It is worth noting that the two approaches can complement each other. By combining wearable sensors with camera vision systems and employing advanced classification algorithms, it is possible to achieve more efficient and comprehensive detection of driver distractions.

3.1 Hardware Solutions

Wang et al. [4] have introduced an innovative computational approach to detect driver distraction early on by analyzing electroencephalographic (EEG) signals that indicate brain activity. While previous studies have primarily focused on distinguishing distracted and non-distracted states, this framework aims to predict the onset and conclusion of driver distraction. During the study, EEG signals were recorded while participants were

asked to drive normally, and the authors developed a customized prediction model that utilizes these EEG signals as input. In order to assist inattentive drivers, the authors also developed a navigation system that provides advance vocal route instructions when the model predicts driver inattention. This allows drivers to avoid looking at maps and instead focus on the road. The proposed model achieved an overall accuracy of 81% for predicting the start of a map viewing session and 70% for predicting its end.

Ben Ahmed and colleagues' [5] study recommends employing contemporary smartphone sensors like accelerometers and gyroscopes to identify driving distractions like texting, reading, and conversing on the phone. Using a driving simulator featuring a steering wheel, pedals, a traffic-simulating broad screen, and features to simulate different weather conditions like day, night, fog, and rain, they carried out an experiment. On the simulator, participants carried out various tasks while the smartphone's sensors captured data. They used a machine learning technique called the Random Forest Classifier combined with a 10-fold cross-validation strategy to address the classification problem utilising this sensory data. The experiment achieved precision of 85%, recall of 84%, and an F-measure of 87% by merging the data from both sensors.

In a study by Sean L. Gallahan and colleagues [6], driver assistance technologies such as Lane-Departure Warning and Blind Spot Detection systems were proposed as a means to enhance driving safety. The researchers utilized the Microsoft Kinect, a device known for its accurate 3D motion capture capabilities, to detect driving-related actions. Specifically, they employed skeletal tracking and facial tracking algorithms developed in C++ to identify various driving motions. To evaluate the system, they attached a Kinect to a VDSL simulator and implemented an auditory alert sound whenever driver distraction was detected. The simulation results indicated a high level of effectiveness, with a 100% success rate in identifying drivers reaching for objects, 33% success rate for detecting phone usage while driving, and 66% success rate for detecting drivers looking at objects outside the vehicle.

3.2 Software Solutions

Drive-Net is a supervised learning algorithm proposed by Mohammed S. Majdi et al. [7] for detecting driver distraction. It combines a random decision forest with a convolutional neural network (CNN) to classify photos of drivers. The authors chose the U-Net architecture as the basis for the CNN model instead of other architectures like Alex-Net because it captures the surrounding context of the object. They made modifications to the U-Net model to make it suitable for their application. The performance of Drive-Net was compared to two other popular machine learning techniques: recurrent neural network (RNN) and multilayer perceptron (MLP). The CNN model consists of two stacked convolution layers, followed by a max-pooling layer and a ReLU layer. The model was trained using various driving scenarios, including distracted driving scenarios. The output of the CNN model is used as input for the Random Forest model to predict driver distraction. Random Forest is employed to prevent overfitting of the CNN model. The suggested model, Drive-Net, was evaluated using a publicly available dataset of pictures from the Kaggle contest, and its accuracy was compared to that of RNN and MLP, achieving an accuracy of 97%.

In a study conducted by Jing Wang et al. [8], a framework was proposed to improve the performance of a model when working with a limited dataset. The framework involved using data augmentation techniques to artificially expand the dataset. The flow of events in this framework is as follows: Firstly, the driving operation key areas were analyzed and identified using the class activation mapping method. Then, a new subdataset called the driving operation area (DOA) dataset was generated by augmenting the dataset using the faster R-CNN model, based on the AUC dataset. Among three models tested, namely YOLO, SSD, and R-CNN, the R-CNN model yielded the highest accuracy of 0.6271. In the third step, a classification model was developed to process data from both the AUC dataset and the newly created DOA dataset. Several classification models, including InceptionV4, Xception, and AlexNet, were experimented with to compare their performance and select the best model. Finally, the trained models were tested on the researchers' dataset to evaluate the classification accuracy. The results showed that using the data augmentation approach, the Xception model achieved a classification

accuracy of 96.97%.

Atran and colleagues (reference [9]) present a computer vision approach for identifying instances of using a cell phone while driving. They utilize a setup consisting of a near infrared camera system positioned on the car's front windshield. The method involves two modules. The first module utilizes the deformable part model (DPM) to locate the driver's face area, while the second module employs a local aggregation model to classify a region of interest (ROI) around the face, specifically targeting mobile phone usage. By combining the features extracted from these modules, they feed the data into an SVM Classifier. The proposed model achieves an accuracy of 86% in detecting instances of mobile phone usage, which is crucial for predicting driver distraction.

A study conducted by Matti Kutila and colleagues [10] presents a tool designed to track driver distraction. The tool combines stereo vision and lane tracking data and applies rule-based and support-vector machine (SVM) classification approaches to determine the level of visual and cognitive burden experienced by the driver. The study shows promising results in identifying cognitive distractions in passenger cars, achieving an accuracy rate of 86%. However, the results for the truck application are less impressive, with an accuracy rate of 68%. Further experiments were conducted to improve the accuracy, and it was found that removing the detection of cognitive distraction in a city context led to an increase in the success rate. This is because it is more challenging to identify cognitive distractions in urban areas where driving demands are higher due to frequent maneuvering.

