Drowsiness Detection using CNN

B. Sreeram Bhuvan
School of Computer Science and Engineering
VIT-AP University
Amaravati, India
sreeram.20bci7058@vitap.ac.in

V. Sunitha
School of Computer Science and Engineering
VIT-AP University
Amaravati, India
sunitha.20bci7049@vitap.ac.in

C. Sai Venkata Yaswanth
School of Computer Science and Engineering
VIT-AP University
Amaravati, India
yaswanth.20bci7076@ vitap.ac.in

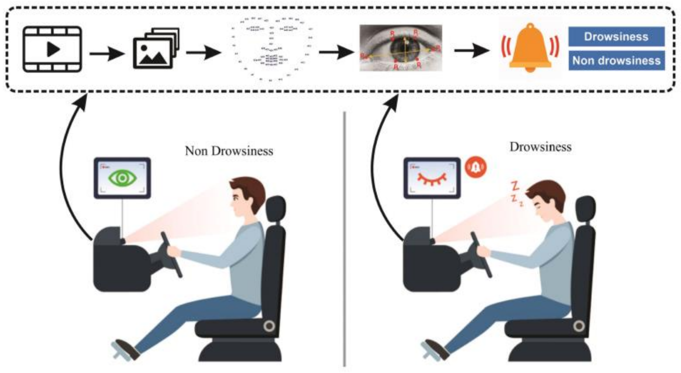
ABSTRACT:

The proposed method for detecting driver drowsiness using machine learning techniques is designed to estimate a driver's condition using basic characteristics such as age, gender, and driving experience, as well as analyzing driving behavior, facial expressions, and bio-signals. The method utilizes various machine learning models to detect drowsiness by analyzing the position of the eyes and extracting detailed features of the mouth. The aim of this system is to improve road safety by predicting and alerting drivers before they fall asleep while driving. The current detection of drowsiness methods are mainly based on machine learning algorithms. Many researchers have followed visual behaviors with machine learning for implementing the drowsiness detection system. The proposed system is reliable and has the potential to reduce the number of accidents caused by driver drowsiness.

Keywords— drowsiness, detection, reliable.

INTRODUCTION

Road accidents caused by drowsy drivers can result in fatalities, serious injuries, and property damage. Drivers who are worn out or tired often cause accidents because they pay less attention, respond slowly, and make bad decisions. In order to maintain road safety, there is an increasing demand for trustworthy and efficient driver sleepiness detection technologies. System for detecting driver sleepiness are created to keep track of the driver's state of awareness and issue a warning if indicators of drowsiness are found. These systems often employ sensors to keep a watch on the driver's physiological or bodily signals, such as heart rate, eye movements, and head position.



The use of artificial intelligence (AI) and machine learning (ML) approaches for detecting driver sleepiness has received a lot of attention recently. An image identification technique known as a convolutional neural network (CNN) has demonstrated considerable potential in recognising facial expressions among other types of images. Using CNNs, it is possible to identify indicators of driver tiredness like as drooping eyelids and altered facial expressions by analysing photographs of the driver's face. By exposing the CNN to a sizable collection of labelled photos, including instances of alert and sleepy drivers, these indications may be recognised.

METHODS

Data gathering: The initial phase entails gathering a sizable dataset of labelled pictures that contains illustrations of both alert and sleepy drivers. To make sure that the system is capable of operating in real-world contexts, these photographs should be taken in a variety of lighting and environmental settings.

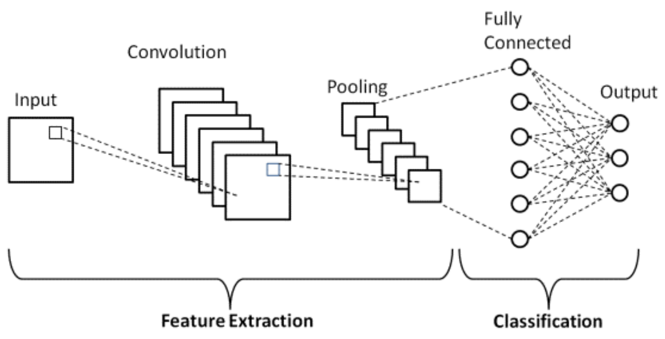
Preprocessing: Following the collection of the dataset, the photos must be cleaned up to get rid of any noise or artefacts. This entails actions like scaling the photographs to a standard resolution, cropping the images to focus on the driver's face, and normalising the pixel values to boost the speed of the network.

