International Conference on Advances in Materials, Computing and Communication Technologies
In Association with International Journal of Scientific Research in Science and Technology
Volume 9 | Issue 1 | Print ISSN: 2395-6011 | Online ISSN: 2395-602X (www.ijsrst.com)

Detecting Driver Drowsiness Using Deep Learning Techniques

S.Aakash[1], N.Viswateja[1], S.Saravana kumar[1], Mr. S. Sasidharan[*2]
[1]Department of IT, Dr. Mahalingam College of Engineering and Technology, Pollachi, Tamil Nadu, India
[*2]Assistant Professor, Department of IT, Dr. Mahalingam College of Engineering and Technology, Pollachi, Tamil Nadu, India

ABSTRACT

Drowsiness detection is a safety technology that can prevent accidents that are caused by drivers who fell asleep while driving. The objective of this project is to build a drowsiness detection system that will detect that a person's eyes are closed for a few seconds. This system will alert the driver when drowsiness is detected. In this Python project, we will be using OpenCV for gathering the images from webcam and feed them into a Deep Learning model which will classify whether the person's eyes are 'Open' or 'Closed'. The approach we will be using for this Python.

Keywords: OpenCV, TensorFlow, Keras, pygame

I. INTRODUCTION

In this Modern era, Vehicle accidents and crashes are so common and abundance all over the world. For this, one of the main reason is drowsiness in driving. Many international reports shows that drowsy driving is primary reason for one third of accidents and crashes occurring. In order to reduce such accidents and crashes and also for the safety of the driver and passengers, driver drowsiness detection systems have been worked on and developed
A countless number of people drive on the highway day and night. Taxi drivers, bus drivers, truck drivers and people traveling long-distance suffer from lack of sleep. Due to which it becomes very dangerous to drive when feeling sleepy.
The majority of accidents happen due to the drowsiness of the driver. So, to prevent these accidents we will build a system using Python, OpenCV, and Keras which will alert the driver when he feels sleepy. Drowsiness detection is a safety technology that can prevent accidents that are caused by drivers who fell asleep while driving.
The objective of this project is to build a drowsiness detection system that will detect that a person's eyes are closed for a few seconds. This system will alert the driver when drowsiness is detected.

II. LITERATURE SURVEY

In this section, we attempt to briefly review the approaches used previously by researchers for vision-based drowsiness detection, along with their limitations. Eyelid closure is considered to be the most reliable indicator of drowsiness, and hence a lot of the systems developed seem to depend on eyelid closure for driver drowsiness detection. However, the following visual characteristics are also an indicator of driver drowsiness: a longer blink duration, slow eyelid movement, fixed gaze, sluggish facial expression, and drooping posture.

Different algorithms have been presented in the past to detect drowsiness. Some use standard cameras, and some IR and stereo cameras Localize the eyes using edge information, and track them using dynamical template matching to detect driver fatigue. In 6 parameters are measured: Blink frequency, nodding frequency, eye closure duration, percent eye closure (PERCLOS), fixed gaze, and face position. These measured parameters are then combined using a fuzzy classifier to detect Driver's drowsiness

The level of drowsiness is distinguished by the blink frequency, where the frequency increases as the person becomes sleepier and vice versa. In a drowsiness detector is presented based on PERCLOS measurement, which is more robust against strong illumination variations. To reiterate, many of these methods depend on eyelid closure to detect drowsiness of the driver.

All of the above mentioned algorithms to detect drowsiness suffer from typical limitations of traditional image processing techniques, and hence may fail in varying illumination conditions, varying user appearance and fast head movements. Recently, Convolutional Neural Networks have truly revolutionized the field of computer vision, outperforming every other algorithm/technique in many applications such as image classification, object detection, emotion recognition, scene segmentation etc.

Hence, convolutional neural networks have been employed to develop drowsiness detection systems in latest research. It is one of the first attempts to have used convolutional neural networks to address this problem. As stated earlier, convolutional neural networks are able to learn an automated and efficient set of features that provide a better representation of the raw data, and hence help us classify the driver as drowsy or non-drowsy very accurately. However, focused only on increasing the accuracy of drowsiness detection, and in real applications speed of the system is also a major concern.

