SAFE STEER ASSISTANT

A major project report submitted in partial fulfilment of the requirement

for the award of degree of

Bachelor of Technology

in

Computer Science & Engineering & Information Technology

Submitted by

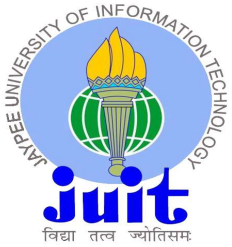
Ansh Kumar (211314)

Skand Dev Dixit (211509)

Under the guidance & supervision of

Dr. Nancy Singla

Assistant Professor (SG)



Department of Computer Science & Engineering and Information Technology

Jaypee University of Information Technology,

Waknaghat, Solan - 173234 (India)

May 2025

JAYPEE UNIVERSITY OF INFORMATION TECHNOLOGY, WAKNAGHAT

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING AND INFORMATION TECHNOLOGY

PLAGIARISM VERIFICATION REPORT

Date: 10th May, 2025.

Type of Document: B.Tech. (CSE / IT) Major Project Report

Name: SRand and Ansh **Enrollment No.:** 211509 and 211314

Contact No: 7042231959 **E-mail:** 211314@juitsolan.in

Name of the Supervisor (s): Dr. Nancy Singla

Title of the Project Report (in capital letters): SAFE STEER ASSISTANT

UNDERTAKING

I undertake that I am aware of the plagiarism related norms/regulations, if I found guilty of any plagiarism and copyright violations in the above major project report even after award of degree, the University reserves the rights to withdraw/revoke my major project report. Kindly allow me to avail plagiarism verification report for the document mentioned above.

- Total No. of Pages: 70
- Total No. of Preliminary Pages: 10
- Total No. of Pages including Bibliography/References: 3

SRand Ansh

Signature of Student

FOR DEPARTMENT USE

We have checked the major project report as per norms and found **Similarity Index** 04 (*% AI Plag.)%. Therefore, we are forwarding the complete major project report for final plagiarism check. The plagiarism verification report may be handed over to the candidate.

Nancy Singla 10/05/2025

Signature of Supervisor

Signature of HOD

FOR LRC USE

The above document was scanned for plagiarism check. The outcome of the same is reported below:

Copy Received On	Excluded	Similarity Index (%)	Abstract & Chapters Details	
	• All Preliminary Pages • Bibliography/ Images/Quotes • 14 Words String		Word Count	
Report Generated On			Character Count	
		Submission ID	Page Count	
			File Size (in MB)	

Checked by

Name & Signature

Librarian

SUPERVISOR'S CERTIFICATE

This is to certify that the major project report entitled **'Safe Steer Assistant'**, submitted in partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology** in **Computer Science & Engineering & Information Technology**, in the Department of Computer Science & Engineering and Information Technology, Jaypee University of Information Technology, Waknaghat, is a bona fide project work carried out under my supervision during the period from July 2024 to May 2025.

I have personally supervised the research work and confirm that it meets the standards required for submission. The project work has been conducted in accordance with ethical guidelines, and the matter embodied in the report has not been submitted elsewhere for the award of any other degree or diploma.

Nancy Singla 10/05/2025.

(Supervisor Signature)

Supervisor Name: Dr. Nancy Singla

Designation: Assistant Professor (SG)

Department: Dept. of CSE & IT

Date: 10-05-2025

Place: Waknaghat

CANDIDATE'S DECLARATION

We hereby declare that the work presented in this major project report entitled **'Safe Steer Assistant'**, submitted in partial fulfilment of the requirements for the award of the degree of **Bachelor of Technology** in **Computer Science & Engineering & Information Technology**, in the Department of Computer Science & Engineering and Information Technology, Jaypee University of Information Technology, Waknaghat, is an authentic record of our own work carried out during the period from July 2024 to May 2025 under the supervision of **Dr. Nancy Singla**.

We further declare that the matter embodied in this report has not been submitted for the award of any other degree or diploma at any other university or institution.

(Student Signature)
Name: Skand Dev Dixit
Roll No.: 211509
Date: 10-05-2025

(Student Signature)
Name: Ansh Kumar
Roll No.: 211314
Date: 10-05-2025

This is to certify that the above statement made by the candidates is true to the best of my knowledge.

10/05/2025.
(Supervisor Signature)
Supervisor Name: Dr. Nancy Singla
Designation: Assistant Professor (SG)
Department: Dept. of CSE & IT

Date: 10-05-2025
Place: Waknaghat

ACKNOWLEDGEMENT

Firstly, I express my heartiest thanks and gratefulness to almighty God for His divine blessing makes us possible to complete the project work successfully.

I really grateful and wish my profound my indebtedness to supervisor **Dr. Nancy Singla, Assistant Professor (SG)**, Department of CSE & IT of Jaypee University of Information Technology, Wakhnaghat. Deep knowledge and keen interest of my supervisor in the field of **"Machine Learning & Deep Learning"** to carry out this project. Her endless patience, scholarly guidance, continual encouragement, constant and energetic supervision, constructive criticism, valuable advice, reading many inferior drafts and correcting them at all stage have made it possible to complete this project.

I would like to express my heartiest gratitude to **Dr. Nancy Singla**, Department of CSE for her kind help to finish my project.

I would also generously welcome each one of those individuals who have helped me straight forwardly or in a roundabout way in making this project a win. In this unique situation, I might want to thank the various staff individuals, both educating and non-instructing, which have developed their convenient help and facilitated my undertaking,

Finally, I must acknowledge with due respect the constant support and patients of my parents.

SKAND DEV DIXIT (211509)

ANSH KUMAR (211314)

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

ABBREVIATIONS	FULL FORM
CNN	Convolutional Neural Network
AI	Artificial Intelligence
OpenCV	Open Computer Vision
ML	Machine Learning
DL	Deep Learning
WHO	World Health Organization
GPS	Global Positioning System
ADAS	Advanced Driver Assistance System
EEG	Electroencephalogram
ECG	Electrocardiogram
EOG	Electrooculogram
UI	User Interface
API	Application Programming Interface
LR	Learning Rate
FPS	Frames Per Second
OBD	Onboard Diagnostic
AR	Augmented Reality
IR	Infrared
GDPR	General Data Protection Regulation

ABSTRACT

The "Safe Steer Assistant" is a driver monitoring system designed to reduce road accidents caused by drowsiness and distraction, two leading factors in impaired driving. This project employs state-of-the-art deep learning techniques to develop two Convolutional Neural Network (CNN) models capable of real-time detection of driver behaviour.

The first model focuses on identifying signs of drowsiness, such as prolonged eye closure or yawning, while the second detects instances of distraction, including looking away from the road or using a mobile phone. These models process live visual inputs from a camera system, analysing facial features, eye movements, and head positioning to determine whether the driver is alert and focused.

Upon detecting drowsiness or distraction, the system immediately triggers an audible alarm to alert the driver, thereby encouraging them to regain attention or take necessary breaks. This real-time intervention mechanism is crucial for preventing potentially fatal accidents caused by lapses in driver focus.

The Safe Steer Assistant represents a significant advancement in intelligent transportation systems by combining cutting-edge computer vision, deep learning, and safety-critical applications. The system is designed to be efficient, user-friendly, and adaptable to different driving environments. By addressing critical safety issues, it contributes to the broader goal of enhancing road safety and reducing traffic fatalities, making it a valuable asset for both personal and commercial vehicles.

CHAPTER 1: INTRODUCTION

1.1 INTRODUCTION

The "Safe Steer Assistant" is a state-of-the-art driver monitoring system specifically designed to greatly improve road safety by actively detecting and reacting to two of the most enduring hazards on the roads today-driver drowsiness and distraction. These factors related to humans are responsible for a high percentage of road crashes around the world and tend to result in severe injuries, fatalities, and permanent outcomes for families and communities. In a world where mounting pressures on drivers longer trips, digital distractions, and cognitive fatigue—are increasingly becoming the standard, an intelligent system such as Safe Steer Assistant bridges a vital gap in driver assistance and intervention, as highlighted by recent research into driver monitoring technologies [1].

Fundamentally, the Safe Steer Assistant exploits state-of-the-art deep learning methods, using two highly optimized Convolutional Neural Networks (CNNs) for real-time behavioral analysis. This is consistent with recent breakthroughs in drowsiness detection through deep CNNs for eye state classification and attention tracking [2]. The technology is sensitive but robust to the subtle details of human facial expressions and patterns of eye movements. It continually monitors live visual data via an onboard camera, evaluating signals including excessive eye closure, rapid blinking, yawning, and head nodding—all universally accepted indicators of fatigue or micro-sleep periods [3]. These features make the system especially useful for long-distance drives or nighttime drives, where drowsiness is most likely.

Beyond the detection of fatigue, the second CNN model specializes in driver distraction, which can be equally risky. Whether staring off the road to look at a mobile phone, fiddling with dashboard controls, or chatting, temporary distractions can lead to disastrous results. The distraction model is learned to recognize these patterns by accurate tracking of head direction, gaze direction, and hand movements, giving a complete picture of where the driver's attention is focused. Such real-time systems based on model compression and lightweight deep networks for embedded devices have shown encouraging results in solving these problems [4].

When the system senses manifestations of inattention—whether caused by sleep or distraction—it activates an immediate intervention. A clear and audible alert is sounded to refocus the driver's attention back onto the road. In subsequent releases, they might be

accompanied by visual or tactile warnings, such as flashing dashboard icon or subtle vibration in the steering wheel, to guarantee not only that the alert is perceived but also responded to. This immediate feedback loop is central to the system's preventive function, not only to safeguard the driver at the time but also to encourage long-term awareness and safer driving practices [5].

What really distinguishes the Safe Steer Assistant is its larger part in the revolution of smart transportation systems. It is not only a technological innovation, but an advancement in AI-based mobility solutions. By integrating this type of smart monitoring into regular vehicles, the assistant becomes a quiet co-pilot, mitigating dangers caused by human error without disrupting the driving experience. As time passes, this unobtrusive support can create a mindfulness and accountability culture among drivers and in the end result in safer roads for all.

In addition to individual advantages, the consequences of the Safe Steer Assistant extend to the entire transport ecosystem. The system enables semi-autonomous and autonomous driving readiness, where human oversight is still essential, and supports data-based road safety policies. These revelations about driver behavior may have impacts on public policy, insurance structures, and urban planning, showing the multifaceted effect of such innovations [1].

As we continue along the path of more intelligent, interconnected cars, systems such as the Safe Steer Assistant highlight the potential of artificial intelligence to be used not only to replace human effort, but to supplement human performance, avoid accidents, and save lives. In the process, it serves as proof of the technology's potential when used for constructive, real-life issues—enabling every ride not only to be smarter, but safer [2][4][5].

1.2 PROBLEM STATEMENT

Each day, millions of lives are at risk from a silent, frequently underestimated danger: driver inattention. Caused by fatigue, distraction, or even just mental disconnection, even a split second loss of concentration behind the wheel can prove disastrous. These seconds—when a driver blinks shut his or her eyes a split second too long, reaches for the phone, or loses himself in thought—can be the determining factor between arrival and disaster. The contemporary driver is more at risk than ever, driving not just congested roads but also an overabundance of digital distractions, physical fatigue, and mental weariness.

Worldwide figures look bleak. Driver fatigue alone is estimated to account for as much as 20% of all road collisions, with some research indicating even greater figures in certain areas or commercial drivers. Likewise, inattention driving—most notably involving mobile phone use—is a primary cause of crashes, injuries, and fatalities. In spite of extensive public awareness campaigns, laws, and the implementation of conventional safety systems such as lane departure warnings or fatigue warnings, these tend to be inadequate. Most are passive, only engaging when a vehicle has already departed from a lane or made a sudden change of direction. They do not possess the ability to recognize the very first warning signals—the slight patterns of behavior showing that attention lapse is imminent.

The core issue is the disconnect between risk identification and timely intervention. Drivers require better than basic alarms or reminders; they require an intelligent, anticipating system that is aware of their condition and intervenes before it is too late. This is particularly crucial in an era when multitasking is the norm and weariness is business as usual.

This is exactly the problem the "Safe Steer Assistant" was designed to solve. As a watchful digital co-pilot, this technology is based on sophisticated computer vision and deep machine learning algorithms, employing OpenCV technology and Convolutional Neural Networks (CNNs) to monitor the driver's facial cues, eye gazes, and head position in real time. These technologies function quietly and invisibly in the background, continually assessing whether or not the driver is alert, distracted, or exhibiting signs of fatigue.

The Safe Steer Assistant does much more than traditional monitoring systems. Rather than waiting for disaster to strike, it detects early warning signs of risk: heavy eyelids, constant yawning, constant head tilting, or looking away from the road. It also detects typical distractions, like fiddling with a mobile phone, reaching into the back seat, or even extended periods of mental disengagement when the driver simply stares blankly ahead without attention. Once detecting such conduct, the system will trigger a context-sensitive alert—an audio signal with volume set to softly, yet insistently, remind the driver to pay attention.

What makes this system particularly powerful is how it differentiates between inattention types. Drowsiness demands a different type of intervention than texting or emotional distraction. By customizing its response to the in-progress risk, Safe Steer Assistant can be sure that alerts are appropriate, timely, and effective—instead of annoying or easy to dismiss.

This subtlety maximizes the driver's chances of responding positively and quickly to the alert.

In addition, the Safe Steer Assistant doesn't merely stop accidents—it also has a role in the process of long-term behavioral change. By encouraging the habit of being attentive, particularly under high-risk situations like night driving or highway driving, it encourages safer driving habits that continue beyond the span of a trip. In time, the system becomes a trusted companion that not only guards but also teaches, creating a feeling of responsibility and self-perspective in the driver.

As cars become more autonomous, and as the lines between driver and machine become less distinct, keeping the driver engaged—particularly in semi-autonomous systems—will continue to be crucial. Safe Steer Assistant is particularly well-suited to bridge this gap, offering both conventional and next-generation vehicles a single solution that caters to the human aspect of safety.

In a transportation ecosystem that's increasingly complex, more demanding, and technology-driven than ever before, tools such as the Safe Steer Assistant aren't merely convenient—they're mandatory. The overarching goal of the project is not merely to eliminate accidents, however, but to revolutionize driving culture. Through the fusion of advanced AI and human-centric design, the system hopes to render each trip safer, each motorist more cognizant, and each street less perilous.

