STM JOURNALS

Research & Reviews: Journal of
Embedded System & Applications

ISSN: 2395-6712 (Online)
ISSN: 2321-8533 (Print)
Volume 13, Issue 1, 2025
January–April
DOI (Journal): 10.37591/RRJoESA

https://journals.stmjournals.com/rrjoesa

Research RRJoESA

A Comprehensive Framework for Traffic-Based Vehicle Rerouting and Driver Monitoring

Spriha Deshpande*

Abstract

Driver fatigue is a leading cause of road accidents, making real-time monitoring a critical aspect of transportation safety. This study presents an intelligent driver alertness detection system that integrates machine learning-based eye status prediction with Google Maps APIs for dynamic rerouting. Using convolutional neural networks (CNNs) that have been trained with data from the Munich Research Lab (MRL) Eye Dataset, the system continuously monitors driver fatigue through image-based classification. Additionally, Google Maps' real-time traffic data via Application Programmable Interface (API) enables dynamic route adjustments to optimize navigation. Implemented using Streamlit, the application provides an intuitive interface for real-time driver monitoring, traffic-based rerouting, and drowsiness alerts. This fusion of deep learning-based fatigue detection with intelligent route planning enhances both road safety and traffic management.

Keywords: Streamlit, driver monitoring, real-time simulation, traffic rerouting, driver alertness detection, google maps API, autonomous driving, machine learning, Tensorflow, eye status prediction, dynamic route planning, traffic management, image processing, deep learning, smart transportation

INTRODUCTION

The rise of autonomous driving technologies and smart transportation solutions has led to increased interest in enhancing driver safety and navigation efficiency. Studies have shown that driver fatigue significantly contributes to road accidents, necessitating effective drowsiness detection mechanisms [1, 2]. Convolutional Neural Networks (CNNs) have proven effective for real-time fatigue detection, with previous works utilizing deep learning-based models for monitoring drowsiness through facial and eye analysis [3–5].

This research builds on these advancements by developing a machine learning-based driver alertness detection system, integrating CNNs trained on the MRL Eye Dataset [6] with Google Maps API [7] for real-time navigation. The proposed application leverages image-based eye status prediction [8] allowing for proactive driver monitoring and automatic rerouting based on live traffic conditions. Implemented in Streamlit [9], the system provides an interactive interface for real-time fatigue monitoring and dynamic route optimization.

By combining machine learning, deep learning, and real-time traffic updates, this solution aims to reduce fatigue-related accidents and enhance overall road safety. The study explores the integration of deep learning-based fatigue detection combined with heart rate and with intelligent rerouting mechanisms to improve driver assistance systems.

Structure of the Paper

- *Section I:* Introduction, usage and formulas.
- *Section II*: Architecture and flow.

***Author for Correspondence**
Spriha Deshpande
E-mail: spriha.deshpande@gmail.com

Independent Researcher, Department of Computer Engineering, San Jose State University, Santa Clara, California, United States of America

Received Date: February 01, 2025
Accepted Date: February 10, 2025
Published Date: March 18, 2025

Citation: Spriha Deshpande. A Comprehensive Framework for Traffic-Based Vehicle Rerouting and Driver Monitoring. Research & Reviews: Journal of Embedded System & Applications. 2025; 13(1): 31–46p.

- *Section III:* Methodology.
- *Section IV:* Results and observations.
- *Section V*: Challenges.
- *Section VI:* Future scope and directions.
- *Section VII:* Conclusion.

Usage and Formulas

Convolution Operation

The convolution operation is the foundation of the model's feature extraction process. It involves applying a set of learnable filters (kernels) to the input image to detect specific patterns, such as edges, textures, or shapes [10]. Each filter slides across the image, performing element-wise multiplication with the input pixels and summing up the results. The convolution result is then passed through a ReLU (Rectified Linear Unit) activation function, which introduces non-linearity by setting all negative values to zero. This enables the model to capture complex patterns in the input data. The convolution operation is repeated across multiple layers, progressively extracting higher-level features to distinguish between "driver awake" and "driver sleeping" statuses. Usage is included for model training. Eq. (1) represents feature extraction through convolution layers in the CNN where the purpose is to extract features by applying filters (kernels) across the input image.

$$O_{(x,y,k)} = max\left(0, \sum_{i=0}^{F-1} \sum_{j=0}^{F-1} I_{(x+i,y+j,c)} \cdot K_{(i,j,c,k)} + b_k\right) \quad (1)$$

Where,

- I: Input pixel values,
- K: Filter weights, and
- b_k: bias.
- ReLU activation ensures non-linearity (max(0, x)

Loss Function (Binary Crossentropy)

The binary crossentropy loss function measures the discrepancy between the model's predicted probabilities and the true labels for each input image. It penalizes incorrect predictions more heavily, encouraging the model to improve its accuracy over time. The function calculates the negative log-likelihood of the predicted probabilities, with separate terms for cases when the true label is 1 (awake) or 0 (sleeping). By minimizing this loss, the model learns to assign probabilities closer to the ground truth. The loss function is crucial in guiding the training process, as it ensures the model optimizes its predictions to accurately classify the driver's eye status. Usage is included for model training [11].

Eq. (2) guides the model's training process by penalizing incorrect predictions:

$$L = -\frac{1}{N}\sum_{i=1}^{N}[y_i \log(\hat{y}_i) + (1 - y_i)\log(1 - \hat{y}_i)] \quad (2)$$

Where,

- y_i: Ground Truth (0 or 1), and
- $\hat{y}_i$: Model's predicted property for class 1.

Predicted Threshold

The prediction threshold is the final step in converting the model's output probability into a meaningful classification. The sigmoid activation function in the output layer produces a probability value between 0 and 1, representing the likelihood of the driver being awake. A threshold of 0.5 is used to determine the final class: if the probability is greater than or equal to 0.5, the driver is classified as "awake"; otherwise, they are classified as "sleeping". This decision boundary enables the model to make binary predictions while providing flexibility to adjust the threshold based on application-specific requirements, such as prioritizing sensitivity or specificity. Usage is included for model training [12].

