DROWSINESS DETECTION SYSTEM USING MACHINE LEARNING

[1]Prakhar Pratap, [2]Diwakar Yagyasen, [3]Nitin Tiwari, [4]Adarsh Vipul, [5]Pratik Kanojia

[1,3,4,5]Student, [2]Professor
Computer Science and Engineering,
Babu Banarasi Das Institute of Technology and Management
Lucknow, Uttar Pradesh, India

***Abstract*— Drowsiness Detection System has been developed using non-intrusive machine vision-based concepts. The system has a small monochrome camera that points directly toward the person's face and monitors the subject's eyes to detect fatigue. In such a case, when fatigue/drowsiness on the subject face is detected, a warning signal is issued to alert. This report describes how to find the eyes and determine if the eyes are open or closed.**

***Index Terms*—Machine learning, Deep learning, Convolutional Neural Network.**

I. INTRODUCTION

A drowsiness detection system is designed to monitor and analyse a person's physical and behavioural signs to determine their level of alertness or drowsiness. This system aims to alert the person when they are becoming drowsy or are at risk of falling asleep, particularly when alertness is critical for safety, such as driving or operating heavy machinery. Drowsy driving is a significant problem worldwide, leading to numerous accidents and fatalities yearly. To address this issue, researchers have been developing drowsiness detection systems that can monitor a driver's physical and behavioural signs to detect signs of drowsiness and alert the driver to take necessary action.

Drowsiness detection systems use a variety of methods, including eye tracking, electroencephalography (EEG) [2], heart rate variability (HRV), and facial recognition, to monitor the driver's condition and assess their level of alertness [4]. Studies have shown promising results for these systems, with accuracies ranging from 91.8% to 94.62% for detecting drowsiness in drivers[1] [4]. A system that uses a convolutional neural network (CNN) to analyse facial features and eye movement achieved an accuracy of 91.8% in detecting drowsiness in drivers[3]. Driver sleepiness, alcoholism, and carelessness are the key contributors to the accident scene. The development of drowsiness detection systems has the potential to improve safety on the roads and in other settings where alertness is critical, such as manufacturing and healthcare.

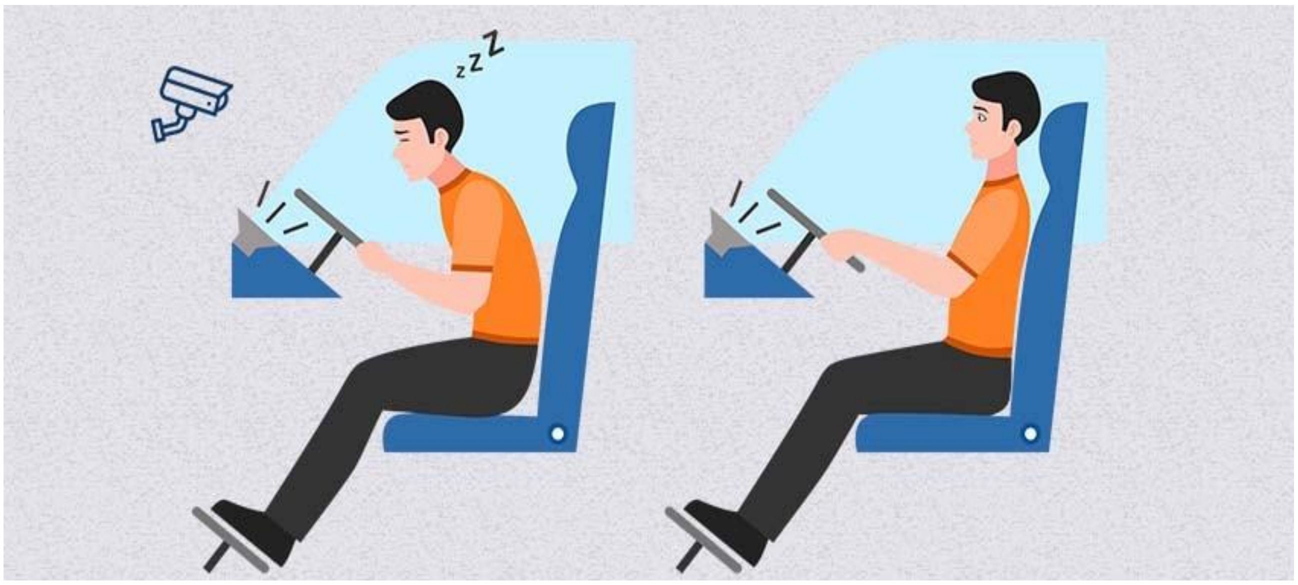
The automotive population is increasing exponentially in our country. The Biggest problem with the increased use of vehicles is the rising number of road accidents. Road accidents are undoubtedly a global menace in our country. Several surveys on road accidents say that driver fatigue causes around 30 percent of accidents[6]. The attention level of drivers degrades because of less sleep, long continuous driving, or other medical conditions like brain disorders. When a driver drives for more than a normal period for humans, then excessive fatigue is caused and also results in tiredness which drives the driver to a sleepy condition or loss of consciousness[10].

