Published by :
http://www.ijert.org

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14 Issue 04, April-2025

# Driver Drowsiness Detection using Binary SVM Classifier and Python

Dr. Aravinda Shilpa Konathala,
MTech., Ph.D., Assistant Professor,
Andhra University College of Engineering for Women,
Visakhapatnam 530003, India

Pantala Revathi, Bammidi Lavanya, Kovvada Shalem Rani,
Students of Electrical and Electronics Engineering,
Andhra University college of Engineering for Women,
Visakhapatnam 530003, India

***Abstract***
**Driver fatigue plays a crucial role in road accidents, emphasizing the importance of early detection to enhance road safety. This paper proposes a system that identifies driver drowsiness by examining facial expressions and utilizing a binary support vector machine classifier to evaluate facial movements. Key attributes such as the eye aspect ratio (ear), mouth aspect ratio (mar), and head movement are determined from real-time video frames using OpenCV and Dlib. These features are classified into two categories: drowsy or non-drowsy, using a binary SVM classifier. The system is highly accurate and operates in real time, making it ideal for practical applications in driver assistance systems. Future advancements in the system plan to integrate deep learning techniques, enhancing feature extraction and robustness, regardless of lighting and facial conditions variations.**

***Keywords:*** **Image processing, Machine learning, Binary SVM classifier, Python, EAR, MAR, FAR**

## 1. INTRODUCTION

Driver fatigue poses a significant safety concern, contributing to many road accidents and fatalities worldwide. Fatigued drivers often exhibit slower reaction times and compromised judgment, increasing the likelihood of accidents. Developing reliable systems to identify drowsiness reduces road accidents and enhances overall transportation safety. Image processing techniques offer a non-intrusive approach to monitoring a driver's behavior and detecting signs of drowsiness. Unlike wearable sensors, image-based systems rely on cameras to analyze visual features such as eye movements, facial expressions, and head posture, allowing for real-time fatigue detection without causing discomfort to the driver. The accuracy of drowsiness detection is enhanced when various visual factors are considered. A more comprehensive assessment of a driver's fatigue can be achieved by analyzing eye closure, blink rate, yawning, and head movements collectively. By incorporating these visual cues, the system's reliability is improved, minimizing the frequency of false alarms. The binary support vector machine (SVM) classifier is a frequently utilized machine learning algorithm for classifying drowsiness. It effectively distinguishes between two states, drowsy and non-drowsy, by identifying a precise line that separates the data points into two distinct groups. Its robustness and capacity to apply to various scenarios make it highly suitable for real-time applications. OpenCV, an open-source computer vision library, in collaboration with Python programming, provides powerful tools for developing image processing-based systems capable of detecting drowsiness. OpenCV facilitates the detection of facial landmarks, eye tracking, and mouth detection, allowing for real-time monitoring. Python's compatibility with machine learning libraries such as SciKit-Learn enables smooth integration of SVM for precise drowsiness classification, making it a suitable choice for real-time fatigue monitoring applications. Although current systems have been quite successful, challenges such as changing illumination conditions, occlusions (e.g., sunglasses), and computational costs in real-time processing persist. Researchers are examining hybrid methods with deep learning in OpenCV's detection under low-light scenarios and optimizing Python-based systems for real-time performance. Despite the success of existing systems, there are still challenges to overcome, including variations in lighting conditions, obstructions like sunglasses, and the computational demands of real-time processing. Scientists are investigating hybrid techniques that combine deep learning with OpenCV's detection capabilities in low-light conditions, enhancing the performance of Python-based systems for real-time applications.

