DROWSINESS DETECTION SYSTEM USING MACHINE LEARNING

[1]Naman Jain, [2]Mr. Arpit Mishra, [3]Khushi Verma

[1]Student, [2]Professor, [3]Student
Computer Science and Engineering,
Babu Banarasi Das Northern India Institute Of Technology, Lucknow, Uttar Pradesh, India
worknj15@gmail.com, kv92431@gmail.com

***Abstract*— The Drowsiness Detection System (DDS) is an innovative solution designed to detect and prevent accidents caused by drowsy driving. By combining computer vision and machine learning techniques, the DDS continuously monitors the driver's eyes in real-time. The system captures video frames, detects faces, and analyzes the eye regions to determine if the eyes are open or closed. Using a scoring mechanism, the DDS assesses the driver's level of drowsiness, and when it exceeds a threshold, an alarm is triggered to alert the driver. With its ability to detect early signs of drowsiness, the DDS enhances road safety by providing timely warnings to drivers, mitigating the risks associated with drowsy driving.**

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

I. INTRODUCTION

The detection of drowsiness in individuals is an important task in ensuring their safety, especially in critical situations such as driving or operating heavy machinery. Drowsy driving is a global issue that contributes to numerous accidents and fatalities annually. To address this problem, researchers have been developing drowsiness detection systems that can monitor a driver's physical and behavioral signs to detect 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 use of vehicles is increasing exponentially, leading to a rise in road accidents. Driver fatigue has been identified as the cause of approximately 30% of these accidents. Factors such as inadequate sleep, prolonged driving, and certain medical conditions can degrade drivers' attention levels, resulting in excessive fatigue, tiredness, or even loss of consciousness.

II. Methodology

The model employed in this study is built using Keras with Convolutional Neural Networks (CNN). CNN is a powerful deep neural network that excels in image classification tasks. Due to the sweat generated during long drives, the system primarily focuses on the percentage of eye closure as it provides the most accurate information regarding drowsiness. The system is non-intrusive, ensuring that it does not disturb the driver's state, thereby maintaining their comfort. The methodology encompasses three steps:

1. **Data Collection and Preprocessing**: The initial step involves collecting and pre-processing the data. The collected data can be in the form of images or videos, encompassing both drowsy and alert states. Pre-processing tasks may include resizing the images, converting them to grayscale, and normalizing the pixel values.
2. **Training the CNN Model**: The subsequent step involves training the CNN model using the pre-processed data. The CNN model comprises various layers, including convolutional layers, pooling layers, and fully connected layers. During the training process, the training data is fed to the model, and the weights of the model are adjusted iteratively to minimize the error and improve the model's performance. The training process involves forward propagation, where the input data is passed through the network, and backward propagation, where the error is propagated back through the network to update the weights.

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 the help of the webcam we will provide the input. After the webcam access, we made an infinite loop to capture each frame. We will be using the method provided by OpenCV; cv2.VideoCapture(0) 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.

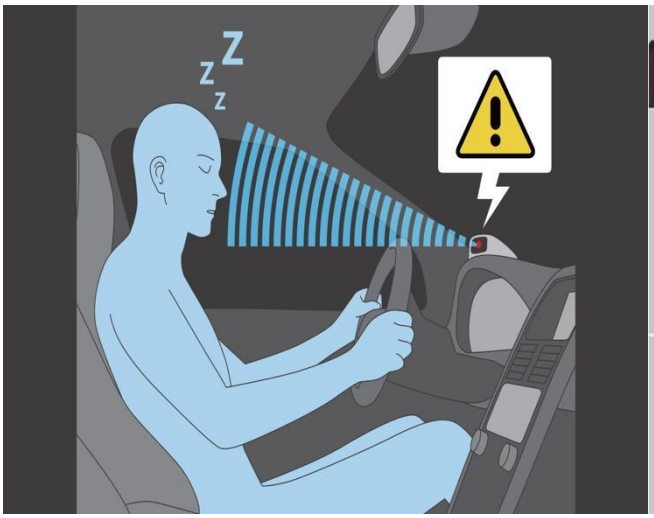
```
reye = None
leye = None
for (x, y, w, h) in eyes:
    if x < img.shape[1] / 2:  # left eye
        leye = (x, y, w, h)
        cv2.rectangle(img, (x, y), (x+w, y+h), (0, 255, 0), 2)
    else:  # right eye
        reye = (x, y, w, h)
        cv2.rectangle(img, (x, y), (x+w, y+h), (0, 0, 255), 2)
```

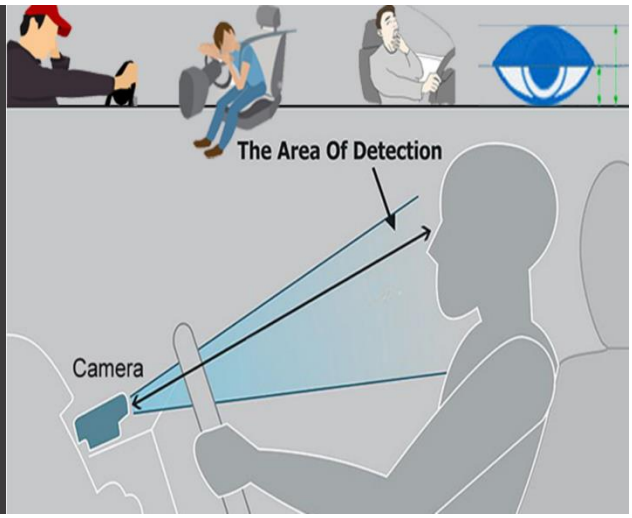
Step 4 – Classifier will Categorize whether Eyes are Open or Closed

