Acta Scientific COMPUTER SCIENCES

Volume 4 Issue 3 March 2022

Research Article

Driver Drowsiness Detection

Hemang Thakur[1], Deval Arora[1], Ramkiran Sampathi[1], Shree Rukmini Thumu[1] and Parisa Naraei[2]*

[1]*Department of Artificial Intelligence and Machine Learning Lambton College Toronto, Canada*

[2]*Cestar College of Business, Health and Technology Research and Innovation Center, Lambton College Toronto, Canada*

***Corresponding Author:** Parisa Naraei, Cestar College of Business, Health and Technology Research and Innovation Center, Lambton College Toronto, Canada.

Received: November 27, 2021

Published: February 24, 2022

Abstract

CNN models are widely used to implement business solutions in the computer vision domain. Inthis paper, we have built a CNN based face detection model along with a facial feature detectionusing dlib library as accurate and light weight available libraries. This allows these models torun efficiently even on low-power devices. As a proof of concept, the paper presents a desktop application of the effective solution in the real-world scenario. In this approach, the images are captured at an interval of 0.1 seconds as the models take roughly 50 to 80 milliseconds to predictthe output for a single image. As a result, the model is able to detect the driver drowsiness withan accuracy of 92.5% taking into account the state of mouth as well as eyes.

Keywords: Face Detection; Drowsy Driving; Convolution Neural Networks; Real Time ImageAnalysis

Introduction

Every year, a number of people die in fatal road accidents around the world. Fatigue and micro sleeping at the wheel are often a cause of serious accidents, and being drowsy while driving is one of the major reasons for this. As per United States Department of Transportation, in 2017 an estimated 91,000 police-reported crashes involved drowsy drivers. These crashes led to an estimated 50,000 people injured and nearly 800 deaths [22]. It is therefore important to have a system that can timely detect drowsiness while driving. Such a system in place can save livesof many, if not all.

Using the machine learning (ML) concepts and principles, an application with a ML model at the backend and a user-friendly interface can be developed to address the problem of drowsy driving. So, the objective of this paper is to introduce a user-friendly model trained to detect the drowsiness in drivers. For this purpose, a thorough research was done on designing user interfaces and this included interface development and testing. User interface acts as a border between the computer (application) and the user [3]. The sole aim of user interfacedesigning is to ensure that with the final version of the UI, any user is able to identify what happens with natural progression for each task. The interface has to be simple enough to avoid ambiguity and complexity of the tasks a user has to follow. On top of that, identifying and understanding the components of GUI and rules that must be applied for effective interface design methodology are remarkable [1].

The model has been trained using various libraries in Python such as keras and numpy. After taking into account the advantages and disadvantages of available UI libraries, Tkinter was chosen. The reason for choosing Tkinter was that it is the only GUI application built into the Python standard library which is cross-platform, meaning the same code runs on Windows, Mac OS X, and Linux

[2,3]. Also, it is open source and has good documentation available [4]. Another benefit of using Tkinter is that it can be easily integrated with OpenCV, a python library that is used to capture images.

While there are many python libraries that can capture images, for this project real-time image processing is important, and for that it is essential that images are captured in real-timeand as efficiently as possible. OpenCV is one such python library that provides algorithmic efficiency mainly to process real-time programs. Moreover, it has been designed in a way that allows it to take advantage of hardware acceleration and multi-core systems to deploy [5].

Machine learning model at the backend is the backbone for the application. Internally, themodel is divided into two sub-models: CNN and shape predictor. CNN is used to detect the facesin the image and define the facial boundary in the form of coordinates which is referred to as the bounding box. Shape predictor is to identify and track the movements of facial features like mouth and eyes in real time [9]. For detecting these features [6], The dlib library is used for face detection and can train custom shape predictors as well as provide pretrained models. Facial features play a vital role in determining whether the person is drowsy or not [8].

Mouth Aspect Ratio (MAR) and Eye Aspect Ratio (EAR) [7] are the mathematical formulas used to detect the drowsiness or fatigue of a person by setting the threshold values. The eye area's aspect ratio is referred to as EAR, and it is widely used to measure the temporal accuracy and speed of left and right eye blinks to detect fatigue [10]. MAR refers to the aspect ratio of the mouth region and is generally used to calculate the distance between the upper and lower lips, which can effectively be used for yawn detection [7]. Once the threshold values of EAR and MAR exceed, the person is considered as sleepy. Based on the result of the model the UI will sound an alarm to make the driver conscious [10].