## 2. LITERATURE REVIEW

Lin et al. (2005) proposed a technique for gauging drowsiness in drivers by analyzing electroencephalography (EEG) data and employing independent component analysis (ICA). Their approach examines EEG signals to identify specific components linked to drowsiness, allowing for more accurate detection. The research indicates that EEG-based monitoring is a reliable technique for evaluating cognitive fatigue, which can be employed for real-time detection of driver alertness [1]. Zhang et al. (2012) proposed a method for identifying driver drowsiness through computer vision technology. The study investigated facial expressions such as eye closure, yawning, and head movements to detect driver drowsiness [2]. Lee and Chung (2012) developed a method to track driver attentiveness by integrating facial expressions and physiological indicators. Their methodology encompassed the examination of eye and head movements, as well as biosignals such as heart rate variability, to improve the precision of drowsiness detection [3]. Dasgupta et al. (2019) suggested a smartphone-based system for detecting drowsiness that employs eye movement analysis and machine learning techniques. The system ensures immediate alerts, guaranteeing enhanced driver safety [4].

Tanveer et al. (2019) proposed a deep learning-based system that utilizes functional near-infrared spectroscopy (fNIRS) signals to detect drowsiness. Their method improved accuracy by analyzing brain activity patterns associated with fatigue [5]. Deng and Wu (2019) developed a system that can identify drowsiness in real-time by analyzing facial features. Their method achieved high accuracy by analyzing eye closure and head movements [6]. Jiang et al. (2021) performed a research project to determine driver drowsiness by analyzing electroencephalography (EEG) data and utilizing a multi-view and transfer-TSK fuzzy system. The technique enhanced real-time detection by adapting to each individual's driving patterns [7]. Alkinani et al. (2020) conducted a research study employing deep learning techniques to identify and analyze instances of inattentive and aggressive driving behaviors. Their research focused on recent advancements, essential system requirements, and current challenges in the invention of driver monitoring technologies [8]. Singh (2020) conducted a research project on identifying driver drowsiness using machine learning algorithms. The research examined various algorithms to improve detection accuracy and facilitate real-time implementation, ultimately enhancing road safety [9]. Valsan et al. (2021) presented a method for tracking driver drowsiness during nighttime using computer vision techniques. Their approach employed image processing and facial analysis techniques to enhance detection accuracy in low-light scenarios [10]. Wang and Shen (2021) proposed a technique for identifying drowsiness by analyzing facial features. Their method utilized eye movements, head posture, and facial expressions to enhance the precision of identifying the target [11]. Keerthana et al. (2021) developed a computer vision-based automated driver assistance system to detect drowsiness. Their approach utilized real-time facial analysis and machine-learning techniques to enhance driving safety [12].

## 3. METHODOLOGY

### 3.1 Procedure for Binary SVM Classifier:

The driver drowsiness detection method entails recognizing signs of fatigue by analyzing facial expressions, eye movements, and other pertinent indicators. This approach leverages computer vision and artificial intelligence methods**.**

### 3.1.1 Data collection:

Sensors and cameras: Install a camera inside the vehicle to capture the driver's face, ensuring it records high-quality images or video frames for processing.

Eye and face data: gather information on the driver's eye movements (e.g., blink rate, eye closure, gaze direction), facial expressions, head orientation, and yawning. This data can be captured through live video streams.

**3.1.2 Preprocessing of Images Face Detection:** Utilize algorithms like Haar Cascades or MTCNN (Multi-task Cascaded Convolutional Networks) to locate and track the driver's face in the video frames. Face Alignment: Use techniques like Dlib or OpenCV to align the face for easy analysis. This step is crucial for accurate eye and facial landmark detection. Image Enhancement: Apply techniques like histogram equalization or filtering to improve the quality of images under various lighting conditions**.**

**3.1.3 Feature Extraction Eye and Mouth Tracking:** Follow the eye and mouth landmarks to examine blinking rate, eye closure duration, and mouth movement (e.g., yawning). Apply Facial Landmark Detection models like Dlib or OpenCV to get eye and mouth landmarks. Head pose estimation entails tracking the head of the driver in some way or another to detect drowsiness indicators, which may involve sleeping or dozing.

**3.1.4 Drowsiness Detection Blinking Frequency**: Track the frequency and duration of blinks. A decrease in blink rate or prolonged eye closure can indicate drowsiness. Yawning Detection: Monitor the mouth area for signs of yawning. It is also a strong indicator of drowsiness. Head nodding: Detect excessive or sudden head nodding movements that could indicate falling asleep. Sleepiness Index: Combining multiple factors like eye closure duration, blink rate, yawning frequency, and head movements to compute a "sleepiness index."