II. Methodology

The model we used is built with Keras using Convolutional Neural Networks (CNN). A convolutional neural network is a special deep neural network that performs extremely well for image classification purposes. Long-time driving brings sweat to the sensors, reducing their capacity to screen precisely; hence the approach will mostly focus on the percentage of eye closure as it provides the most accurate information on drowsiness. It is also non-intrusive in nature and thus does not affect the driver's state, and the driver feels comfortable with this system. The methodology includes three steps:

1. **Data Collection and Pre-processing**: The first step is to collect the data and pre-process it. The data collected can be in the form of images or videos, and it should include both drowsy and alert states. Pre-processing may involve resizing the images, converting them to grayscale, and normalizing the pixel values.
2. **Training the CNN Model**: The second step is to train the CNN model using the pre-processed data. The CNN model consists of several layers, including convolutional layers, pooling layers, and fully connected layers. During the training process, the model learns to identify the features that distinguish drowsy and alert states based on the eye closure percentage. The model is trained using a set of labelled data, and the weights of the model are adjusted to minimize the error between the predicted and actual labels.
3. **Testing and Evaluation**: The final step is to test the performance of the CNN model on new data that it has not seen before. The model's accuracy is evaluated using metrics such as precision, recall, and F1 score. If the model's performance is satisfactory, it can be deployed for real-time drowsiness detection. Otherwise, the model may need to be retrained with more data or fine-tuned to improve its performance.

The proposed system works well under low light conditions if the camera delivers better output. The system can differentiate between normal eye blinking and drowsiness and detect drowsiness while driving.



Our model is built with Keras using Convolutional Neural Networks (CNN). A CNN consists of an input layer, an output layer, and a hidden layer which can have multiple numbers of layers. A convolution operation is performed on these layers using a filter that performs 2D matrix multiplication on the layer and filter. The CNN model architecture consists of the following layers:
● Convolutional layer; 32 nodes, kernel size 3
● Convolutional layer; 32 nodes, kernel size 3
● Convolutional layer; 64 nodes, kernel size 3
● Fully connected layer; 128 nodes

Step 1 – Take Image as Input from a Camera
With a webcam, we will take images as input. So to access the webcam, we made an infinite loop to capture each frame. We use the method provided by OpenCV; cv2.VideoCapture(0) is used to access the camera and set the capture object (cap). cap.read() will read each frame and store the image in a frame variable.

```
 import cv2
cap = cv2.VideoCapture(0)  (# create a video capture object for the default camera #)
while True:
    ret, frame = cap.read()  (# read a frame from the camera #)
    if not ret:  (exit the loop if the frame could not be read)
        break
```

Step 2 – Detect Face in the Image and Create a Region of Interest (ROI)
To detect the face, we need first to convert the image into grayscale as the OpenCV algorithm takes grey images in the input for object detection. We don’t need colour information to detect the objects. We will use a haar cascade classifier to detect faces. This line is used to set our classifier face = cv2.CascadeClassifier(‘ path to the haar cascade xml file’). Then we perform the detection using faces = face_cascade.detectMultiScale(grey). It returns an array of detections with x,y coordinates and height, the width of the boundary box of the object. Now we can iterate over the faces and draw boundary boxes for each face.