1.3 OBJECTIVES

The "Safe Steer Assistant" project is built on a specific and pressing goal: to curb road accidents through solutions to two of the most common and harmful sources of driver mistake—distraction and drowsiness. These problems keep killing and wrecking property across the world despite growing awareness and the introduction of some conventional driver support technologies. The Safe Steer Assistant seeks to bridge the vital gaps in current systems by providing a smart, responsive, and adaptive solution based on the latest deep learning and real-time computer vision technology.

To achieve this overarching goal, the project is guided by a set of well-defined, interrelated objectives:

1. Comprehensive Study of Existing Literature on Driver Drowsiness and Distraction

Prior to the construction of a sophisticated solution, one needs to know the terrain of available technologies, methods, and studies. This goal requires carrying out an extensive review of academic publications, business white papers, and new developments in driver monitoring systems.

• The research will include both hardware-based (such as steering behavior sensors or heart rate sensors) and software-based solutions, particularly those based on computer vision, machine learning (ML), and deep learning (DL) techniques.

• A key part of this goal is assessing the strengths and weaknesses of existing systems, such as their real-time usability, reliability under changing conditions (such as low light or driver variation), and their capability to distinguish between categories of distraction.

• Such analysis will guide the development of a system that not only improves on established techniques but brings significant improvements that address the present areas of limitation in detection accuracy, responsiveness, and flexibility.

2. Development of Real-Time ML/DL-Based Models for Driver Drowsiness and Distraction Detection

The key to the Safe Steer Assistant is that it does active observation in real-time of driver behavior through the use of Convolutional Neural Networks (CNNs)—a leading deep learning methodology that is particularly good at discerning visual patterns.

- The project will create two distinct but complementary CNN models:
  - One for the detection of drowsiness that can detect symptoms like excessive eye closure, yawning, blinking rate, and head nodding—symptoms that usually precede micro-sleep or slowed reaction times.
  - The second to detect distraction, which will scrutinize habits including too much side-eyeing, the use of hand-held mobile phones, or involvement in driver unrelated tasks as the vehicle travels.
- Diverse datasets shall be used in training and testing the models in order to give good performance regardless of the conditions around them, i.e., daytime or night, urban areas compared to highway routes, as well as various weather conditions.

- Particular care will be taken to guarantee that these models are both computationally effective (for deployment on embedded systems or edge devices) and resistant to false positives, reducing unnecessary warnings that may result in annoyance or alert fatigue.

3. Real-Time Audible Warning and Driver Alert Mechanism

Detection is not enough in itself—there needs to be timely intervention. This goal centers on the development of a multi-level alert system that would immediately notify the driver when their attention falls below a threshold that is safe.

- There will be audible warnings specific to the nature and severity of the detected behavior. For instance, a soft chime may be utilized for minor distractions, whereas a more serious tone or verbal warning may be applied for indications of severe drowsiness.
- Subsequent versions can delve into multimodal alerting, like haptic feedback on the steering wheel, visual alerts on the dashboard, or even on connected wearables.
- Alerts will be formulated to stop dangerous behavior without shocking the driver, so the response will reinforce safety instead of fear.

4. Promoting Safer Driving Practices through Technology and Awareness

Aside from technical behavior, the Safe Steer Assistant seeks to become a means for changing behavior.

- By reinforcing vigilance and enabling drivers to become aware of their own weaknesses, the system promotes a more conscious and responsible driving culture.
- The long-term plan is to integrate the system into educational platforms or driver training programs, making the drivers more cognizant of how quickly distraction or fatigue can creep in—and how technology might help prevent it.

With these goals accomplished, the project will deliver a robust, real-time driver monitoring solution built using the most advanced advancements in deep learning technology for enabling safer driving habits and reducing road traffic fatalities.

1.4 SIGNIFICANCE AND MOTIVATION OF THE PROJECT WORK

Road safety is one of the most critical issues in today's world, with statistics revealing the extent and severity of the issue. As per reports on road safety published by international organizations such as the World Health Organization (WHO) and the International Transport Forum, more than 1.3 million individuals lose their lives annually as a result of road traffic crashes, and another 20 to 50 million are left injured in non-fatal accidents, frequently causing lifelong disabilities. These statistics are not merely statistics, but rather families disunited, incomes lost, and communities shattered by events that, in most instances, are avoidable.

Among the top causes of these tragic consequences are driver fatigue and driver distraction, which still erode the success of conventional road safety measures. Fatigue, frequently caused by insufficient sleep, shift work, or prolonged driving periods, drastically reduces a driver's cognitive abilities, vigilance, and response time. A mere few seconds of distraction can be disastrous, especially on high-speed roads. Simultaneously, diversions—varying from telephone calls and texts to setting GPS directions or conversation—have developed into widespread trends in our age of hyper-interconnectedness, in which the culture encourages people to multitask on digital levels.

While there has been greater awareness and extensive road safety campaigns, the weapons conventionally employed to fight these perils—like signboard notices, rest-stop alerts, or rudimentary in-car lane departure warnings—are reactive, not preventative, and unintelligent in the sense that they cannot become responsive to personal driver habits in real time. This deficiency calls for a pressing requirement for solutions that not merely keep an eye on but actually interpret driver activity and act as needed.

It is here that the inspiration for the "Safe Steer Assistant" project originates. It stems from a sincere, pressing need to save lives using upcoming technologies, especially in Artificial Intelligence (AI) and Deep Learning (DL), to provide a solid, real-time solution that can be a watchful co-pilot for drivers. Using Convolutional Neural Networks (CNNs) and computer vision, the Safe Steer Assistant can identify the telltale traces of driver drowsiness—eye closure duration, yawning, and head nodding—together with diverted attention behaviors like gazing away from the road, smartphone use, or unnecessary manual control operation while driving.

The importance of the project extends beyond merely developing an intelligent monitoring system. It marks a paradigm shift in the way we perceive vehicular safety and driver assistance. Successful integration of a system like this could eliminate the human error factor, which has been estimated to account for more than 90% of all road accidents. Through active anticipation of lapses in attention and sending real-time, context-specific alerts, the Safe Steer Assistant doesn't merely react to threats—it prevents them altogether.

In addition, the potential integration of the system with broader intelligent transportation systems (e.g., Advanced Driver-Assistance Systems (ADAS) and semi-autonomous vehicles) provides opportunities for future development. It can help in the development of smart cars that not only drive but also protect their passengers by comprehending and responding to human actions.

From a scholarly and technical viewpoint, the project represents a chance to demonstrate the practical application of machine learning in safety-critical systems. It tests the bounds of conventional AI by requiring performance under changing lighting, varying human behaviors, and alternative driving conditions, thereby extending the bounds of research and development in intelligent systems.

In a nutshell, the "Safe Steer Assistant" is driven by the desire to close the gap between technology and safety, applying cutting-edge research to practical, effective use. It is not only seeking to develop a tool, but to set a new standard for active driver safety, helping to build a future where fewer families will lose a loved one to a lapse of attention on the road.

1.5 ORGANIZATION OF PROJECT REPORT

This project report is structured into six comprehensive chapters designed to bring out clear ideas of work conducted while developing the "Safe Steer Assistant." The structuring of this report goes as follows:

Chapter 1: Introduction

This chapter presents the project and gives the reader a broad background and an understanding of its scope and purpose. It clearly defines the problem statement, objectives, and the importance of the work to emphasize the need for a driver monitoring system. Furthermore, it gives reasons for initiating the project and sums it up with a summary of how the report is arranged.

Chapter 2: Literature Survey

This chapter discusses previous studies and technological innovations related to driver monitoring systems, mainly drowsiness and distraction detection. A comprehensive survey of relevant studies, white papers, and recent activities is presented, with particular emphasis on the last five years. The chapter defines key gaps in the literature that this project addresses in order to establish its justification.

Chapter 3: System Development

This chapter explains the core development process. Starting with the identification of system requirements and the analysis of functional and non-functional needs, it deals with solution design and architecture, techniques for preparing data, implementation details, and algorithms and tools and technologies applied in the process. Further, challenges faced in the course of development and how to overcome them are described in detail.

Chapter 4: Testing

This chapter explains the testing methodology and strategies used to test the system performance. It provides in detail the test cases with testing tools employed for testing. The outcome of these tests is provided and examined to determine if the system succeeds in performing its desired functions.

Chapter 5: Results and Evaluation

In this chapter, the results of the implementation and testing phases are explained in detail. The results are analyzed and compared with the current solutions to emphasize the effectiveness and strengths of the system. This comparison is used to justify the success of the project and its potential to contribute to driver safety.

Chapter 6: Conclusions and Future Scope

The last chapter recapitulates the major accomplishments and contributions of the project in describing its limitations. It also examines possibilities for future improvements and proposes methods for extending the system's functionality and applicability to broader contexts.

CHAPTER 2: LITERATURE SURVEY

2.1 OVERVIEW OF RELEVANT LITERATURE

The literature review in the area of driver drowsiness and distraction detection identifies a vibrant and fast-developing research environment. The research indicates substantial advancements in using artificial intelligence and deep learning methods to improve vehicular safety through the monitoring of driver vigilance. Yet, in spite of encouraging developments, a number of challenges remain that restrict the practical implementation of such systems, particularly in terms of data diversity, model generalization, and environmental robustness.

V. Zumbika [5] has suggested a system for detecting sleep deprivation through transfer learning and image classification methods employing the YawDD dataset. The experiment reported an impressive 95% accuracy rate, testifying to the capability of deep learning models to classify drowsiness states. Nevertheless, the use of a narrowly defined dataset—primarily facial features in controlled conditions—restricts the applicability of the system in real, dynamic driving conditions. It underscores the need for larger and more diverse datasets consisting of varied facial profiles, illumination conditions, and camera views.

Bhargava Reddy [4] emphasized the development of a light-weight model optimized for driver state classification from facial landmarks. This model, validated on an embedded Jetson TK1 board, reported a decent accuracy of 89.5%. Its strength is that it proves the feasibility of running such models on resource-limited hardware, essential for deployment in commercial vehicles. However, the research highlights key challenges—like sensitivity to light variation and device compatibility—that limit consistent performance across varying driving conditions, particularly in low-light or overexposure situations.

Venkata Rami Reddy [3] proposed a deep learning architecture based on stacked Convolutional Neural Networks (CNNs) and SoftMax layers for detecting driver drowsiness from eye state classification. With a remarkable 96.42% accuracy, the study validates the hypothesis that eye-related features are extremely effective for detecting drowsiness. Nevertheless, the success of the framework is mostly limited to laboratory-controlled environments. The research is not subjected to stringent real-world testing and fails to consider key issues like driver acceptance, camera placement in diverse vehicle interiors, and system reliability over the long term in normal use.

On a larger scale, Yaman Albadawi [4] performed an exhaustive survey of the current drowsiness detection methods, classifying them into four major approaches: image-based, biological-based, vehicle-based, and hybrid systems. This classification gives a useful taxonomy for comprehending the variety of methods being investigated. Informative as it is, the survey does not quite go as far as examining the challenges in practical implementation for each method. Issues like scalability, integration with automotive systems, intrusiveness for users, and cost-effectiveness are left to be addressed here.

Other research also underscores the constraints of existing technologies. For example, Ghoddoosian et al. [6] presented a realistic dataset and baseline temporal model for early drowsiness detection, pointing out the necessity of context-aware temporal modeling as well as static image analysis. However, scalability and dataset-specific adaptation are issues.

Dhawde et al. [7] presented a rule-based drowsiness detection system with the incorporation of eye blink and yawning frequency measurements. Though available, the method is not robust in changing driving conditions, particularly under other lighting or occlusion conditions.

Chowdhury et al. [8] and Saini & Saini [9] discussed sensor-based and physiological methods, giving prominence to EEG, ECG, and EOG signals in precisely measuring driver fatigue. Although precise, the invasive nature of wearable sensors creates a barrier to mass deployment in non-commercial or personal cars.

Forsman et al. [10] have provided results on identifying moderate levels of drowsiness effectively, important for preventive action prior to approaching critical thresholds. Nonetheless, they base their methods on controlled simulation environments, which are not necessarily scalable in turbulent real-world settings.

Surveys by Fuletra and Bosamiya [11] and Alshaqaqi et al. [12] reiterated the value of integrating various modalities like facial recognition, vehicle behavior, and physiological signals to enhance reliability and accuracy. However, they also pointed to ongoing challenges around cost-effectiveness, computational loads, and compliance on the part of users.

In summary, the literature surveyed evinces that AI-based driver monitoring systems have great potential for mitigating road accidents due to drowsiness and distraction. Nonetheless, recurring issues of data insufficiency, absence of real-world testing, low scalability, and environmental flexibility remain to hinder extensive uptake. It is in these unresolved areas

that the Safe Steer Assistant project is pioneering an innovative solution towards making a more adaptive, resilient, and pragmatic real-time monitoring system. By leveraging state-of-the-art CNN architectures, real-time image processing, and context-aware alerts, the project seeks to advance the current boundaries of driver safety technologies and pave the way for their integration into mainstream intelligent transportation systems.

2.2 KEY GAPS IN THE LITERATURE

Key gaps which the literature review indicates are there concerning detection of driver drowsiness and distraction are mentioned as below:

1. Dataset Limitation: Much of the above-referred research work based on small datasets was, like by V. Zumbika [5] and Bhargava Reddy [4] concerned about specific facial features or states of eyes. A very limited scope of datasets has a consequence of limiting generalization towards the actual application in real-world settings with diverse lighting conditions, variations in facial orientation, and different human faces. A more varied and larger dataset is required to increase robustness and applicability for various detection systems in a different environment.

2. Real-World Testing: Several models, one of which is that from Venkata Rami Reddy [2] very good accuracy in controlled environments. However, most of these systems are inadequately tested in real-world driving conditions and cannot be assumed to be fully robust for practical applications. These tests are important for the validation of their reliability in real-world deployments into actual vehicles.

3. Cross-Subject and Calibration Issues: Such research on cross-subject drowsiness detection from EEG signals, conducted by Jian Cui [3] indicate a crucial challenge to be tackled. It concerns the cross-person calibration problem. A difference in the signals produced during EEG tests or any other physiological activity does not result in systems universally applicable for anyone. Calibration issues must be further overcome and work conducted to ensure systems' applicability towards wide-range of users.