CNN architecture selection: The driver drowsiness detection system's next step is to choose a suitable CNN architecture. This entails selecting a network that can be trained using the gathered dataset and has demonstrated strong performance in image recognition tests.

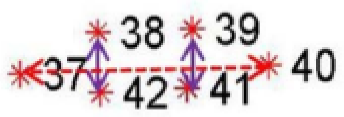
Training: Using a supervised learning strategy, the CNN is trained on the labelled dataset. The network gains the ability to recognise drowsiness-related patterns in the photos, such as drooping eyelids and altered facial expressions, throughout training.

Evaluation: After the network has been trained, its performance is assessed on a different dataset. Measuring measures like recall, accuracy, precision, and F1-score is involved in this.

Implementation: Using the trained CNN, the driver sleepiness detection system is finally put into practise. Usually, the system uses a camera to take pictures of the driver's face, which are then sent to CNN for analysis. Real-time visual analysis by the network alerts the user if indicators of intoxication are found.



Methodology:



The horizontal and vertical distance of the eye is calculated as
EAR= (38-42) +(39-41)/(2*(40-37))

EyeBlink_Detect= (EARLeft_Ey e +EARRight_Eye)/2

```
If(EyeBlink_Detect<0.24){
    eye closed;
}
Else{
    eye opened;
}
```

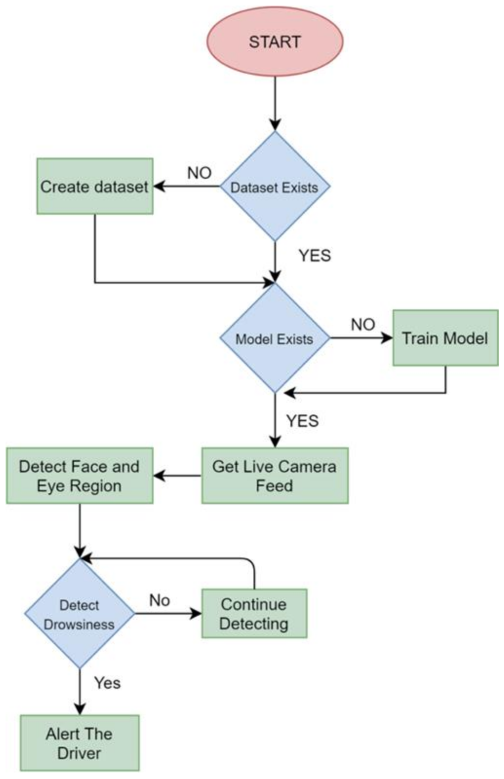
Number of blinks are between 15-30 per min.

For Right Eye:

```
for (x,y,w,h) in right_eye:
r_eye=frame[y:y+h,x:x+w]
count=count+1
r_eye =
cv2.cvtColor(r_eye,cv2.COLOR_BGR2GRAY)
r_eye = cv2.resize(r_eye,(24,24))
r_eye= r_eye/255
r_eye= r_eye.reshape(24,24,-1)
r_eye = np.expand_dims(r_eye,axis=0)
rtrain = model.predict(r_eye)
rpred=np.argmax(rtrain,axis=1)
if(rpred[0]==1):
lbl='Awake'
if(rpred[0]==0):
lbl='Drowsy'
break
```

For Left Eye:

```
for (x,y,w,h) in left_eye:
l_eye=frame[y:y+h,x:x+w]
count=count+1
l_eye =
cv2.cvtColor(l_eye,cv2.COLOR_BGR2GRAY)
l_eye = cv2.resize(l_eye,(24,24))
l_eye= l_eye/255
l_eye=l_eye.reshape(24,24,-1)
l_eye = np.expand_dims(l_eye,axis=0)
ltrain = model.predict(l_eye)
lpred=np.argmax(ltrain,axis=1)
if(lpred[0]==1):
lbl='Awake'
if(lpred[0]==0):
lbl='Drowsy'
break
```



Implementation:

Step 1- Input:

With a webcam, we will take images as input. Therefore, we created an infinite loop that would record every frame in order to access the camera. We employ the OpenCV-provided cv2 technique.To access the camera and set the capture object (cap), use VideoCapture(0). Each frame will be read by cap.read(), and the picture will be saved in a frame variable.

