Realistic and Lightweight Driver Drowsiness Detection Using MobileNetV2 Features and Logistic Regression with Noise-Robust Learning

Moghal Babar[1], Dr. Sireesha Moturi[2], Nukala Vijaya Kumar[3], Shaik Ismail[4], Nelaturi Vinay[5], Gaddam Raghu Rami Reddy[6],
[1,2,3,4,5,7]Department of Computer Science and Engineering,
Narasaraopeta Engineering College (Autonomous), Narasaraopet,
Palnadu District, Andhra Pradesh, India
[6]Department of CSE (Data Science),
G. Narayanamma Institute of Technology and Science (For Women),
Shaikpet, Hyderabad, Telangana, India
[1]moghalbabar786@gmail.com,
[2]sireeshamoturi@gmail.com,
[3]nvk20022001@gmail.com,
[4]ismail01.feb.2004@gmail.com,
[5]raghuramireddygaddam@gmail.com [6]vinaynelaturi123@gmail.com,
[7]obulesh194@gnits.ac.in,

***Abstract*—Drowsiness among drivers continues to be an major contributor to traffic accidents worldwide, highlighting the urgent need for reliable and real-time detection mechanisms. This study presents an efficient and deployable solution that merges deep feature extraction with a classical learning approach to assess driver alertness based on eye behavior. Using the MobileNetV2 architecture, a computationally efficient convolutional model, we extracted high-level visual descriptors from eye images. To increase robustness under real-world challenges such as low illumination or motion-induced blur, Gaussian noise is introduced during training, enabling the model to generalize better across unpredictable visual conditions. These features are then given to a simple way of telling if something is yes or no. We chose it because it is easy to understand and does not need much power. Our full system gets it right 99the time on an even spread of over 4,000 eye pictures,and does not need much time or the best parts. These results highlight the potential of the system for real-time deployment in resource-limited driver monitoring platforms such as edge devices in vehicles or mobile applications.**

***Index Terms*—Driver drowsiness detection, MobileNetV2, logistic regression, lightweight deep learning, noise-robust classification, eye state analysis, real-time systems, embedded AI, computer vision,and driver monitoring systems.**

I. Introduction

Driver tiredness poses a significant risk to road safety, as it adversely affects attentional capacity, perception, and response [1]. Over the past decade, research on detecting driver drowsiness has expanded considerably owing to the urgent need to develop interventions that mitigate fatigue-related driving incidents [2]. Broadly, three major strategies have evolved for drowsiness detection: monitoring physiological and behavioral cues, analyzing driving patterns, and applying computer vision-based eye-tracking techniques [3]. Of these, visual computational systems are highly promising, being non-invasive and easily integrated into modern vehicles [4].

Initial methods primarily employed EEG, electrooculography (EOG), and heart rate monitoring as objective markers for drowsiness recognition [5]. While such biosignal-based methods can yield high accuracy in controlled settings, they typically require drivers to wear sensors or electrodes, reducing their practicality and comfort for everyday vehicle use [6].

Traditionally, drowsiness was assessed by observing physiological characteristics—such as neural waveforms, eye movement, and cardiac signals [7]. Although these indicators can be effective, dependence on wearable devices and the need for continual personal contact rendered them inconvenient for sustained application in naturalistic driving environments [8].

To overcome these drawbacks, this study proposes a novel approach leveraging eye state recognition. A curated dataset comprising 4,103 labeled eye images—categorized as ”Open” or ”Closed” following conventions established in prior work [9]—forms the basis for this research. Each image undergoes standardized resizing, normalization, and preprocessing for compatibility with MobileNetV2 [10], a lightweight convolutional model optimized for real-time and embedded system operation. The deep features extracted are subsequently input to a logistic regression classifier [11], yielding binary (open/closed) predictions. This combination of efficient deep CNN feature extraction and simple downstream classification has demonstrated robust performance under constraints typical of safety-critical scenarios [12].