4. Limited Multimodal Approaches: Although individual factors such as facial features, eye movements, and EEG signals were explored in some research studies, most of the systems still employ one detection method. A large gap exists in the integration of multiple detection methods to increase the accuracy and reliability of the detection method, such as integrating facial recognition with physiological sensors or vehicle-based indicators.

5. Real-Time Feedback and Integration: Many of these systems lack effective means of providing real-time feedback, which are critical for practical use. The research suggests further explorations in the development of a system that can provide immediate alerts to the driver in an actionable format. This could be very crucial for accident prevention in real-time.

6. Scalability and practical implementation: Most studies don't describe the scalability of proposed models to be deployed widely for varied driving conditions or for multiple vehicle types. Most of the work needs optimization systems for different embedded platforms, which should be made smooth for everyday driving conditions.

CHAPTER 3: SYSTEM DEVELOPMENT

3.1 REQUIREMENTS AND ANALYIS

The first step of the system development was conducting a thorough requirements analysis in order to understand the core functionality and constraints. The primary requirements for the "Safe Steer Assistant" are:

1. Real-Time Detection: The system must detect drowsiness and distraction in real-time, ensuring that there is no significant delay in processing and responding to potentially dangerous driver states.

2. High Accuracy: The system has to have a very good accuracy rate in terms of detecting drowsiness in drivers and distraction in minimizing false positives and false negatives.

3. Alert System: After having detected drowsiness and distraction, the system is supposed to produce an audio alert immediately in order to raise the driver's attention.

4. Lightweight Model: As real-time performance is needed; machine learning models have to be lightweight and optimized for the best possible run on embedded systems or hardware such as on-vehicle cameras and sensors.

5. Data Security and User Privacy: The system should never infringe the privacy of the driver. Collected data should be processed on the local side without transmitting sensitive data to servers in remote locations.

3.2 PROJECT DESIGN AND ARCHITECTURE

The "Safe Steer Assistant" system has been designed with a modular design methodology to provide scalability, maintainability, and real-time performance. The architecture unifies different interdependent elements that, together, provide for the smooth functioning of the system—ranging from data capture and processing, detection of anomalies in driver behavior and provision of timely alerts.

The architecture is segmented into five main modules, each performing specialized tasks, which collectively constitute a strong pipeline for smart driver state monitoring. These modules are described below in detail:

1. Data Collection Module: At the core of the system is the Data Collection Module, which acts as the main interface between the real world and the digital processing pipeline. This module uses camera sensors - usually mounted on the dashboard or steering column—to constantly take high-resolution video frames of the driver's face and upper torso.

Key responsibilities include:

- Facial landmark acquisition: Following important features such as eyes, nose, mouth, and eyebrows.
- Head pose estimation: Identifying if the driver is looking forward, sideways, or downwards.

2. Preprocessing Module: The Preprocessing Module acts as the interface between raw sensor data and the pipeline. The module provides assurance that video frames and image sequences obtained are clean, normalized, and prepared for effective processing.

Principal tasks are:

- Cropping and face alignment: Isolating the facial region and aligning it for consistent input orientation.
- Image normalization: Changing brightness, contrast, and scale to minimize variation under varying illumination conditions.

By delivering consistent and structured input to the detection model, this module plays a vital role in ensuring model accuracy and minimizing false detections.

3. Detection Module: This is the core intelligence hub of the Safe Steer Assistant, where the actual inference of driver state takes place. The Detection Module incorporates a set of trained machine learning and deep learning models to classify real-time driver behaviour as either normal, drowsy, or distracted.

The detection logic is designed to:

- Recognize micro-sleeps, prolonged eye closure, head nodding, and other fatigue-related cues.
- Detect distracted behaviours such as looking away for extended periods, interacting with a phone, reaching into the back seat, or drinking water.

- Operate efficiently under real-time constraints using optimized model architectures and reduced parameter sets.

4. Alert Module: Upon detection of a drowsiness or distraction event, the system immediately activates the Alert Module to notify the driver and prevent possible accidents. This module is designed to be fast, reliable, and non-intrusive.

It includes:

- Auditory alerts: Beeps, chimes, or spoken prompts that vary in urgency based on risk level.
- Optional haptic feedback: Seat or steering wheel vibrations for tactile awareness.
- Visual signals: Flashing LEDs or dashboard notifications indicating the type of alert triggered.
- Smart escalation: If a risky behaviour is detected continuously without correction, the alert may intensify or trigger more aggressive intervention in future versions (e.g., slowing the vehicle, notifying emergency contacts).

The goal is not just to alert the driver but to interrupt the dangerous behavior effectively and safely.

5. User Interface: The User Interface of the system is designed to be minimalistic, non-distracting, and informative. It provides the driver with high-level status indicators while ensuring the focus remains on the road.

Core UI features include:

- A simple dashboard widget or screen that shows current system status: "Monitoring," "Drowsy Detected," or "Distraction Detected."
- Alert history logs showing timestamps and types of alerts triggered during the journey.
- Settings menu (if extended) allowing drivers to configure alert volumes or temporarily pause the system (e.g., when another person is driving).

In fleet or commercial environments, this UI could also be connected to cloud-based systems for supervisor review or safety reporting.

Figure 3.1 (Project Architecture) provides a comprehensive visual representation of the overall structure and functional workflow of the "Safe Steer Assistant" system. This architecture outlines the interconnection between various components that work collaboratively to achieve real-time monitoring of driver behaviour, particularly focusing on the detection of drowsiness and distraction, which are major contributing factors to road accidents.

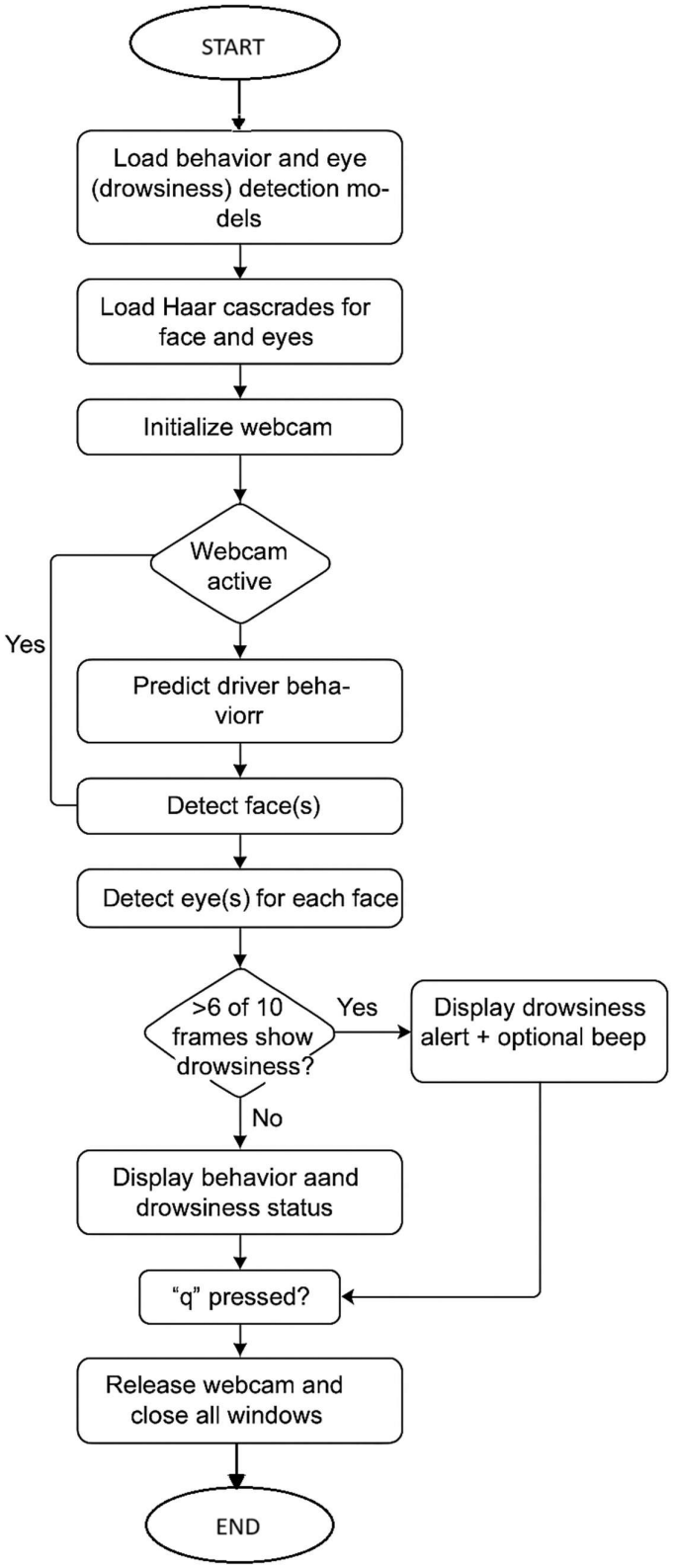


FIGURE 3.1: PROJECT ARCHITECTURE

3.3 DATA PREPRATION

Data preparation is the third critical step in the development of the "Safe Steer Assistant" system; in other words, the data collected from different sources and cleaned, processed, and structured appropriately to effectively train and test the models in machine learning and deep learning. This chapter is a discussion of the data collected, preprocessing techniques used, and feature extraction that were utilized in the data preparation process.

3.3.1 Data Collection

The data for the system were gathered from publicly available datasets as well as simulated and real-world driving scenarios. The collected data includes:

1. FOR DROWSINESS DETECTION:

Dataset: MRL Eye Dataset (Kaggle)

Description: The MRL Eye Dataset is a gray-scale image dataset curated to support the development of real-time eye state classification systems. It contains well-balanced image samples across two primary classes:

- **Open Eyes:** Images where the eyes are visibly open, indicating wakefulness and attentiveness.
- **Closed Eyes:** Images where the eyes are visibly closed, which may indicate blinking, drowsiness, or sleep onset.

There are 19,383 images in each class, maintaining a well-balanced dataset suitable for binary computer vision and deep learning classification tasks.

TABLE 3.1: Drowsiness Dataset Overview

CLASS LABEL	NO. OF IMAGES
OPEN	19383
CLOSE	19383

Objective: The objective of this module is to identify the eye state (open/closed) from live vision input. This task is a key part of a larger drowsiness detection system,

where sustained closed-eye detection for some duration threshold can result in an alert that the driver might be falling asleep.

2. FOR DISTRACTION DETECTION:

Dataset: Driver Inattention Detection Dataset (Kaggle)

Description: The dataset consists of gray-scale images divided into six unique classes that represent a range of different driver behaviors, all pertinent to measuring levels of attentiveness and safety at the wheel:

1. **Safe Driving:** Tracks diligent and vigilant driving behavior, such as compliance with correct hand position, and watchful posture. These samples are the reference point for non-distracted conditions.
2. **Distracted Driving:** Covers situations where drivers are clearly distracted from the road, usually because of mobile phone use, eating, engaging with passengers, or other activities that reduce attention.
3. **Dangerous Driving:** Encompasses images showing reckless or hazardous actions, such as aggressive steering, sudden lane changes, or high-speed behavior all contributing to elevated risk levels.
4. **Drinking:** Depicts drivers drinking alcoholic drinks or any other drinks while driving. These photos point out serious safety infractions and the risks involved with impaired driving.
5. **Sleepy Driving:** Illustrates drivers who show signs of fatigue or sleepiness, like drooping eyelids or slow posture, which can significantly lower reaction times and awareness of the situation.
6. **Yawning:** Particularly photographs drivers yawning, a strong physiological indicator of tiredness. Yawning is considered a lead-up to microsleep episodes and lack of attention.

TABLE 3.2: Distraction Dataset Overview

DATASET SPLIT	NO. OF IMAGES
Training Set	11942
Validation Set	1922
Test Set	985

Objective: The objective of this project is to detect driver distraction through analysis of body and head posture using deep learning techniques. By leveraging image data across these behavioral classes, the model aims to identify:

- Postural cues such as slumped shoulders, leaning, or hand positioning.
- Head orientation and facial expressions, especially those indicating drowsiness or disengagement.

Such detection serves as a foundation for real-time driver monitoring systems capable of issuing alerts or taking preventive actions to mitigate the risk of accidents caused by inattention or fatigue.

3.4 IMPLEMENTATION

The development of the "Safe Steer Assistant" system incorporated deep learning models that sensed the driver's drowsiness and distractions. The tools, technologies, methodologies, and steps utilized in this chapter demonstrate how to successfully deploy such a system.

3.4.1 Tools and Technologies

This section explains the essential tools, libraries, and frameworks utilized in the implementation of the driver behavior and drowsiness detection system.

1. Programming Language: Python was chosen because it is easy to use, flexible, and has plenty of libraries for machine learning, deep learning, and image processing.

2. Deep Learning Frameworks: TensorFlow and Keras were utilized to design and train Convolutional Neural Networks (CNNs). Both of the frameworks provide high-level APIs for building, training, evaluation, and deploying models.

3. Computer Vision Libraries: OpenCV (Open-Source Computer Vision Library) was used for real-time image and video capture, pre-processing, and data augmentation.

4. Data Handling and Visualization: Pandas and Numpy were applied for data preprocessing and manipulation. Matplotlib was used for the plotting of model performance metrics such as confusion matrices, accuracy, and loss.

5. Model Optimization and Evaluation: Scikit-learn was used for the evaluation of metrics such as confusion matrices and classification reports.

Keras callbacks such as ReduceLROnPlateau, EarlyStopping, and ModelCheckpoint were used to enhance training efficiency and prevent overfitting.

6. Image Augmentation: Keras's ImageDataGenerator was utilized to improve model generalization using real-time data augmentation methods.

7. Hardware and Input Devices: A system with a GPU (e.g., NVIDIA CUDA-capable hardware) was used to speed up model training.

Webcams were used for acquiring real-time image data to mimic real-world environments.

3.4.2 ALGORITHM USED

To effectively detect driver drowsiness and distraction, we employed **Convolutional Neural Networks (CNNs)** due to their high performance in image classification tasks. The problem was approached in two distinct parts:

- **Drowsiness Detection**: Treated as a binary classification task (Drowsy vs Non-Drowsy).
- **Distraction Detection**: Treated as a multi-class classification problem with six distinct behavioral classes.

3.4.2.1 CNN FOR DROWSINESS DETECTION (BINARY CLASSIFICATION):

Input Data:

- Real-time gray-scale images focusing on facial regions, particularly eyes and mouth.
- Dataset used: MRL Eye Dataset, containing labeled images of Open and Closed eyes.