Learning from the existing work, in our investigation, we have conducted extensive modifications to the preexisting dataset, specifically the Statefarm Dataset, in order to enhance its suitability for our research objectives. These modifications will be thoroughly elucidated in the methodology section. Notably, we have employed the YOLOv8 object detection model, a highly acclaimed and sophisticated framework, to effectively identify instances of cell phone usage. This approach mitigates the occurrence of misclassifications and ensures precise and accurate detection of cell phone-related distractions within

the dataset. Furthermore, we have harnessed the power of the VGG19 model, a widely renowned deep neural network architecture, to detect and classify other forms of distractions depicted in the dataset. By leveraging these cutting-edge models, we aspire to achieve a robust and accurate identification of various types of distractions, thereby contributing to a comprehensive analysis of driver behavior.

Chapter 4

Architecture and System Design

4.1 Software Overview

4.1.1 Architecture Diagram

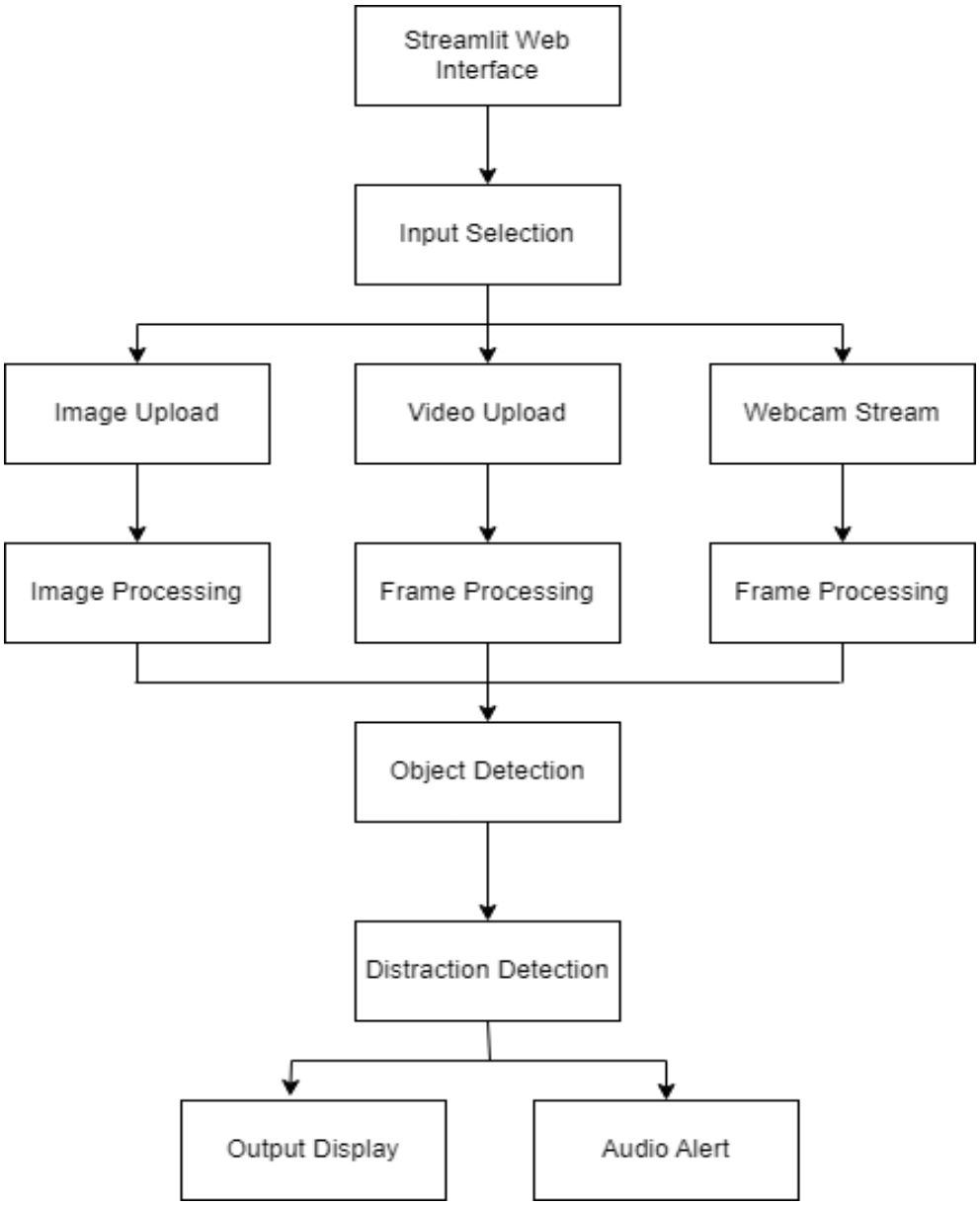

Figure 4.1: Architecture Diagram

The system architecture is illustrated in Fig. 4.1. This architecture comprises various components that work together to facilitate the driver distraction detection system. The central component is the Streamlit web interface, which serves as the user-facing part of the system.