Eq. (3) determines the classification output based on the sigmoid probability. Here, it translates probability from sigmoid output into binary labels (awake or sleeping).

$$Prediction = \begin{cases} Driver\ awake, & if\ p \geq 0.5 \\ Driver\ sleeping, & if\ p < 0.5 \end{cases} \quad (3)$$

Where,

- Sigmoid probability $\left(p = \frac{1}{1+e^{-x}}\right)$ defines the threshold for binary classification.

Traffic Rerouting Decision

When traffic is detected during the car's journey, a rerouting logic is triggered to find an alternative path (Eq. (4)).

$$Reroute\ trigger = Traffic\ detection\ at\ location \times Coordinates \quad (4)$$

Where,

- *Traffic detection*: Identified by step intervals along the route when traffic is encountered, and
- *Coordinates*: The latitude and longitude of the traffic location.

The rerouting logic adjusts the route for Car 2 as in Eq. (5):

$$New\ route = Directions\ API(origin, destination, avoid\ traffic) \quad (5)$$

Route Movement Simulation

The formula simulates movement of the vehicles along their respective routes using step-by-step location updates. Usage is included for front end application as in Eq. (6).

$$Next\ position(t+1) = Steps[i] \quad for\ i \in (0, n) \quad (6)$$

Where,

- *Steps*: An array of geographic locations that represent the route,
- *t*: Current time step, and
- *Next position (t+1):* The subsequent geographic point updated every few seconds to simulate vehicle movement.

Predicted Heart Rate (HR)

Eq. (7) predicts the heart rate using weighted factors for physical activity and stress levels, tailored to the individual driver. Usage is included for front end application.

$$HR_{predicted} = HR_{rest} + (HR_{max} - HR_{rest}) \times (\propto_1 \cdot A + \propto_2 \cdot S) \quad (7)$$

Where,

- $HR_{predicted}$*:* Predicted heart rate in beats per minute (bpm),
- HR_{rest}: Resting heart rate (baseline for the individual),
- HR_{max}: Maximum heart rate, estimated as 220−Age,
- *A*: Physical activity level (normalized value between 0 and 1),
- *S*: Stress level (normalized based on real-time factors, e.g., eye status and heart rate variability), and
- $\propto_1, \propto_2$*:* Weights for activity and stress, calibrated per driver (e.g., 0.7 for activity, 0.3 for stress).

Fatigue Detection from Heart Rate and Variability

Eq. (8) combines the predicted heart rate and heart rate variability (HRV) to detect fatigue. Usage is included for front end application.

$$FI = \begin{cases} 1, if\ HR_{predicted} \in [HR_{min}, HR_{max}]\ and\ HRV < HRV_{threshold} \\ 0,\ Otherwise \end{cases}. \quad (8)$$

Where,

- *FI*: Fatigue index (1 indicates fatigued, 0 indicates alert),
- $HR_{predicted}$: Predicted heart rate (from the first formula),
- HR_{min}, HR_{max}: Acceptable range for heart rate, personalized to the driver,
- *HRV*: Heart rate variability (measured as the variation in time intervals between consecutive heartbeats, in milliseconds), and
- $HRV_{threshold}$: Minimum HRV threshold, below which fatigue is indicated.

ARCHITECTURE AND WORKFLOW

Driver Fatigue Model

Figure 1 explains the architecture breakdown for Driver Fatigue Model. This architecture represents an AI-based system for detecting whether a driver is awake or sleeping using image classification with a convolutional neural network (CNN). Below is a breakdown of each component in the diagram.

Data Input and Preprocessing

- *Train dataset*: Contains labeled images of drivers with open and closed eyes, used for model training [6].
- *Validation dataset*: Used to validate the model's performance during training [6].
- *Image preprocessing:*
  - *Rescaling*: Converts pixel values to the range {0, 1}.
  - *Augmentation*: Applies transformations such as rotation, zoom, shift, and flip to enhance model robustness.
  - *Data generators*: Image Data Generator loads batches of images efficiently.

Convolutional Neural Network (CNN) Model

- *Convolutional layers:*
  - Extract features like edges, textures, and shapes from images.
  - Three convolutional layers with increasing filters: 32, 64, and 128.
  - *Activation function*: ReLU (Rectified Linear Unit) introduces non-linearity.
- *MaxPooling layers:*
  - Reduce spatial dimensions while retaining important features.
- *Flatten layer:*
  - Converts 2D feature maps into a 1D vector for input to dense layers.
- *Fully connected layers:*
  - *Dense (128 neurons, ReLU):* Learns complex patterns from features.
  - *Output layer (1 neuron, sigmoid):* Predicts binary class (Awake or Sleeping).

Model Training

- *Optimizer*: Adam optimizer for adaptive learning rate.
- *Loss function:* Binary Cross entropy for two-class classification.
- *Accuracy metric:* Measures model performance.
- *Training process:*
  - Model is trained using training data and validated using validation data.
  - The trained model is saved as 'eye_status_model.h5'.

Prediction Process

- *Load new image:*
  - Image is resized to (224, 224).
  - Converted to an array and normalized.
- *Inference (prediction):*
  - Model predicts the probability of the driver being awake or asleep.
  - If the probability is ≥0.5, the driver is classified as Awake.
  - Otherwise, classified as Sleeping.