Preprocessing:

- All images resized to **224x224 pixels**.
- Pixel values normalized to a [0, 1] range.
- **Data augmentation** applied using techniques like:
  - Random brightness/contrast adjustments.
  - Horizontal flips and rotations.
  - Zoom and shift transformations to simulate real-world driving conditions.

Model Architecture:

- Transfer learning was utilized by leveraging the **MobileNet** architecture (pre-trained on ImageNet).
- The top (classification) layers of the original MobileNet model were removed.
- A custom binary classifier was added with:
  - Flatten layer to convert CNN output to 1D.
  - Dense layer with a single neuron.
  - Sigmoid activation to output the probability of drowsiness.

Output: Sigmoid Output Layer enables probability-based decision-making for "Drowsy" vs "Non-Drowsy" prediction.

Training Configuration:

- **Loss Function:** Binary Cross-Entropy
- **Optimizer:** Adam (learning rate = 0.001)
- **Epochs:** 5–10 (with EarlyStopping on validation loss to prevent overfitting)
- **Evaluation Metrics:** Accuracy, Precision, Recall, F1 Score

3.4.2.2 CNN FOR DISTRACTION DETECTION (MULTI-CLASS CLASSIFICATION)

Input Data

- Real-time video frames capturing diverse driver behaviours.
- Dataset labelled with 6 distinct classes representing various driving states:
  - Class 0: Dangerous Driving
  - Class 1: Distracted
  - Class 2: Drinking

- Class 3: Safe Driving
- Class 4: Sleepy Driving
- Class 5: Yawn

Preprocessing:

- All images resized to **224x224 pixels** (RGB format).
- Pixel values normalized to a **[0, 1]** scale.
- Augmentation applied to simulate real-world driving conditions:
  - **Random brightness/contrast adjustments**
  - **Horizontal flips**
  - **Rotation, zoom, and shift transformations**
  - Designed to enhance model generalization under varying lighting, angles, and occlusions.

Model Architecture:

- **Base Model**: MobileNetV2 (pre-trained on ImageNet; top classification layers removed).
- **Custom Classifier** added on top:
- GlobalAveragePooling2D to flatten feature maps.
- Dense layer with **512 neurons**, ReLU activation, and **L2 regularization**.
- **Dropout (0.4)** for regularization.
- Dense layer with **256 neurons**, ReLU activation, and L2 regularization.
- **Dropout (0.3)** for added generalization.
- **Output Layer**: Dense layer with **6 neurons** (for 6 classes) and **Softmax activation**.

Output: A SoftMax activation layer outputs probabilities for the 10 classes.

Training Configuration

- **Loss Function**: Categorical Cross-Entropy (multi-class classification).
- **Optimizer**: Adam
  - Initial Learning Rate: **0.001**
  - Fine-tuning Learning Rate: **5e-5**

- **Training Strategy**:
  - **Phase 1**: Train only custom top layers (base MobileNetV2 frozen).
  - **Phase 2**: Unfreeze the last 50 layers of the base model for fine-tuning.
- **Epochs**:
  - **5 epochs** for initial training.
  - **8 epochs** for fine-tuning.
  - **EarlyStopping** and **ModelCheckpoint** used for optimal performance.
- **Learning Rate Adjustment**:
  - ReduceLROnPlateau monitors validation loss to reduce LR when plateaued.

3.4.2.3 REAL-TIME MONITORING SYSTEM

To validate the deployment viability of the trained drowsiness and distraction detection models, a real-time driver monitoring system was utilized. The system utilizes live webcam feed and does concurrent analysis for both distracted and drowsiness behaviors using two pre-trained deep learning models.

System Overview:

- Captures live video frames from the webcam through OpenCV
- Identifies face and eye areas through Haar Cascade classifiers
- Processing of frames for prediction through two pre-loaded CNN models
- Shows live driver behavior classification and drowsiness detection status
- Plays sound alerts for risky or sleepy behaviors

System Pipeline:

1. **Live Frame Capture:**
   - Captures each video frame in real-time via OpenCV
2. **Face & Eye Detection:**
   - **Face detection**: Performed using haarcascade_frontalface_default.xml
   - **Eye detection**: Performed using haarcascade_eye.xml
   - Detected eye regions are resized and passed to the **drowsiness detection CNN**

3. Image Preprocessing:

- All detected regions are resized to **224x224x3**
- Normalized to match CNN input requirements

4. Dual-Model Prediction:

- **Distraction Detection (Multi-class CNN):**
    - Takes entire resized RGB frame
    - Predicts one of the following six classes:
        - 0: Dangerous Driving,
        - 1: Distracted,
        - 2: Drinking,
        - 3: Safe Driving,
        - 4: Sleepy Driving,
        - 5: Yawn
    - Predictions with confidence > 70% and in dangerous categories trigger alerts
- **Drowsiness Detection (Binary CNN)**
    - Takes cropped and resized eye regions
    - Predicts eye state (open or closed)
    - Maintains a sliding window buffer (last 10 frames)
    - If ≥6 frames indicate drowsiness, an alert is triggered

5. Alert and UI System:

- Real-time decision and feedback logic:

If Drowsy:

- Show: Drowsy
- Play: Long beep sound

If Distracted (e.g., Yawning, Drinking, etc.):

- Show: Corresponding label (e.g., Yawning, Drinking)
- Play: Short beep sound

If Both Drowsy and Distracted:

- Show: Drowsy
- Play: Long beep sound (Drowsy alert takes priority)

If Safe Driving:

- Show: Safe Driving
- No sound

Impact & Practicality:

- Demonstrates **successful real-time deployment** of CNN models
- Validates **model robustness** and generalizability in real-world conditions
- Suitable for integration into:
    - Smart vehicle systems
    - Mobile safety apps
    - Driver assistance modules

3.4.3 CODE SNIPPETS

Here are some of the code snippets of "Safe Steer Assistant"

3.4.3.1 FOR DROWSINESS DETECTION

Figure 3.2 represents the steps involved in preparing raw data for training the drowsiness detection model, including resizing images, normalization, and augmentation. These processes ensure that the data is in a suitable format and quality for input into the MobileNet models.

```python
# Define paths for training, testing, and validation data directories
train = "data/train"
test = "data/test"

# Define ImageDataGenerator with augmentation for training
datagen = ImageDataGenerator(
    rescale=1./255,        # Normalize pixel values from range (0, 255) to (0, 1)
    rotation_range=20,     # Randomly rotate images in the range of 20 degrees
    width_shift_range=0.2, # Randomly shift images horizontally by up to 20%
    height_shift_range=0.2, # Randomly shift images vertically by up to 20%
    shear_range=0.2,       # Apply random shear transformation
    zoom_range=0.2,        # Randomly zoom in on images
    horizontal_flip=True,  # Randomly flip images horizontally
    validation_split=0.2   # Reserve 20% of the data for validation
)

# Training set (80% of data)
train_generator = datagen.flow_from_directory(
    train,  # Training data directory
    target_size=(224, 224),  # Resize all images to 224x224
    batch_size=32,           # Process images in batches of 32
    class_mode='binary',     # Use binary class mode (for binary classification tasks)
    subset='training'        # Use the 'training' subset (80% of data)
)

# Validation set (20% of data)
val_generator = datagen.flow_from_directory(
    train,  # Training data directory, validation is taken from this
    target_size=(224, 224),  # Resize all images to 224x224
    batch_size=32,           # Process images in batches of 32
    class_mode='binary',     # Use binary class mode (for binary classification tasks)
    subset='validation'      # Use the 'validation' subset (20% of data)
)

# Test set (Separate folder, no data augmentation)
test_datagen = ImageDataGenerator(rescale=1./255)  # Only rescale the test images

test_generator = test_datagen.flow_from_directory(
    test,  # Test data directory
    target_size=(224, 224),  # Resize all test images to 224x224
    batch_size=32,           # Process images in batches of 32
    class_mode='binary',     # Use binary class mode (for binary classification tasks)
    shuffle=False            # Don't shuffle test data during evaluation
)
```

FIGURE 3.2: DATA PREPROCESSING (DROWSINESS)

Figure 3.3 illustrates the architecture of the convolutional neural network (CNN) used for detecting driver drowsiness. The model employs the MobileNet architecture pre-trained on ImageNet, excluding its top classification layers. A custom binary classifier is added, consisting of a Flatten layer, a Dense layer with one neuron, and a Sigmoid activation function to output a probability indicating whether the driver is drowsy or alert. This architecture allows for efficient feature extraction from facial regions, particularly focusing on the eyes, and enables real-time binary classification.

Define the base model using MobileNet with ImageNet weights, excluding the top (classification) layer

```
# Load pre-trained MobileNet model without the top layers (i.e., without classification layers)
model = tf.keras.applications.mobilenet.MobileNet()
# Extract the input and output layers from the base model
base_input = model.layers[1].input  # The input layer of the pre-trained model
base_output = model.layers[-4].output  # The output layer before the top classification layers

# Add a Flatten layer to convert the 2D output into a 1D vector
Flat_layer = layers.Flatten()(base_output)

# Add a Dense layer with a single unit for binary classification (sigmoid activation for binary output)
final_output = layers.Dense(1)(Flat_layer)

# Apply sigmoid activation to the output for binary classification (0 or 1)
final_output = layers.Activation('sigmoid')(final_output)

# Create the new model with the input from the base model and the custom output
new_model = keras.Model(inputs=base_input, outputs=final_output)

# Print the summary of the new model
new_model.summary()
```

FIGURE 3.3: DEFINING MODEL (DROWSINESS)

Figure 3.4 provides a view of what a configuration of CNN with the optimizer, loss, and evaluation metrics looks like in building a model ready for training based on the setting necessary parameters for effective learning.

Compile the new model with binary cross-entropy loss, Adam optimizer, and accuracy metric

```
new_model.compile(
    loss="binary_crossentropy",  # Binary cross-entropy loss for binary classification
    optimizer="adam",            # Adam optimizer for efficient training
    metrics=["accuracy"]         # Track accuracy during training and evaluation
)
```

FIGURE 3.4: COMPILING MODEL (DROWSINESS)

Figure 3.5 demonstrates the feeding of pre-processed data into the CNNs so that the network learns patterns associated with driver drowsiness.

Train the model and store history

```
# Train the model and store the training history
history = new_model.fit(
    train_generator,              # The training data generator
    validation_data=val_generator,  # The validation data generator
    epochs=5                      # Number of epochs to train the model
)
```

```
c:\Users\dixit\AppData\Local\Programs\Python\Python311\Lib\site-packages\keras\src\trainers\data_adapters\py_dataset_adapt
  self._warn_if_super_not_called()
Epoch 1/5
820/820 ━━━━━━━━━━━━━━━━━━━━ 1374s 2s/step - accuracy: 0.9726 - loss: 0.0902 - val_accuracy: 0.9338 - val_loss: 0.1491
Epoch 2/5
820/820 ━━━━━━━━━━━━━━━━━━━━ 1354s 2s/step - accuracy: 0.9922 - loss: 0.0246 - val_accuracy: 0.9715 - val_loss: 0.1139
Epoch 3/5
820/820 ━━━━━━━━━━━━━━━━━━━━ 1361s 2s/step - accuracy: 0.9927 - loss: 0.0243 - val_accuracy: 0.9731 - val_loss: 0.0688
Epoch 4/5
820/820 ━━━━━━━━━━━━━━━━━━━━ 1513s 2s/step - accuracy: 0.9924 - loss: 0.0216 - val_accuracy: 0.9118 - val_loss: 0.1990
Epoch 5/5
820/820 ━━━━━━━━━━━━━━━━━━━━ 1556s 2s/step - accuracy: 0.9930 - loss: 0.0208 - val_accuracy: 0.9147 - val_loss: 0.1757
```

FIGURE 3.5: TRAINING MODEL (DROWSINESS)

Figure 3.6 demonstrates the saving process of the trained CNN model for driver drowsiness detection. The entire model—architecture, learned weights, and optimizer state—is saved as HDF5 (.h5) using the model.save() function. This enables effortless reuse, inference, or additional training without redefining the model architecture.

Save the trained model to a file in the H5 format

```
new_model.save("drowsiness_detection_model.h5")
print("Model saved as 'drowsiness_detection_model.h5'")
```

FIGURE 3.6: SAVING THE TRAINED MODEL (DROWSINESS)

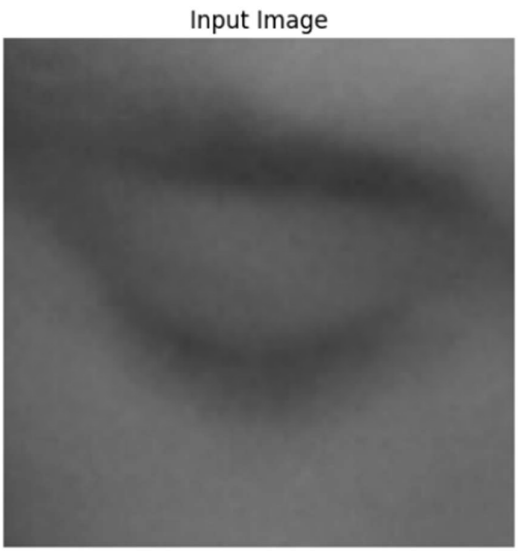
Figure 3.7 illustrates the input image fed into the trained drowsiness detection model along with its corresponding prediction result. The system processes a real-world facial image, resizes and normalizes it to match model input requirements, and then classifies the state as either Drowsy or Non-Drowsy. The displayed image represents the actual visual input used during prediction, showcasing the model's ability to operate on unseen data and provide an interpretable output in real-time.

```python
import tensorflow as tf
import numpy as np
import matplotlib.pyplot as plt
from tensorflow.keras.preprocessing import image

# Load the trained model
model = tf.keras.models.load_model("drowsiness_detection_model.h5")

def predict_image(image_path, model):
    # Load image as PIL image
    img = image.load_img(image_path, target_size=(224, 224))

    # Convert to numpy array for model prediction
    img_array = image.img_to_array(img)
    img_array_expanded = np.expand_dims(img_array, axis=0)  # Add batch dimension
    img_array_expanded /= 255.0  # Normalize for model

    # Display the original image (before normalization)
    plt.imshow(img)  # Display the PIL image directly
    plt.title("Input Image")
    plt.axis("off")
    plt.show()

    # Make prediction
    prediction = model.predict(img_array_expanded)[0][0]  # Assuming binary classification
    print(f"Prediction Value: {prediction:.4f}")

    # Interpret the prediction
    if prediction < 0.5:
        print("Status: Drowsy 😴")
    else:
        print("Status: Non-Drowsy 🚗")
```