In their work Masoumeh Tashakori, et al [13] enhanced the accuracy of the driver detection model by 82% and precision by 84%. The authors proposed using the temperature gradient as a sensitive indicator of drowsiness. It is a non-invasive and robust approach against changes in the ambient light. A similar approach was adopted by Mahmoodi, M., and Nahvi, A. (2019) [17] and Rahman N.A.A., Mustafa M., Sulaiman N., Samad R., Abdullah N.R.H. (2022) [21] collecting the data from surface electromyography and conducting k-nearest neighbor classifiers to predict drowsiness by 90% accuracy, 82% precision, 77% sensitivity, and 92% specificity. In another similar work by Houshmand, S., Kazemi, R., and Salmanzadeh, H. (2021) [18] electroencephalography (EEG) signals are used along with the Convolutional Neural Network (CNN). CNN model classifies EEG signals and automatically learns features of the early drowsy state, and achieve an accuracy of 94%.

A different category of work uses driver images and not on the data obtained from sensors attached to the driver's seat or elsewhere. Emashharawi M.J.S., Khalifa O.O., Abdul Malik N., Abdul Malek N.F. (2022) [19] and Bhatia U., Tshering, Kumar J., Choubey D.K. (2022) [20] in their respective work, used computer vision to detect driver drowsiness, In the paper, Athira Babu [14] considered head pose along with face detection to detect drowsiness, and this got an accuracy of 94.51%. Similar to this, Ajinkya Rajkar [15] in their paper used a deep learning model with an accuracy of 96% to detect the drowsiness of the driver. The paper focuseson the eye region and mouth region in order to get enhanced accuracy. In their work Mohit Dua [16] obtained the final result with an accuracy of 85% but this just gives us positive or negative answers using the softmax classifier.

Dataset

For this project the dataset that we used consists of images which cover all angles of face and different regions of the world, and this helps in predicting in a better way. Our proposed project not only considers eye landmarks but also takes care of mouth landmarks which makes this project distinguishable. The approach taken ensures that the model is lightweight and this enables the system to run faster without lagging on less powerful devices as well.

For the training of the Machine Learning models for this project, we have used the ibug 300-W dataset [23]. The total size of the dataset is 1.72 GB and the total number of files in the dataset are 3,105. Out of this 2,433 were used for training the model and remaining 672 were used for testing purposes.

This dataset provides enough images to train the models as well as covering the facial features from various angles and perspectives which allows the models to detect facial features regardless of the angle the person is facing the driver seat. Besides, the images are of goodclarity and resolution.

Proposed architecture

In this section, the details of the proposed framework for the driver drowsiness detection system are discussed. The methods are subjected to testing on live video sequences recorded with a laptop webcam. Before deciding the driver's vigilance level, the proposed system goes through a few measures. The face is first extracted from video frames using the CNN model. Thelocation of the eyes and mouth is then identified through a shape predictor using Dlib. Finally,we use Mouth Aspect Ratio (MAR) and Eye Aspect Ratio (EAR) for detecting fatigue and drowsiness.

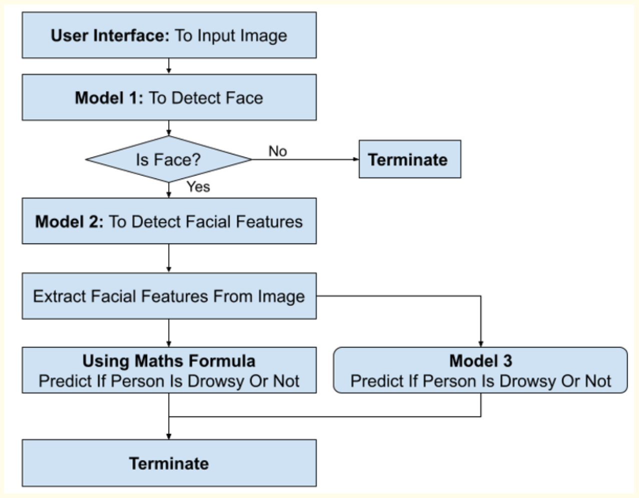
User Interface: To Input Image
Model 1: To Detect Face
Is Face?
No
Terminate
Yes
Model 2: To Detect Facial Features
Extract Facial Features From Image
Using Maths Formula Predict If Person Is Drowsy Or Not
Model 3 Predict If Person Is Drowsy Or Not
Terminate

Figure 1: Model Flowchart.