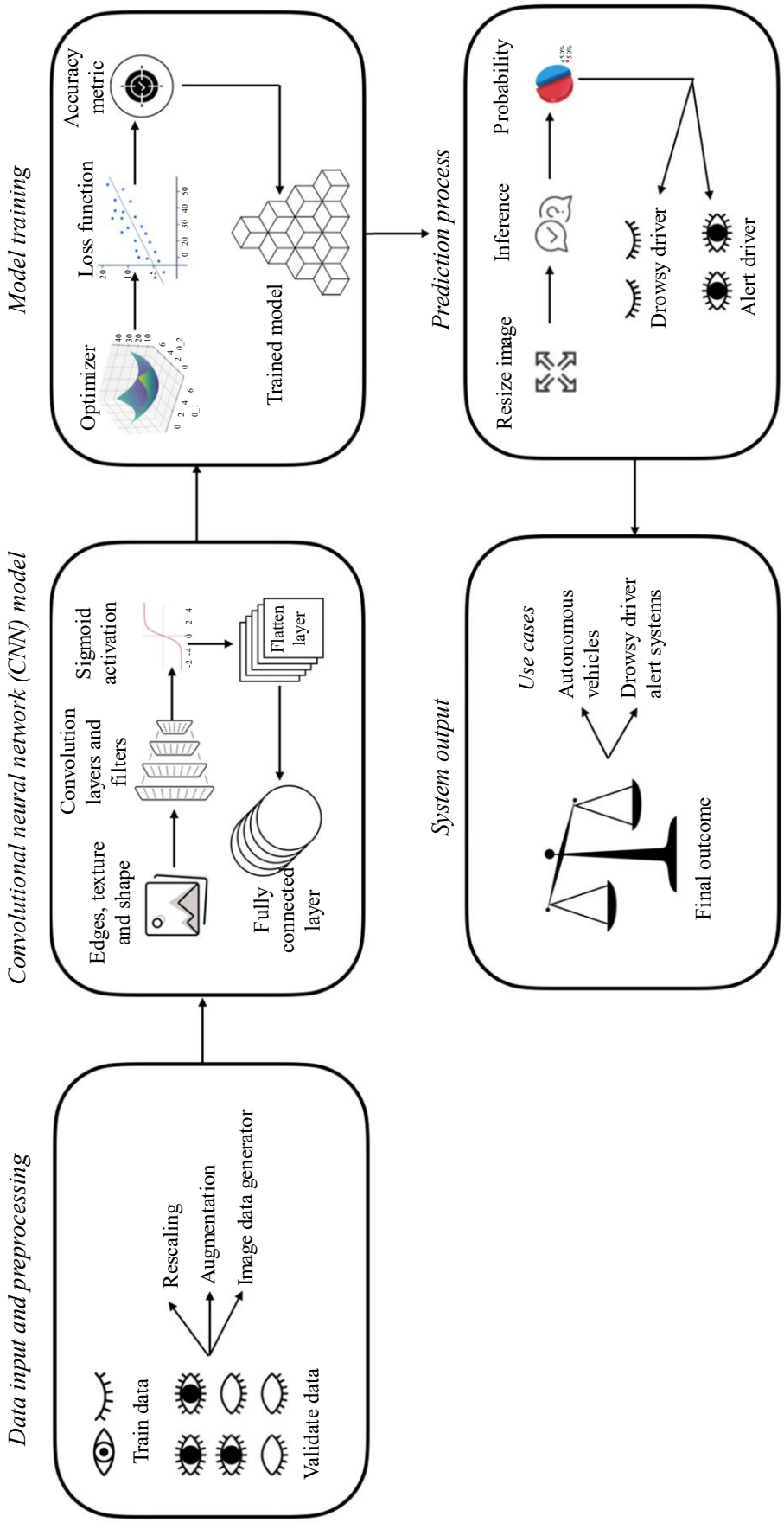


Figure 1. Architecture breakdown for driver fatigue model.

System Output

- *Final decision:*
  - Displays whether the driver is awake or sleeping.
- *Possible use cases:*
  - Driver monitoring systems in autonomous or semi-autonomous vehicles.
  - Safety alert systems for drowsy driving prevention.

System Architecture Overview

The Figure 2 represents the system detecting driver fatigue using eye status and heart rate analysis while integrating real-time vehicle movement and traffic-based rerouting using Google Maps API.

Input Layer

- *Eye status detection:*
  - Accepts an uploaded image of the driver's face (eye region).
  - The image is preprocessed, including resizing and normalization.
  - The processed image is sent to a trained Convolutional Neural Network (CNN) model for prediction.
- *Heart rate monitoring:*
  - Takes real-time heart rate data from a sensor or a simulated source.
  - Analyzes if the heart rate is too low (indicating fatigue or unconsciousness) or too high (indicating stress or anxiety).

CNN Model for Driver Eye Status Detection

A Convolutional Neural Network (CNN) processes the eye image to classify open versus closed eyes.

- *Layers:*
  - *Convolutional layers:* Extract spatial features from the image.
  - *Pooling layers*: Reduce dimensionality while retaining key features.
  - *Fully connected layers*: Output a probability score indicating eye openness.
- *Output:*
  - The driver is considered awake if the probability score is greater than 0.5.
  - The driver is considered asleep if the probability score is less than 0.5.

