Driver Drowsiness Detection System Using Behavioural Analysis and Deep Learning

Kovvuri Karthik Varma, Maddineni Jyothish Prakash Ranga, Mallidi Pavan Kumar
Dept. of Artificial Intelligence and Data Science, Dhanalakshmi Srinivasan University
Trichy, Tamil Nadu, India

***Abstract*—Driver drowsiness is a leading cause of road traffic accidents worldwide, contributing to thousands of fatalities and injuries every year. This paper presents the design and implementation of a real-time Driver Drowsiness Detection System that integrates computer vision with deep learning to monitor driver alertness continuously. The system employs OpenCV Haar cascade classifiers for facial and ocular region detection from live webcam video streams, and a custom Convolutional Neural Network (CNN) to classify each detected eye as "Open" or "Closed." A time-based PERCLOS (Percentage of Eye Closure) mechanism triggers an audible alarm when eye closure persists for three or more consecutive seconds. Yawning detection is incorporated as an additional behavioral indicator using a dedicated dataset. The entire pipeline is deployed as a Streamlit web application enabling browser-accessible, real-time monitoring. Evaluation on a diverse test set demonstrates strong classification accuracy, near real-time frame processing rates, and reliable alarm response. The system requires no specialized hardware, making it an affordable and scalable safety solution for passenger vehicles, fleet management, and public transportation.**

***Keywords*—Driver Drowsiness Detection, Convolutional Neural Network, Computer Vision, Haar Cascade, PERCLOS, Eye State Classification, Yawning Detection, Road Safety, Stream lit, Fatigue Monitoring, Deep Learning, OpenCV.**

I. INTRODUCTION

Road safety remains one of the most pressing public health challenges of the twenty-first century. The World Health Organization (WHO) estimates that approximately 1.35 million people die in road traffic accidents each year, with driver fatigue implicated in 10–20% of all serious crashes [1]. Unlike alcohol impairment or distraction, the onset of drowsiness is gradual and often imperceptible to the driver, making it particularly dangerous. Fatigue reduces cognitive processing speed, impairs judgment, slows reaction time, and ultimately leads to microsleeps—brief involuntary episodes of sleep lasting 1–30 seconds—during which the vehicle continues at full speed without any driver input.

Existing countermeasures range from road-side rumble strips to legislative limits on driving hours for commercial vehicles. While effective at a population level, these measures provide no protection in the critical moment when an individual driver transitions from alertness to incapacitation. Real-time automated drowsiness detection systems are therefore essential complements to passive road safety infrastructure.

Drowsiness detection approaches fall into three broad paradigms: (i) physiological signal-based methods, which measure EEG, ECG, or galvanic skin response; (ii) vehicle dynamics-based methods, which monitor lane deviation, steering wheel entropy, and acceleration patterns; and (iii) behavior-based vision methods, which analyze facial cues such as eye closure, blink dynamics, yawning, and head pose. Vision-based approaches are particularly attractive for mass-market deployment because they require only a low-cost camera, impose no physical sensors on the driver, and can operate continuously without driver awareness.

This paper proposes a robust, real-time behavior-based drowsiness detection system that leverages the PERCLOS metric operationalized through Haar cascade face and eye localization, a three-layer CNN for eye state binary classification, a yawn detection sub-pipeline, a time-threshold alarm mechanism, and a Stream lit-based deployment framework.
The remainder of the paper is organized as follows: Section II reviews related work. Section III describes the dataset preparation. Section IV details the system architecture and methodology. Section V presents the CNN model and training procedure. Section VI discusses experimental evaluation. Section VII presents a comparative analysis. Section VIII concludes the paper with directions for future research.

II. RELATED WORK

Drowsiness detection has been an active research area for over two decades. Early systems relied heavily on physiological signals. Lal and Craig [2] demonstrated that EEG delta and theta wave power increases correlate strongly with fatigue-induced performance degradation. While these systems achieve high sensitivity, the requirement for electrode attachment makes them impractical in naturalistic driving environments.

Vehicle-based approaches avoid sensor attachment by exploiting the relationship between driver state and vehicle behavior. Eskandarian and Mortazavi [3] proposed a model using lateral acceleration and steering wheel movements to detect fatigue-related driving degradation. However, these methods exhibit high response latency, detecting drowsiness only after vehicle behavior has already deteriorated, which may be too late to prevent accidents.

Vision-based behavioral analysis has gained prominence due to advances in real-time computer vision. The seminal work of Viola and Jones [4] introduced Haar feature-based cascade classifiers that enabled real-time face detection on consumer hardware—a prerequisite for practical in-vehicle vision systems. Soukupova and Cech [5] proposed the Eye Aspect Ratio (EAR) metric derived from facial landmarks as a robust indicator of eye closure and blink events, demonstrating reliable drowsiness detection in a variety of conditions.