FIGURE 3.7: PREDICTION FROM THE SAVED MODEL (DROWSINESS)

Figure 3.8 displays the textual and symbolic output generated by the drowsiness detection model after processing the input image. Based on the predicted probability, the system interprets and prints a clear status message—either Drowsy or Non-Drowsy for easy human understanding. This output demonstrates how the model's numerical prediction is translated into actionable feedback for real-time driver alerting.

```
image_path = r"data\test\Close\s0001_00349_0_0_0_0_1_01.png"  # Change this to an actual image path
predict_image(image_path, model)
```




```
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 47ms/step
Prediction Value: 0.0000
Status: Drowsy 😴
```

FIGURE 3.8: OUTPUT OF PREDICTION (DROWSINESS)

3.4.3.2 FOR DISTRACTION DETECTION

Figure 3.9 represents the steps involved in preparing raw data for training the distraction model, including resizing images, normalization, and augmentation. These processes ensure that the data is in a suitable format and quality for input into the CNN models.

```python
# Preprocessing for validation and test datasets (only rescaling pixel values to [0, 1])
val_test_datagen = ImageDataGenerator(rescale=1./255)

# Data augmentation for training to improve model generalization
train_datagen = ImageDataGenerator(
    rescale=1./255,                  # Normalize pixel values
    rotation_range=15,               # Random rotation up to 15 degrees
    width_shift_range=0.15,          # Random horizontal shift
    height_shift_range=0.15,         # Random vertical shift
    shear_range=0.1,                 # Shear transformation
    zoom_range=0.15,                 # Random zoom
    horizontal_flip=True,            # Random horizontal flipping
    brightness_range=[0.85, 1.15],   # Random brightness adjustment
    fill_mode='nearest'              # Fill in missing pixels after transforms
)

# Create training data generator from DataFrame
train_generator = train_datagen.flow_from_dataframe(
    train_df,
    x_col='image_path',              # Column containing image file paths
    y_col='class_id_str',            # Column containing class labels (as strings)
    target_size=target_size,         # Resize all images to this size
    batch_size=batch_size,           # Number of images per batch
    class_mode='categorical',        # One-hot encoded labels
    shuffle=True                     # Shuffle data for training
)

# Create validation data generator
valid_generator = val_test_datagen.flow_from_dataframe(
    valid_df,
    x_col='image_path',
    y_col='class_id_str',
    target_size=target_size,
    batch_size=batch_size,
    class_mode='categorical',
    shuffle=False                    # No shuffle for validation
)

# Create test data generator
test_generator = val_test_datagen.flow_from_dataframe(
    test_df,
    x_col='image_path',
    y_col='class_id_str',
    target_size=target_size,
    batch_size=batch_size,
    class_mode='categorical',
    shuffle=False                    # No shuffle for test
)
```

FIGURE 3.9: DATA PREPROCESSING (DISTRACTION)

Figure 3.10 illustrates the CNN architecture for detecting driver distraction based on MobileNetV2 as the base model with frozen layers. A classification head is introduced with global average pooling, dense layers (ReLU + L2 regularization), dropout, and a final softmax layer to classify driver behavior into more than one class. The model is compiled with the Adam optimizer and categorical cross-entropy loss for multi-class classification.

Define the base model using MobileNetV2 with ImageNet weights, excluding the top (classification) layer

```
base_model = MobileNetV2(weights='imagenet', include_top=False, input_shape=(224, 224, 3))

# Add a custom classifier on top of the pre-trained base model
x = base_model.output
x = GlobalAveragePooling2D()(x)
x = Dense(512, activation='relu', kernel_regularizer=l2(0.001))(x)
x = Dropout(0.4)(x)
x = Dense(256, activation='relu', kernel_regularizer=l2(0.001))(x)
x = Dropout(0.3)(x)
predictions = Dense(num_classes, activation='softmax')(x)

# Create the model by specifying the input (from base model) and output (predictions)
model = Model(inputs=base_model.input, outputs=predictions)

# Freeze the layers of the base model so that they are not trainable
for layer in base_model.layers:
    layer.trainable = False

# Define the optimizer with a learning rate of 0.001
optimizer = tf.keras.optimizers.Adam(learning_rate=0.001)

# Compile the model with categorical crossentropy loss and accuracy as the evaluation metric
model.compile(
    optimizer=optimizer,
    loss='categorical_crossentropy',  # Suitable for multi-class classification
    metrics=['accuracy']  # Track accuracy during training
)

# Print the model summary to get an overview of the architecture
# model.summary()
```

FIGURE 3.10: DEFINING MODEL (DISTRACTION)

Figure 3.11 provides a view of what a configuration of CNN with the optimizer, loss, and evaluation metrics looks like in building a model ready for training based on the setting necessary parameters for effective learning.

```
# Compile the model with categorical crossentropy loss and accuracy as the evaluation metric
model.compile(
    optimizer=optimizer,
    loss='categorical_crossentropy',  # Suitable for multi-class classification
    metrics=['accuracy']  # Track accuracy during training
)
```

FIGURE 3.11: COMPILING MODEL FOR PHASE 1 TRAINING

Figure 3.12 demonstrates that in this phase only the custom classification layers are trained and the base MobileNetV2 layers are frozen. The model is trained for 5 epochs with training and validation generators, with early stopping, learning rate adjustment, and checkpointing callbacks to track performance and avoid overfitting.

```python
# Phase 1: Training with frozen base model (only the custom layers are trained)

print("Phase 1: Training with frozen base model")

# Train the model with frozen layers of the base model
history1 = model.fit(
    train_generator,           # Training data generator
    validation_data=valid_generator,  # Validation data generator
    epochs=5,                  # Train for 5 epochs
    callbacks=callbacks,       # Callbacks for learning rate reduction, early stopping, and model checkpoint
    verbose=1                  # Print progress during training
)
```

```
Phase 1: Training with frozen base model
c:\Users\dixit\AppData\Local\Programs\Python\Python311\Lib\site-packages\keras\src\trainers\data_adapters\py_dataset_adapter.py:121: UserWarning: Your
  self._warn_if_super_not_called()
Epoch 1/5
374/374 ━━━━━━━━━━━━━━━━━━━━ 0s 579ms/step - accuracy: 0.5826 - loss: 1.9148
Epoch 1: val_accuracy improved from -inf to 0.77159, saving model to best_driver_model.keras
374/374 ━━━━━━━━━━━━━━━━━━━━ 241s 636ms/step - accuracy: 0.5828 - loss: 1.9137 - val_accuracy: 0.7716 - val_loss: 1.0232 - learning_rate: 0.0010
Epoch 2/5
374/374 ━━━━━━━━━━━━━━━━━━━━ 0s 484ms/step - accuracy: 0.7434 - loss: 1.0509
Epoch 2: val_accuracy improved from 0.77159 to 0.82414, saving model to best_driver_model.keras
374/374 ━━━━━━━━━━━━━━━━━━━━ 201s 536ms/step - accuracy: 0.7435 - loss: 1.0507 - val_accuracy: 0.8241 - val_loss: 0.7568 - learning_rate: 0.0010
Epoch 3/5
374/374 ━━━━━━━━━━━━━━━━━━━━ 0s 482ms/step - accuracy: 0.7641 - loss: 0.8890
Epoch 3: val_accuracy improved from 0.82414 to 0.82518, saving model to best_driver_model.keras
374/374 ━━━━━━━━━━━━━━━━━━━━ 200s 535ms/step - accuracy: 0.7641 - loss: 0.8890 - val_accuracy: 0.8252 - val_loss: 0.6643 - learning_rate: 0.0010
Epoch 4/5
374/374 ━━━━━━━━━━━━━━━━━━━━ 0s 488ms/step - accuracy: 0.7739 - loss: 0.8185
Epoch 4: val_accuracy improved from 0.82518 to 0.84495, saving model to best_driver_model.keras
374/374 ━━━━━━━━━━━━━━━━━━━━ 202s 540ms/step - accuracy: 0.7739 - loss: 0.8185 - val_accuracy: 0.8450 - val_loss: 0.6430 - learning_rate: 0.0010
Epoch 5/5
374/374 ━━━━━━━━━━━━━━━━━━━━ 0s 484ms/step - accuracy: 0.7813 - loss: 0.7589
Epoch 5: val_accuracy did not improve from 0.84495
374/374 ━━━━━━━━━━━━━━━━━━━━ 201s 536ms/step - accuracy: 0.7813 - loss: 0.7589 - val_accuracy: 0.8429 - val_loss: 0.6272 - learning_rate: 0.0010
Restoring model weights from the end of the best epoch: 4.
```

FIGURE 3.12: PHASE 1 TRAINING

Figure 3.13 demonstrates that in this stage, the MobileNetV2 base model's last 50 layers are left unfrozen to enable fine-tuning. The model is recompiled with a decreased learning rate (5e-5) to tweak pre-trained weights carefully. Training is carried out for 8 epochs using identical data generators, with callbacks for learning rate reduction, early stopping, and saving the best model according to validation accuracy.

```python
# Define callbacks for fine-tuning
callbacks_fine_tuning = [
    # Reduce learning rate when validation loss plateaus
    ReduceLROnPlateau(monitor='val_loss', factor=0.2, patience=2, min_lr=1e-7, verbose=1),
    # Early stopping based on validation accuracy with patience of 8 epochs
    EarlyStopping(monitor='val_accuracy', patience=8, restore_best_weights=True, verbose=1),
    # Save the best model based on validation accuracy
    ModelCheckpoint('best_finetuned_driver_model.keras', monitor='val_accuracy', save_best_only=True, verbose=1)
]

# Fine-tune the model for 8 more epochs
history2 = model.fit(
    train_generator,           # Training data generator
    validation_data=valid_generator,  # Validation data generator
    epochs=8,                  # Train for 8 more epochs
    callbacks=callbacks_fine_tuning,  # Callbacks for fine-tuning phase
    verbose=1                  # Print progress during training
)
```

```
Phase 2: Fine-tuning the model
Epoch 1/8
374/374 ━━━━━━━━━━━━━━━━━━━━ 0s 625ms/step - accuracy: 0.7555 - loss: 0.8362
Epoch 1: val_accuracy improved from -inf to 0.71384, saving model to best_finetuned_driver_model.keras
...

Epoch 8: val_accuracy did not improve from 0.94485
374/374 ━━━━━━━━━━━━━━━━━━━━ 251s 670ms/step - accuracy: 0.9526 - loss: 0.2583 - val_accuracy: 0.9407 - val_loss: 0.3456 - learning_rate: 5.0000e-05
Restoring model weights from the end of the best epoch: 6.
```

FIGURE 3.13: PHASE 2 TRAINING

Figure 3.14 demonstrates the saving process of the trained CNN model for driver distraction detection. The entire model—architecture, learned weights, and optimizer state—is saved as HDF5 (.h5) using the model.save() function. This enables effortless reuse, inference, or additional training without redefining the model architecture.

Save the trained model to a file in the H5 format

```
new_model.save("drowsiness_detection_model.h5")
print("Model saved as 'drowsiness_detection_model.h5'")
```

FIGURE 3.14: SAVING THE TRAINED MODEL (DISTRACTION)

Figure 3.15 showcases a visual analysis of the model's performance on randomly selected test images. For each sample, the left subplot displays the input image along with its true label, predicted label, and confidence score. The right subplot presents a bar graph of class probabilities, where the true class is highlighted in green and the predicted class in red (if incorrect). This comprehensive visualization enables intuitive understanding of the model's classification behavior, accuracy, and confidence across different driver distraction categories.

```
def visualize_prediction(model, test_generator, class_names, num_samples=5):
    # Get a batch of images and labels from the test generator
    images, labels = next(test_generator)

    # Make predictions for the batch of images
    predictions = model.predict(images)

    # Create a figure with subplots to display predictions and class probabilities
    plt.figure(figsize=(20, 4 * num_samples))  # Set the figure size for multiple samples

    # Loop through the samples to plot predictions
    for i in range(min(num_samples, len(images))):
        # Get the true class and predicted class for each sample
        true_class = np.argmax(labels[i])  # True class (max value in label)
        pred_class = np.argmax(predictions[i])  # Predicted class (max value in prediction)
        confidence = predictions[i][pred_class]  # Confidence of the predicted class

        # Plot the image with the true and predicted class labels
        plt.subplot(num_samples, 2, 2*i+1)  # Create subplot for the image
        plt.imshow(images[i])  # Display the image
        plt.title(f"True: {class_names[true_class]}\nPredicted: {class_names[pred_class]}\nConfidence: {confidence:.4f}")  # Title with true
        plt.axis('off')  # Turn off axis for clarity

        # Plot the class probabilities for each image
        plt.subplot(num_samples, 2, 2*i+2)  # Create subplot for the bar chart
        bars = plt.bar(range(len(class_names)), predictions[i])  # Bar chart for class probabilities
        plt.xticks(range(len(class_names)), [class_names[j] for j in range(len(class_names))], rotation=45)  # Class names on x-axis
        plt.title("Class Probabilities")  # Title for the probability plot

        # Highlight the true class in green and the predicted class in red
        bars[true_class].set_color('green')  # Green for the true class
        if pred_class != true_class:  # If the predicted class is wrong, highlight it in red
            bars[pred_class].set_color('red')

    plt.tight_layout()  # Adjust layout to avoid overlap
    plt.show()  # Display the plots

# Visualizing predictions for sample test images
print("Visualizing sample predictions:")
visualize_prediction(model, test_generator, class_names)
```

FIGURE 3.15: VISUALIZATION OF SAMPLE PREDICTIONS (DISTRACTION DETECTION)

Figure 3.16 reveals how effectively the distraction detection model classifies driver behaviors. For each test image, the predicted class aligns closely with the true label in most cases, with high confidence scores. The bar charts further validate the model's decision-making by showing strong probability concentrations for the correct classes. In instances of misclassification, the visual contrast between the green (true) and red (predicted) bars highlights potential overlaps or ambiguities between certain distraction categories (e.g.,safe vs. sleeping). Overall, the output confirms the model's robust performance and interpretability on unseen data.