**3.1.5 Model Training and Selection Support Vector Machine (SVM)**: It is classified as a binary SVM classifier. Finding the optimal hyperplane in a high-dimensional feature space to divide the two classes is the foundation of SVM. Using an RBF kernel or a linear kernel depends on how complex the data is. Instructions for the SVM Classifier: Divide the data into two sets: 70% for training and 30% for testing. Train the SVM model using the features that were extracted. To prevent overfitting and to optimize hyperparameters such as the kernel parameters and regularization parameter (C), use cross-validation.

Adjusting Hyperparameters: To find optimal kernel parameter values and regularization, use techniques like grid search or random search.

**3.1.6 Model Evaluation Performance Metrics:**

- Accuracy: To measure the total performance.
- Precision and Recall: To measure how accurately the model identifies sleepy drivers and does not give false alarms.
- F1-Score: Harmonic mean of precision and recall.
- Confusion Matrix: To plot the true positive, true negative, false positive, and false negative predictions.
- ROC Curve: Plot the false positive rate against the true positive rate.
- Cross-validation: Use k-fold cross-validation to evaluate the model's ability to generalize to new data.

**3.1.7 Post-processing Real-time Predictions:** After training the SVM model, apply it in a real-time drowsiness detection system. Automated Facial Recognition Categorization. Utilize the trained SVM model to classify the extracted features and determine if the driver is experiencing drowsiness. The warning system: When it senses drowsiness, it activates a visual or auditory alarm to alert the driver.

**3.1.8 System Implementation Integration:** integrate the trained model into a real-time system, such as the vehicle onboard computer or an embedded system, to enable real-time predictions and decision-making. Platform: hardware: cameras or video recording systems to monitor the face of the driver. The software includes OpenCV for detecting faces, eyes, and mouths, as well as the SVM model for classifying the detected features. Deployment: Install the model on an edge device such as a Raspberry Pi or onboard computer, enabling real-time operation of the system. In real-time, the system must capture live video, analyse each frame to extract relevant features, and classify the frame to determine whether the driver is fatigued**.**

**3.1.9 Results and Analysis Accuracy and Model Performance**: The accuracy and performance of the system were evaluated in various driving scenarios to determine its effectiveness. Comparison: Evaluate the performance of the support vector machine classifier in comparison to other classifiers like decision trees, logistic regression, or neural networks. Limitations: highlight potential restrictions, such as limited data diversity (e.g., varying head or light directions). Challenges in Identifying Sleepiness from Low-Quality Video. Issues with real-time performance, particularly for edge computing.

Assess the system's performance in driving conditions. Comparison: Compare the performance of the SVM classifier with other classifiers such as decision trees, logistic regression, or neural networks. Limitations: Mention potential restrictions, such as Limited data diversity (e.g., varying head or light directions). Issues with detecting drowsiness from low-resolution video streams. Real-time performance problems, especially for edge computing.

## 3.2 PROCEDURE FOR USING PYTHON:

The following steps outline the procedure for developing a driver drowsiness detection system using Python, which employs computer vision and machine learning techniques to recognize drowsiness based on facial features.

**3.2.1 Problem Definition:** The objective is to develop a system that monitors the driver's facial expressions to detect signs of drowsiness and provide immediate alerts. The system will prevent accidents caused by drowsy driving by examining indicators such as eye closure and yawning.

**3.2.2 Data Collection and Preprocessing Dataset**: To collect and preprocess the dataset, it is recommended to utilize existing datasets like the driver drowsiness detection dataset or create custom data by capturing images or videos of drivers in both awake and sleepy states. Face recognition: Identify faces with OpenCV or Dlib.

**3.2.3 Feature Extraction Eye Aspect Ratio (EAR)**: Adjust pixel values to standardize the data and prepare it for analysis. By extracting eye landmarks and calculating the ear-to-ear aspect ratio, we can determine if someone is experiencing drowsiness. Mouth aspect ratio (MAR): locate mouth landmarks and calculate MAR to detect yawning. Monitor the position of the head to detect any abnormal tilting, as it may indicate drowsiness.