III. METHODS AND DESCRIBTIONS

In this Python project, we will be using OpenCV for gathering the images from webcam and feed them into a Deep Learning model which will classify

Whether the person's eyes are 'Open' or 'Closed'.

Step 1 : Take image as input from a camera.
Step 2 : Detect the face in the image and create a Region of Interest (ROI).
Step 3 : Detect the eyes from ROI and feed it to the classifier.
Step 4 : Classifier will categorize whether eyes are open or closed.
Step 5 : Calculate score to check whether the person is drowsy.

IV. PROPOSED SYSTEM

The model we used is built with Keras using Convolutional Neural Networks (CNN). A convolutional neural network is a special type of deep neural network which performs extremely well for image classification purposes. A CNN basically 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

The final layer is also a fully connected layer with 2 nodes. In all the layers, a Relu activation function is used except the output layer in which we used Softmax.

Let's now understand how our algorithm works step by step.

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 that will capture each frame. We use the method provided by OpenCV, cv2.VideoCapture(0) to access the camera and set the capture object (cap). cap.read() will read each frame and we store the image in a frame variable.

Step 2 – Detect Face in the Image and Create a Region of Interest (ROI)

To detect the face in the image, we need to first convert the image into grayscale as the OpenCV algorithm for object detection takes gray images in the input. We don't need color information to detect the objects. We will be using cascade classifier to detect faces. This line is used to set our classifier face = cv2.CascadeClassifier ('path to cascade xml file'). Then we perform the detection using faces = face.detectMultiScale(gray). 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.

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 leye and reye respectively then detect the eyes using left_eye = leye.detectMultiScale(gray). Now we need to extract only the eyes 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.
l_eye only contains the image data of the eye. This will be fed into our CNN classifier which will predict if eyes are open or closed. Similarly, we will be extracting the right eye into r_eye

Step 4 – Classifier will categorize whether Eyes are open or closed

We are using CNNvv classifier for predicting the eye status. To feed our image into the model, we need to perform certain operations because the model needs the correct dimensions to start with. 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 Score to check whether Person is Drowsy

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

A threshold is defined for example if score becomes greater than 15 that means the person's eyes are closed for a long period of time. This is when we beep the alarm using sound.play()