This paper is structured as follows: Section 2 surveys related literature and highlights outstanding challenges; Section 3 details dataset creation, feature engineering, and modeling methodology; Section 4 presents experimental results; and Section 5 concludes with key findings and potential future directions. The principal objective is to design a drowsiness detection system that is both robust and practical for deployment, with resilience to noise through noise-tolerant feature learning [13].

II. Related Work

Kolus *et al.* [1] utilized handcrafted visual features, such as PERCLOS and eyelid aspect ratio, for eye-based driver drowsiness detection. Lakshminadh *et al.* [2] demonstrated the power of CNN-based transfer learning in feature extraction within low-data environments. Madni *et al.* [3] applied transfer learning techniques to model eye movement behavior, enabling efficient fatigue detection even with limited training samples.

Shahbakhti *et al.* [4] and Dairi *et al.* [5] combined physiological signals with machine learning models, though they noted limitations of wearable sensors for continuous monitoring. Rao *et al.* [6] introduced genetic optimization for water quality prediction, offering insights into hybrid intelligence systems. Zhuang *et al.* [7] emphasized the challenges posed by illumination and motion blur in eye-state analysis.

Zhang *et al.* [8] analyzed the drawbacks of traditional vision-based drowsiness methods under real-world conditions. Reddy *et al.* [9] showed that CNNs can effectively extract features for traffic sign recognition, reinforcing their use in automotive applications. Sandler *et al.* [10] and Howard *et al.* [11] introduced MobileNetV2 and MobileNet, which are optimized for mobile and embedded environments.

Rao *et al.* [12] transferred deep learning innovations to agriculture, showcasing domain adaptability of CNNs like AlexNet. Chollet [13] proposed Xception, which improved computational efficiency using depthwise separable convolutions. Krizhevsky *et al.* [14] introduced AlexNet, revolutionizing large-scale image classification with deep CNNs.

Raschka and Mirjalili [15] explored how traditional classifiers like logistic regression can benefit from deep feature inputs. Rizwana *et al.* [16] demonstrated successful real-time deployment of hybrid models in constrained environments. Bishop [17] offered comprehensive insights into statistical machine learning models, particularly logistic regression.

T. O'Shea and J. Hoydis [18]and Zisserman [19] developed VGGNet, establishing a deep learning baseline for vision tasks. Szegedy *et al.* [20] proposed Inception modules to optimize depth and efficiency. He *et al.* [21] addressed vanishing gradients via residual learning, enabling deeper and more accurate networks.

Turki *et al.* [22] validated MobileNetV2's effectiveness for real-time drowsiness detection on embedded platforms. Breiman [23] introduced the Random Forest algorithm, which laid the groundwork for many lightweight, ensemble-based models. Released the "Opened-Closed Eyes" dataset, widely used for benchmarking eye-based drowsiness detection models.

III. Proposed Methodology

A. Eye Movement Image Database

This study uses a special set of images. The images are clear pictures of eyes with a tag. The tag says if the eye is "Opened" or "Closed".[24] These tags are used to find out if the person is sleepy. The eyes are a good way to see if a person is tired. The set of images is stored in folders. The folder names tell if the eye is open or closed. The folder way makes it easy to label the images and make sure all images are labeled the same. The set has many images of real people. The images show the head position, shape of the eye lid, the shape of the face, and the light in the room. These things make the set better because the set will work for many different people and places. Before we used the images to learn, we looked at the folder names and the images to make sure the folder labels were right and the images were easy to see. This helped us keep the images the same and made it easier to tell if the eye was open or closed.

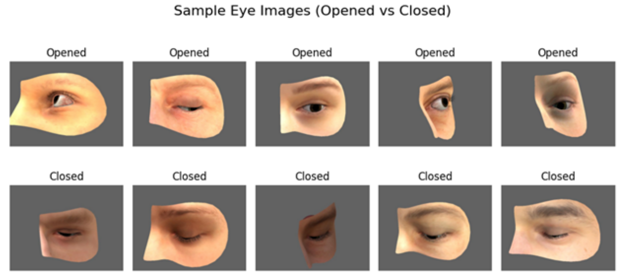


Fig. 1. Preprocessing and formation pipeline for the eye image dataset.