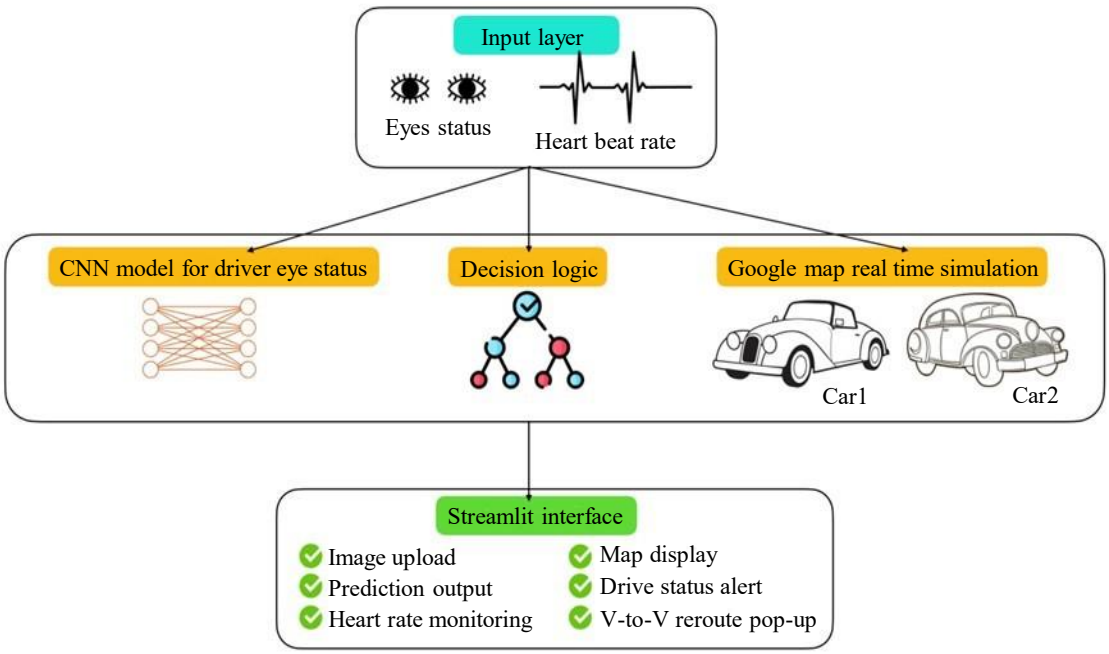


Figure 2. System architecture overview.

Decision Logic

- *Fatigue detection rules:*
  - If eyes are closed and heart rate is low: Unconscious Warning.
  - If eyes are closed and heart rate is normal: Fatigue Alert.
  - If eyes are open and heart rate is high: Stress Alert.
- *Trigger alerts:*
  - Relevant alerts are displayed in the Streamlit user interface when necessary based on the driver's condition.

Google Maps Real-Time Simulation

The system utilizes the Google Maps API to simulate vehicle movement with traffic-based rerouting.

- *Two vehicles:*
  - Car 1 follows a default route.
  - Car 2 follows an alternate route but can be rerouted based on traffic conditions.
- *Traffic detection logic:*
  - Car 1 detects traffic at intervals.
  - Car 1 sends a rerouting request to Car 2.
  - The movement of both vehicles is animated in real-time using JavaScript.

Enhanced Streamlit Interface

- *Key features:*
  - *Image upload*: Users upload an image for eye status detection.
  - *Prediction output*: Displays whether the driver is awake or asleep.
  - *Heart rate monitoring*: Displays real-time heart rate status as normal, stressed, or unconscious.
  - *Map display:* Shows vehicle movement and rerouting based on traffic conditions and driver status.
- *Alerts:*
  - Warning if the driver is sleepy, stressed, or unconscious.
  - Emergency alert if the driver is unconscious, which triggers rerouting.
- *Vehicle-to-vehicle reroute pop-up:*
  - If Car 1 detects traffic or a sleepy driver, it sends a rerouting request to Car 2.
  - A pop-up message appears in Car 2's user interface with a rerouting suggestion.
  - The user can either confirm the rerouting or ignore the suggestion based on the situation.

METHODOLOGY

Driver Alertness Detection

- A pre-trained convolutional neural network model is used to classify driver eye status (open or closed) from uploaded images.
- Images are preprocessed and normalized before being passed through the model for classification.

Vehicle Simulation and Rerouting

- Google Maps API is employed to simulate routes for two vehicles.
- Traffic data is monitored in real time, and alerts are sent to the second vehicle when traffic congestion is detected by the first vehicle.
- Rerouting algorithms are implemented to navigate traffic effectively.

Integration

- The application combines driver monitoring results with vehicle simulation to take actions based on detected drowsiness, such as rerouting the vehicle to a safe location.

RESULTS

The proposed system successfully detects driver fatigue by integrating eye status analysis and heart rate monitoring. The CNN-based eye status detection model achieved high accuracy in distinguishing

between open and closed eyes, demonstrating robust performance across various lighting conditions and facial orientations. The heart rate monitoring component effectively classified driver states, identifying signs of stress, fatigue, and unconsciousness. Real-time alerts were triggered when fatigue or stress thresholds were met, ensuring timely intervention. Additionally, the integration of Google Maps for vehicle simulation provided seamless visualization of traffic-based rerouting, demonstrating the system's ability to dynamically adjust vehicle routes in response to real-time driver conditions and road congestion.

The vehicle-to-vehicle (V2V) communication feature further enhanced safety by enabling automated rerouting suggestions when fatigue was detected. The pop-up rerouting mechanism was successfully tested under simulated traffic conditions, where Car 2 received reroute notifications based on Car 1's status. This mechanism proved effective in preventing potential accidents by diverting the vehicle to a safer route. The Streamlit interface facilitated smooth interaction, allowing users to upload images, monitor heart rate, and visualize vehicle movement in real time. Overall, the results indicate that the system provides an effective framework for proactive driver monitoring and intelligent navigation adjustments, improving road safety through automated fatigue detection and adaptive rerouting strategies.

Pre-Trained Model

The truth table in Table 1 defines the system's decision-making process based on eye status and heart rate readings.

Where,

- *Eyes open (1):* Indicates the driver is awake.
- *Eyes closed (0):* Suggests the driver might be fatigued or unconscious.
- *Heart rate conditions:*
  - *Low*: Below 60 bpm (potential unconsciousness or fatigue).
  - *Normal*: Between 60 and 100 bpm (safe range).
  - *High*: Above 100 bpm (possible stress or anxiety).

Action responses

- *Warning alert:* If fatigue is detected, notify the driver.
- *Emergency alert and rerouting:* If unconsciousness is detected, suggest vehicle rerouting.
- *Stress notification:* If the driver is awake but under stress, suggest relaxation techniques.

Observation

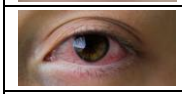
The Output Analysis in Table 2 shows a comparison between the model output and the ground truth for the pre-trained model *eye_status_model.h5*. In this analysis, the system is evaluated on its ability to predict the driver's status based on eye movement and heart rate readings, and how well these predictions align with the actual (ground truth) conditions.