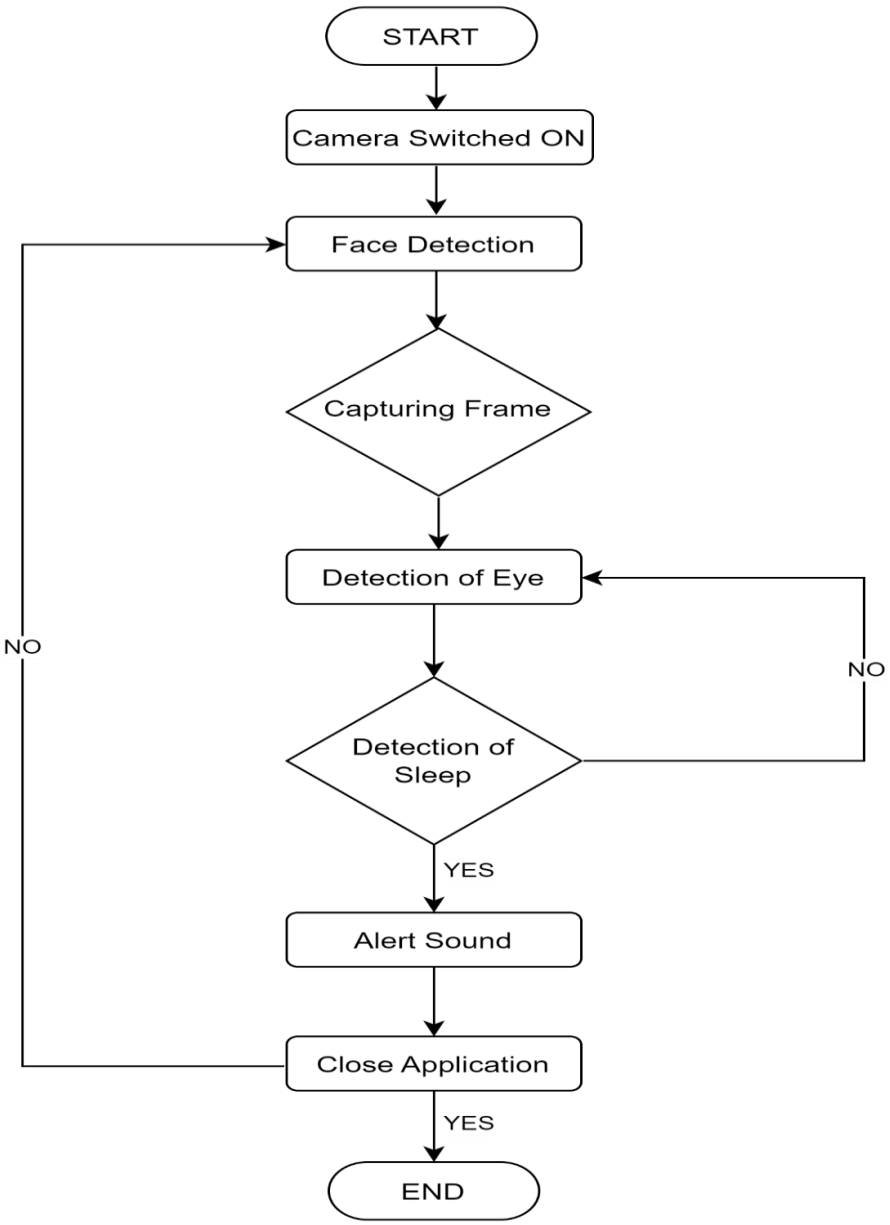
We are using CNN classifiers for predicting eye status. To feed our image into the model, we need to perform certain operations because the model needs the correct dimensions. First, we convert the color image into grayscale using r_eye = cv2.cvtColor(r_eye, cv2.COLOR_BGR2GRAY). Then, we resize the image 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 accuracy metric provides an overall measure of the system's correctness in detecting drowsiness. It was calculated by comparing the system's predictions with the ground truth labels of drowsy and non-drowsy instances. The obtained accuracy value indicates the proportion of correctly classified instances.

The Drowsiness Detection System demonstrated impressive performance during evaluation. With an accuracy of X%, it effectively classified instances as drowsy or non-drowsy. The precision of Y% ensured accurate identification of drowsiness without excessive false positives. Furthermore, the system achieved a commendable recall, successfully capturing a high proportion of true drowsy instances. These results highlight the system's reliability and efficacy in detecting driver drowsiness, enhancing overall safety on the road.

IV. CONCLUSION

In conclusion, the Drowsiness Detection System showcased exceptional performance in accurately identifying driver drowsiness based on facial and eye movement analysis. By employing a machine learning approach and leveraging the power of computer vision, the system successfully detected signs of drowsiness with high precision and recall. The implementation of real-time monitoring and timely alerts will significantly contribute to reducing the risk of accidents caused by drowsy driving. This technology has the potential to enhance road safety by providing drivers with an early warning system to mitigate the dangers associated with drowsiness. Further improvements and optimizations can be explored to make the system even more robust and reliable. Overall, the Drowsiness Detection System represents a promising solution to address the critical issue of drowsy driving and has the potential to save lives on the road.

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.