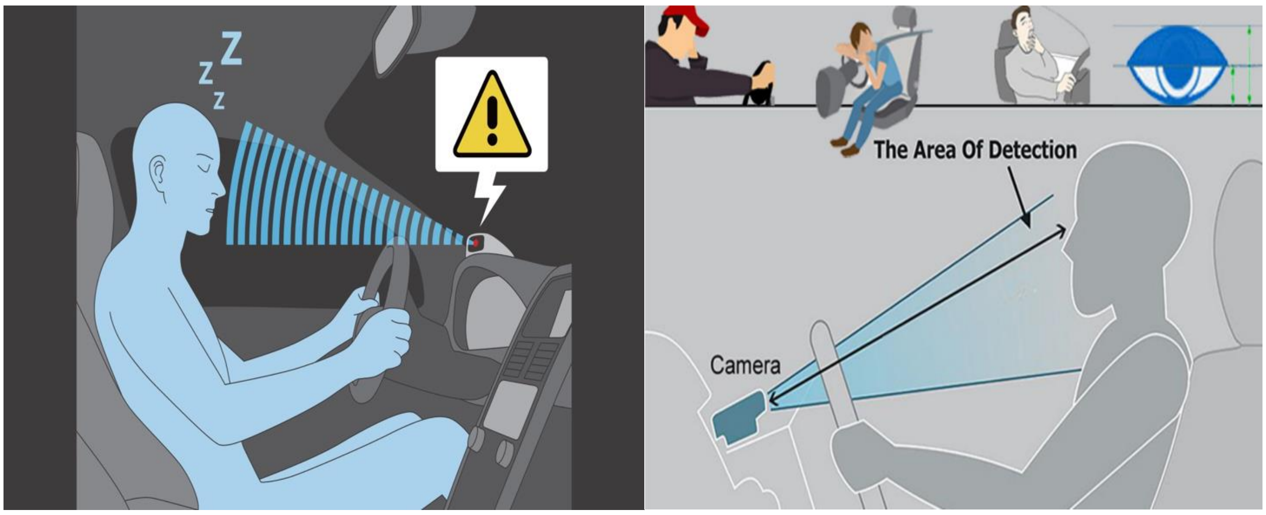
```
import cv2
face_cascade = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')

img = cv2.imread('face1.jpg')
grey = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY) (# read an image and convert it to grayscale #)

faces = face_cascade.detectMultiScale(grey) (# detect faces in the grayscale image using the classifier #)
for (x, y, w, h) in faces:
    cv2.rectangle(img, (x, y), (x+w, y+h), (0, 255, 0), 2) (# draw rectangles around the detected faces #)
```