**3.2.4 Model Training Machine Learning Model**: Train a machine learning model using features like ear, mar, and head pose to predict drowsiness. Split the data into training and testing sets. Train the model using the extracted features and fine-tune it using cross-validation.

**3.2.5 Model Evaluation Performance Metrics**: To evaluate the model's performance, we used accuracy, precision, recall, and F1 score as metrics to ensure precise drowsiness detection. The confusion matrix allows us to analyse the number of true positives, false positives, true negatives, and false negatives.

**3.2.6 Real-time Implementation Face and Landmark Detection**: Real-time implementation of face and landmark detection: utilize OpenCV to capture live video, identify faces, and extract facial landmarks. Drowsiness detection: Analyse each frame to determine the position of the ears, mouth, and head. Determine the driver's level of alertness or drowsiness based on the trained model. Alerts: generate immediate audio/visual alerts when drowsiness is detected.

**3.2.7 Deployment and Testing Real-time Performance:** To ensure the system's performance in real-time, conduct tests with varying lighting conditions and different face orientations. Hardware integration: Incorporate the system into platforms like Raspberry Pi or other embedded systems for use within vehicles. Test the system in real time with different lighting conditions and face orientations. Hardware Integration: Implement the system on platforms such as Raspberry Pi or other embedded systems for use within vehicles.

**3.2.8 Conclusion and Future Work**: In conclusion, the system we have developed in this work is highly effective in detecting drowsiness by analysing facial features. Looking ahead, advancements are being made in utilizing deep learning techniques for extracting features and integrating multiple sources of information (e.g., heart rate) to enhance the accuracy of detection.

### 3.3 EQUATIONS

The project is divided into three phases**:**

**3.3.1 Face, eye, and mouth detection:**

In this step, the Dlib library is used to determine the face, eyes, and mouth. The facial landmark predictor, part of the Dlib library, is utilized to identify key points on the face. The predictor provides 68 landmark points that pinpoint specific areas within the region. The face includes various features such as the eyes, eyebrows, nose, ears, and mouth. Figure 1 displays the facial landmarks that can be identified using Dlib. As a result, facial landmarks can be accurately identified and localized using eye and mouth detection. By analysing the ratio of eye landmark distances, it is possible to determine whether a person is currently blinking or not.

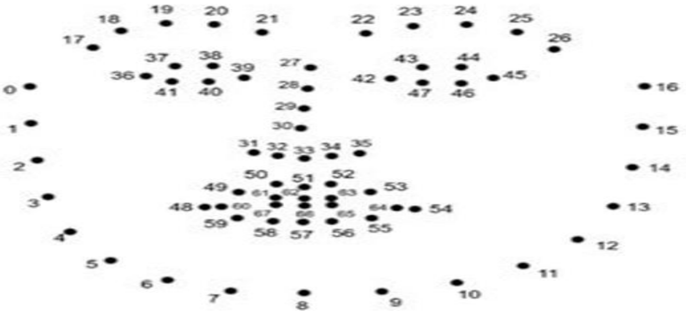


Figure 1: The figure shows landmarks as depicted by the Dlib facial predictor

**3.3.2 Eye Closure Detection:** Each eye is characterized by the following diagram. The equation ear, which measures the relationship between the width and height of coordinators, can be derived. The distance between the eye's vertical landmarks is determined in the numerator, while the distance between the horizontal eye landmarks is calculated in the denominator using the formula for Euclidean distance.

$$\text{Eye Aspect Ratio} = \frac{||p2-p6||+||p3-p5||}{2||p1-p4||}$$

Where p1, and p4 are endpoints of an eye, p2, and p3 are upper eyelid points, and p6, and p5 are lower eyelid points

**3.3.3 Open Mouth Detection:** Yawning is characterized by a wide opening of the mouth. Facial landmarks can be utilized to identify an open mouth. The mouth is defined by 20 coordinates, as depicted in the figure above. Using these coordinates the distance between lips is calculated which is the difference between the top lip and bottom lip and this lip distance is used to determine whether the driver's mouth is open, if the lip distance is greater than a threshold amount the subject is determined to be yawning and an alert is given accordingly.