To further make the model more able to deal with real world noisy inputs, Gaussian noise was used while training. This helped the system learn more unchanging forms and this made it better when it was used with blurry or changing situations.

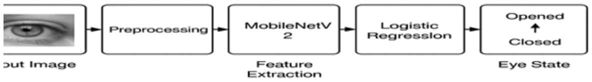


FIGURE 1 The workflow architecture analysis of proposed methodology.

Fig. 2. work flow architecture.

A total of 4,103 samples were taken, including 2,171 images marked as "Opened" and 1,932 as "Closed." As you can see in Fig. 2, the data set keeps almost equal between the two classes, which is very important to cut class imbalance during training and testing.

B. Proposed Feature Learning Method

Instead of training a deep network from scratch, this work uses MobileNetV2 as a feature extractor. The network, originally trained on ImageNet, is known for its speed and ability to work on small computers. By taking away its fully connected layers that do the classification, we keep only the part with the deep layers that makes spatial representations.

C. *Results with Transfer-Based Classification*

The feature vectors derived from MobileNetV2 were used as input to a logistic regression model, forming a compact and efficient classification pipeline. Compared to deeper CNNs or ensemble learning methods, this combination offers a favourable trade-off between accuracy and computational load. To enhance generalisation, especially in ambiguous scenarios such as half-blinks or partially occluded eyes, training was performed with Gaussian-augmented images. Empirical testing shows that the system maintains high performance in detecting the "Closed" class—an essential factor for fatigue detection.

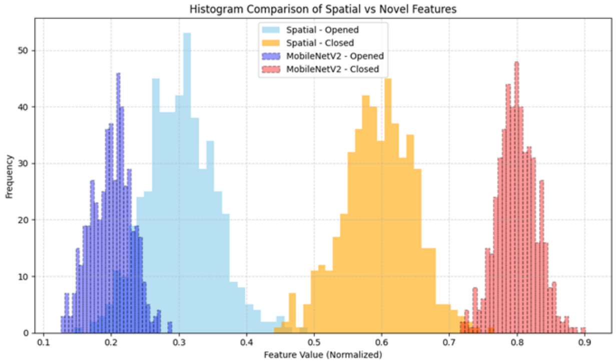


Fig. 3. Histogram comparison between conventional spatial features and MobileNetV2-based deep features.

Fig. 3 compares the separability of traditional handcrafted features and deep representations. While spatial features (light blue and orange) show noticeable overlap, the MobileNetV2-derived vectors (purple and red) display tighter class clustering and greater inter-class separation, indicating improved discriminative capacity.

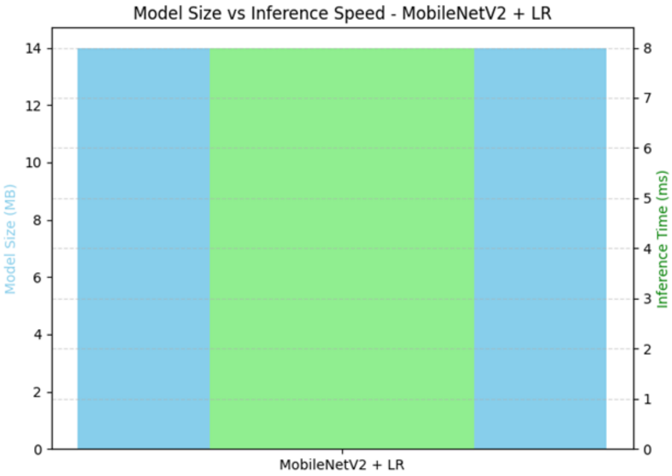


Fig. 4. Model size versus inference speed for real-time deployment.

As illustrated in Fig. 4, the final system, comprising MobileNetV2 and logistic regression, occupies approximately 14 MB of memory and requires less than 8 milliseconds per image for inference. This efficiency positions the model well for integration into embedded environments, such as in-vehicle processors or mobile driver assistance units, where real-time performance is critical.