Step 2-Detect face in the image and create ROI:

In order to detect faces in an image, the first step is to convert the image to grayscale. This is because the algorithm used for face detection in OpenCV requires grayscale images as input, and color information is not necessary for object detection. The haar cascade classifier is then applied to detect faces, using the line face = cv2.CascadeClassifier('path to our haar cascade xml file'). The classifier returns an array of detections, including the x and y coordinates, height, and width of each detected face. We can then loop through these detections and draw bounding boxes around each detected face to highlight it. By following these steps, we can successfully detect and highlight faces in an image using OpenCV.

Step 3- Detect the Left and Right Eyes from ROI:

The method for detecting eyes is the same as that for detecting faces. We first put the cascade classifier for the eyes in the leye and reye, respectively, and then use left_eye = leye to find the eyes.detectMultiScale(gray). We now need to isolate the eyeballs' data from the entire picture. This may be done by removing the eye's border box, after which we can use this code to extract the eye's picture from the frame.

Step 4- Categorize the Eye Status:

When using a CNN classifier to predict the status of the eyes in an image, certain operations need to be performed to ensure that the image is in the correct format for the model. First, the color image is converted to grayscale using the OpenCV function cv2.cvtColor. The image is then resized to 24x24 pixels to match the dimensions of the model's input layer. Next, the data is normalized by dividing all pixel values by 255, which improves convergence during training. Finally, the dimensions of the image are expanded to match the input dimensions of the classifier. The trained model is loaded using the function load_model from the Keras library. Once the model is loaded, the left and right eyes in the image are predicted separately using the predict_classes method, which returns a binary classification of either "open" or "closed". A prediction of 1 indicates that the eyes are open, while a prediction of 0 indicates that the eyes are closed. By following these steps, the CNN classifier can accurately predict the status of the eyes in an image, which can be useful for detecting drowsiness in drivers.

- 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

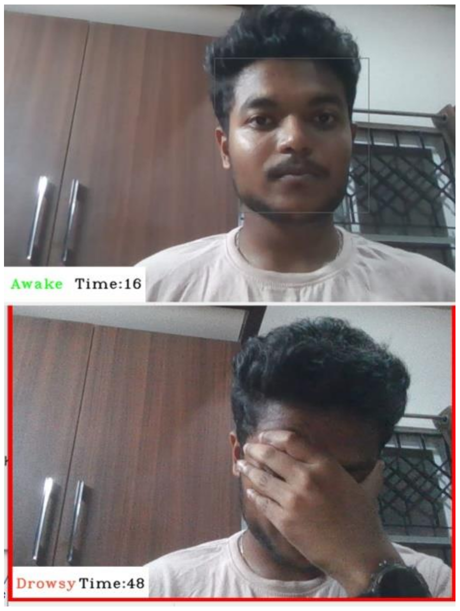
lpred = model.predict_classes(l_eye). If the value of lpred[0] = 1, it states that eyes are open, if value of lpred[0] = 0 then, it states that eyes are closed.

Step 5- Calculate Time/Score:

The score is a value used to measure how long a person's eyes have been closed. When both eyes are detected as closed, the score is incremented, and when they are open, the score is decremented. The current score is displayed in real-time using the cv2.putText() function, which draws the result on the screen. This allows for continuous monitoring of the driver's eye status and can be used to detect drowsiness in real-time.

Step 6- Output:

After Classifying both Eyes using ROI, each label with respect to that eye decides whether the eye is opened or not. If the label is said to be closed for more than 15(score)sec it activates the alarm and creates red box that appears inside the frame and it gradually increases until the eyes are open.



Discussion:

The use of CNN in driver drowsiness detection is a promising technology that can significantly improve road safety by continuously monitoring the driver's eye status and providing timely alerts to prevent accidents caused by drowsiness. However, the effectiveness and reliability of the system depend on several key factors. The CNN model used to detect drowsiness should be accurately trained and tested to classify eye images correctly, even under challenging conditions like poor lighting or when the driver wears glasses. In addition, the system's efficiency is crucial, requiring low latency and fast processing for real-time detection. Furthermore, user-friendliness is a critical aspect of the system design, with an intuitive interface and non-distracting alerts to avoid stress and keep the driver focused on the road.