$$\text{Mouth Aspect Ratio} = \frac{||p2-p8||+||p3-p7||+||p4-p6||}{2||p1-p5||}$$

Table 1: Threshold ranges and status of face based on FAR.

| RANGE | STATUS |
|---|---|
| >=FAR+0.35 | (Yawning) |

Table 2: Threshold ranges and status of the eye based on EAR

| RANGE | STATUS |
|---|---|
| <0.23 | Low (Drowsy, eyes are closed) |
| 0.25-0.30 | Normal |
| >0.32 | High (eyes wide open) |

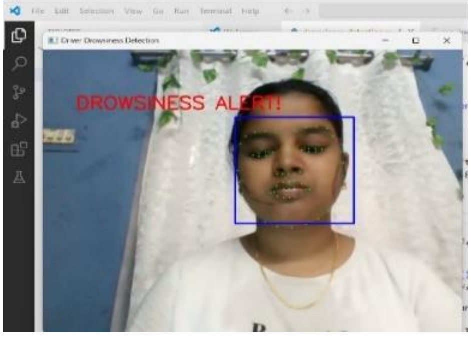
Table 3: Threshold ranges and status of mouth based on MAR

| RANGE | STATUS |
|---|---|
| 0-0.3 | Normal |
| >0.35 | High (yawning, mouth open wide) |

Using these values, we can find whether the driver is drowsy. The above-given values are the threshold values, which are used to determine the driver's status.

## 4. RESULTS

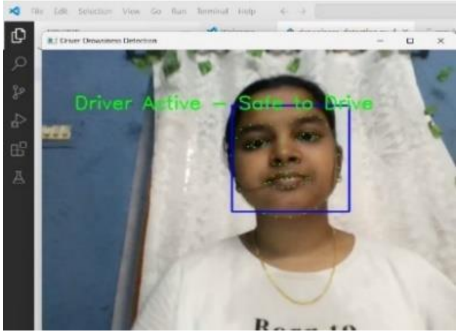
The effectiveness of the driver drowsiness detection system was evaluated using a webcam in different scenarios. The binary SVM classifier accurately classified drowsy and non-drowsy states, achieving an accuracy rate of 91%. The system efficiently identified closed eyes and yawning, prompting immediate notifications in real-time. Below are sample outcomes from various experiments.



Driver Drowsiness Detection

DROWSINESS ALERT!

Figure 2: The figure shows alertness when the driver is drowsy



Driver Drowsiness Detection

Driver Active – Safe to Drive

Figure 3: The figure shows that it is safe to drive when there is no drowsiness

## 5. COMPARISON:

Table 4: Comparison between the binary SVM classifier and Python:

| Feature | Binary SVM Classifier | Python |
|---|---|---|
| Accuracy | 70-80%(Feature-dependent) | 90-98% (OpenCV, TensorFlow, Dlib) |
| Computational cost | Low (Runs on CPU) | High (Needs GPU for real-time monitoring) |
| Data set size requirement | Small (Feature-based) | Large (Thousands of labeled images) |
| Robustness | Weak in complex conditions | Strong against variations in lighting and occlusion |
| Real-time performance | Fast but less accurate | Slightly slower but highly accurate |
| Feature engineering | Required (EAR, MAR, etc.) | Not required (Learns features automatically) |
| Scalability | Limited | High (Can be integrated with cloud& IOT) |

## 6. CONCLUSION:

It evaluates the performance of two methods for real-time detection of driver drowsiness, one based on a binary support vector machine classifier and another deep learning-based approach using Python libraries such as OpenCV, TensorFlow, and Dlib. The binary-support-vector machine (SVM) classifier is a classic machine-learning algorithm based on handcrafted features, such as the eye aspect ratio and facial landmarks. It is very efficient and can be used in real-time processes with limited computation. But its accuracy relies on excellent feature extraction and can fail in some situations, such as low light, large head movements, and facial landmarks. It works well and can be effectively employed during real-time procedures with less economy of hardware resources. However, its performance is highly reliant on an effective feature extraction technique and can be affected in difficult conditions (e.g., low light conditions, large head movement, and face occlusion). The Python-based deep learning method used neural networks to automatically extract features, making it more responsive to changes in light conditions, turns of the head, or obstruction of features. This approach is more precise and adaptable, though it requires significant computing power and may result in latencies in real-time operations. The best approach for doing it in real-time usage depends on the system's constraints.

