Alertdriver: Early Detection Of Fatigue States In Drivers During Long Driving Stretches

D. DINESH[1], S. PRAKASAM[2]
[1, 2] Dept of CSA
[1, 2] MCA,Sri Chandrasekharendra Saraswathi Viswa Mahavidyalaya
(SCSVMV) University

Abstract- *The majority, of the accidents were happening perpetually due to driver drowsiness over the decades. Automation has been playing key role in many fields to provide conformity and improve the quality of life of the users. Though various drowsiness detection systems have been developed during lost decade based on many factors, still the systems were demanding an improvement in terms of efficiency, accuracy, cost, speed and availability, etc. In this paper, proposed an integrated approach depends on the Eye closure status along with the calculation of the new proposed vector FAR (Facial Aspect Ratio) . This helps to find the status of the closed eyes or opened and any frame finds that has hand gestures like nodding or covering opened mouth with a hand as innate nature of humans when trying to control the sleepiness. The system also integrated the methods and textural-based gradient patterns to find the driver's face in various directions identify the sun glasses on the driver's face and the scenarios like hands-on eyes while nodding were also recognized and addressed.*

Keywords- Computer Vision, Eye State Classification, Fatique Detection, Face and Eye Detection, Web-Cam Surveillance, Real-time Monitoring.

I. INTRODUCTION

A large number of people across the world want to buy vehicles. Drowsy driving is one of the leading causes of road accidents worldwide, accounting for thousands of injuries and fatalities each year. Fatique impairs a driver's attention, reaction time, and decision-making ability, posing serious threats to public safety. As many of these incidents occurs due to unnoticed or unaddressed driver drowsiness, there is a growing need for intelligent systems that can monitor and alert drivers in real-time before accidents happen. This project proposes a real-time Driver Drowsiness Detection System using computer vision and deep learning technologies. The system is designed to be non-intrusive, cost-effective, and capable of functioning with standard hardware such as webcam. A pre-trained Convolutional Neural Network (CNN) classifies the eye state, and the system keeps track of how long the eyes remain closed. If the driver's eyes are detected as closed for a specific number of consecutive frames, the system interpretes it as a sign of drowsiness and triggers and audible alarm. This immediate response helps the driver regain alertness or take corrective action, potentially preventing serious accidents. The system leverages tools like Open CV, Keras and Python threading to ensures real-time performance and responsiveness. By providing an affordable and scalable solution, this project aims to enhance road safety and contribute to the growing field of intelligence driver assistance systems (ADAS).

II. IDENTIFY,RESEARCHANDCOLLECT IDEA

Drowsiness is one of the primary causes of road accidents, particularly among commercial drivers, night-shift workers, and individuals driving long distances. According to studies by global traffic safety organizations, driver fatigue is responsible for a significant percentage of road accidents, injuries, and fatalities every year. Detecting drowsiness early can prevent these incidents and save lives. The idea behind the **Drowsiness Detection System** is to use technology to monitor the user's alertness in real time and trigger a warning before an accident occurs.

Problem Identification : Traditional safety mechanisms like seatbelts, airbags, or anti-lock braking systems react only after an accident starts or occurs. However, drowsiness detection is **preventive**, identifying potential danger based on human behavior. Human monitoring of drivers is impractical and inefficient, especially in individual or small vehicle contexts. Hence, there is a growing need for automated systems that can monitor human fatigue through observable signs, such as **eye behavior**, in real time.

By identifying patterns like **frequent blinking**, **eye closure**, or **head nodding**, a system can alert the driver before their alertness drops dangerously low. The goal of this project is to implement a **real-time drowsiness monitoring solution** using widely available hardware and open-source software tools, making it accessible, scalable, and cost-effective.
Research and Feasibility Study :

During the phase, various methods of detecting drowsiness were studied:

1. Behavioral Methods – Monitor physical behaviors such as yawning, eye closureand facial expressions.
2. Physiological Methods : Measure heart rate, brain waves (EEG) and pulse through wearable devices.
3. Vehicle-based Methods : Detect sudden steering movements, lane deviations and braking patterns.