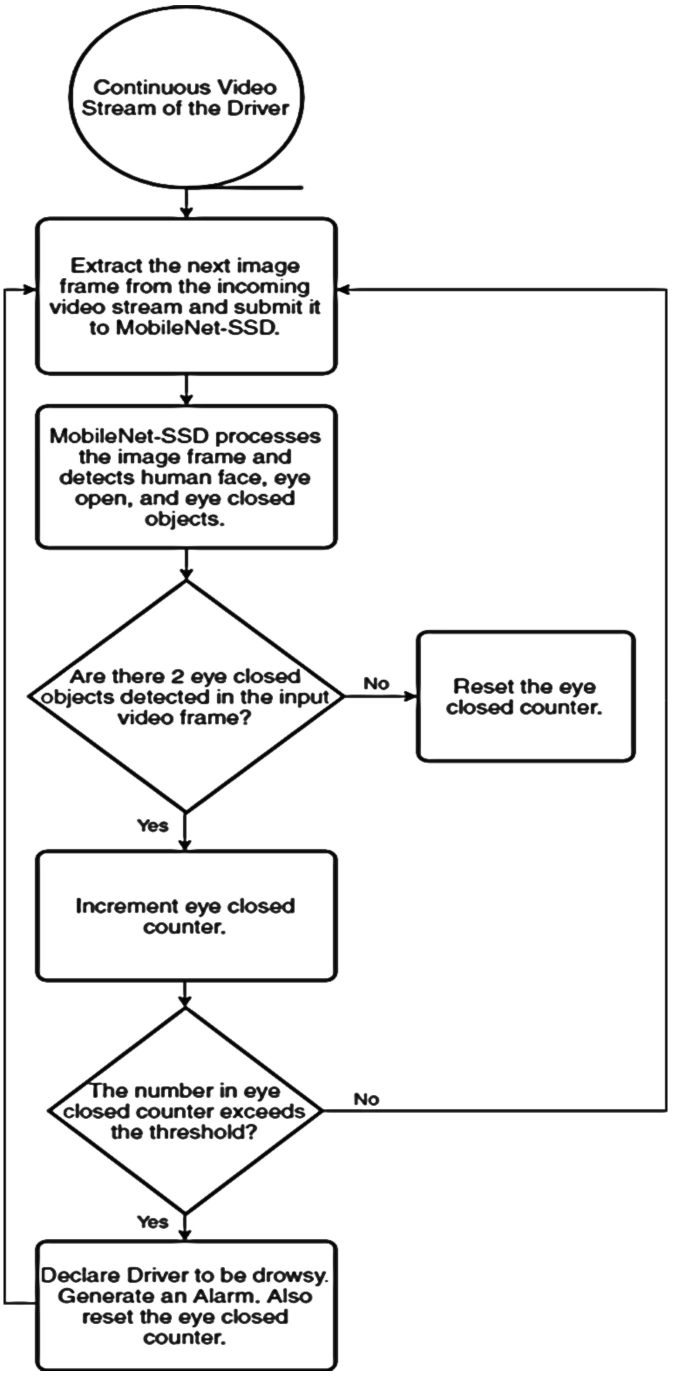
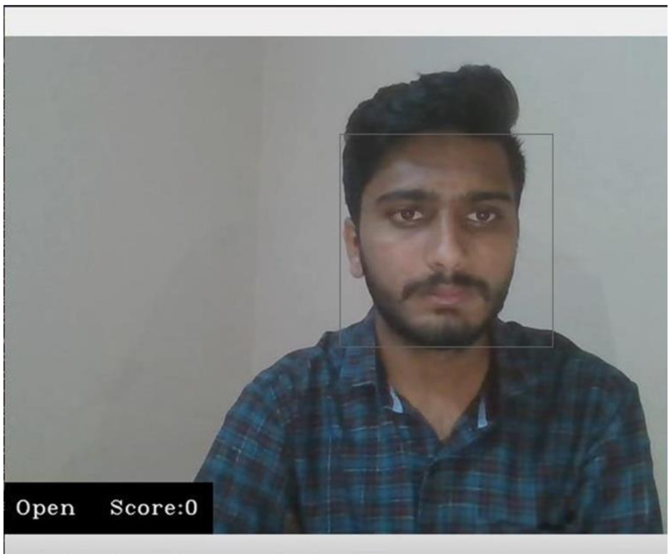
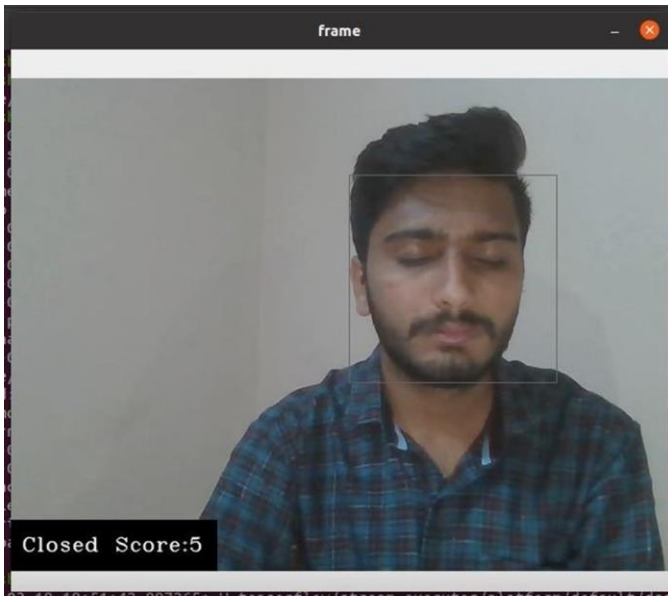
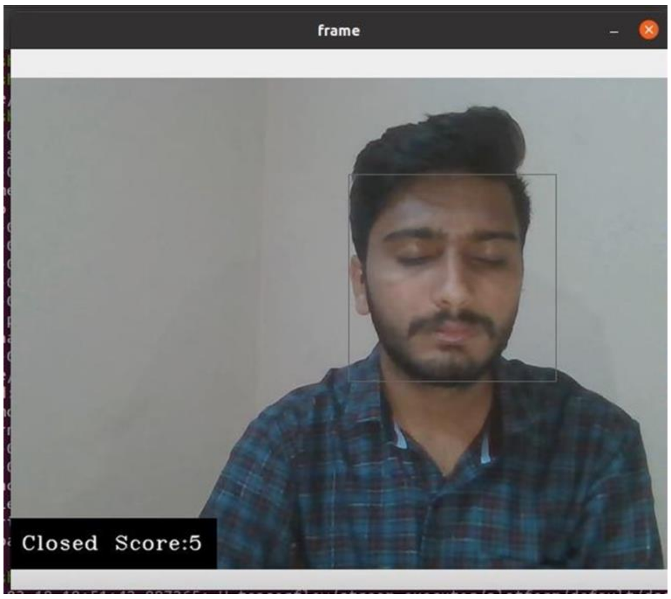