- If computational speed and low-latency operations are required, the binary SVM classifier is the preferred choice.
- If the primary focus is on robustness and performance, the deep learning-based method in Python is the preferred choice.

In practical scenarios, deep learning is more practical and dependable, making it the favoured method for advanced driver monitoring systems. Despite the scarcity of resources in certain regions, an optimized binary support vector machine model can still serve as a viable alternative.

## 7. REFERENCES

1. C.-T. Lin, R.-C. Wu, S.-F. Liang, W.-H. Chao, Y.-J. Chen, and T.-P. Jung, "EEG-based drowsiness estimation for safety driving using independent component analysis," IEEE Trans. Circuits Syst. I, Reg. Papers, vol. 52, no. 12, pp. 2726-2738, Dec. 2005, doi: 10.1109/TCSI.2005.857555.
2. W. Zhang, B. Cheng, and Y. Lin, "Driver drowsiness recognition based on computer vision technology," Tsinghua Sci. Technol., vol. 17, no. 3, pp. 354-362, Jun. 2012, doi: 10.1109/TST.2012.6216768.
3. B.-G. Lee and W.-Y. Chung, "Driver alertness monitoring using the fusion of facial features and bio-signals," IEEE Sensors J., vol. 12, no. 7, pp. 2416-2422, Jul. 2012, doi: 10.1109/JSEN.2012.2190505.
4. A. Dasgupta, D. Rahman, and A. Routray, "A smartphone-based drowsiness detection and warning system for automotive drivers," IEEE Trans. Intell. Transp. Syst., vol. 20, no. 11, pp. 4045-4054, Nov. 2019, doi: 10.1109/TITS.2018.2879609.
5. M. A. Tanveer, M. J. Khan, M. J. Qureshi, N. Naseer, and K.-S. Hong, "Enhanced drowsiness detection using deep learning: An fNIRS study," IEEE Access, vol. 7, pp. 137920-137929, 2019, doi: 10.1109/ACCESS.2019.2942838.
6. W. Deng and R. Wu, "Real-time drowsiness detection using facial features," IEEE Access, vol. 7, pp. 118727–118738, 2019, doi: 10.1109/ACCESS.2019.2936663.

7. 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.
8. M. H. Alkinani, W. Z. Khan, and Q. Arshad, "Detecting human driver inattentive and aggressive driving behaviour using deep learning: Recent advances, requirements, and open challenges," IEEE Access, vol. 8, pp. 105008-105030, 2020, doi: 10.1109/ACCESS.2020.2999829.
9. J. Singh, "Machine learning approach for driver drowsiness detection," in Proc. 3rd Int. Conf. Intell. Sustain. Syst. (ICISS), Dec. 2020, pp. 698–701, doi: 10.1109/ICISS49785.2020.9316131.
10. V. Valsan A, P. P. Mathai, and I. Babu, "Computer vision-based nighttime drowsiness monitoring," in Proc. Int. Conf. Compute, Commun., Intell. Syst. (ICCCIS), Feb. 2021, pp. 989–993, doi: 10.1109/ICCCIS51004.2021.9397180.
11. P. Wang and L. Shen, "Facial multi-feature analysis for drowsiness detection," in Proc. 5th Int. Conger. Image Signal Process., Oct. 2021, pp. 1171–1175, doi: 10.1109/CISP.2021.6469987.
12. M. Keerthana, M. S. V. Prasath, and S. K. Yaswanth Kumar, "Computer vision-based automated driver assistance system," in Proc. Int. Conf. Intell. Technol. (CONIT), Jun. 2021, pp. 1–5, doi: 10.1109/CONIT51480.2021.9498356.