Within the Streamlit interface, users can select the type of input they want to use for distraction detection. There are three options available: Image Upload, Video Upload, and Webcam. The Image Upload component allows users to upload still images for analysis, while the Video Upload component enables the upload of video files. The Webcam component allows real-time video streaming from the user's webcam as an input source.

Each of these input components is responsible for capturing the respective input data and passing it to the subsequent processing stages.

The Image Processing, Video Processing, and Webcam components take the captured input and preprocess it to enhance the quality and ensure compatibility with the subsequent stages. These processing stages may involve tasks such as resizing, normalization, and format conversion to ensure consistency and optimal performance.

The processed input data is then forwarded to the Object Detection component. This component utilizes the YOLO (You Only Look Once) model, specifically the YOLOv8 model, for detecting objects of interest within the input data. Object detection allows the system to identify and localize relevant objects, such as cell phones, within the frames.

After object detection, the system proceeds to the Distraction Detection stage. Here, a separate model, namely the VGG19-based small model, is loaded to perform classification and determine the type of distraction exhibited by the driver. The model has been trained on different distraction classes, including activities like using cell phones, operating the radio, drinking, reaching behind, hair and makeup, and talking to passengers.

Once the distraction is classified, the system can provide real-time feedback to the user. This feedback may include visual indicators overlaid on the input frames, displaying the detected distractions and their corresponding labels. Additionally, an audio alert is triggered when distractions deemed unsafe, such as using a cell phone while driving, are detected. The audio alert aims to draw the driver's attention to the potentially hazardous behavior.

The Output display component receives the processed and analyzed data from the Distraction Detection stage and presents it to the user within the Streamlit interface. This output display showcases the frames with overlaid indicators and provides relevant information, such as the current frames per second (FPS) rate, to keep the user informed about the system's performance.

In summary, the system architecture encompasses a web interface, various input options, preprocessing stages, object detection, distraction classification, and output display with audio alerting. This comprehensive architecture enables real-time driver distraction detection, enhancing safety and promoting responsible driving behavior.

4.1.2 Data Flow Diagram

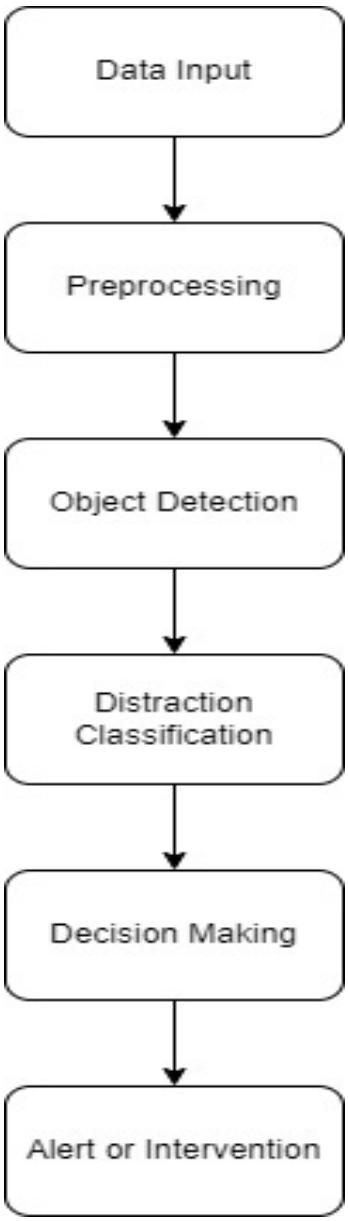

Figure 4.2: Data Flow Diagram

As Fig. 4.2 illustrates, the flow of data in the project happens in the following steps:

1. Data Input: The system receives input data in the form of a video stream captured from a camera mounted inside the vehicle. This video stream provides a continuous feed of the driver's perspective.

2. Preprocessing: The incoming video frames are preprocessed to optimize them for

input to the detection models. This preprocessing step involves resizing the frames and normalizing them.

3. Object Detection: The preprocessed video frames are fed into the YOLOv8 Object Detection model. The YOLOv8 model processes each frame and identifies any instances of cell phone usage by detecting and localizing the presence of cell phones in the driver's field of view.

4. Distraction Classification: The VGG19 model analyzes the detected regions of interest and classifies them into different distraction categories..

5. Decision Making: Based on the output of the VGG19 model, the system can make informed decisions about the level of distraction exhibited by the driver. This can include determining whether a distraction is present, identifying the type of distraction, and assessing the severity or potential risk associated with it.

6. Alert or Intervention: If the system detects a significant distraction that poses a potential danger to the driver and others on the road, appropriate alerts or interventions can be triggered. These alerts can take the form of visual cues, audible warnings, or haptic feedback, depending on the design of the driver assistance system.

4.1.3 System Design

The system design (Fig. 4.3) of our project revolves around detecting driver distractions in real-time. The flow of the system can be outlined as follows:

1. Input Acquisition: The system acquires input from a webcam, video file, or image source. This input represents the driver's perspective.

2. Object Detection using YOLOv8: The acquired input is processed through the YOLOv8 Object Detection model. The YOLOv8 model focuses on identifying instances of cell phone usage in the driver's field of view.

3. Cell Phone Usage Detection: The YOLOv8 model analyzes the input and identifies whether cell phone usage is present. If cell phone usage is detected, the system

generates an output prediction on the video frame, indicating the presence of cell phone usage.