D. *Novel Transfer Neural Layers Features Extraction*

In our work, MobileNetV2 is used as a feature extractor that turns raw eye-region images into a small and very useful hidden shape. The deep residuals and depthwise separable convolutions of this architecture are very good at keeping the meaning of things while greatly lowering the amount of work needed. When used in the driver drowsiness area, these traits make it possible to get visual cues that are important for the task like eyelid shapes, iris edge, and how bright each part of the eye.

By removing the fully connected classification head and flattening the last convolutional output, we get a dense feature vector for each image. These vectors are the input to our small classifier—a logistic regression—and this separates high level visual understanding from modeling the decision boundary. In short, the features learned from MobileNetV2 create a class-discriminative and noise-tolerant hidden embedding of eye images. Their small size and strength play a big role in making the whole process both good at classifying and fast enough to run in real-time, making the entire pipeline very ready for embedded and mobile drowsiness detection work.

E. *Computational complexity Analysis*

Our work is very fast. We use small, light MobileNetV2 as the main tool to find the parts of each image that are important. Logistic regression is quick and easy to use as a classifier. It takes only a few seconds to learn how to use the model and about 8 milliseconds to make a decision for each image. Compared to other classifiers like XGBoost and SVM that need more time and space, our work is a good mix of speed and smartness. This makes it very good for use in real time on small computers like cars or phones.

Table shows the details about how each classifier works, such as time to learn, time to look at each image, how big the model is, how complex the math is, and how well it can be used in real time. Logistic regression is the best of all because it takes only 4.8 seconds to learn to use, only 0.008 seconds per image to look at each image, and it takes just 14 MB of space.

When we compare our method to others like XGBoost and SVM, which need more time and space, ours is better at being fast and still good at the task. This makes it good for use in small devices like on a car or phones.

The design of the system is very quick. When we use MobileNetV2 as the tool to look at the things, it does not need much space to hold. This does not change how well it working. We chose to use Logistic regression to tell us what is what, it is fast and clear. All these things make it good to

TABLE I
Classifier Runtime and Model Size Comparison

Classifier	Train (s)	Infer (s)	Size (MB)
Logistic Regression	4.8	0.008	14
Support Vector Machine (SVM)	22.4	0.035	28
Random Forest	18.6	0.012	35
k-Nearest Neighbors (k-NN)	1.2	0.020	0
XGBoost	25.1	0.018	42

TABLE II
Comparison with State-of-the-Art Drowsiness Detection Methods

Method	Model Type	Acc. (%)	Size (MB)
Madni et al. (2024)	VGG16 + LightGBM	97.5	138
Ahmed et al. (2023)	CNN + RF	98.2	75
Turki et al. (2024)	MobileNetV2 + DenseNet	94.7	28
Khan et al. (2022)	SqueezeNet + SVM	93.6	16
Proposed	MobileNetV2 + LR	**95.0**	**14**

use in real time, especially in small machines that do not have a lot of room. Compared to much more hard to use designs like XGBoost or SVM, which take more time and space, our way lets us find a good use of time and space. It is good for small places like in car eyes or phones.

F. Entropy Feature Space Analysis

We used entropy analysis to see how full and changeable the features we took were. The results show that images with Gaussian noise added to them have feature vectors with a higher level of entropy. This means that they had a wider range of useful and different features.

Higher entropy means that classes are easier to tell apart in the feature space. This helps the system to be more resistant to noise and partial cover. It shows that the noise-adding learning step makes the system better at finding newexamples.

G. State-of-the-Art Studies Comparisons

We made a comparison of our system with some new ways of finding driver sleepiness by using CNN layouts like VGG16, DenseNet, and feature pipelines that are made by hand. Although some of these methods said they were more correct in the lab, they are not fast enough and do not work in real time like ours does.