Continuous Video Stream of the Driver
Extract the next image frame from the incoming video stream and submit it to MobileNet-SSD.
MobileNet-SSD processes the image frame and detects human face, eye open, and eye closed objects.
Are there 2 eye closed objects detected in the input video frame?
No
Reset the eye closed counter.
Yes
Increment eye closed counter.
The number in eye closed counter exceeds the threshold?
No
Yes
Declare Driver to be drowsy. Generate an Alarm. Also reset the eye closed counter.

Fig: Drowsiness Detection System Methodology







V. DATASET

The dataset used for this model is created by us. To create the dataset, we wrote a script that captures eyes from a camera and stores in our local disk. We separated them into their respective labels 'Open' or 'Closed'. The data was manually cleaned by removing the unwanted images which were not necessary for building the model. The data comprises around 7000 images of people's eyes under different lighting conditions. After training the model on our dataset, we have attached the final weights and model architecture file "models/cnnCat2.h5".Now, you can use this model to classify if a person's eye is open or closed.

VI. CONCLUSION

Based on the results on the test dataset and real-world testing in a variety of illumination and occlusion conditions, we conclude that the proposed methodology of treating the task of drowsiness detection as an object detection task is practical and reliable. Moreover, this technique works in real time on an Android device, which makes it very accessible. This would be beneficial as drowsiness related accidents have a high chance of occurrence during night-time driving.

VII. REFERENCES

[1]. Mohn, T.: Around 5,000 People Were Killed Last Year Due to Drowsy Driving, https://www.forbes.com/sites/tanyamohn/2016/08/08/nearly-83-6-millionamerican-drivers-are-sleep-deprived-new- report-highlights-dangers-highcost/#3bbc82664007.

[2]. Rapaport, L.: Drowsy drivers often behind fatal crashes, https://www.reuters.com/article/us-health-driving-sleep/drowsy-drivers-oftenbehind-fatal-crashes-idUSKBN15P2PM.

[3]. Facts and Stats : Drowsy Driving – Stay Alert, Arrive Alive, http://drowsydriving.org/about/facts-and-stats/.

[4]. Saini, V., Saini, R.: Driver Drowsiness Detection System and Techniques : A Review. Int. J. Comput. Sci. Inf. Technol. 5, 4245–4249 (2014).

[5]. Bhatt GHPatel PG, P.P., Trivedi GHPatel PG, J.A.: Various Methods for Driver Drowsiness Detection : An Overview. Int. J. Comput. Sci. Eng. 9, 70–74 (2017).