4. Distraction Classification using VGG19: In cases where cell phone usage is not detected, the input is then passed to the VGG19 model for further analysis. The VGG19 model classifies the input and identifies other distractions such as eating, drinking, smoking, or interacting with objects.

5. Distraction Prediction: Based on the VGG19 model's classification, the system generates an output prediction on the video frame. This prediction highlights the detected distractions or provides an overall assessment of the driver's state, in the absence of cell phone usage.

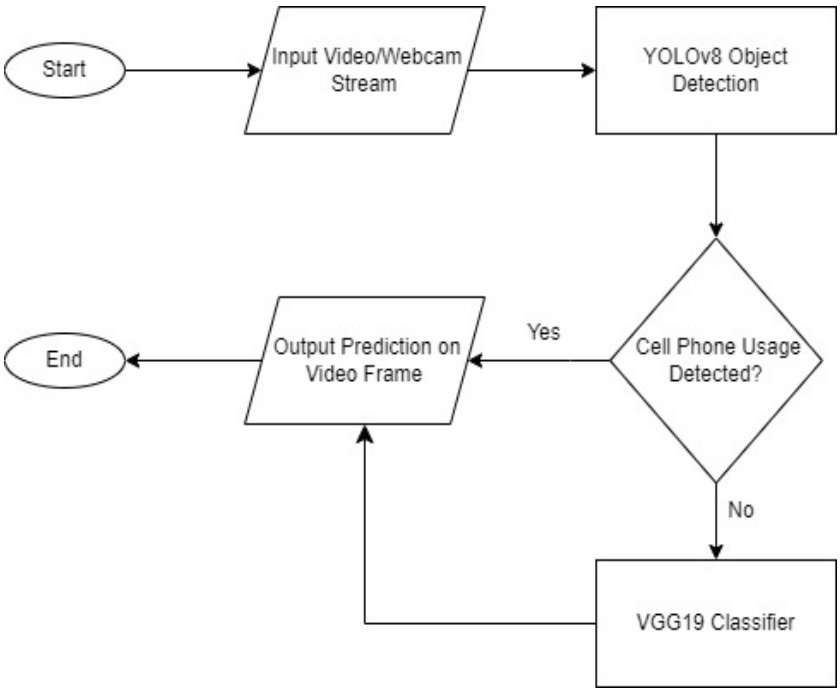

Figure 4.3: System Design

The designed system aims to provide real-time detection and classification of driver distractions. By utilizing the YOLOv8 model for cell phone usage detection and the VGG19 model for other distractions, the system can accurately identify and predict distractions, contributing to enhancing driver safety and assisting in the prevention of potential accidents.

Chapter 5

Implementation

5.1 Implementation Platform

5.1.1 Hardware

1. Platform: Google Colab Virtual Machine for Training
2. CPU: Intel Xeon CPU for Training and Intel i7-8565U for Experiment
3. GPU: Tesla K80 for Training and Nvidia MX200 for Experiment
4. Memory: 13 GB RAM for Training and 16GB for Experiment
5. Video Memory: 12 GB VRAM for Training and 2GB VRAM for Experiment

5.1.2 Software

1. Operating System: Google Colab VM (Linux) for Training and Windows 11 for Experiment
2. Software Used: Tensorflow, OpenCV, Pygame
3. Programming Language: Python 3
4. Server: Streamlit

5.2 Development Steps

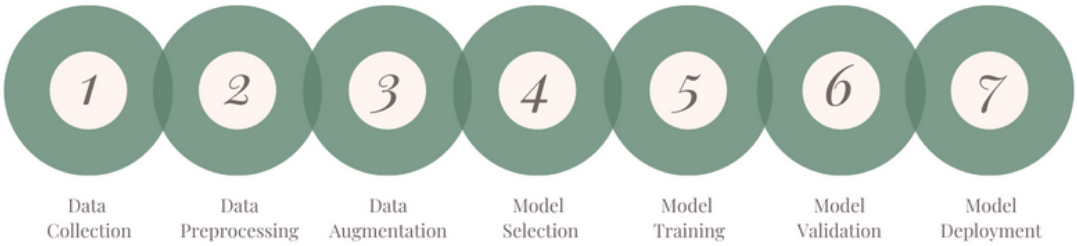

Figure 5.1: Development Steps

As Fig. 5.1 illustrates, the development steps of the project are:

1. Data Collection - Collecting a large and diverse dataset of driver behavior is crucial to ensure that the model can effectively distinguish between distracted and non-distracted driving examples. This dataset should encompass various scenarios, such as mobile phone usage, eating, and interacting with in-car systems, to capture a wide range of distractions that drivers may encounter.

2. Dataset Preparation - To improve the model's generalization and robustness, the collected dataset needs to be carefully organized into train, test, and validation sets. This division ensures that the model is trained on a diverse range of examples and evaluated on unseen data, enabling it to learn patterns and make accurate predictions on new instances of driver behavior.

3. Data Augmentation - Augmenting the dataset is essential to expand its size and increase its diversity. By applying random transformations, such as rotations, flips, and brightness adjustments, to the existing data, new variations are created, enabling the model to learn to recognize distractions from different perspectives and lighting conditions.

4. Model Selection - Choosing a suitable deep learning model architecture, such as a Convolutional Neural Network (CNN), is crucial for effective driver distraction detection. CNNs are known for their ability to capture spatial features and patterns, making them well-suited for image-based tasks like identifying distractions in video frames.