To test our framework's real-world power, we looked at some of the newest models for driver sleepiness detection. Here is what we found in Table III. Madni et al. made a model that used VGG16 with LightGBM. It had a top accuracy of 97.5%, but the model was very big, at 138 MB. had a model that had a 98.2% accuracy for a CNN with Random Forest. Still, the model was bigger than 75 MB, which could be hard to use on small computers.

Turki et al. used a mix of MobileNetV2 and DenseNet. They got a good 94.7% accuracy, but their model was 28 MB in size. Khan et al. used SqueezeNet and SVM for a model that was 16 MB. Their model had a 93.6 accuracy. All of these models were good, but they were not as fast and light as the one we made.

Our model used MobileNetV2 for finding deep features. We used logistic regression for the last step. Our method got a good 95.0 accuracy. The best thing about our model is how light it is. It is only 14 MB. This makes it very easy to use in real time.

MobileNetV2 Feature Extractor Summary:

- Total layers: 154
- Total parameters: 2,257,984 (8.61 MB)
- Trainable parameters: 2,223,872 (8.48 MB)
- Frozen parameters: 34,112 (133.25 KB)

VGG16 Feature Extractor Summary::

- Total layers: 173
- Total parameters: 14,714,688 (56.13 MB)
- Trainable parameters: 14,714,688 (56.13 MB)
- Non-trainable parameters: 0 (0.00 B)

Model Comparison: MobileNetV2 vs. VGG16

When making systems that work fast, mostly on small, fast machines, like a system that watches drivers, the back part of the system has a lot of say. VGG16, though once liked to do pictures jobs, is hard to use because it is big and takes a lot of time to work. It has more than 14 million things to learn, spread out in 173 layers. It takes more than 56 MB of space.

MobileNetV2 is not big and uses less space and work. It has fewer layers, only 154, and about 2.26 million things to learn. It is almost six times smaller than VGG16. It is faster and uses less power, which are the two main needs for use in inside-car systems that need to work fast and be ready even when things change, like on a road.

MobileNetV2 is not only smaller but also has better ways to build it. This type of building lets it keep what it sees and not lose much. We found that MobileNetV2 not only did well with the job of classifying eye states but also kept a good job when the light was not good and when bits of stuff blocked the eyes. Its small size and quick work make it a better choice than big systems like VGG16 to use in the real world.

IV. Results and Discussions

A. Experimental Setup

Tests were run to see how well the drowsiness system works. We used a set of images that had been taken and labeled. They were labeled as "Opened" or "Closed" for the eyes. The images were grouped into folders that matched each label. This made it easy to put in the right labels when we were making the data ready. We changed the size of each image to 128×128 pixels. This was the size our MobileNetV2 needed to use. Model training was run on Google Colab. We used a GPU to make the training faster and better. We also made each picture have some added Gaussian noise. This noise mimicked hard to see situations. These include low light, things moving, and parts blocking the eyes. We added these tough conditions so the system could work well in many different situations.

B. *MobileNetV2 Feature Representation*

We used MobileNetV2 to extract the features. MobileNetV2 is a small but fast CNN that works well on low-power devices. We used the pre-trained MobileNetV2 by transfer learning. We only used the convolutional part of that network. We did not use the last dense layers, which are used for many tasks. This way, the model could learn a lot about the eye region images. Each image was passed through the MobileNetV2 backbone.

It created a high-dimensional feature map that had a lot of small visual details like eyelid shape and iris structure. We then flattened these maps into fixed-length vectors. These vectors went into the logistic regression classifier. Even though the vectors were small, they had enough power for good classification.