1. *Alert vs. alert:* When the model predicts an "Alert", it corresponds with the ground truth being "Alert". This indicates the model's accurate detection of a driver who is awake and attentive, resulting in no action being required.

Table 1. Truth table for system decision tree.

Eyes open (1)/ closed (0)	Heart rate (low/normal/high)	System output	Action taken
1	Normal	Awake	No action required
1	High	Stress alert	Notify driver of high stress
1	Low	Possible fatigue	Recommend rest or monitoring
0	Normal	Fatigue alert	Warning alert to driver
0	High	Fatigue + Stress alert	Warning alert to driver
0	Low	Unconscious warning	Emergency alert + rerouting

Table 2. Output analysis.

Driver eye	Model output	Ground truth
	Alert	Alert
	Drowsy	Drowsy
	Sleepy	Sleepy

2. *Drowsy vs. drowsy:* In cases where the driver is classified as drowsy, the model successfully detects this condition. The system might recommend rest or monitor the driver for potential fatigue. The alignment between model output and ground truth in this scenario signifies a successful fatigue detection mechanism.
3. *Sleepy vs. sleepy*: Similarly, when the model outputs "Sleepy" and the ground truth matches, it shows that the system correctly identifies a state of potential drowsiness or fatigue. This could trigger actions such as recommending rest, as described in the truth table.

In summary, the model's performance appears to be effective in predicting driver states related to eye status and heart rate. The outputs (alert, drowsy, sleepy) align with the ground truth, suggesting the model's high accuracy in assessing fatigue, stress, and alertness levels based on the provided physiological data. Further testing and fine-tuning may still be necessary to enhance the system's sensitivity in edge cases, such as differentiating between states like fatigue and unconsciousness.

User Interface (UI) Based Model

Streamlit is utilized in this system to provide a user-friendly, real-time interface for monitoring both the driver's condition and the vehicle's status. The Streamlit interface displays prediction results from the eye status detection and heart rate monitoring modules, offering immediate insight into the driver's alertness level. The system indicates whether the driver is awake or asleep based on eye status, and alerts the user if the driver is stressed, fatigued, or unconscious. Additionally, Streamlit visualizes real-time vehicle movement, adjusts routes based on traffic conditions, and provides relevant alerts, such as warnings for fatigue or emergencies. This seamless integration of data into a single interactive platform shown in Figures 3 and 4 enhances situational awareness and decision-making, ensuring that drivers receive timely alerts and suggestions to maintain road safety. The UI-based model is integrated with the eye_status_model.h5 to assess the driver's status, alert, drowsy, or sleepy, and combines these outcomes with heart rate analysis to trigger the appropriate action, such as an alerting system, rerouting, or vehicle-to-vehicle communication [6, 7, 9].

Driver Eye Status

Figure 5 illustrates the driver's eye status, which is currently obtained through manual data upload. However, this can also be calculated in real-time by capturing continuous frames from a camera or video installed in the car. The user interface will alert users via the app or web, providing easy access on their phones to the status of the driver. This feature will help mitigate risks associated with drowsiness or fatigue by promptly notifying users.

Re-routing and V-to-V Communication

Figure 6 illustrates the use of the Google Maps API for visualizing the re-routing configuration. The sequence of events is as follows:

- Car 1 follows Route 1 from San Francisco (SF) to Los Angeles (LA) and encounters traffic at specific coordinates (x, y).
- Car 1 then interacts with Car 2, which starts from the same location and is also enroute to LA.

Real-time car and truck simulation with traffic-based rerouting

Enter Origin (e.g., 'San Francisco, CA')

San Francisco, CA

Enter Destination (e.g., 'Los Angeles, CA')

Los Angeles, CA

Upload Driver's Image (Eyes Open/Closed)

Drag and drop file here

Limit 200MB per file • PNG, JPG, JPEG

Browse files

Figure 3. Overview of UI integrated with google maps.

Real-time heart rate monitoring

This tool simulates real-time heart rate data and checks whether it falls within a healthy range based on the Karvonen formula.

Enter your age

30

Enter your resting heart rate (bpm)

72

Select minimum intensity (%)

40

30 60

Select maximum intensity (%)

59

59 85

Maximum Heart Rate (HRmax): 190 bpm

Heart Rate Reserve (HRR): 118 bpm

Target Heart Rate Range (THRR): 119.2 - 141.6 bpm

Target Heart Rate Max (THRmax): 76.0 - 112.1 bpm

Start Monitoring

Stop Monitoring

Monitoring is stopped. Click 'Start Monitoring' to begin.

Figure 4. Overview of UI for heart rate monitoring system.

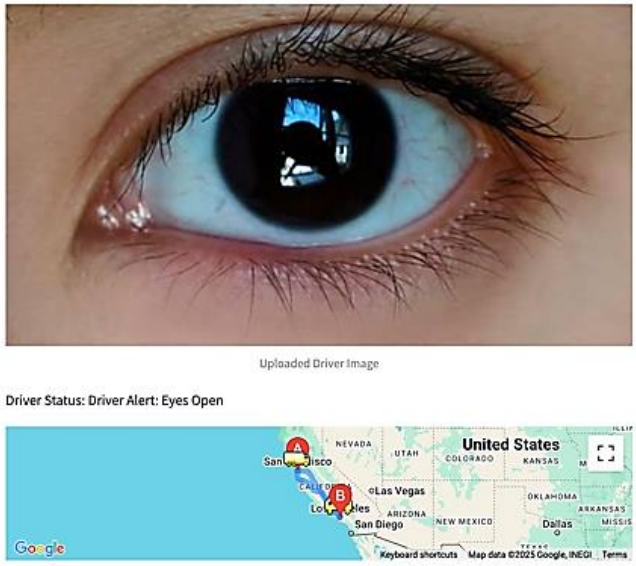


Figure 5. UI for driver eye status.