5. Model Training - The selected model is trained on the preprocessed and augmented dataset using optimization algorithms. Transfer learning can be leveraged by initializing the model with pre-trained weights from a related task, such as image classification, and fine-tuning it on the driver distraction dataset to speed up training and improve performance.

6. Model Validation - The trained model's performance is assessed using a separate validation dataset. By evaluating metrics such as accuracy, precision, recall, and F1 score, the model's effectiveness in detecting driver distractions can be measured.

Adjustments to the model architecture or hyperparameters can be made based on the validation results to optimize its performance.

7. Model Deployment - Integrating the trained model into a real-time driver assistance system allows for the detection of driver distractions on the fly. By processing video input from onboard cameras or other sensors, the model can continuously analyze the driver's behavior and provide timely alerts or interventions to enhance safety and mitigate the risk of accidents caused by distractions.

5.3 Dataset

In our research project, we encountered practical challenges with the dataset size and lack of diversity, which prompted us to incorporate the Statefarm dataset [6] (Fig. 5.2) to enrich our investigation into driver distraction detection. The Statefarm dataset consisted of approximately 22,000 images, providing a comprehensive collection of driving scenarios. We carefully divided the dataset into train, test, and validation sets, with around 17,000 images used for training and the remaining images allocated to the testing and validation sets. This distribution ensured a robust evaluation of our model's performance on unseen data.

Figure 5.2: Glimpse of the Dataset

To make the dataset more suitable for Indian driving conditions, we introduced a modification by horizontally flipping every image. This adjustment aimed to match the driving context prevalent in India, where vehicles are right-hand driven. By incorporating this adaptation, we aimed to enhance the relevance and accuracy of our model's training.

During our analysis, we identified the issue of data leakage when randomly splitting the dataset, which resulted in inflated validation accuracy scores. To address this, we adopted a novel dataset splitting approach based on driver IDs. This approach ensured that images of 18 drivers were exclusively included in the training set, while four drivers each were assigned to the testing and validation sets. By mitigating data leakage, we obtained a more reliable evaluation of our model's generalization capabilities, resulting in a realistic validation accuracy score of 79.53%.

Our modifications also involved the exclusion of redundant classes related to mobile phone usage. Instead, we employed a pretrained object detection model specifically designed for detecting mobile phone usage, enhancing the efficiency and accuracy of our approach.

Chapter 6

Testing

To evaluate the performance of our driver distraction detection system, we followed the following testing procedure:

1. Training Optimization: During the training phase, we experimented with different parameters and settings to optimize the performance of our model. We made adjustments to various aspects such as learning rate, batch size, optimizer, and regularization techniques. Each training run involved iterations over the dataset, with the goal of improving the model's accuracy, precision, and recall for driver distraction detection.

2. Input Preparation: For testing, we utilized a combination of unseen images and videos (Fig. 6.1) to assess the robustness and generalization capabilities of our model. We employed a Streamlit app as an interface to feed the test images and videos into the model. Additionally, we recorded a video of ourselves in a car while performing various distracted activities to simulate real-world scenarios and evaluate the system's performance.

3. Individual and Batch Testing: During testing, we fed the test images and videos to the model both individually and in batches. For individual testing, we provided one test image or frame from a video to the model at a time, allowing us to analyze the model's performance on a per-image basis. For batch testing, we grouped multiple test images or frames together and fed them as a batch to the model, enabling us to evaluate the model's performance in a more comprehensive manner.

4. Considerations and Settings: Throughout the testing phase, we considered various factors to ensure accurate and reliable evaluation. We ensured that the input images and videos were of the same format and resolution as the training data to maintain consistency. Moreover, we set the appropriate model parameters and configurations consistent with the training phase to maintain the integrity of the model's architecture and optimize performance.

5. Performance Analysis: After feeding the test images and videos into the model, we analyzed the model's predictions (Fig. 6.2) and performance. We recorded the model's outputs, including the detected driver distractions and associated confidence scores, to assess the accuracy and reliability of the system. We compared the model's predictions with the ground truth labels or manual annotations to determine the correctness of the detections.

6. Real-World Simulation: To further evaluate the system's effectiveness in real-world scenarios, we recorded a video of ourselves engaging in various distracted activities while driving. We fed this video to the model and analyzed its performance in detecting and classifying the distracted behaviors accurately.

Figure 6.1: Frame of an Input Video

Figure 6.2: Prediction Made on Test Input Frame

By following this testing procedure, we were able to thoroughly evaluate the performance of our driver distraction detection system. The combination of unseen images, videos, and real-world simulation allowed us to assess the robustness, generalization, and accuracy of the model in diverse scenarios.

Chapter 7

Experimentation and Results

7.1 Experimentation

Upon successful acquisition of the meticulously trained model, our focus shifted towards the practical implementation of a real-time experiment, wherein we leveraged a cutting-edge web camera module to enable seamless streaming of video input. This dynamic setup allowed for immediate frame analysis and accurate predictions, empowering us to delve deeper into the realm of real-time driver behavior monitoring.

However, it is crucial to acknowledge a noteworthy constraint that emerged during the implementation phase: the absence of a dedicated graphics processing unit (GPU). Without the computational prowess and parallel processing capabilities offered by a GPU, we were compelled to rely solely on the central processing unit (CPU) to handle the complex computational tasks. This inherent limitation posed significant challenges and impacted the system's overall performance.