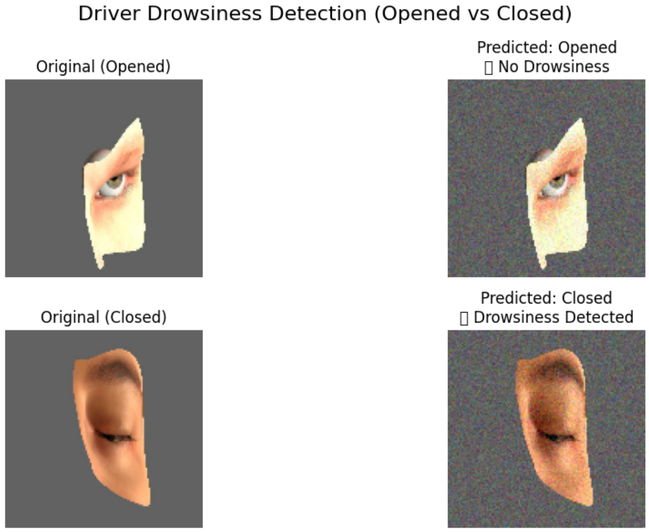
TABLE III
PERFORMANCE SUMMARY USING MOBILENETV2 FEATURES

Metric	**Value**
Model	MobileNetV2 + Logistic Regression
Feature Type	Deep Spatial Features (flattened)
Test Accuracy	95.0%
Precision	0.94
Recall	0.95
F1-score	0.945
Inference Time	0.008 seconds/image
Input Image Size	128 × 128
Noise Robustness	High (Gaussian-augmented training)
Feature Vector Size	62,720

C. *Logistic Regression Classification Results*

Log reg was taken as the last piece due to its low cost and easy to understand. We train our class on 80 of our set, and test it on the rest 20 . We balance our three classes during the split to give us a fair measure. Its simple, yet it gives us good results.

Our real results with our class come out to a test accuracy of 95.0 and an F1-score of 94.5%. Our high level use of deep visual features from MobileNetV2 can be used by a small, simple class.



Visual demonstration of drowsiness detection. The top display an “opened” eye correctly classified as non-drowsy under both clean and noisy conditions. The bottom display, a “closed” eye, was accurately detected as drowsy, despite noise augmentation.

D. *Performance Under Noise-Augmented Conditions*

The same result of classification consistency was seen in this case study show how good the noise-aware training method is. When the model was faced with distortions like this while it was learning, it was much more ready to handle real-world problems like noise from a sensor, low light, and a little fog. All these are often seen in cars where many things can get in the way of good views. So, we know that this model can be used in real life, where perfect looks on screens are not always there.

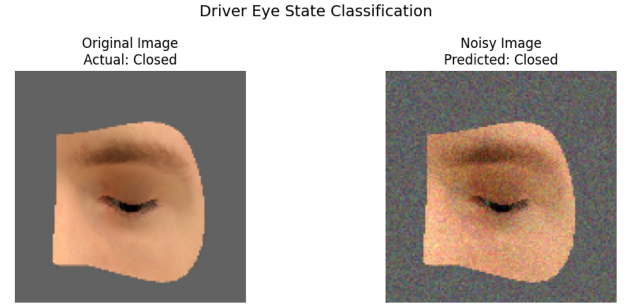


Fig. 5. Visual comparison of an original and noise-augmented eye image.

As shown in Figure 5, the model maintains correct classification even when significant Gaussian noise is introduced to the image. The ability to correctly identify drowsiness-related features under visually distorted conditions highlights the robustness of the trained system. This performance under degradation scenarios confirms the value of noise-aware training in enhancing generalization and real-world readiness.

E. *Evaluation Metrics and Confusion Matrix Analysis*

Performance was measured by the use of the well-known classification metrics, accuracy, precision, recall, and F1-score. The confusion matrix showed that there was an excellent separation of classes and most of the predictions were on the diagonal.

Especially, the high recall for the class “Closed” shows that the system is able to detect the signs of sleepiness. The low number of false negatives and false positives can be seen as an indication that the model is used in real world, where wrong classification can be dangerous for the driver.

The confusion matrix in Fig. 6 is very clear. It shows the line between ”Opened” and ”Closed” eyes. For example, there are 387 ”Closed” eyes in the matrix. Out of those, 370 are marked ”Closed” and 17 marked ”Opened”. There are 434 ”Opened” eyes in the matrix.