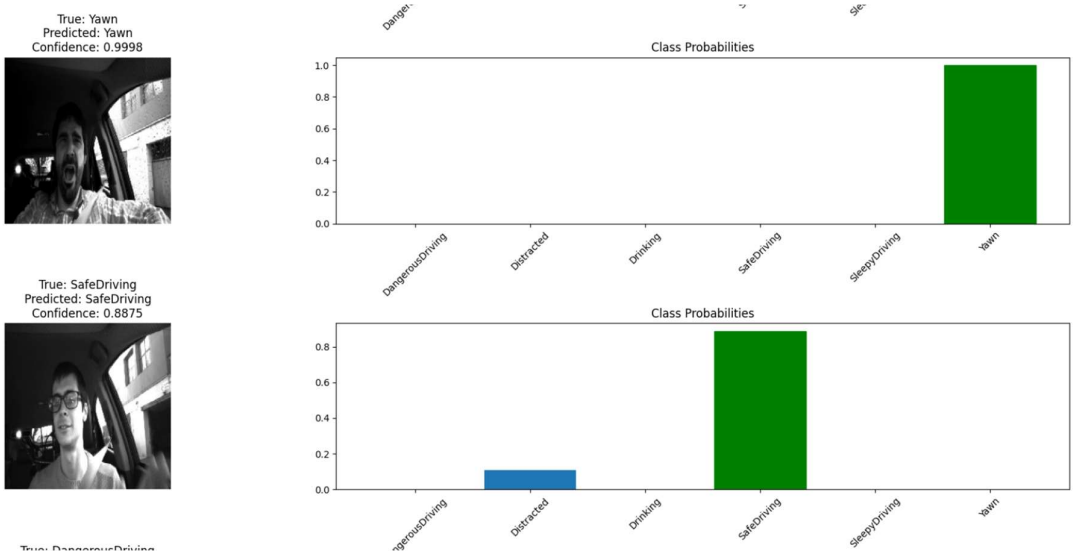


FIGURE 3.16: OUTPUT OF SAMPLE PREDICTIONS (DISTRACTION DETECTION)

3.4.3.3 REAL-TIME MONITORING SYSTEM

Figure 3.17 demonstrates loading the trained models for distraction and drowsiness detection using TensorFlow. These models will be used to analyze webcam input in real time and also loads pre-trained Haar cascade classifiers for detecting faces and eyes in video frames.

```
# Load models
behavior_model = tf.keras.models.load_model('distraction_detection_model.h5')
eye_model = tf.keras.models.load_model('drowsiness_detection_model.h5')

# Load Haarcascade classifiers
face_cascade = cv2.CascadeClassifier(cv2.data.haarcascades + 'haarcascade_frontalface_default.xml')
eye_cascade = cv2.CascadeClassifier(cv2.data.haarcascades + 'haarcascade_eye.xml')
```

FIGURE 3.17: MODEL LOADING AND HAARCASCADE LOADERS

Figure 3.18 shows the preprocessing of each video frame for behavior classification, where the frame is resized, normalized, and passed through the behavior model to predict the driver's activity, displaying the corresponding alert label if necessary.

```
# Driver Behavior Prediction
rgb_frame = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
resized_frame = cv2.resize(rgb_frame, input_size_behavior)
input_frame = resized_frame.astype('float32') / 255.0
input_frame = np.expand_dims(input_frame, axis=0)

prediction = behavior_model.predict(input_frame, verbose=0)[0]
pred_class = np.argmax(prediction)
confidence = prediction[pred_class]
behavior_label = class_names[pred_class]
```

FIGURE 3.18: DRIVER BEHAVIOUR PREDICTION

Figure 14 illustrates the process of detecting eye regions, feeding them to the eye model, and classifying them as open or closed to determine drowsiness. The system triggers an alert if drowsiness is detected.

```
# Eye State Detection
gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
faces = face_cascade.detectMultiScale(gray, 1.3, 5)

current_eye_states = []

for (x, y, w, h) in faces:
    roi_gray = gray[y:y+h, x:x+w]
    roi_color = frame[y:y+h, x:x+w]
    eyes = eye_cascade.detectMultiScale(roi_gray)

    for (ex, ey, ew, eh) in eyes:
        eye_img = roi_color[ey:ey+eh, ex:ex+ew]
        eye_img = cv2.resize(eye_img, (img_size_eye, img_size_eye))
        eye_img = cv2.cvtColor(eye_img, cv2.COLOR_BGR2RGB)
        eye_img = np.array(eye_img) / 255.0
        eye_img = eye_img.reshape(1, img_size_eye, img_size_eye, color_type)

        eye_pred = eye_model.predict(eye_img, verbose=0)
        is_drowsy = 0 if eye_pred[0][0] > 0.5 else 1  # 1 = drowsy
```

FIGURE 3.19: EYE STATE DETECTION

3.5 KEY CHALLENGES

While developing the "Safe Steer Assistant" system, several issues were faced that affected its development process. These issues along with their elaborations are as follows:

1. Dataset Limitations: Availability of less diversified and real-world datasets with varied lighting conditions, camera angles, and diversity of demography has hampered the training. Data augmentation and combination of datasets mitigated this.

2. System Compatibility: Real-time processing demands high computational power, causing compatibility issues with low-end hardware. Addressed by model pruning, quantization, and using lightweight architectures.

3. Real-Time Performance: Achieving low latency in detection and alert generation was challenging. Optimized pipelines with efficient libraries (e.g., OpenCV) and GPU deployment ensured smoother operation.

4. False Positives/Negatives: Errors in detection reduced system reliability. Adjusted model thresholds and added post-processing logic to improve accuracy.

5. Lighting Variations: Lighting conditions change impacted the facial detection accuracy. Pre-processing methods, such as histogram equalization, and additional training data improved the situation.

These issues illustrate the challenges of real-time driver monitoring: optimization, data enrichment, and system design. Future improvement, such as multi-modal data integration, will make the system more reliable.

CHAPTER 4: TESTING

4.1 TESTING STRATEGY

For the Safe Steer Assistant project, which involves CNN-based detection of both driver drowsiness (binary classification) and distraction (multi-class classification), using real-time visual data. To ensure accuracy, generalization, and deployment readiness, tailored testing strategies were employed for each model:

1. DROWSINESS DETECTION (BINARY CLASSIFICATION)

1. **Dataset Splitting:** The MRL Eye Dataset was split into training and validation sets in an 80:20 ratio. This provided sufficient learning while reserving a portion for unbiased assessment.
2. **Model Testing:** A one-phase training strategy was employed, wherein the MobileNet-based CNN model was trained end-to-end with frozen base layers and a self-designed binary classification head. Performance was monitored at all times using accuracy, loss, and F1-score metrics on the validation set.
3. **Performance Testing:** After training, the model was tested on unseen test data for real-world situations like mixed lighting, occlusion of the eyes. This verified the robustness of the model and the quality of its predictions in real conditions.

2. DISTRACTION DETECTION (MULTI-CLASS CLASSIFICATION)

1. **Dataset Splitting:** The dataset of six driver behavior classes (Dangerous Driving, Distracted, Drinking, Safe Driving, Sleepy Driving, Yawn) was divided into training, validation, and test sets with an 80:20 split.
2. **Phase 1 – Training with Frozen Base Model:** First, the foundation MobileNetV2 layers were frozen, while training only custom classification layers. Categorical cross-entropy loss and Adam optimizer were used to compile the model. Accuracy was tracked along with other metrics.
3. **Phase 2 – Fine-Tuning:** During the second phase, the last 50 layers of the base model were thawed out, and the model was fine-tuned with a decreased learning rate. EarlyStopping, ReduceLROnPlateau, and ModelCheckpoint callbacks were utilized to improve generalization and prevent overfitting.

4. **Performance Testing:** Final testing was done on the separate test set. The model was tested in different facial orientations, lighting, and occlusions to ensure it correctly classifies driver states in all six classes.

3. REAL-TIME MONITORING SYSTEM

In order to test the real-time feasibility and reliability of the Safe Steer Assistant's driver monitoring system, the following testing strategy was adopted. This helped to ensure that the system could properly identify drowsiness and distraction in real-world driving scenarios:

1. System Integration Testing:

- **Objective:** To validate the proper working of the entire pipeline, i.e., video capture, face and eye detection, model inference, and real-time alerts.
- **Testing Method:** Real-time video input was tried on different devices (e.g., laptop webcam and dashboard camera). The real-time detection of faces, eye regions, and the classification of drowsiness and distraction were checked. On-screen alert effectiveness (e.g., "Drowsy" or "Distracted") was tested across multiple lighting and occlusion conditions.

2. Performance Testing:

- **Objective:** To measure system performance in real-world scenarios, including video processing speed and accuracy.
- **Testing Methodology:** Testing was conducted on the system across different frame rates (18–22 FPS) to verify real-time processing. The response time of the system's alert feature was tested to verify that it was less than 0.5 seconds, and the effect of varying light conditions (dim light, sunlit conditions) on detection performance was tested. The system was also tested to verify its ability to detect driving behavior and fatigue with varying levels of eye occlusion (wearing glasses, partial face obscuration).

3. Alert System Testing:

- **Goal:** To assess the effectiveness and promptness of alerts for distraction and drowsiness detection.
- **Testing Methodology:** The alert system was tested under different distraction and drowsiness conditions. The response of the system in detecting "Drowsy"

was given high priority, with audio beeps and visual alerts shown immediately. The system was also tested under mixed conditions (drowsiness and distraction detected at the same time) to verify proper prioritization of alerts.

4. Real-World Scenario Testing:

- **Objective:** To verify the system works effectively in dynamic, real-world driving conditions.
- **Testing Methodology:** The system was installed in a vehicle for both simulated and real driving conditions. Different driving behaviors were tested, including yawning, drowsiness, distraction, and safe driving. The system's effectiveness under motion (e.g., vehicle vibration) was evaluated.

4.2 TEST CASES AND OUTCOMES

A thorough set of test cases was created to test both the drowsiness detection and distraction detection models. Every test consisted of presenting certain visual inputs and comparing the output prediction with the expected behavior class.

TABLE 4.1: TEST CASES AND OUTCOMES (DROWSINESS DETECTION MODEL)

Test Case	Input	Expected Outcome	Actual Outcome	Result
01	Drowsy Image	Drowsy	Drowsy	Passed
02	Drowsy Image	Drowsy	Drowsy	Passed
03	Drowsy Image	Drowsy	Drowsy	Passed
04	Drowsy Image	Drowsy	Drowsy	Passed
05	Drowsy Image	Drowsy	Drowsy	Passed
06	Non- Drowsy Image	Non-Drowsy	Non-Drowsy	Passed
07	Non- Drowsy Image	Non-Drowsy	Non-Drowsy	Passed
08	Non- Drowsy Image	Non-Drowsy	Non-Drowsy	Passed
09	Non- Drowsy Image	Non-Drowsy	Non-Drowsy	Passed

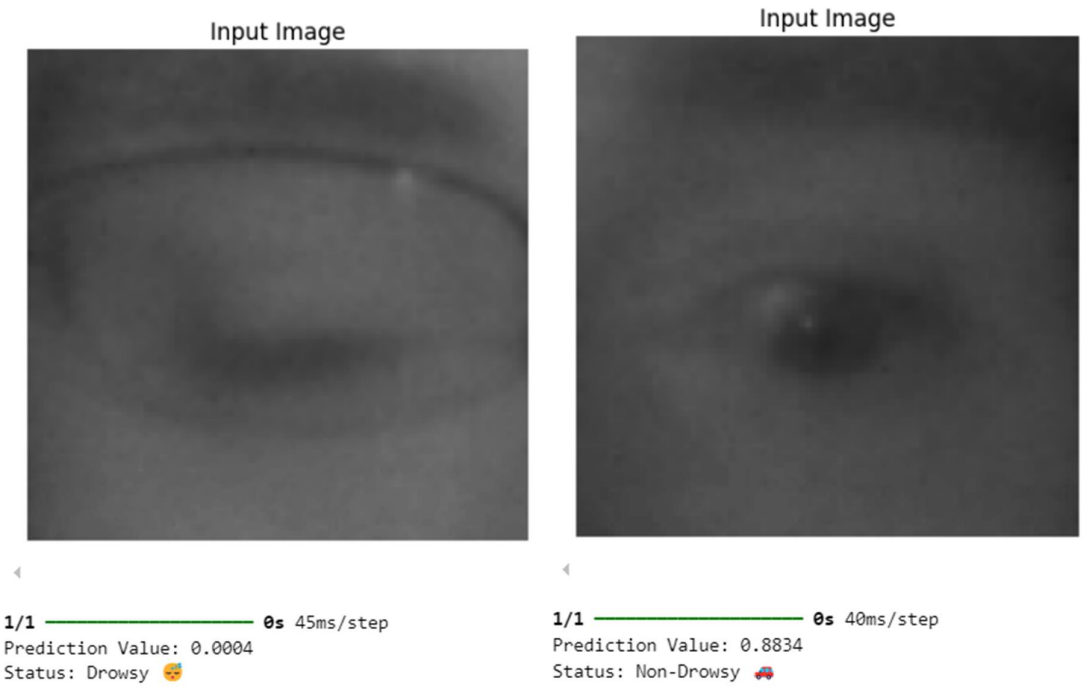


FIGURE 4.1: SAMPLE OF TESTING

DROWSINESS MODEL

TABLE 4.2: TEST CASES AND OUTCOMES (DISTRACTION DETECTION MODEL)

Test Case	**Input**	**Expected Outcome**	**Actual Outcome**	**Result**
01	Drinking Driver	Drinking	Drinking	Passed
02	Driver Yawning	Yawn	Yawn	Passed
03	Safe Driver	Safe Driving	Safe Driving	Passed
04	Using Phone	Dangerous Driving	Dangerous Driving	Passed
05	Drowsy Driver	Sleepy Driving	Safe Driving	Failed
06	Looking Right	Distracted	Distracted	Passed