The absence of a GPU resulted in considerable performance overhead, manifesting in maximum frame rates of approximately 2.5 to 3 frames per second (fps) for input videos and a slightly lower range of 1 to 1.5 fps for live webcam input. These modest frame rates fell short of our expectations and imposed notable constraints on the system's real-time responsiveness and efficiency.

To address this intrinsic limitation and unlock the true potential of our system, it became evident that access to a powerful GPU was imperative. By capitalizing on the parallel processing capabilities and optimized computations offered by a robust GPU, we could revolutionize the system's performance and elevate it to unprecedented heights. With a GPU at our disposal, we could significantly enhance the frame rates, enabling real-time video analysis with remarkable precision and reduced computational overhead.

Thus, the integration of a high-performance GPU stood as a pivotal step in our endeavor to optimize system functionality, ensuring that we could harness the full capabilities of our meticulously trained model. With enhanced computational power and improved parallel processing, our system could seamlessly handle the demanding tasks of real-time frame analysis and accurate prediction, paving the way for a comprehensive and insightful analysis of driver behavior in diverse scenarios.

Figure 7.1: Streamlit User Interface

To facilitate user interaction and seamless integration, we developed a user interface in the form of a Streamlit application as illustrated in Fig. 7.1 and Fig. 7.2. This application allowed users to either upload pre-recorded videos or utilize webcam input for real-time analysis, enhancing the usability and accessibility of our system.

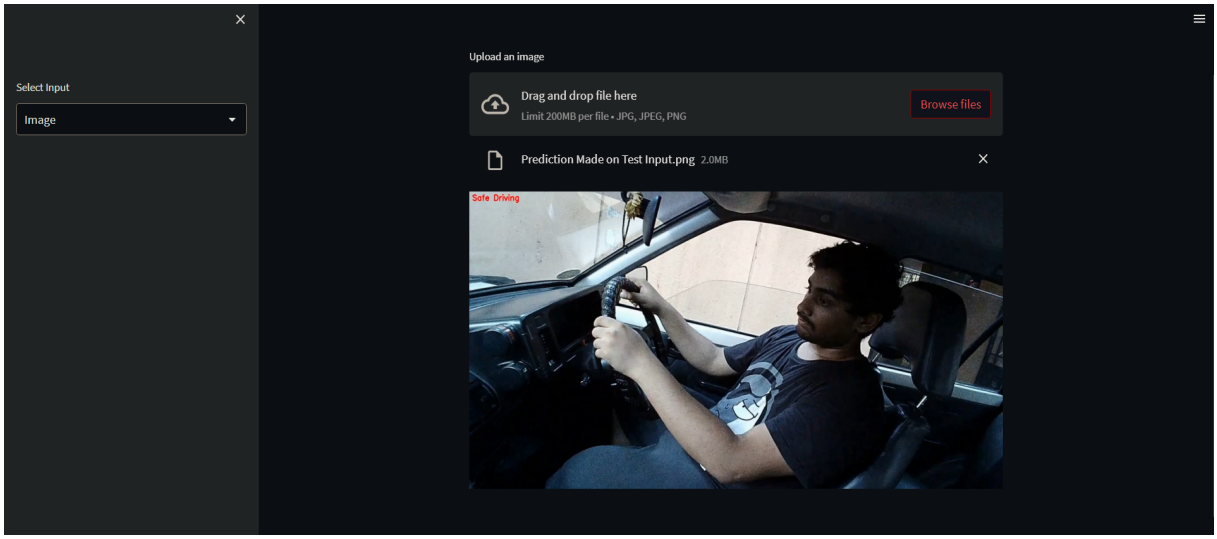

Figure 7.2: Output

7.2 Results

In the training phase, our model underwent training for 10 epochs, and the best-performing model based on validation accuracy was saved. The results obtained demonstrated an approximate validation accuracy of 79.53%. To provide a thorough analysis of the model's performance, we generated various visualizations, which are depicted in the attached figures.

Fig. 7.3 illustrates the training versus validation accuracy plot, showcasing the progression of accuracy values throughout the training epochs. This plot allows for a closer examination of the model's learning behavior and helps identify any potential overfitting or underfitting tendencies.

To assess the model's performance across different distraction classes, we created a confusion matrix, as depicted in Fig. 7.4. The confusion matrix provides a comprehensive overview of the model's ability to accurately classify instances into their respective distraction categories. Each row in the matrix corresponds to the true labels, while each column represents the predicted labels. Analyzing the values within the matrix enables

an evaluation of the model's performance in correctly identifying various types of distractions.