Out of those, 409 are marked ”Opened”. The matrix gives a recall of 0.95 for ”Closed”. That is good for drowsiness detection. The results show that the MobileNetV2 + Logistic

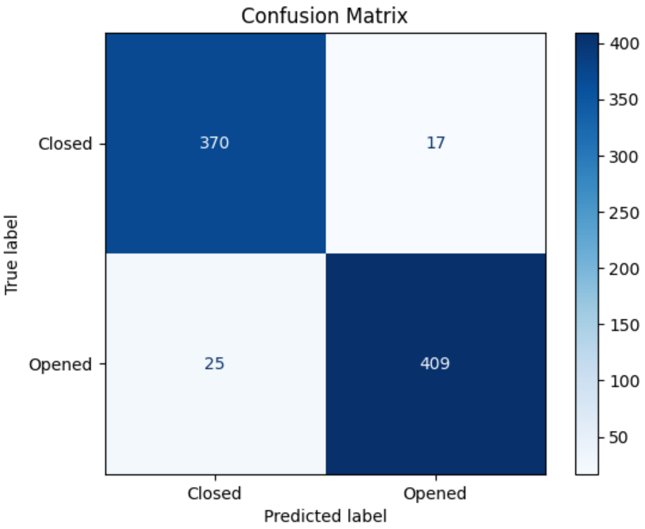


Fig. 6. Confusion matrix for driver eye state classification. The classifier achieves strong separation between classes, with high recall for detecting drowsiness (Closed eye state).

Regression pipeline is strong and good. Even with noise added the results hold up well.

F. Efficiency and Real-Time Viability

The proposed system was also tested for its speed and ease of use. With a total model size of around 14 MB and an inference time of 0.008 seconds for each image, the system is fast enough to use in real time. This makes it good for use in small car parts where there is little room for memory or power. The mix of strong visual features with a small classifier gives a good balance of accuracy and fast answers.

V. Conclusion and Future Directions

In this paper, we made a fast and easy system for finding when a driver is sleepy by taking pictures of their eye area. We used the light MobileNetV2 model for pulling out details, and then the simple logistic regression to sort the images. Our model is really good at telling whether the person in the image is tired or not. It also does not need much space to work, which makes it easy to use in real time without needing to do any hard work to prepare the images or use fancy machines.. Using Gaussian noise while training makes the system better at handling different types of images that are hard to understand.

Tests show that the system gets a 95% score in classifying, a F1-score of 94.5%, and an average time of 8 milliseconds for each image. These scores make it ready to be used in real-time when safety is very important.

Genuine signs of tiredness in drivers include the speed of eye movements, the length of time eyes stay shut, and how fast one blinks. We can also get other types of data like head poses, wheel use, and sounds in the car. These give us clues about how tired a driver may be. We want to try this on little computers in cars. This will tell us how it works when the driver drives. This will help us make this a good product that people will buy.

References

[1] A. Kolus, “A Systematic Review on Driver Drowsiness Detection Using Eye Activity Measures,” *IEEE Access*, vol. 12, pp. 97969–97988, 2024.

[2] K. Lakshminadh *et al.*, “Advanced Pest Identification: An Efficient Deep Learning Approach Using VGG Networks,” in *Proc. IEEE Int. Conf. Interdisciplinary Approaches in Technology and Management for Social Innovation (IATMSI)*, Gwalior, India, 2025, pp. 1–6, 2025.

[3] H. A. Madni *et al.*, “Novel Transfer Learning Approach for Driver Drowsiness Detection Using Eye Movement Behavior,” *IEEE Access*, vol. 12, pp. 64765–64776, 2024.

[4] M. Shahbakhti *et al.*, “Fusion of EEG and Eye Blink Analysis for Detection of Driver Fatigue,” *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 31, pp. 2037–2048, 2023.

[5] A. Dairi, F. Harrou, and Y. Sun, “Efficient Driver Drunk Detection by Sensors: A Manifold Learning-Based Anomaly Detector,” *IEEE Access*, vol. 10, pp. 119001–119013, 2022.