Future Plans: Future plans for the driver drowsiness detection system include improving its accuracy through more comprehensive training of the CNN model and integrating other data sources such as facial expressions and heart rate. Another area of development is integrating the system with other advanced technologies like vehicle-to-vehicle communication and autonomous driving systems to enhance the prevention of accidents caused by driver drowsiness. Additionally, AI and machine learning algorithms can be used to personalize the system to individual drivers' habits and preferences. Finally, more advanced features such as detection of distraction and aggressive driving can be added to improve road safety and make driving a safer and more enjoyable experience.

Conclusion:

Driver drowsiness detection systems that use CNNs have shown great promise in improving road safety and preventing accidents caused by drowsy driving. By analyzing images of the driver's face in real-time, these systems can detect signs of drowsiness and provide warnings to the driver, preventing accidents before they occur.

The methodology of using CNNs for driver drowsiness detection involves collecting a labeled dataset, preprocessing the images, selecting an appropriate CNN architecture, training the network, evaluating its performance, and implementing the system in real-time. The optimal architecture will depend on factors such as the size of the dataset, the complexity of the task, and the computational resources available.

Overall, driver drowsiness detection using CNNs is a promising application of deep learning that has the potential to save countless lives by preventing accidents caused by drowsy driving. With ongoing research and development, these systems will continue to improve and become an increasingly important tool in road safety.

References:

[1] Ann Williamson and Tim Chamberlain. Review of on-road driver fatigue monitoring devices. NSW
Injury Risk Management Research Centre, University of New South Wales, , July2013.
[2] C. Hentschel, T. P. Wiradarma, and H. Sack, Fine tuning CNNS with scarce training data-adapting
imagenet to art epoch classification, in Proceedings of the IEEE International Conference on Image
Processing (ICIP), Phoenix, AZ, USA, September 2016.
[3] K. He, X. Zhang, S. Ren, and J. Sun, Deep residual learning for image recognition, in Proceedings of
the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, June 2016.
[4] ZuopengZhao,LanZhang,Hualin Yan, Yi Xu, and Zhongxin Zhang, Driver Fatigue Detection Based on
Convolutional Neural Networks Using EM -CNN, 2020.
[5] MiluPrince,NehaSanthosh,NehaSanthosh, Reshma Sudarsan & Ms.AnjusreeV.K,Rajagiri School of
Engineering And Technology - Eye Movement Classification Using CNN.
[6] AnjithGeorge and AurobindaRoutrayDept. of Electrical Engg., IIT Kharagpur Kharagpur, 721302,
India – "Real-time Eye Gaze Direction Classification Using Convolutional NeuralNetwork".
[7] HaoxiangLi ,ZheLin, XiaohuiShen , Jonathan Brandt , Gang HuStevensInstitute of Technology
Hoboken, NJ 07030- A Convolutional Neural Network Cascade for FaceDetection.

[8] Venkata Rami Reddy Chirra,Srinivasulu Reddy Uyyala& Venkata Krishna Kishore Kolli, Department
of Computer Applications, National Institute of Technology, Tiruchirappalli 620015, India – “Deep
CNN: A Machine Learning Approach for Driver Drowsiness Detection Based on Eye State
[9] V.D.Ambeth Kumar et.al., , “Performance Improvement Using an Automation System for
Segmentation of Multiple Parametric Features Based on Human Footprint” for the Journal of
Electrical Engineering & Technology , vol. 10, no. 4, pp.1815-1821 , 2015.
[http://dx.doi.org/10.5370/JEET.2015.10.4.1815]
[10] Ambeth Kumar.V.D et.al. .A Survey on Face Recognition in Video Surveillance. Lecturer Notes on
Computational and Mechanism, Vol. 30, pp: 699-708, 2019
[11] Ambeth Kumar.V.D .Precautionary measures for accidents due to mobile phone using IOT.Clinical
eHealth, Volume 1, Issue 1, March 2018, Pages 30-35.
[12] Ambeth Kumar.V.D et,al. Enhancement in Footprint Image using Diverse Filtering Technique.
Procedia Engineering journal, Volume 8, No.12, 1072-1080, 2012.
[doi:10.1016/j.proeng.2012.01.965]