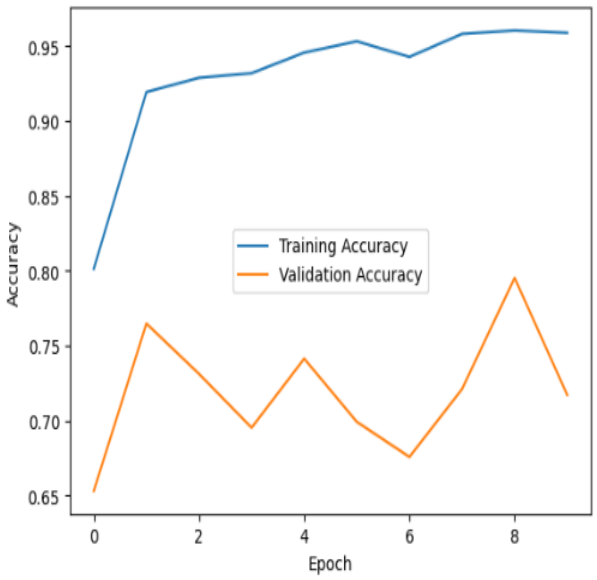

Figure 7.3: Plot of Progression of Training vs Validation Accuracy

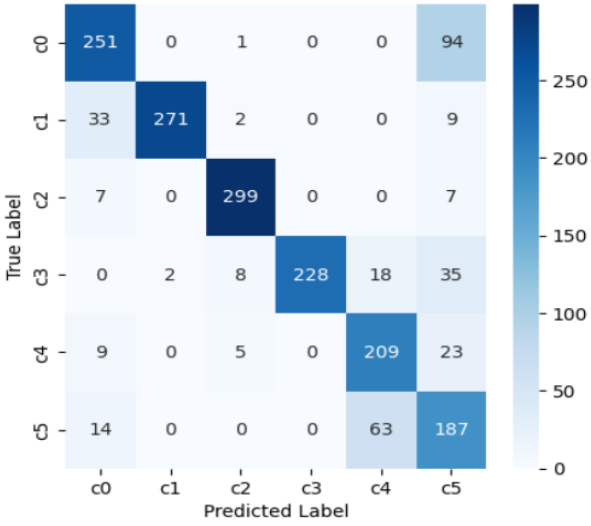

Figure 7.4: Confusion Matrix

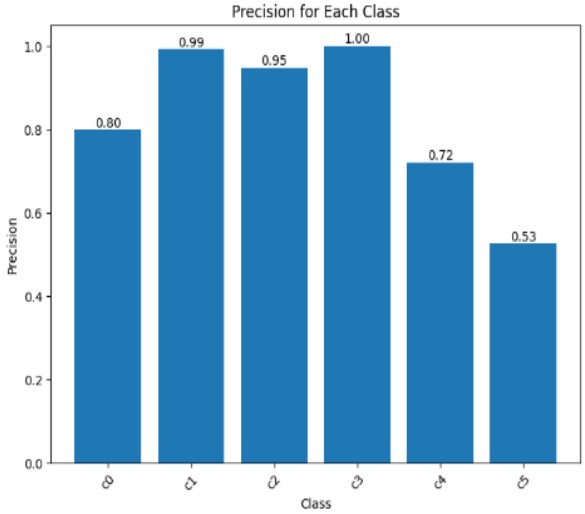

Figure 7.5: Plot of Precision for each class

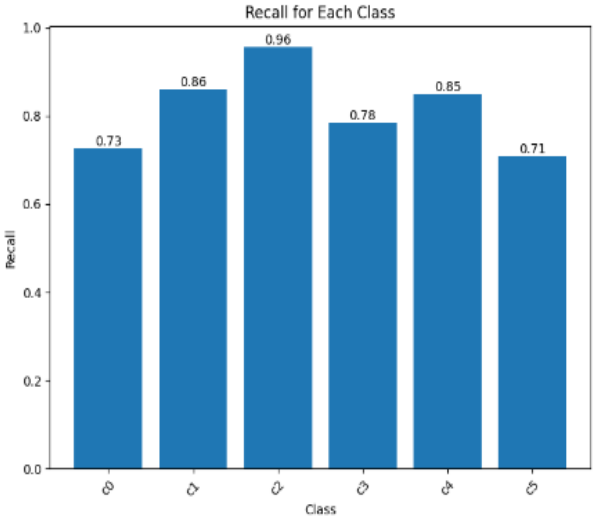

Figure 7.6: Plot of Recall for each class

Additionally, we generated bar graphs, as shown in Figures 7.5 and 7.6, to illustrate the precision and recall scores for each class. Precision measures the accuracy of positive predictions, while recall assesses the model's ability to correctly identify instances of a particular class. Examining these metrics for individual classes provides a detailed understanding of the model's performance across different types of distractions.

Figure 7.7: Predictions made by the model on the frames of an input video

Fig. 7.7 visually portrays the culmination of our model's predictive prowess as it meticulously analyzes and generates insightful predictions based on the individual frames extracted from an input video.

7.2.1 Comparison with Other Models

Table 7.1: Comparison of Model Performance

Model	**Validation Accuracy (%)**
MobileNet	~73%
VGG16	~76%
VGG19	~79%

Upon conducting a thorough comparison of multiple models (Table 7.1), namely VGG16, MobileNet, and VGG19, it became evident that the VGG19 model exhibited notable advantages over the other architectures. The VGG19 model consistently demonstrated superior performance in terms of accuracy rates throughout the training process, displaying remarkable stability in its predictions. This stability can be attributed to the deeper architecture of VGG19, which enables it to capture more intricate and abstract features from the input images.

Compared to the VGG16 model, the VGG19 model surpassed it in terms of validation accuracy, achieving a higher maximum validation accuracy of approximately 73%. This improvement can be attributed to the additional convolutional layers present in the VGG19 architecture. These extra layers allow for more complex and detailed feature extraction, enhancing the model's ability to discern subtle patterns and nuances within the dataset.