Among these , behavioral methods using image processing and Deep learning were chosen for this project due to :

- Their non-invasive nature (no sensors attached to the body)
- Availability of webcams on most devices
- Suitability for real-time application

OpenCV was selected for image and video processing because of its efficiency and widespread use in the computer vision community. For eye state classification, a **Convolutional Neural Network (CNN)** was chosen due to its high accuracy in image-based pattern recognition tasks. Haarcascade classifiers were used for fast and efficient **face and eye detection**.

Python was the language of choice due to its large ecosystem of libraries and ease of implementation for both machine learning and computer vision tasks.

Idea Collection and Design Insight : The core idea is to use the webcam to continuously monitor the user's face and detect the state of the eyes. The working pipeline is as follows:

1. **Capture webcam video frames** in real-time.
2. Use **Haarcascade classifiers** to detect the **face** and **eyes** in each frame.
3. Extract and preprocess the eye region (resizing, normalization).
4. Pass the cropped eye image to a **trained CNN model** that classifies the eye as open or closed.
5. Maintain a **counter** for the number of consecutive frames with both eyes closed.
6. If the counter exceeds a threshold (e.g., 5 frames), **trigger an alarm sound** to alert the user.
7. Use **multithreading** to play the alarm sound without freezing the webcam feed.

III. WRITEDOWNYOURSTUDIESAND FINDINGS

The development of the Drowsiness Detection System began with a study of the increasing rate of accidents caused by driver fatigue and drowsiness. Research showed that one of the main contributing factors to drowsy driving is the lack of alertness and reduced eye activity. Based on this, the project aimed to develop a real-time system that can track the eyes of a user and identify signs of drowsiness using computer vision and deep learning.

Human Behavior and Drowsiness Indicators

The first area of study focused on understanding **how drowsiness manifests physically**. Research revealed that eye behavior—specifically **eye closure**, **blink rate**, and **duration**—are reliable indicators of fatigue. When a person is drowsy, they tend to blink more slowly, and their eyes may remain closed for longer periods.

Existing Drowsiness Detection Approaches

Several methods exist for detecting drowsiness:

- **Physiological methods** using EEG and heart rate monitoring.
- **Behavioral methods** using video analysis of eye and facial movements.
- **Vehicle-based methods** monitoring steering and brake patterns.

Technology Stack

- **OpenCV** was selected for real-time image capture and face/eye detection.
- **Haarcascade classifiers** were used to detect facial features due to their speed and efficiency.
- A **Convolutional Neural Network (CNN)** model was trained to classify eyes as either "Open" or "Closed".
- **Multithreading** was implemented to handle alarm triggering without freezing the video feed.
- **Python** was the main programming language due to its versatility and strong support for AI and computer vision libraries.

Findings During Implementation

A. Face and Eye Detection

- Haarcascade classifiers performed well in detecting frontal faces and eyes in good lighting.

- Detection accuracy dropped in dim lighting or if the face was partially covered (e.g., by glasses or hair).
- Fine-tuning the scale factor and minNeighbors parameters improved detection stability.

B. CNN-Based Eye Classification

- The trained CNN model was able to classify cropped eye images with high accuracy (~95%) under normal lighting.
- The model struggled with blurry or low-resolution input, indicating the need for preprocessing.
- Normalizing pixel values and maintaining a consistent image size (145x145 pixels) helped maintain prediction consistency.

C. Real-Time Performance

- Average frame processing rate was around 25–30 FPS on a mid-range system.
- Alarm sound played within 1 second of detecting 5+ closed-eye frames, which was acceptable for real-time safety alerts.
- Multithreading successfully prevented video freezing while playing the alarm.

D. False Alarms and Blink Handling

- A common challenge was distinguishing between normal blinking and actual drowsiness.
- Introducing a frame counter to track **continuous eye closure** effectively reduced false positives.
- The system was tuned to tolerate brief closures (blinks) but trigger alerts if eyes remained closed for 5 or more frames..