Modeling

In this section, we describe the Machine Learning models used in the implementation of this project. We have used 2 separate models to develop this project, namely the face detection model and the facial feature detection model.

For the face detection model, we have used the pre-trained frontal face detector from the Dlib library. Contrary to the name, this model is able to detect the faces not just from the front but also from various other angles effectively. Besides, this model is also lightweight which enables the system to run faster without lagging on less powerful devices as well.

For the facial feature detection model, we have used the semi pre-trained model from the Dlib library's shape predictor function. To tune the hyperparameters of the model i.e. to find the set of hyperparameters that will give us the best possible accuracy and stability, we used a brute force approach which entails manually looping over a collection of hyperparameter sets and recording the accuracy from each trial. Since this model wasn't compatible with the Grid search and Random search methods from the scikit-learn library, we had to design the entire method from scratch. It took roughly 3 - 4 hours to run the model on each set of the hyperparameters andabout 3 days to complete the entire task of hyperparameter tuning. The results were manually noted down each time. The final set of hyperparameters that we obtained is mentioned in the following paragraph.

To train this model, we have used the training subset of the data which contains images ofthe faces from various angles as well as a file describing the positions where the facial features are located. We can choose what facial features we want the model to detect and in our project we have chosen to detect all the possible facial features i.e. eyes, nose, eyebrows and mouth.

To train the shape predictor model, we have used the following hyperparameters and values which we obtained from the hyperparameter tuning step,

- Tree depth = 3
- Nu = 0.1
- Cascade depth = 10
- Feature pool size = 150
- Num test splits = 350
- Oversampling amount = 5
- Oversampling translation jitter = 0.

Here, "tree depth" is the depth of the tree used in each cascade, "nu" is the regularization parameter, "cascade depth" is the number of cascades used to train the model, "feature pool size"is the number of pixels used to generate the features for the random trees at each cascade, "num test splits" is the number of split features sampled at each node, "oversampling amount" is used to apply some deformations applied to the training samples and "oversampling translation jitter" is used to apply some translation deformations to the given bounding boxes.

The model was tested on a testing subset of the data. The training and testing accuracy that we achieved was 93.2% and 92.5% respectively.

EAR and MAR calculations

As mentioned earlier, to determine if the driver is feeling sleepy or not, we have used specific mathematical formulas, specifically Eye Aspect Ratio (EAR) and Mouth Aspect Ratio (MAR).

The eye area's aspect ratio is referred to as EAR, and it is widely used to measure the temporal accuracy and speed of left and right eye blinks to detect fatigue. Below is the mathematical formula used to calculate the Eye Aspect Ratio (EAR).

$$EAR = \frac{||p2 - p6|| + ||p3 - p5||}{2||p1 - p4||}$$

Where p_1, p_2, p_3, p_4, p_5 and p_6 are the 6 landmark points representing the eye.