Similarly, when compared to the MobileNet model, the VGG19 model also showcased superior performance. The MobileNet model achieved a maximum validation accuracy of around 76%, falling slightly short of the VGG19 model's performance. This disparity can be attributed to the deeper architecture of VGG19, which enables it to capture a broader range of image features and exhibit better generalization capabilities.

In summary, the VGG19 model offers advantages such as enhanced accuracy, stability in predictions, and the ability to capture intricate visual features due to its deeper architecture. These factors contribute to its superior performance when compared to both VGG16 and MobileNet models, making it the preferred choice for our proposed work.

Chapter 8

Conclusion and Future Work

8.1 Conclusion

- This research paper has presented a novel approach to driver distraction detection using computer vision and deep learning techniques.
- By customizing and augmenting the dataset, employing advanced models such as VGG16 and YOLOv8, and leveraging real-time video streaming, significant advancements have been achieved in accurately identifying various types of driver distractions.
- The results demonstrate the potential of the methodology in enhancing road safety by enabling timely detection and intervention.

8.2 Future Work

- Expanding the dataset to include a wider range of distractions and diverse driving scenarios would enhance the model's generalization capabilities.
- Exploring the integration of other sensors such as in-car cameras and physiological sensors could provide a more comprehensive understanding of driver behavior.
- Incorporating a more diverse range of lighting conditions and augmenting the dataset with nighttime scenarios would improve the system's performance under low-lighting or nighttime conditions.
- Integrating the driver distraction detection system with Advanced Driver Assistance Systems (ADAS) by fusing computer vision with other sensor modalities like radar, lidar, and in-car sensors holds great potential for enhancing driver assistance and safety.
- This integration would enable real-time detection and timely intervention, augmenting the overall capabilities of ADAS and contributing to a safer driving ecosystem.

In conclusion, by addressing these challenges and pursuing future research directions, the effectiveness and practicality of the driver distraction detection system can be further enhanced, paving the way for safer and more intelligent driving experiences.

References

[1] Li Li, Boxuan Zhong, Clayton Hutmacher, Yulan Liang, William J. Horrey, Xu Xu, *Detection of driver manual distraction via image-based hand and ear recognition*, *Accident Analysis & Prevention*, Volume 137, 105432, ISSN 0001-4575, `https://doi.org/10.1016/j.aap.2020.105432`, 2020.

[2] B. Qin, J. Qian, Y. Xin, B. Liu, and Y. Dong, *Distracted Driver Detection Based on a CNN With Decreasing Filter Size*, *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 7, pp. 6922-6933, July 2022.

[3] Chen, Ju-Chin, Chien-Yi Lee, Peng-Yu Huang, and Cheng-Rong Lin. *Driver behavior analysis via two-stream deep convolutional neural network. Applied Sciences*, 10(6), 1908, 2020.

[4] Wang, S., Zhang, Y., Wu, C., Darvas, F., and Chaovalitwongse, W. A. *Online Prediction of Driver Distraction Based on Brain Activity Patterns. IEEE Transactions on Intelligent Transportation Systems*, vol. 16, no. 1, pp. 136-150, doi: 10.1109/TITS.2014.2330979. Feb. 2015.

[5] Ben Ahmed, Kaoutar and Goel, Bharti and Bharti, Pratool and Chellappan, Sriram and Bouhorma, Mohammed. *Leveraging Smartphone Sensors to Detect Distracted Driving Activities. IEEE Transactions on Intelligent Transportation Systems*, pp. 1-10, doi: 10.1109/TITS.2018.2873972, 2018.

[6] Gallahan, S. L. et al. *Detecting and mitigating driver distraction with motion capture technology: Distracted driving warning system. IEEE Systems and Information Engineering Design Symposium*, pp. 76-81., 2013.

[7] Majdi, M. S., Ram, S., Gill, J. T., and Rodríguez, J. J. *Drive-Net: Convolutional Network for Driver Distraction Detection. 2018 IEEE Southwest Symposium on Image Analysis and Interpretation (SSIAI)*, pp. 1-4, doi: 10.1109/SSIAI.2018.8470309, 2018.

[8] Wang, Jing, Wu, ZhongCheng, Li, Fang, and Zhang, Jun. *A Data Augmentation Approach to Distracted Driving Detection. Future Internet*, 13(1), 1, https://doi.org/10.3390/fi13010001, 2021.

[9] Artan, Y., Bulan, O., Loce, R. P., and Paul, P. *Driver Cell Phone Usage Detection from HOV/HOT NIR Images.* In *2014 IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp. 225-230, doi: 10.1109/CVPRW.2014.42, 2014.

[10] Kutila, M., Jokela, M., Markkula, G., and Rue, M. R. *Driver Distraction Detection with a Camera Vision System.* In *2007 IEEE International Conference on Image Processing*, pp. VI - 201-VI - 204, doi: 10.1109/ICIP.2007.4379556, 2007.

[11] Ezzouhri, A., Charouh, Z., Ghogho, M., and Guennoun, Z. *Robust Deep Learning-Based Driver Distraction Detection and Classification. IEEE Access*, vol. 9, pp. 168080-168092, doi: 10.1109/ACCESS.2021.3133797, 2021.