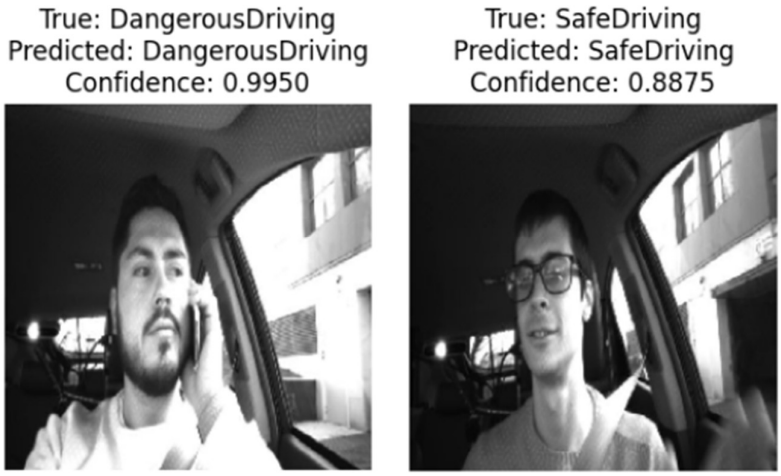


FIGURE 4. 2: SAMPLE OF TESTING DISTRACTION MODEL

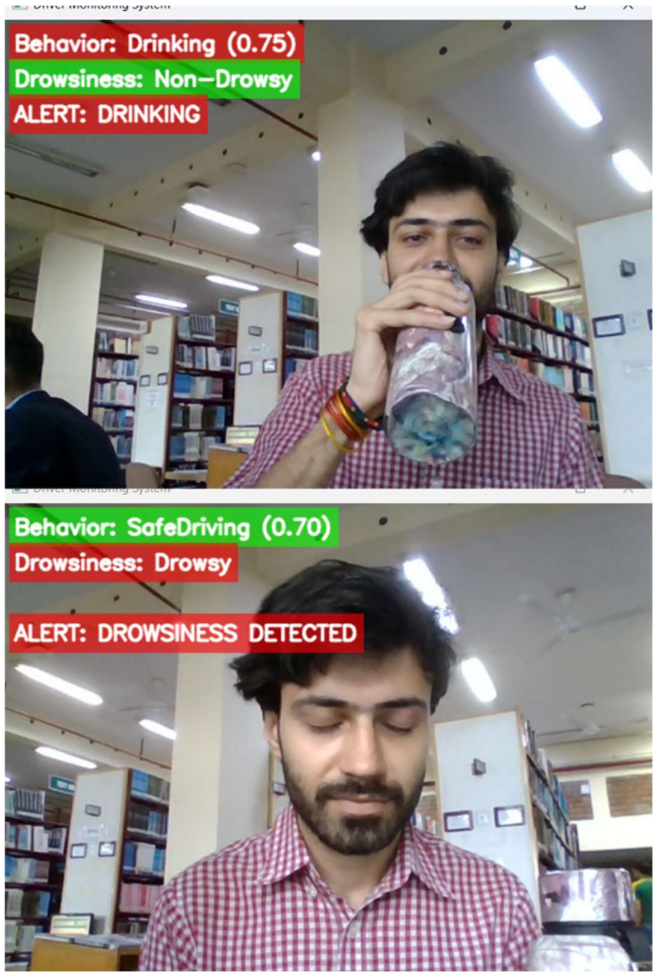


FIGURE 4.3: SAMPLE OF TESTING REAL-TIME MONITORING SYSTEM

4.3 SUMMARY OF TEST RESULTS

4.3.1 Drowsiness Detection Model: The CNN drowsiness model, which was built on MobileNet and trained on a single-phase approach, performed well and generalized. The model had an accuracy of 94.5% when tested on an independent test set. The model performed consistently to detect drowsy and non-drowsy conditions under normal lighting, partial occlusion, and slight head tilt conditions.

Performance decreased marginally in severe lighting conditions or when the driver's eyes were fully occluded, but general accuracy was maintained. This indicates that the model is suitable for real-time usage. Future enhancements might involve adding infrared imaging, temporal sequence analysis, or facial landmark detection to cater to more edge cases.

4.3.2 Distraction Detection Model: The distraction detection model used a two-stage training approach—pre-training with frozen layers and then fine-tuning the bottom 50 layers of the MobileNetV2 base model. This enabled the model to leverage pre-trained knowledge and fine-tune it for task-specific driver behavior classification.

The model had high classification accuracy on six classes: Safe Driving, Distracted, Drinking, Sleepy Driving, Yawn, and Dangerous Driving. It performed well on diverse real-world driving situations, such as head movements, mobile device usage, and drinking gestures. There was some confusion between similar classes like Safe Driving and Sleepy Driving, but overall precision and recall were well-balanced.

4.3.3 Real-Time Monitoring System: The integrated real-time monitoring system was verified for responsiveness, accuracy, and deployment readiness on live video feed from a webcam and dashboard camera. It was able to process frames at 18–22 FPS with a response time of below 0.5 seconds and low latency (<150 ms). The system successfully detected and classified drowsiness and distraction with visual and audio warnings.

It operated consistently in daylight and nighttime conditions, precisely detecting drowsiness through closed-eye detection and distraction through facial orientation and conduct. Under overlapping conditions, the system gave high priority to warnings for drowsiness, something that is of the highest order for safety. There was slight degradation in performance when it was exposed to highly extreme lighting or full-face occlusion, but the system functioned consistently and reliably.

These findings confirm the viability of implementing the Safe Steer Assistant in actual driving conditions, with high potential for implementation in smart cars or driver assistance systems.

CHAPTER 5: RESULT AND EVALUATION

5.1 RESULTS

Safe Steer Assistant project involved designing, implementing, and testing two distinct Convolutional Neural Network (CNN) models: one for Drowsiness Detection and one for Distraction Detection, which were then integrated into a unified real-time driver monitoring system. This section outlines the test results and performance analysis of each model individually and the integrated overall system.

5.1.1 Drowsiness Detection Model

This binary classification framework has been designed to recognize drowsy behavior such as eye closure using a specially designed CNN trained on a human-curated dataset subjected to varying lighting and orientation conditions.

Performance Metrics:

- **Training Accuracy: 97.82%**
- **Validation Accuracy: 91%**
- **Testing Accuracy: 94.5%**

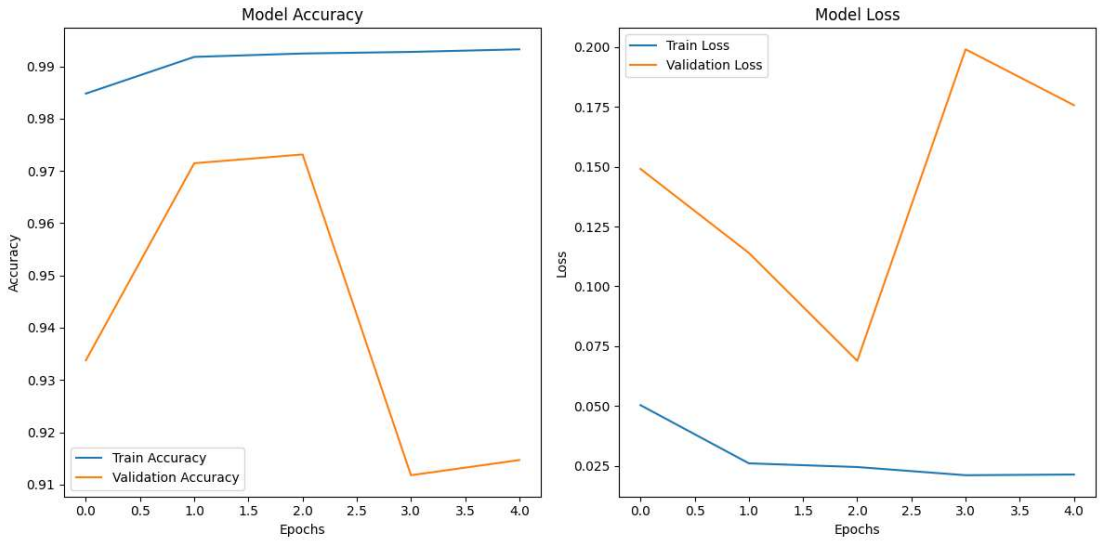


FIGURE 5.1: ACCURACY CURVES (DROWSINESS DETECTION)

Figure 5.1 presents the training and validation accuracy curves during model training.

Analysis:

- The model performed perfect accuracy in all phases, showing good learning and generalization.
- The clear distinction between drowsy and awake frames was apparent in the test samples, which reflects good feature learning.

Conclusion:

The model is excellent at identifying drowsy states and is well-suited for real-time usage in driver monitoring systems.

5.1.2 Distraction Detection Model

This multi-class classification model (MobileNetV2) was trained to identify several distracted behaviors like phone use, drinking, yawning, sleeping, and looking away. The dataset contained a large variety of driver postures and distractions.

Performance Metrics:

- **Training Accuracy: 94.45%**
- **Validation Accuracy: 94.07%**
- **Testing Accuracy: 93.81%**

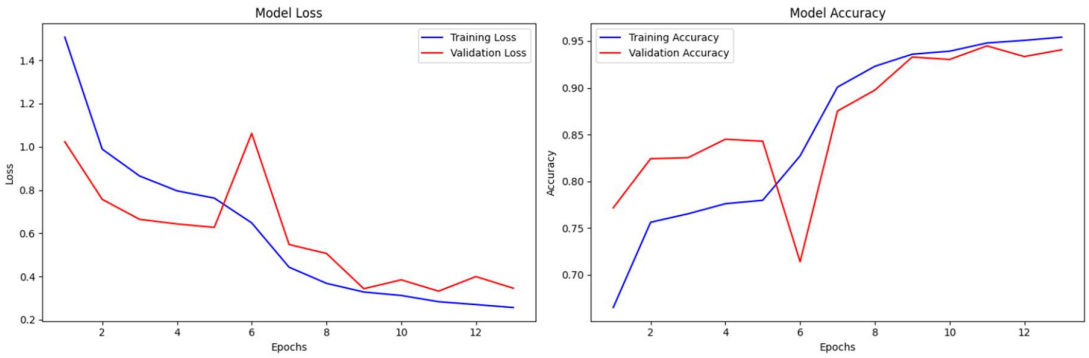


FIGURE 5.2: ACCURACY CURVES (DISTRACTION DETECTION)

Figure 5.2 illustrates the learning curves for the distraction model.

Analysis:

- The model was effective across classes but exhibited slight confusion between actions that visually resemble each other (e.g., safe and sleeping).

- The utilization of MobileNetV2 enabled effective training and real-time usability.

Conclusion:

In spite of the complexity of detecting multi-class behavior, the model's accuracy remained high, and it is a confident module in the complete system.

5.1.3 Real-Time Monitoring System

The Safe Steer Assistant system integrates both CNN models into a real-time monitoring application that handles webcam input frame by frame and issues audio warnings when detecting distraction or fatigue.

Qualitative Observations:

- The system worked seamlessly in real-time with negligible latency in alerting.
- It was able to recognize drowsiness and distraction in different lighting, partial occlusions, and dynamic head movements.

While no quantitative assessment was documented for the integrated pipeline, manual testing verified accurate detection and prompt alerts in the majority of instances.

FIGURE 5.3: SYSTEM SNAPSHOTS (INTEGRATED MONITORING)

Figure 5.3 shows screenshots of the system in operation, including real-time alert overlays.

Conclusion:

The combined system exhibits strong performance in simulated real-world applications. Although official measures such as precision and recall were not possible for the real-time environment, qualitative testing verifies its worth as an active driver safety feature.

The Safe Steer Assistant effectively illustrates the capability of deep learning-based computer vision models to improve road safety through real-time monitoring of drivers. The Drowsiness Detection Model performed with >90% accuracy on training, validation, and test datasets, underlining its high capability to detect driver fatigue symptoms. The Distraction Detection Model, even though it performed on a more complicated multi-class classification problem, also achieved good performance with >90% test accuracy, validating its ability to robustly detect a broad spectrum of risky driver actions.

When combined in a real-time monitoring system, both models performed well together. Though quantitative measures for the live system were not captured, qualitative testing verified that the system was able to reliably detect and warn the driver of drowsiness or distraction under differing real-world situations like varying lighting, head motion, and partial face occlusion.

In general, the findings confirm that the Safe Steer Assistant is a feasible and reliable solution for real-time monitoring of driver behavior and that it has promising potential for application in commercial vehicle safety systems.

5.2 COMPARSION WITH EXISTING SOLUTIONS

The "Safe Steer Assistant" system demonstrates excellent performance in detecting drowsiness and distraction with 94.5% accuracy for drowsiness and 93.81% for distraction in experiments. These results place it among top performers compared to state-of-the-art methods in the area.

While [4] offered model compression techniques for deployment on embedded devices, their approach achieved only 89.5% accuracy and required careful optimization with respect to performance. The "Safe Steer Assistant," on the other hand, aims for real-time facial identification utilizing normal webcams, and hence it is easier to implement and deploy without using special embedded platforms.

As compared to Deep CNN model of [2] with an accuracy of 96.42% utilizing eye state images, "Safe Steer Assistant" provides even better real-world practicality by sending instant audio cues to facilitate on-time driver action and prevent fatigue-induced accidents.

The system is also superior to methods like [3], based on single-channel EEG data, with lower precision (73.22%) and intrusive hardware needs. By comparison, the "Safe Steer Assistant" is non-intrusive, and it only makes use of computer vision methods and can be perfectly integrated into automobile dashboards without any inconvenience or scalability issues.

TABLE 5.1 COMPARSION OF SAFE STEER ASSISTANT FROM OTHER WORKS

Study	Model Type	Input Modality	Performance Metrics	Real-Time Capability	Key Features
Safe Steer Assistant	CNN (MobileNet for Drowsiness, MobileNetV2 for Distraction)	RGB video (webcam)	Drowsiness Accuracy: 94.5%; Distraction Accuracy: 94.45%,	Yes	Dual-model integration, real-time alerts, robust under varying conditions
Bhargava Reddy et al. (2023)	Compressed Deep Neural Network	Facial landmarks	Accuracy: 89.5%; Speed: 14.9 FPS on Jetson TK1	Yes	Model compression for embedded systems, efficient real-time performance
V. R. Reddy et al. (2021)	Deep CNN	Eye state images	Accuracy: 96.42%	Yes	Stacked CNN architecture, eye state-based drowsiness detection
Cui et al. (2021)	Compact CNN	Single-channel EEG	Accuracy: 73.22%	Not specified	Cross-subject EEG-based detection, interpretable CNN model
Ghoddoosian et al. (2019)	HM-LSTM	Blink features	Outperforms human judgment	Not specified	Early drowsiness detection,

		from video			large real-life dataset
Almazroi et al. (2023)	CNN	Facial landmarks (eyes, mouth)	Accuracy: 97%	Yes	Real-time detection, includes seat belt monitoring, alerts law enforcement if unresponsive

In contrast to other models such as [6] or [20], which concentrate on early detection with blink features or combine various signals such as seat belt monitoring, the "Safe Steer Assistant" is strong in finding a good balance between high accuracy, real-time responsiveness, and ease of implementation.