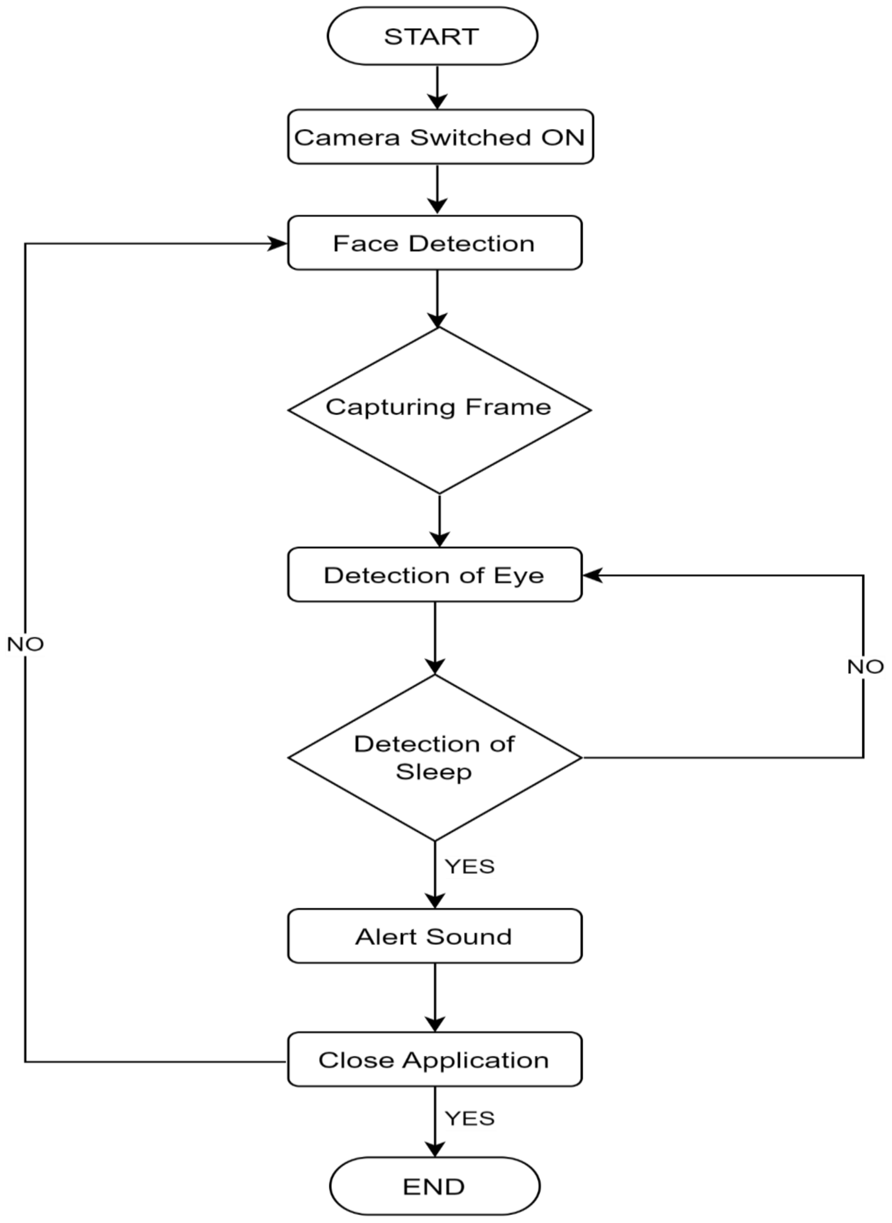
Step 3 – Detect the eyes from ROI and feed it to the classifier
The same procedure to detect faces is used to detect eyes. First, we set the cascade classifier for eyes in reye and leye, respectively, then detect the eyes using left_eye = leye.detectMultiScale(grey). Now we must extract only the eye's data from the full image. This can be achieved by extracting the boundary box of the eye, and then we can pull out the eye image from the frame with this code.

```
reye = None
leye = None
for (x, y, w, h) in eyes:
  if x < img.shape[1] / 2:  # left eye
    leye = (x, y, w, h)
    cv2.rectangle(img, (x, y), (x+w, y+h), (0, 255, 0), 2)
  else:  # right eye
    reye = (x, y, w, h)
    cv2.rectangle(img, (x, y), (x+w, y+h), (0, 0, 255), 2)
```

Step 4 – Classifier will Categorize whether Eyes are Open or Closed
We are using CNN classifiers for predicting eye status. To feed our image into the model, we need to perform certain operations because the model needs the correct dimensions. First, we convert the color image into grayscale using r_eye = cv2.cvtColor(r_eye, cv2.COLOR_BGR2GRAY). Then, we resize the image to 24*24 pixels as our model was trained on 24*24 pixel images cv2.resize(r_eye, (24,24)). We normalize our data for better convergence r_eye = r_eye/255 (All values will be between 0-1). Expand the dimensions to feed into our classifier. We loaded our model using model = load_model('models/cnnCat2.h5'). Now we predict each eye with our model lpred = model.predict_classes(l_eye). If the value of lpred[0] = 1, it states that eyes are open; if the value of lpred[0] = 0, it states that eyes are closed.

Step 5 – Calculate Score to Check whether a Person is Drowsy
The score is a value we will use to determine how long the person has closed his eyes. So if both eyes are closed, we will keep increasing the score, and when eyes are open, we decrease the score. We are drawing the result on the screen using the cv2.putText() function, which will display the real-time status of the person.