IV. GETPEERREVIEWED

1. Lighting Sensitivity
- The Haarcascade classifiers and camera input depend heavily on ambient lighting.
- In low light or backlit conditions, face and eye detection accuracy drops significantly.

2. Poor Detection with Glasses or Sunglasses
- The system may struggle to detect eyes if the user wears glasses, especially reflective or dark-tinted ones.
- Eye classification performance also drops due to distorted or obscured input images.

3. Limited Pose Tolerance
- Haarcascade detection works best with frontal face orientation.
- If the user turns their head slightly (e.g., looking sideways), the eyes may not be detected, reducing system reliability.

4. Single-Person Focus
- The system is designed to track only one face at a time (usually the largest one in frame).
- It is not suitable for environments where multiple people need to be monitored simultaneously.

5. Blink Detection Ambiguity
- Normal blinking can sometimes be misclassified as drowsiness, especially if a user blinks slowly.
- While frame counting helps reduce false alarms, it's not always perfect under varied blinking behavior.

6. No Learning or Adaptability
- The CNN model used is static-once trained, it cannot adapt to different eye shapes, skin tones, or lighting conditions unless retrained.
- This reduces its generalization ability across diverse users.

7. Dependence on Webcam Quality
- Low-resolution or laggy webcams can result in blurry eye images, which degrade prediction accuracy.
- The system may fail on lower-end hardware with slow frame rates.

8. Alarm Limitations
- The alarm is a simple static audio file played using playsound.
- There is no customization, volume control, or visual alert for users with hearing impairments.

9. No Long-Term Data Logging
- The system provides no dashboard or logging features to track user behavior over time.
- In safety-critical settings like fleet monitoring, lack of logs limits its usefulness.

10. No Detection of other Drowsiness Symptoms

- The system only considers eye state and ignores other important indicators like: Yawning, Head Nodding and Slouched posture.

V. IMPROVEMENT AS PER REVIEWER COMMENTS

1. Improve Detection Under Poort Lighting
   - Integrate adaptive histogram equalization (CLAHE) or gamma correction to enhance image brightness dynamically.
   - Explore using infrared (IR) cameras or low-light webcam modules in future versions.
   - Implement an auto-detection module that notifies the user if lighting is insufficient for accurate detection.

2. Handle Users Wearing Glasses
   - Replace Haarcascade classifiers with more robust facial landmark detection libraries such as Mediapipe or Dlib, which work better with glasses.
   - Augment the training dataset with more samples of eyes behind glasses to improve CNN generalization.
3. Enhance Face and Eye Detection Accuracy
   - Switch to deep learning-based face detectors like MTCNN or YOLO face detection to allow for side-angle detection.
   - Implement head pose estimation to warn the user if their face turns away from the camera.
4. Prevent False Alarms During Blinking
   - Implement **blink duration measurement** and adjust the threshold dynamically.
   - Use **moving average techniques** or time-based smoothing to differentiate quick blinks from prolonged eye closures.

5. Make Alarm System More User Friendly
   - Add **volume control**, **snooze option**, and support for **voice alerts**.
   - Introduce **visual alerts** (e.g., flashing screen) for users who are hard of hearing.
   - Allow users to **select different alarm tones** from a settings file.

6. Support for Multiple Users
   - Enhance multi-face tracking using **OpenCV's multi-object tracker** or YOLO with person ID assignment.
   - Limit alarm triggers to the primary user by default (e.g., the largest detected face).

7. Add Logging and Reporting Feature
   - Add a module to log detection events, including timestamps, eye status, and alarm triggers.
   - Generate daily or weekly drowsiness reports for analysis, especially useful in fleet or workplace safety settings.

8. Mobile / Embedded System Compatibility
   - Optimize the CNN model using TensorFlow Lite or ONNX for mobile deployment.
   - Prototype the system on platforms like Raspberry Pi for edge computing use cases.