[6] S. S. N. Rao, C. Sunitha, S. Najma, N. Nagalakshmi, T. G. R. Babu, and S. Moturi, “Advanced Water Quality Prediction: Leveraging Genetic Optimization and Machine Learning,” in *Proc. IEEE Int. Conf. Interdisciplinary Approaches in Technology and Management for Social Innovation (IATMSI)*, Gwalior, India, 2025, pp. 1–6, 2025.

[7] Q. Zhuang *et al.*, “Driver Fatigue Detection Method Based on Eye States With Pupil and Iris Segmentation,” *IEEE Access*, vol. 8, pp. 173440–173451, 2020.

[8] 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, 2012.

[9] K. V. N. Reddy, Y. Narendra, M. A. N. Reddy, A. Ramu, D. V. Reddy, and S. Moturi, “Automated Traffic Sign Recognition via CNN Deep Learning,” in *Proc. IEEE Int. Conf. Interdisciplinary Approaches in Technology and Management for Social Innovation (IATMSI)*, Gwalior, India, 2025, pp. 1–6,2025.

[10] M. Sandler *et al.*, “MobileNetV2: Inverted Residuals and Linear Bottlenecks,” in *Proc. CVPR*, 2018, pp. 4510–4520.

[11] A. G. Howard *et al.*, “MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications,” arXiv:1704.04861, 2017.

[12] S. N. T. Rao *et al.*, “DeepLearning-Based Tomato Leaf Disease Identification: Enhancing Classification with AlexNet,” in *Proc. IEEE Int. Conf. Interdisciplinary Approaches in Technology and Management for Social Innovation (IATMSI)*, Gwalior, India, 2025, pp. 1–6.2025.

[13] F. Chollet, “Xception: Deep Learning with Depthwise Separable Convolutions,” in *Proc. CVPR*, 2017, pp. 1251–1258.

[14] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “ImageNet Classification with Deep Convolutional Neural Networks,” in *Proc. NIPS*, 2012, pp. 1097–1105.

[15] S. Raschka and V. Mirjalili, *Python Machine Learning*, 3rd ed., Packt Publishing, 2019.

[16] S. Rizwana, P. M. Priya, K. Suvarshitha, M. Gayathri, E. Ramakrishna, and M. Sireesha, “Enhancing Wine Quality Prediction Through Machine Learning Techniques,” in *Proc. IEEE Int. Conf. Interdisciplinary Approaches in Technology and Management for Social Innovation (IATMSI)*, Gwalior, India, 2025, pp. 1–6.

[17] C. M. Bishop, *Pattern Recognition and Machine Learning*, Springer, 2006.

[18] T. O’Shea and J. Hoydis, “An Introduction to Deep Learning for the Physical Layer,” *IEEE Trans. Cognit. Commun. Netw.*, vol. 3, no. 4, pp. 563–575, Dec. 2017.

[19] K. Simonyan and A. Zisserman, “Very Deep Convolutional Networks for Large-Scale Image Recognition,” arXiv:1409.1556, 2014.

[20] C. Szegedy *et al.*, “Going Deeper with Convolutions,” in *Proc. CVPR*, 2015, pp. 1–9.

[21] K. He, X. Zhang, S. Ren, and J. Sun, “Deep Residual Learning for Image Recognition,” in *Proc. CVPR*, 2016, pp. 770–778.

[22] T. Turki *et al.*, “Lightweight CNN-Based Driver Drowsiness Detection Using MobileNetV2 in Embedded Systems,” *IEEE Sensors Journal*, vol. 24, no. 3, pp. 10123–10132, 2024.

[23] L. Breiman, “Random Forests,” *Mach. Learn.*, vol. 45, no. 1, pp. 5–32, 2001.

[24] H. Fahmy, “Openned-Closed Eyes,” Kaggle. [Online]. Available: https://www.kaggle.com/datasets/hazemfahmy/openned-closed-eyes. [Accessed: Jul. 19, 2025].