[6]. Using, S., Eog, E.: Development of Vehicle Driver Drowsiness Detection System Using Electrooculogram (EOG). In: 1st International Conference on Computers, Communications, and Signal Processing With Special Track on Biomedical Engineering (CCSP), Kuala Lumpur, Malaysia. pp. 165–168 (2005).

[7]. Takei, Y., Furukawa, Y.: Estimate of driver's fatigue through steering motion. 2005 IEEE Int. Conf. Syst. Man Cybern. 2, 1–6 (2005).

[8]. Lee, A.: Comparing Deep Neural Networks and Traditional Vision Algorithms in Mobile Robotics. Swart. Coll. (2015).

[9]. Lowe, D.G.: Distinctive Image Features from Scale-Invariant Keypoints. Int. J. Comput. Vis. (2004).

[10]. Lazebnik, S., Schmid, C., Ponce, J.: Beyond bags of features: spatial pyramid matching for recognizing natural scene categories. IEEE Comput. So-ciety.

[11]. Dalal, N., Triggs, B.: Histograms of Oriented Gradients for Human Detection. In: Conference on Computer Vision and Pattern Recognition (2005).

[12]. Bittner, R., Hána, K., Poušek, L., 6puþnd, 3dyho, Schreib, P., Vysoký, P.: Detecting of Fatigue States of a Car Driver. ISMDA. 260–274 (2000).

[13]. Bergasa, L.M., Nuevo, J.: Real-time system for monitoring driver vigilance. IEEE Int. Symp. Ind. Electron. III, 1303– 1308 (2005).

[14]. Suzuki, M., Yamamoto, N., Yamamoto, O., Nakano, T., Yamamoto, S.: Measurement of

driver's consciousness by image processing - A method for presuming driver's drowsiness by eye-blinks coping with individual differences. Conf. Proc. - IEEE Int. Conf. Syst. Man Cybern. 4, 2891–2896 (2007).

[15]. Technology | Smart Eye, http://smarteye.se/technology/.

[16]. Fan, C.: Driver Fatigue Detection Based. Proc. 2004 IEEE, Int. Conf. Networking, Sensiing Control. 7–12 (2004).

[17]. Abtahi, S., Hariri, B., Shirmohammadi, S.: Driver Drowsiness Monitoring Based on Yawning Detection. In: IEEE International Instrumentation and Measurement Technology Conference, Binjiang, Hangzhou, China (2011).

[18]. Danisman, T., Bilasco, I.M., Djeraba, C., Ihaddadene, N.: Drowsy driver detection system using eye blink patterns. 2010 Int. Conf. Mach. Web Intell. ICMWI 2010 - Proc. 230–233 (2010).

[19]. Bronte, S., Bergasa, L.M., Almaz, J., Yebes, J.: Vision-based drowsiness detector for Real Driving Conditions. (2012).

[20]. Krizhevsky, A., Sutskever, I., Hinton, G.E.: ImageNet Classification with Deep Convolutional Neural Networks. In: Neural Information Processing Systems (2012).

[21]. He, K., Zhang, X., Ren, S., Sun, J.: Deep Residual Learning for Image Recognition. In: Conference on Computer Vision and Pattern Recognition (2016).

[22]. Ren, S., He, K., Girshick, R., Sun, J.: Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. In: Neural Information Processing Systems. pp. 1–14 (2015).

[23]. Shelhamer, E., Long, J., Darrell, T.: Fully Convolutional Networks for Semantic Segmentation. IEEE Trans. Pattern Anal. Mach. Intell. 39, 640–651 (2017).

[24]. Dwivedi, K., Biswaranjan, K., Sethi, A.: Drowsy driver detection using representation learning. Souvenir 2014 IEEE Int. Adv. Comput. Conf. IACC 2014. 995–999 (2014).

[25]. Bhargava Reddy, Ye-Hoon Kim, Sojung Yun, Chanwon Seo, J.J.: Real-time Driver Drowsiness Detection for Embedded System Using Model Compression of Deep Neural Networks. In: Conference on Computer Vision and Pattern. Recognition Workshops (CVPRW). pp. 438–445 (2017).