9. Increase Model Accuracy with More Training Data
   - Retrain the CNN model with a larger and more diverse dataset that includes:
     - Different skin tones
     - Eye shapes
     - Lighting variations
     - Users with and without glasses
   - Apply data augmentation techniques (rotation, brightness adjustment, flipping) to increase the robustness of the model.

10. Add User Interface (GUI) for Better Usability
    - Develop a simple Graphical User Interface (GUI) using libraries like Tkinter, PyQt, or Kivy.
    - GUI features to include:
      - Start/Stop Monitoring button
      - Real-time status indicator (e.g., "Monitoring", "Drowsy")
      - Alarm settings (volume, tone)
      - Display of last detection result or log summary.

VI. CONCLUSION

The Driver Drowsiness Detection System presented in this project demonstrates a practical and efficient solution for enhancing road safety by identifying early signs of driver fatigue. By leveraging real-time computer vision techniques and a deep learning model, the system successfully detects the state of a driver's eyes through a webcam feed and alerts them when drowsiness is suspected.

The project uses Haar cascade classifiers for detecting facial and eye features and a trained Convolutional Neural Network (CNN) to classify eye states as open or closed. When both eyes are detected as closed over a defined frame threshold, an alarm is triggered to awaken or alert the driver. This approach offers a non-intrusive and cost-effective method of monitoring driver alertness, without the need for complex hardware or wearable devices.

In conclusion, this project not only highlights the power of AI in preventive safety applications but also serves as a foundation for developing advanced driver-assistance systems (ADAS) aimed at reducing fatigue-related accidents and saving lives on the road.It combines the strengths of both indirect and direct monitoring techniques and provides a scalable framework for further enhancement. Future work will aim to improve accuracy under diverse environmental conditions, expand the dataset with real-world driving scenarios, and explore integration with other sensors such as infrared cameras and eye-tracking devices. Some promising future improvements include the integration of yawning detection, head pose estimation, and blink rate analysis, which offer complementary indicators of fatigue.

APPENDIX

The appendix includes supplementary information that supports the research, such as system configurations, software tools used, code snippets, and design diagrams referenced throughout the study. This section is useful for readers who want to replicate or build upon the system.

A. System Configuration

Hardware:

- Processor: Intel Core i5
- RAM: 8 GB
- Hard Disk: 500 GB

Software:

- Operating System: Windows 10 (64-bit)
- Front End : HTML, CSS
- Scripts : Python Language
- Tool : Python Idle

B. Functional Modules Overview

1. Dataset Collection
   - Source: Closed eyes in the wild (CEW)
   - Types: Labeled images for drowsy and alert state
   - Size: ~2,400 images

2. Preprocessing Steps
   - Image resizing to 145×145 px
   - Normalizing pixel values (0–1 range)
   - Converting to array and reshaping
   - Ensures the input format is compatible with the CNN model.

3. Feature Detection
   - Haarcascade Classifiers (Viola-Jones Algorithm) : Used for: Detecting face, left eye, and right eye
   - Convolutional Neural Network (CNN) : While CNNs are used mainly for classification, they also extract high-level visual features from the eye image such as:
     - Shape of eyelids
     - Eyelash and iris patterns
     - Texture and contrast
4. Classification
   - Algorithm: Convolutional Neural Network (CNN)
   - Labels: Edges, Shapes, Eyelid patterns
   - Model Input: Open Eyes, Closed Eyes, Frame Count , Alarm sound.
5. Real-Time Workflow
   - Webcam Initialization → Frame Capture → Face and Eye Detection→Image Preprocessing for CNN input → Eye State Classification →Frame Count Logic → Alert Trigger→ Visual Feedback Display → Exit.