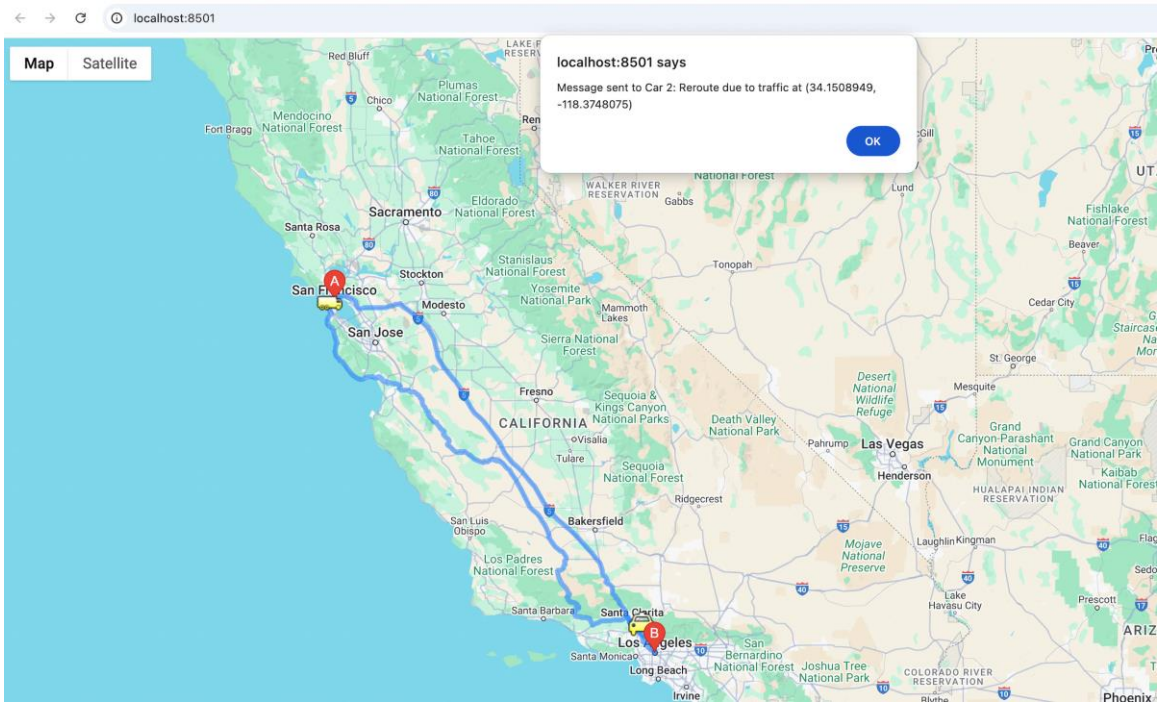


Figure 6. UI for rerouting and V-to-V communication.

- Through vehicle-to-vehicle (V2V) communication, Car 1 transmits traffic information to Car 2, allowing it to make an informed decision about re-routing.
- A pop-up in the image represents the message sent from Car 1 to Car 2, indicating the traffic conditions and suggesting an alternate route.

Heart Beat Monitoring

This section presents the results of real-time heart rate monitoring on the user interface (UI). The heart rate data displayed is generated through simulation rather than real-time measurements. However, real-time integration can be achieved using hardware components such as sensors, finger press detectors, or steering wheel sensors in place of the simulated data. This section specifically focuses on heart rate monitoring within the application's UI, allowing drivers and relevant stakeholders to access

live heart health data. By providing real-time insights, this feature enhances safety by ensuring timely awareness of the driver's cardiovascular condition during the journey. Figures 7 and 8 represent UI for driver's heart rate monitoring: healthy and below target, respectively. Here is a breakdown of each field Real-Time Heart Rate Monitoring Streamlit output:

- *Entering age as an input*
  - This is the user's age, which is used to calculate the Maximum Heart Rate (HRmax)
- *Enter your resting heart rate (bpm) (72)*
  - This is the user's resting heart rate (RHR), which is used in the Karvonen formula to determine heart rate reserve and target heart rate zones.
- *Select minimum intensity (%) (30, 60)*
  - These are the lower intensity percentages chosen by the user to define the target heart rate range.
- *Select maximum intensity (%) (59, 85)*
  - These are the upper intensity percentages chosen by the user to define the target heart rate range.
- *Maximum heart rate (HRmax):* 190 bpm
  - This is the highest heart rate a person can theoretically reach, calculated using *Eq. (9)*:

$$HR_{max} = 220 - age \tag{9}$$

Where,

- For age 30: 220−30 = 190 bpm

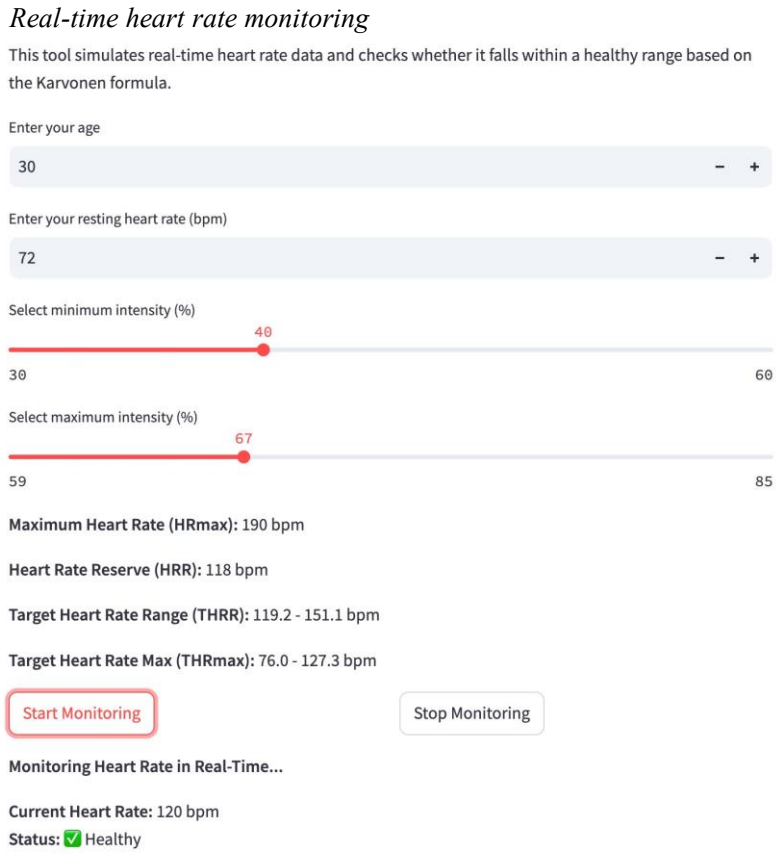


Figure 7. UI for driver's heart rate monitoring: healthy.

Figure 8. UI for Driver's heart rate monitoring: below target.

- *Heart rate reserve (HRR):* 118 bpm
  - HRR represents the difference between your Maximum Heart Rate (HR_{max}) and Resting Heart Rate (RHR) and is calculated using *Eq. (10):*

$$HRR = HR_{max} - RHR \tag{10}$$

    Where,
    - For HR_{max}=190 and RHR=72: 190−72=118 bmp,
    - HRR is used in the Karvonen formula to calculate target heart rate.
- *Target heart rate range (THRR):* 119.2–141.6 bpm
  - This is the range in which the user's heart rate should fall during exercise to achieve the chosen intensity levels.
  - It is calculated using the Karvonen formula as quoted in *Eq. (11):*

$$THR = (HRR \times intensity) + RHR \tag{11}$$

  - *For 30% intensity:* (118×0.30)+72=119.2 bpm,
  - *For 60% intensity:* (118×0.60)+72=141.6 bpm.
  - This means the user should aim to keep their heart rate between 119.2 and 141.6 bpm for moderate-intensity exercise.
- *Target heart rate max (THRmax):* 76.0–112.1 bpm
  - This appears to be another target heart rate calculation, likely for a different range or a different method of calculation (possibly incorporating different scaling factors or thresholds).

CHALLENGES

Despite the successful implementation of the proposed driver fatigue detection and rerouting system, several challenges were encountered during development and testing:

- *Variability in lighting conditions:*
  - The CNN-based eye status detection model demonstrated robustness across different lighting conditions; however, extreme variations in brightness and shadows occasionally impacted accuracy. Low-light environments and glare from headlights or sunlight caused inconsistencies in detecting open or closed eyes. Enhancing the model's adaptability to diverse lighting scenarios remains a key area for improvement.
- *Real-time data processing constraints:*
  - The integration of eye-tracking and heart rate monitoring required real-time data acquisition and processing. Latency issues were observed when handling continuous image frames and heart rate readings simultaneously. Optimizing computational efficiency, particularly for deployment on resource-limited embedded systems or edge devices, posed a challenge.
- *Accuracy of heart rate monitoring:*
  - While the simulated heart rate monitoring functioned effectively, real-time implementation using wearable, or in-car sensors introduced variability. Factors such as sensor placement, movement artifacts, and external noise affected heart rate accuracy, potentially leading to false positives or negatives in stress and fatigue classification.
- *Integration with google maps and V2V communication:*
  - Implementing real-time rerouting based on driver fatigue involved challenges in synchronizing Google Maps API responses with vehicle-to-vehicle (V2V) communication. Ensuring timely rerouting decisions required an efficient data transmission mechanism to minimize delays in vehicle coordination.
- *Model generalization and adaptability:*
  - The eye status detection model was trained on a dataset representing a variety of facial orientations and ethnicities, but real-world application revealed limitations in handling occlusions, such as sunglasses, facial hair, or masks. Improving model generalization to diverse driver demographics remains an ongoing challenge.

- *User interface responsiveness:*
  - The Streamlit-based user interface provided an intuitive interaction experience, but occasional lags were noted when handling large amounts of real-time data. Enhancing the UI's responsiveness and scalability, particularly in high-traffic scenarios, is necessary for a seamless user experience.
- *Ethical and privacy concerns:*
  - Continuous driver monitoring raises concerns about data privacy and ethical considerations. Storing and processing real-time physiological and facial data requires strict adherence to data protection regulations, ensuring secure handling and user consent mechanisms.

Addressing these challenges through improved model training, optimized algorithms, and enhanced hardware integration will further refine the system's accuracy, efficiency, and practical applicability in real-world driving environments.

FUTURE SCOPE AND DIRECTION

The proposed driver fatigue detection and rerouting system holds significant potential for further development and broader applications. Future advancements in the following areas can enhance its accuracy, efficiency, and real-world impact:

- *Enhanced deep learning models:*
  - Leveraging advanced architectures like Vision Transformers (ViTs) or hybrid CNN-RNN models can improve feature extraction and fatigue classification.
  - Expanding the dataset to include diverse facial features, lighting conditions, and occlusions (e.g., sunglasses, masks) will enhance model robustness.
- *Integration with biometric sensors:*
  - Incorporating high-precision biometric sensors such as EEG (electroencephalography) and ECG (electrocardiography) can provide deeper physiological insights into driver fatigue.
  - Using galvanic skin response (GSR) sensors can further refine stress and fatigue detection mechanisms.
- *Edge AI for real-time processing:*
  - Deploying the system on edge devices like embedded AI modules will minimize latency and enable real-time analysis.
  - Optimizing the model for low-power embedded systems will facilitate seamless in-vehicle implementation.
- *Smart traffic integration and adaptive rerouting:*
  - Future enhancements can integrate the system with intelligent traffic management and IoT-enabled infrastructure to enable dynamic rerouting based on real-time fatigue levels.
  - Vehicle-to-Everything (V2X) communication will enhance safety by allowing connected and autonomous vehicles to respond proactively to driver fatigue signals [13].
- *Multi-modal sensor fusion:*
  - Combining eye-tracking with additional behavioral cues such as yawning detection and head movement analysis will improve detection accuracy.
  - Integrating data from multiple sensors (cameras, LiDAR, infrared) will ensure robust fatigue assessment across diverse environmental conditions.
- *AI-driven personalized fatigue prediction:*
  - Implementing machine learning models that adapt to individual driver patterns will enable more precise and early fatigue detection.
  - Developing a personalized fatigue scoring system based on historical driving behavior will enhance proactive interventions.
- *Seamless integration with automotive safety systems:*
  - Collaborating with automotive manufacturers to incorporate the system into advanced driver-assistance systems (ADAS) will support broader industry adoption.