The PERCLOS metric, standardized by Wierwille et al. [6], defines drowsiness as occurring when the eyes are at least 80% closed for more than 80% of any given minute. This measure has since been widely validated as one of the most reliable non-intrusive indicators of driver fatigue. Subsequent systems have operationalized PERCLOS using image processing pipelines of varying complexity.

Deep learning has substantially improved the robustness of eye state classification. Weng et al. [7] employed a multi-scale CNN trained on the ZJU Eye-Blink dataset to achieve high accuracy under illumination variation. Guo et al. [8] proposed a lightweight MobileNet-based architecture for embedded deployment. More recently, attention-based transformer architectures have shown promise for multi-cue fatigue detection integrating eye state, yawning, and head pose simultaneously.

The proposed system synthesizes the strengths of these approaches: Haar cascades provide computationally efficient ROI localization, a custom CNN delivers robust eye state classification, and a yawning detection sub-pipeline broadens the behavioral cue set, all within an accessible, zero-hardware-overhead deployment framework.

III. DATASET PREPARATION

Two independent datasets are used to train the eye state classifier and the yawning detector respectively, following a standard preprocessing and augmentation pipeline.

A. Eye State Dataset

The eye dataset is organized into two classes: Open_Eyes and Closed_Eyes. Grayscale images are sourced from publicly available drowsiness and blink datasets and supplemented with augmented samples. Each image is resized to 64×64 pixels. The combined dataset contains images across both classes, ensuring balanced class representation to prevent classifier bias.

B. Yawning Dataset

The yawn dataset comprises two classes: yawn and not_yawning. Images are captured under varying head poses and lighting conditions to promote model robustness. The same 64×64 grayscale preprocessing pipeline is applied for consistency. Both datasets undergo an 80/20 stratified train-test split with a fixed random seed of 42 to ensure reproducibility.

C. Preprocessing Pipeline

Preprocessing includes pixel intensity normalization to the [0, 1] range, conversion to single-channel grayscale, and reshaping to (N, 64, 64, 1) tensors for CNN input. No external data augmentation transforms such as rotation or flipping are applied, as the Haar cascade crop-and-resize operation naturally introduces sufficient intra-class variation through variable detection box sizes.

IV. SYSTEM ARCHITECTURE AND METHODOLOGY

The proposed system is structured as a five-stage real-time pipeline: (1) video frame acquisition, (2) face and eye region localization using Haar cascades, (3) CNN-based eye state classification, (4) PERCLOS-inspired timer-threshold logic, and (5) alarm-based driver alert. Figure 1 illustrates the end-to-end processing pipeline.

A. Video Frame Acquisition

Live video is captured at the native camera frame rate using OpenCV's cv2.VideoCapture(0) interface. Each captured BGR frame is immediately converted to grayscale using cv2.cvtColor to reduce computational overhead. Grayscale frames are sufficient for both Haar cascade detection and CNN inference, as texture and intensity—rather than color—are the dominant discriminating features for eye and face regions.

B. Hierarchical Face and Eye Localization

OpenCV's pre-trained Haar cascade classifiers are applied hierarchically.

The haarcascade_frontalface_default.xml classifier operates on the full grayscale frame with a scale factor of 1.3 and minimum neighbor count of 5, producing a list of face bounding boxes. For each detected face, a Region of Interest (ROI) is cropped and the haarcascade_eye.xml classifier is applied with a scale factor of 1.1 and minimum neighbor count of 3, yielding individual eye bounding boxes. This hierarchical approach restricts the eye detection search space, substantially reducing false positives and improving processing throughput.

C. CNN-Based Eye State Classification

Each detected eye ROI is extracted from the grayscale face ROI, resized to 64×64 pixels using bilinear interpolation, normalized by dividing by 255.0, and reshaped to (1, 64, 64, 1) for CNN input. The CNN outputs a two-class softmax probability vector; the argmax determines the label—0 for Open, 1 for Closed. If any eye in the frame is classified as Open, the driver is considered alert for that frame.

D. PERCLOS Timer-Threshold Logic

A global timer variable records the timestamp at which continuous eye closure begins. On each frame, if all detected eyes are classified as Closed, elapsed time is computed as the difference between the current timestamp and the closure start time. If this elapsed time meets or exceeds the configurable threshold (default: 3 seconds), the system flags a drowsiness event. If any eye is detected as Open, the timer resets to None. This logic directly operationalizes the PERCLOS concept within a streaming video framework.

E. Auditory Alert Mechanism

Upon drowsiness detection, pygame.mixer.music.play(-1) triggers continuous looped playback of a WAV alarm file. The alarm persists until the driver's eyes are detected as open, at which point pygame.mixer.music.stop() is called. The "DROWSY ALERT!" warning is simultaneously rendered on the video frame using cv2.putText for on-screen notification. The FPS counter, computed as the reciprocal of per-frame processing time, is overlaid to monitor system performance in real time.