Flowchart Showing Process

III. Results

The drowsiness detection system implemented in the driver abnormality monitoring system has numerous advantages and can be used for various applications. One of the primary benefits is the detection of drowsiness, which can help prevent accidents caused by driver fatigue. The system is particularly useful for heavy vehicles like trucks, which often have long driving periods, and commercial vehicles like buses, where many people rely on public transport for traveling. By implementing the system in these vehicles, passengers can have added safety measures during their commute.

The system can also be used for overloaded cranes and mobile cranes, where the risk of accidents due to driver fatigue is high. By detecting drowsiness, the system can help prevent accidents and improve workplace safety. Another advantage of the system is that it does not require monitoring cameras or any other devices attached or aimed at the driver. This makes it a practical and easy-to-use solution for various applications. Overall, the driver abnormality monitoring system's drowsiness detection system is an effective technology that can help decrease road accidents and improve workplace safety. The practicality of the system and its ability to detect drowsiness without additional devices or cameras make it a valuable tool for various industries.

.

IV. Conclusion

The driver abnormality monitoring system is designed to detect drowsiness while driving and prevent accidents caused by sleepiness. The system uses a Drowsiness Detection System that is based on eye closure of the driver, which can differentiate between normal eye blink and drowsiness and detect the drowsiness while driving.

The system works by capturing images of the driver's face and eyes, and processing them to obtain information about the head and eye positions. The processing algorithms analyse the images and judge the driver's alertness level based on continuous eye closures. If the system determines that the driver's eyes have been closed for too long, a warning signal is issued to alert the driver.

The system can also detect the level of alertness even when the driver is wearing spectacles, and can function effectively under low light conditions if the camera delivers better output. This makes the system versatile and reliable in various driving conditions.
Overall, the driver abnormality monitoring system is an advanced technology that can help improve road safety by preventing accidents caused by driver drowsiness..

REFERENCES:

1. Li, M., Li, Z., Yang, Y., & Lu, Q. (2021). Real-Time Driver Drowsiness Detection System Based on Multimodal Analysis. Sensors, 21(13), 4463.
2. Bhandari, S., Dhanwani, A. D., & Khadkikar, S. B. (2021). Drowsiness Detection using EEG Signals: A Review. International Journal of Advanced Intelligence Paradigms, 16(1), 1-24.
3. Liu, Y., Feng, J., & Wang, Z. (2021). Driver Drowsiness Detection using Convolutional Neural Networks. IEEE Access, 9, 54674-54682.
4. Patil, V. C., & Deshmukh, R. B. (2020). A Review of Drowsiness Detection Techniques for Cognitive Assessment. International Journal of Computer Applications, 175(26), 19-25
5. Hoque, M. A., Hossain, M. S., & Taslim, S. M. (2020). Driver Drowsiness Detection using Multimodal Features and Random Forest Classifier. In Proceedings of the 2020 11th International Conference on Intelligent Systems, Modelling and Simulation (pp. 428-433). IEEE.
6. Damien Leger, The Cost of Sleep-Related Accidents: A Report for the National Commission on Sleep Disorders Research, Sleep, Volume 17, Issue 1, January 1994, Pages 84–93.
7. H. Su and G. Zheng, "A Partial Least Squares Regression-Based Fusion Model for Predicting the Trend in Drowsiness," in *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, vol. 38, no. 5, pp. 1085-1092, Sept. 2008, doi: 10.1109/TSMCA.2008.2001067..
8. F. Friedrichs and B. Yang, "Camera-based drowsiness reference for driver state classification under real driving conditions," *2010 IEEE Intelligent Vehicles Symposium*, 2010, pp. 101-106, doi: 10.1109/IVS.2010.5548039.K. Elissa, “Title of paper if known,” unpublished.
9. M.J. Flores J. Ma Armingol A. de la Escalera, “Driver drowsiness detection system under infrared illumination for an intelligent vehicle” Published in IET Intelligent Transport Systems 5(2011): 241-251. Received on 13th October 2009 Revised on 1st April 2011
10. Cheng, Qian & Wang, Wuhong & Jiang, Xiaobei & Hou, Shanyi & Qin, Yong. (2019). Assessment of Driver Mental Fatigue Using Facial Landmarks. IEEE Access. PP. 1-1. 10.1109/ACCESS.2019.2947692.M. Young, The Technical Writer's Handbook. Mill Valley, CA: University Science, 1989.