C. Code Snippet

```
import cv2
cap = cv2.VideoCapture(0)
while True:
   ret, frame = cap.read()
   if not ret:
      break
   frame = cv2.flip(frame, 1)
   gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
   cv2.imshow("Drowsiness Detection", frame)
   if cv2.waitKey(1) & 0xFF == ord('q'):
      break
```

```
cv2.destroyAllWindows()
```

D. Alert Conditions

- Drowsiness Alert: Error Message when Eyes closed for 5+ frames
- CNN Prediction : Cropped closed images output closed
- Alarm Trigger : Alarm sound plays when Eyes closed for 5+ frames

VII. ACKNOWLEDGMENT

I would like to express my sincere gratitude to my project guide, Dr. S. Prakasam, for his invaluable guidance, constant encouragement, and dedicated support throughout the duration of this research work. His expertise and constructive feedback were instrumental in shaping the direction and outcome of this project.

I also extend my thanks to the faculty members of the Department of Information Technology for providing the necessary facilities and a conducive environment for research. Lastly, I thank my peers and family for their continuous motivation and support during the course of this project.

REFERENCES

[1] Y. Jiang, Y. Zhang, C. Lin, D. Wu, and C.-T. Lin, ‘‘EEG-based driver drowsiness estimation using an online multi-view and transfer TSK fuzzy system,’’ IEEE Trans. Intell. Transp. Syst., vol. 22, no. 3, pp. 1752–1764, Mar. 2021, doi: 10.1109/TITS.2020.2973673.

[2] L. Zhang, Z. Zhang, and C. Guan, ‘‘Accelerating privacy-preserving momentum federated learning for industrial cyber-physical systems,’’ Complex Intell. Syst., vol. 7, no. 6, pp. 3289–3301, Dec. 2021, doi: 10.1007/s40747-021-00519-2.

[3] R. M. Salman, M. Rashid, R. Roy, M. M. Ahsan, and Z. Siddique, ‘‘Driver drowsiness detection using ensemble convolutional neural networks on YawDD,’’ 2021, arXiv:2112.10298.

[4] M.K. Hussein, T. M. Salman, A. H. Miry, and M. A. Subhi, ‘‘Driver drowsiness detection techniques: A survey,’’in Proc.1st Babylon Int. Conf. Inf. Technol. Sci. (BICITS), Apr. 2021, pp. 45–51.

[5] K. Fujiwara, and M. Kano, ‘‘Real-drivingimplementable drowsy driving detection method using heart rate variability based on long short-term memory and autoencoder,’’ IFAC-PapersOnLine, vol. 54, no. 15, pp. 526–531, 2021.

[6] N. N. Pandey and N. B. Muppalaneni , ‘‘Real-time drowsiness identification based on eye state analysis,’’ in Proc. Int. Conf. Artif. Intell. Smart Syst. (ICAIS), Mar. 2021, pp. 1182–1187.

[7] A. Babu and S. A. Jerome, ‘‘ICMFKC with optimize XGBoost classification for breast cancer image screening and detection,’’ Multimedia Tools Appl.,pp. 1-28, Jan. 2024.

[8] European New Car Assessment Programme (Euro NCAP), “2025 Roadmap—In Pursuit of Vision Zero,” Leuven, Belgium, Sep. 2017. [Online]. Available: https://cdn.euroncap.com/media/30700/euroncap-roadmap-2025-v4.pdf

[9] European Union, “Legislative Acts and Other Instruments,” Brussels, Belgium, Oct. 2019. [Online]. Available: https://data.consilium.europa.eu/doc/document/PE-82-2019-INIT/en/pdf

[10] ABI Research, “Camera-Based Driver and Occupant Monitoring Systems,” 2020. [Online]. Available: https://www.abiresearch.com/market-research/product/7778112-camera-based-driver-and-occupantmonitorin/

[11] ABI Research, “Driver Monitoring Systems,” 2014. [Online]. Available: https://www.abiresearch.com/market-research/product/1018540-driver-monitoring-systems/

[12] J. C. Stutts, J. W. Wilkins, and B. V. Vaughn, “Why do people have drowsy driving crashes? Input from drivers who just did,” AAA Foundation for Traffic Safety, Washington, DC, USA, Technical Report, Nov. 1999.

[13] European Commission, Directorate-General for Transport, “Fatigue,” Brussels, Belgium, Feb. 2018