F. Streamlit Deployment Interface

The entire detection pipeline is wrapped in a Streamlit application. Users initiate monitoring via a "Start Detection" button, which launches the detect_drowsiness() function. The live annotated video feed is rendered in the browser using st.empty().image(), updating on every processed frame. A "Stop Detection" button halts the capture loop and silences any active alarm. The compile=False flag in tf.keras.models.load_model() resolves a known IndexError associated with certain TensorFlow versions when loading legacy .h5 model files.

V. CNN MODEL AND TRAINING

Two structurally identical CNN models are trained independently: one for eye state classification (Open/Closed) and one for yawning detection (Yawn/Not Yawning). The shared architecture was designed to balance representational capacity with computational efficiency for real-time CPU inference.

A. Architecture Design

Each CNN comprises three convolutional blocks, each consisting of a Conv2D layer followed by ReLU activation and 2×2 max pooling. Filter depth progressively doubles across blocks (32, 64, 128) to capture features of increasing semantic complexity—from edges and textures in early layers to high-level structural patterns in deeper layers. The flattened feature vector is passed through a 128-unit dense layer with ReLU activation and a Dropout layer (rate = 0.5) for regularization, before the final two-unit softmax output layer.

TABLE I. CNN MODEL ARCHITECTURE

Layer	Activation	Description
Input	—	64×64×1 grayscale image
Conv2D (32 filters, 3×3)	ReLU	Low-level edge & texture features
MaxPooling2D (2×2)	—	Spatial downsampling → 31×31
Conv2D (64 filters, 3×3)	ReLU	Mid-level structural features
MaxPooling2D (2×2)	—	Spatial downsampling → 14×14
Conv2D (128 filters, 3×3)	ReLU	High-level semantic features
MaxPooling2D (2×2)	—	Spatial downsampling → 6×6
Flatten	—	4,608-dimensional feature vector
Dense (128 units)	ReLU	Non-linear classification head
Dropout (rate=0.5)	—	Regularization, reduces overfitting
Dense (2 units)	Softmax	P(Open), P(Closed) — output

B. Training Configuration

Both models are compiled with the Adam optimizer (default learning rate 0.001) and categorical cross-entropy loss. Training runs for 100 epochs with a batch size of 32 on an 80/20 stratified train-test split. One-hot encoding via to_categorical() converts integer labels to two-class probability vectors. The compile=False loading flag ensures compatibility across TensorFlow versions during inference. Trained models are serialized to the HDF5 format (.h5) using Keras's model.save() method.

VI. EXPERIMENTAL RESULTS AND DISCUSSION

The proposed system was evaluated on a commodity laptop (Intel Core i5, 8 GB RAM, integrated GPU) running Ubuntu 20.04. No dedicated GPU acceleration was utilized during inference, demonstrating the system's viability on standard consumer hardware.

The CNN eye state classifier achieved a test accuracy of approximately 92–95% across multiple training runs, reflecting strong generalization from the balanced train-test split. Classification loss converged within approximately 40 epochs, after which validation loss stabilized without significant overfitting, attributable to the Dropout regularization layer. The yawning classifier similarly converged to high accuracy, providing a complementary fatigue cue beyond blink-based monitoring.

End-to-end per-frame latency—including Haar cascade detection, CNN inference, and frame rendering—ranged from 35 to 55 milliseconds, corresponding to effective frame rates of 18–28 FPS. This throughput exceeds the minimum requirement for real-time drowsiness monitoring (typically cited as 15 FPS), confirming the system's suitability for live deployment without GPU acceleration.

The three-second eye closure threshold was empirically validated as an appropriate drowsiness indicator, consistent with PERCLOS literature. Informal testing across multiple subjects confirmed that the threshold avoids false alarms from normal blink events (typically 150–400 ms duration) while reliably detecting genuine drowsiness onset. The alarm response latency from threshold breach to audio playback initiation was measured at under 100 milliseconds, ensuring prompt driver notification.
Table II summarizes the key performance metrics of the proposed system under standard indoor lighting conditions.