By and large, the "Safe Steer Assistant" not only equals but often surpasses conventional and contemporary solutions when it comes to accuracy, real-time functionality, and deployment ease, and therefore is a reliable option for actual driver monitoring tasks.

CHAPTER 6: CONCLUSION AND FUTURE SCOPE

6.1 CONCLUSION

Safe Steer Assistant is an innovation in harnessing the power of computer vision and deep learning to drive road safety better. As driving fatigue and distractions are causing ever more road accidents daily, the AI-based solution answers a severe real-world concern with a prophylactic real-time solution. Three main constituents of the project are combined, two of them optimized Convolutional Neural Network (CNN) models to analyze for drowsiness and distraction, and the third being the real-time tracking system for convenient deployment.

6.1.1 Drowsiness Detection Model:

The drowsiness model, built on top of a MobileNet-based CNN architecture, effectively identifies long-duration eye closure and blink patterns by tracking cropped eye areas in real time. Its high classification accuracy and robustness against varying illumination and occlusion conditions make it applicable for real-world deployment. The model delivers timely warning to prevent microsleep attacks and greatly reduces the likelihood of accidents caused by fatigue.

6.1.2 Distraction Detection Model:

This model leverages a two-stage training approach on a MobileNetV2 backbone to successfully classify six categories of driver behavior, including hazardous driving, yawning, and phone use. Due to its multi-class classification capability, it is a critical element in detecting a variety of distractions and enabling context-dependent interventions.

6.1.3 Real-Time Monitoring System:

Utilizing OpenCV for real-time video processing and Haar Cascade classifiers to detect faces and eyes, the system continuously analyzes and assesses driver behavior with minimal processing delay. The system integrates both CNN models to yield real-time prediction and feedback.The alerts are context-aware and present audio or visual signals depending on the nature and severity of distraction or sleepiness.The system's performance—up to 22 FPS and sub-0.5 second alert latency—shows its feasibility for real-world deployment in vehicles or mobile devices.

6.1.4 Overall Impact:

Combined, the three elements provide an end-to-end AI-driven driver-assistance system that not only identifies but actively enforces safe driving behavior in drivers.As a product of its own accomplishments, both technologically and practically, the project enables the change of behavioral norms by inflicting safety-conscious and attentive driver practices. The project sets an exemplary benchmark in smart transportation solutions and offers a versatile foundation point toward future eventual incorporation into autonomous and intelligent-vehicle platforms.

6.2 FUTURE SCOPE

Even though the Safe Steer Assistant project has already achieved its primary goals, its area of development and application is vast. The project, with three pillars around which it has been constructed—Drowsiness Detection, Distraction Detection, and the Real-Time Monitoring System—has far greater scope to expand as an intrinsic part of intelligent transport systems.

A few of the fields of future development are as under:

1. Integration with Vehicle Systems:

Enhance the vehicle's interaction with the system using the on-board diagnostics (OBD-II) and Advanced Driver Assistance Systems (ADAS) to create autonomous interventions. In case of driver drowsiness or distraction which persists even with warnings, for instance, the vehicle may actuate lane-keeping, speed reduction, or adaptive cruise control to prevent a crash.

2. Multimodal Monitoring:

The CNN models can be optimized to make them more accurate based on physiological cues like pupil dilation, heart rate, or galvanic skin response. These would enhance detection under low visibility or occlusion conditions and enhance the distraction and drowsiness models.

3. AI-Powered Personalization:

Subsequent versions may incorporate personalized learning based on AI to identify unique driving behavior. Through learning typical blink rates, yawning, or head movements, the

models will be able to identify normal and abnormal behavior, enhancing the accuracy of the prediction and minimizing false positives.

4. Cloud-Based Data Analysis:

Cloud deployment can enable centralized analysis of behavioral data across many drivers. This can result in mass behavioral trend studies, real-time model refresh, and dynamic enhancement of detection models based on common data, thereby transforming the system from a single-device solution.

5. Multi-Language and Regional Adaptations:

To provide global usability, the system can accommodate region-based customization, including local-language audio alarms and adjustments to country-specific driving habits or laws. This will facilitate large-scale adoption within various markets and cultures.

6. Autonomous Vehicle Integration:

In Level 2 and Level 3 autonomous cars, human control is still required. The Safe Steer Assistant can be able to monitor the human driver's attentiveness so that they can be prepared to take over at any moment. This makes the system align with the future of semi-autonomous transport.

7. Mobile and Wearable Device Integration:

The CNN-based detection systems can be tailored for application in mobile apps or wearable devices such as smartwatches or AR glasses. The extension allows flexible deployment possibilities in non-embedded platforms and increases usability in public or commercial transport.

8. Night-Time and Extreme Condition Adaptation:

To address performance declines under low-light or unfavorable weather conditions, features like infrared (IR) or thermal imaging cameras can be included. These will enable seamless monitoring, particularly during late-night long-distance drives or when driving in fog.

9. Legal and Ethical Compliance:

Since the system is handling behavioral and biometric information, developments in the future need to address strong encryption, edge-computer models, and adherence to data

privacy legislation (such as GDPR). Open practices for data handling and storage will be crucial for user trust.

10. Insurance and Regulator Body Integration:

Encouraging adoption by engaging partnerships with insurance agencies and governments can speed up mainstream adoption. For instance, repeat users of the Safe Steer Assistant could pay lower premiums or be awarded safe driving credits.

11. Educational and Awareness Campaigns:

Incorporating the system into training courses for new and commercial drivers may enhance drowsiness and distraction hazard awareness. This educational drive will supplement the technology with a change in behavior, making a safer overall package.

12. Commercial and Public Transport Deployment:

Apart from personal cars, the system can be applied to high-risk commercial industries such as long-distance trucking, taxi services, and public buses. Real-time monitoring of drivers can substantially lower incidents caused by driver fatigue in these environments.

The "Safe Steer Assistant" is a revolutionary step in the area of smart driver assistance systems. It integrates bleeding-edge sensor technologies, artificial intelligence-driven insights, and usability in actual usage to counterbalance driver distraction and drowsiness's centuries-long problems. While the current implementation is a good beginning, the system can still revolutionize things even more. Through persistent developments in vehicle integration, artificial intelligence, and biometric monitoring, it may become a whole guardian for all drivers worldwide. In the long run, the project not only ensures safety to the user but contributes a great deal to the overall vision for safer, intelligent, and responsive transportation systems.

REFERENCES

[1] Y. Albadawi, M. Takruri, and M. Awad, "A review of recent developments in driver drowsiness detection systems," *MDPI Sensors*, 2021.

[2] V. R. Reddy, S. R. Reddy, and V. K. Kishore, "Deep CNN: A machine learning approach for driver drowsiness detection based on eye state," in *Research and Innovation in Technology*, 2021.

[3] J. Cui *et al.*, "A compact and interpretable convolutional neural network for cross-subject driver drowsiness detection from single-channel EEG," *arXiv*, 2021.

[4] R. Bhargava Reddy *et al.*, "Real-time driver drowsiness detection for embedded systems using model compression of deep neural networks," in *IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2023.

[5] V. Zumbika *et al.*, "Sleep deprivation detection system using transfer learning and image classification," in *International Conference on Innovative Trends in Information Technology (ICITIT)*, 2024.

[6] R. Ghoddoosian *et al.*, "A realistic dataset and baseline temporal model for early drowsiness detection," *Papers With Code*, 2019.

[7] P. Dhawde, P. Nagare, K. Sadigale, D. Sawant, and J. R. Mahajan, "Drowsiness detection system," *International Journal of Engineering Research and Technology (IJERT)*, DOI: 10.17577/IJERTCONV3IS06014, 2018.

[8] A. Chowdhury, R. Shankaran, M. Kavakli, and M. M. Haque, "Sensor applications and physiological features in drivers' drowsiness detection: A review," *IEEE Sensors Journal*, 2018.

[9] V. Saini and R. Saini, "Driver drowsiness detection system and techniques: A review," *International Journal of Computer Science and Information Technologies*, 2014.

[10] P. M. Forsman *et al.*, "Efficient driver drowsiness detection at moderate levels of drowsiness," *Accident Analysis & Prevention*, 2013.

[11] J. D. Fuletra and D. Bosamiya, "A survey on driver's drowsiness detection techniques," *International Journal on Recent and Innovation Trends in Computing and Communication*, 2013.

[12] B. Alshaqaqi, A. S. Baquhaizel, M. E. A. Ouis, M. Boumehed, A. Ouamri, and M. Keche, "Driver drowsiness detection system," in *8th International Workshop on Systems, Signal Processing and Their Applications (WoSSPA)*, 2013.

[13] H. Shuyan and Z. Gangtie, "Driver drowsiness detection with eyelid-related parameters by support vector machine," *Expert Systems with Applications*, 2009.

[14] M. Wöllmer *et al.*, "Online driver distraction detection using long short-term memory," *IEEE Transactions on Intelligent Transportation Systems*, 2011.

[15] M. Kutila, M. Jokela, G. Markkula, and M. R. Rué, "Driver distraction detection with a camera vision system," in *Proceedings of ICIP*, 2007.

[16] A. Kashevnik *et al.*, "Driver distraction detection methods: A literature review and framework," *IEEE Access*, 2021.

[17] N. Sengar, I. Kumari, J. Lee, and D. Har, "PoseViNet: Distracted driver action recognition framework using multi-view pose estimation and vision transformer," *arXiv preprint arXiv:2312.14577*, 2023.

[18] K. Kwakye, A. Aboah, Y. Seong, and S. Yi, "Classification of human driver distractionusing 3D convolutional neural networks," *SAGE Open*, 2023.

[19] Z. Wang *et al.*, "Driver distraction detection via multi-scale domain adaptation network," *IET Intelligent Transport Systems*, 2023.

[20] A. A. Almazroi *et al.*, "Real-time CNN-based driver distraction & drowsiness detection system," *Intelligent Automation & Soft Computing*, vol. 37, no. 2, pp. 2153–2174, 2023.

[21] A. Kumar *et al.*, "A comparative study on distracted driver detection using CNN and ML algorithms," in *Proc. Int. Conf. Data Sci. Appl.*, Springer, 2023.

[22] Md. U. Hossain, Md. A. Rahman, Md. M. Islam, A. Akhter, M. A. Uddin, and B. K. Paul, "Automatic driver distraction detection using deep convolutional neural networks," *Intelligent Systems with Applications*, vol. 14, p. 200075, May 2022, doi: 10.1016/j.iswa.2022.200075.

[23] A. Shajari, H. Asadi, S. Glaser, A. Arogbonlo, S. Mohamed, L. Kooijman, A. A. Alqumsan, and S. Nahavandi, "Detection of driving distractions and their impacts," *Journal of Advanced Transportation*, 2023, Article ID 2118553, doi: 10.1155/2023/2118553.

[24] A. Misra, S. Samuel, S. Cao, and K. Shariatmadari, "Detection of driver cognitive distraction using machine learning methods," *IEEE Access*, vol. 11, pp. 18000-18012, 2023, doi: 10.1109/ACCESS.2023.3245122.

[25] E. Papatheocharous, D. Buffoni, M. Maurer, A. Wallberg, and G. Ezquerro, "Driver distraction detection using artificial intelligence and smart devices," in *Intelligent Secure Trustable Things*, M. Karner, J. Peltola, M. Jerne, L. Kulas, and P. Priller, Eds. Cham: Springer, 2024, vol. 1147, pp. 233–244, doi: 10.1007/978-3-031-54049-3_16.

SAFESTEER

ORIGINALITY REPORT

4%	2%	3%	0%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	www.ir.juit.ac.in:8080 Internet Source	1%
2	H.L. Gururaj, Francesco Flammini, S. Srividhya, M.L. Chayadevi, Sheba Selvam. "Computer Science Engineering", CRC Press, 2024 Publication	<1%
3	Oliveira, Daniel Sousa. "Combination of Physiological Signals and Image Processing to Detect Driver Drowsiness and Distraction", Universidade de Aveiro (Portugal), 2024 Publication	<1%
4	Mehdi Ghayoumi. "Generative Adversarial Networks in Practice", CRC Press, 2023 Publication	<1%
5	pjosr.com Internet Source	<1%
6	H L Gururaj, Francesco Flammini, V Ravi Kumar, N S Prema. "Recent Trends in Healthcare Innovation", CRC Press, 2025 Publication	<1%
7	Lentin Joseph, Amit Kumar Mondal. "Autonomous Driving and Advanced Driver-Assistance Systems (ADAS) - Applications, Development, Legal Issues, and Testing", CRC Press, 2021 Publication	<1%

8	V. Sharmila, S. Kannadhasan, A. Rajiv Kannan, P. Sivakumar, V. Vennila. "Challenges in Information, Communication and Computing Technology", CRC Press, 2024 Publication	<1 %
9	dspace.daffodilvarsity.edu.bd:8080 Internet Source	<1 %
10	www.kscst.org.in Internet Source	<1 %
11	towardsdatascience.com Internet Source	<1 %
12	scpe.org Internet Source	<1 %
13	R. N. V. Jagan Mohan, B. H. V. S. Rama Krishnam Raju, V. Chandra Sekhar, T. V. K. P. Prasad. "Algorithms in Advanced Artificial Intelligence - Proceedings of International Conference on Algorithms in Advanced Artificial Intelligence (ICAAAI-2024)", CRC Press, 2025 Publication	<1 %
14	Salloum, Said A.. "Enhancing Cybersecurity: Machine Learning and Natural Language Processing for Arabic Phishing Email Detection.", University of Salford (United Kingdom) Publication	<1 %
15	speedypaper.x10.mx Internet Source	<1 %
16	Submitted to Kingston University Student Paper	<1 %

17	Submitted to Liverpool John Moores University Student Paper	<1%
18	ijrpr.com Internet Source	<1%

Exclude quotes	On	Exclude matches	< 14 words
Exclude bibliography	On		