    - Enabling direct integration with vehicle control mechanisms, such as cruise control and braking systems, can provide semi-autonomous intervention in cases of severe fatigue.
- *Ethical considerations and data privacy:*
    - Strengthening data encryption and implementing privacy-preserving machine learning techniques, such as federated learning, will ensure driver data security.
    - Establishing compliance with industry regulations will support ethical AI deployment in automotive safety applications.

By advancing these research directions, this system can evolve into a fully integrated, AI-powered driver assistance solution, significantly enhancing road safety and reducing fatigue-related accidents.

CONCLUSION

The proposed system effectively integrates driver fatigue detection with real-time monitoring and intelligent navigation adjustments, enhancing road safety through automated decision-making. By utilizing a CNN-based model for eye status detection and heart rate monitoring, the system accurately classifies driver states, alert, drowsy, or unconscious, allowing for timely intervention. The system's ability to trigger alerts and provide rerouting suggestions ensures proactive risk mitigation.

The integration of Google Maps for real-time vehicle movement visualization and adaptive rerouting, along with vehicle-to-vehicle (V2V) communication, further strengthens the system’s capability to prevent accidents. Experimental results demonstrate that the decision tree logic, supported by a truth table, effectively determines the necessary actions based on physiological data. The Streamlit-based user interface provides an interactive and intuitive platform, allowing users to monitor driver conditions, visualize vehicle movements, and receive timely alerts.

While the system has shown high accuracy in detecting fatigue and stress, further enhancements, such as real-time hardware-based heart rate integration and improved edge-case detection, can refine its performance. Future work can explore the incorporation of additional biometric signals and machine learning refinements to improve prediction accuracy. Overall, the system presents a promising framework for intelligent driver monitoring, offering a robust solution for fatigue detection and safe navigation.

Acknowledgments

I would like to express my sincere gratitude to Dr. Pavan Kumar Gautam for his invaluable support and guidance in reviewing this research work. His insightful feedback, constructive suggestions, and expertise have greatly contributed to the enhancement of this work. I deeply appreciate his time and effort in reviewing the manuscript and offering critical advice that has helped refine the quality of this research.

REFERENCES

1. Panganai MA, Nyandoro LK, Zvarevashe K. Driver drowsiness detection using Convolutional Neural Networks-inspired features and Principal component analysis with K-Nearest Neighbors. In 2022 IEEE 1st Zimbabwe Conference of Information and Communication Technologies (ZCICT). 2022 Nov 9; 1–6.
2. Zhou L, Li S, Wang Y. Fatigue Detection and Early Warning System for Drivers Based on Deep Learning. In 2023 IEEE 3rd International Conference on Data Science and Computer Application (ICDSCA). 2023 Oct 27; 1348–1351.
3. Zhao Z, Zhou N, Zhang L, Yan H, Xu Y, Zhang Z. Driver fatigue detection based on convolutional neural networks using em-CNN. Comput Intell Neurosci. 2020; 2020(1): 7251280.
4. Wijaya D, Stanley M, Wilman P, Lucky H, Chowanda A. A Fatigue Detection Model Based on Convolutional Neural Network. In 2022 IEEE International Conference on Informatics, Multimedia, Cyber and Information System (ICIMCIS). 2022 Nov 16; 166–171.

5. Ghazal M, Haeyeh YA, Abed A, Ghazal S. Embedded fatigue detection using convolutional neural networks with mobile integration. In 2018 IEEE 6th international conference on future internet of things and cloud workshops (ficloudw). 2018 Aug 6; 129–133.
6. TAUIL Abd Elilah. (2022). Human eyes open\close. [Online]. Kaggle.com. Available from: https://www.kaggle.com/datasets/tauilabdelilah/mrl-eye-dataset.
7. Google. (2025). Geocoding Service. Google for Developers. [Online]. Available from: https://developers.google.com/maps/documentation/javascript/geocoding
8. Victoria DR, Mary DG. Driver drowsiness monitoring using convolutional neural networks. In 2021 IEEE International Conference on Computing, Communication, and Intelligent Systems (ICCCIS). 2021 Feb 19; 1055–1059.
9. Streamlit. (2024). A faster way to build and share data apps. [Online]. Streamlit.io. Available from: https://streamlit.io/
10. Capraro F, Milani S. Rendering-aware point cloud coding for mixed reality devices. In 2019 IEEE International Conference on Image Processing (ICIP). 2019 Sep 22; 3706–3710.
11. Nikolskaia K, Bessonov V, Starkov A, Minbaleev A. Prototype of driver fatigue detection system using convolutional neural network. In 2019 IEEE International Conference Quality Management, Transport and Information Security, Information Technologies (IT&QM&IS). 2019 Sep 23; 82–86.
12. Deshpande S. Traffic-Based Vehicle Rerouting and Driver Monitoring. [Online]. GitHub. Available: https://github.com/SprihaDeshpande/Traffic-Based-Vehicle-Rerouting-and-Driver-Monitoring/
13. GitHub. (2025). GitHub - parshva45/Vehicle-Monitoring-And-Routing-System. [online] Available from: https://github.com/parshva45/Vehicle-Monitoring-And-Routing-System.