TABLE II. SYSTEM PERFORMANCE METRICS

Metric	Value
Eye State Classification Accuracy	~93.4%
Yawn Detection Accuracy	~91.7%
Average Inference Latency (per frame)	~42 ms

Effective Frame Rate	~23 FPS
Eye Closure Alarm Threshold	3 seconds
Alarm Response Latency	< 100 ms
False Positive Rate (normal blinks)	< 2%
Hardware Requirement	Standard Webcam + CPU

VII. COMPARATIVE ANALYSIS

Table III compares the proposed system against representative state-of-the-art drowsiness detection approaches across key evaluation dimensions: detection accuracy, real-time capability, hardware requirements, and identified limitations. The proposed system achieves a favorable balance of accuracy, real-time performance, and hardware accessibility. Its principal advantage over physiological and vehicle-dynamics methods is the complete elimination of additional sensors. Compared to landmark-based methods, it avoids dependency on facial landmark libraries such as dlib, which can degrade under non-frontal head poses. Compared to MobileNet-based deep learning systems, it operates without GPU acceleration, making it deployable on a broader class of devices.

TABLE III. COMPARISON WITH STATE-OF-THE-ART METHODS

Method	Accuracy	Real-Time	Hardware	Limitation
EEG-based [2]	Very High	High	Intrusive sensors	Laboratory only
Vehicle dynamics [3]	Moderate	Low	None	Post-degradation
EAR + Landmarks [5]	High	Medium	Camera	Landmark sensitivity
CNN (MobileNet) [8]	High	High	Camera + GPU	High compute cost
Proposed System	High	High	Webcam only	Webcam + CPU

VIII. CONCLUSION AND FUTURE WORK

This paper presented a comprehensive real-time Driver Drowsiness Detection System combining Haar cascade-based face and eye localization, a custom CNN for binary eye state classification, a yawning detection sub-pipeline, and a PERCLOS-inspired timer-threshold alarm mechanism, all deployed through a Streamlit web interface. The system operates entirely on standard consumer hardware without any specialized sensors, achieving classification accuracy exceeding 92%, effective frame rates above 18 FPS, and alarm response latencies under 100 milliseconds.

The modular pipeline architecture enables independent improvement of each subsystem. The Streamlit deployment model provides a low-barrier entry point for practical adoption in small fleet management operations, driver training applications, and research prototyping. The system's reliance on grayscale imagery and lightweight CNN inference ensures compatibility with low-power embedded processors, expanding its deployment envelope beyond laptop-class hardware.

Several limitations merit acknowledgment. The Haar cascade classifiers exhibit reduced sensitivity under extreme lateral head poses and low-illumination conditions. The binary eye state CNN does not model blink dynamics across frames, which could enable more nuanced fatigue staging. The current system also lacks a mechanism to distinguish genuine drowsiness from prolonged blinking or brief eye closures caused by environmental factors such as bright sunlight entering the vehicle.

Future research will pursue the following enhancements: (i) adoption of Vision Transformer (ViT) or EfficientNet architectures for improved accuracy under adverse lighting and pose variation; (ii) integration of head pose estimation via 3D facial landmark models to detect nodding—another key drowsiness indicator; (iii) multi-frame temporal modeling using LSTM or 3D CNN architectures to capture blink dynamics and micro-sleep onset patterns; (iv) edge deployment on Raspberry Pi or

NVIDIA Jetson Nano for embedded in-vehicle integration; (v) personalized baseline calibration to account for inter-driver variability in natural blink rate and eye aperture; and (vi) explainability analysis using Grad-CAM to visualize the CNN's discriminative attention regions, building driver trust in the system's alerts.

REFERENCES

[1] World Health Organization, "Global Status Report on Road Safety 2018," WHO, Geneva, 2018.

[2] S. K. L. Lal and A. Craig, "A critical review of the psychophysiology of driver fatigue," Biological Psychology, vol. 55, no. 3, pp. 173–194, 2001.

[3] A. Eskandarian and A. Mortazavi, "Evaluation of a smart algorithm for commercial vehicle driver drowsiness detection," in Proc. IEEE Intelligent Vehicles Symposium, 2007, pp. 553–559.

[4] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in Proc. IEEE CVPR, vol. 1, 2001, pp. 511–518.

[5] T. Soukupova and J. Cech, "Real-time eye blink detection using facial landmarks," in Proc. 21st Computer Vision Winter Workshop (CVWW), 2016, pp. 1–8.

[6] W. W. Wierwille, S. S. Wreggit, C. L. Kirn, L. A. Ellsworth, and R. J. Fairbanks, "Research on vehicle-based driver status/performance monitoring," NHTSA Technical Report DOT HS 808 640, 1994.

[7] C. Weng, Y. Sun, and T. Tan, "Driver drowsiness detection via a hierarchical temporal deep belief network," in Proc. ECCV Workshops, 2016, pp. 117–133.

[8] J. Guo, Z. Zhou, and J. Sun, "Drowsiness detection based on MobileNet and long short-term memory," in Proc. IEEE ICSAI, 2018, pp. 1–5.

[9] F. Chollet et al., "Keras," GitHub, 2015. [Online]. Available: https://github.com/keras-team/keras.

[10] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in Proc. NIPS, 2012, pp. 1097–1105.