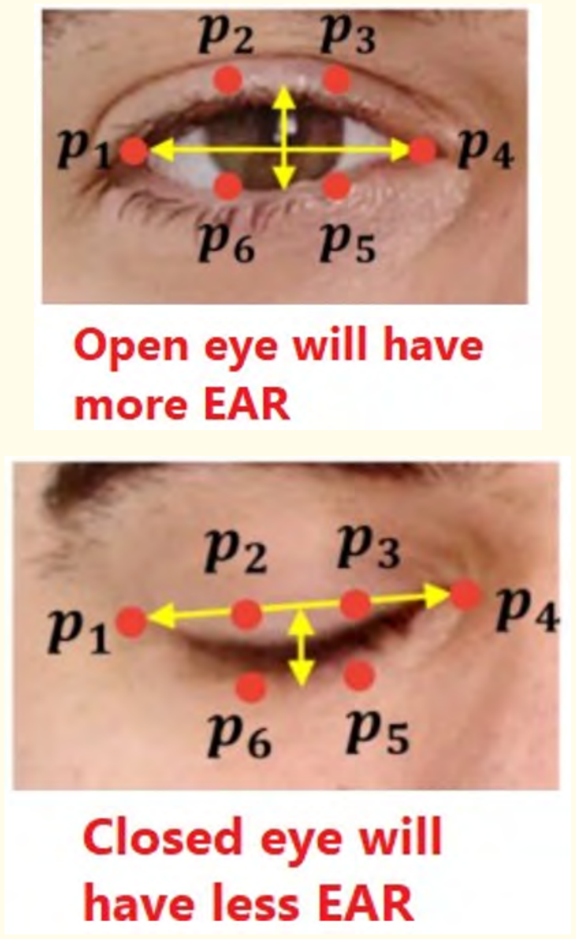

Figure 2: EAR Calculation For Open Eye(L) and Close Eye(R).

Similarly, MAR refers to the aspect ratio of the mouth region and is generally used to calculate the distance between the upper and lower lips, which can effectively be used for yawn detection. Below is the mathematical formula used to calculate the Mouth Aspect Ratio (MAR).

$$MAR = \frac{||p2 - p8|| + ||p3 - p7|| + ||p4 - p6||}{2||p1 - p5||}$$

Where p_1, p_2, p_3, p_4, p_5, p_6, p_7 and p_8 are the 8 landmark points representing the eye.

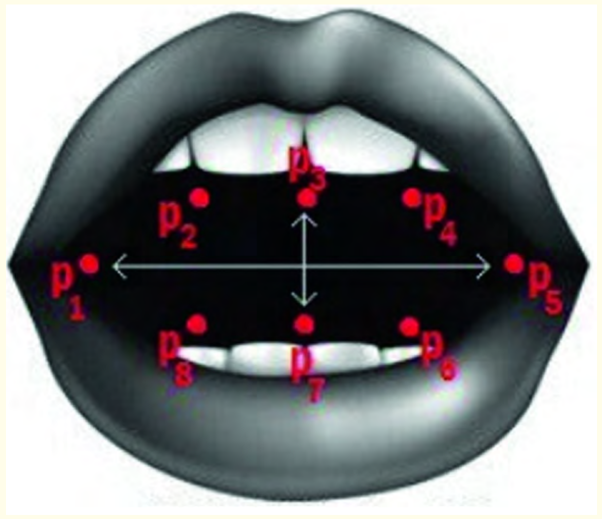

Figure 3: MAR Calculation for Mouth.

Results

In this section, we will discuss the experimental results and the user interface that we built as a proof of concept. The primary function of the user interface is to capture images in real time. After capturing the image, the UI will pass that image to the model for analysis. Based on the result of the model the UI will sound an alarm as appropriate. Functionalities included in the User Interface are Start-Detector, Stop alarm, and Stop-Detector.

In the images below we can see that the model has been integrated with the userinterface. The application is able to detect a single face as well as multiple faces. The user interface has functionalities such as starting the application, stopping the application and snoozing the alarm as and when required.

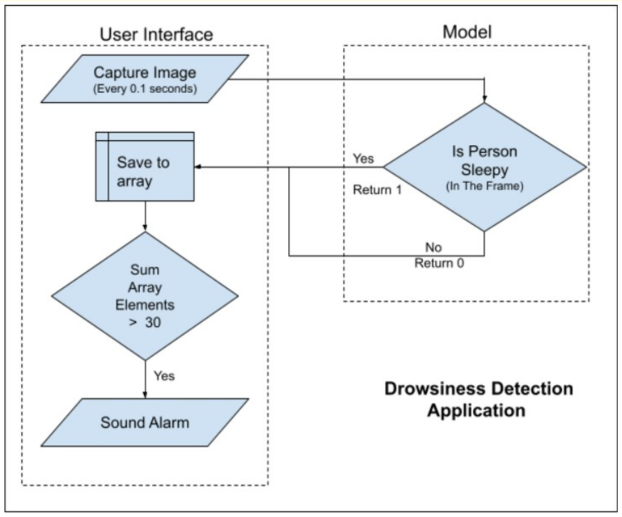
User Interface
Model
Capture Image (Every 0.1 seconds)
Save to array
Is Person Sleepy (In The Frame)
Yes
Return 1
No
Return 0
Sum Array Elements > 30
Yes
Sound Alarm
Drowsiness Detection Application

Figure 4: User Interface and Model Integration.

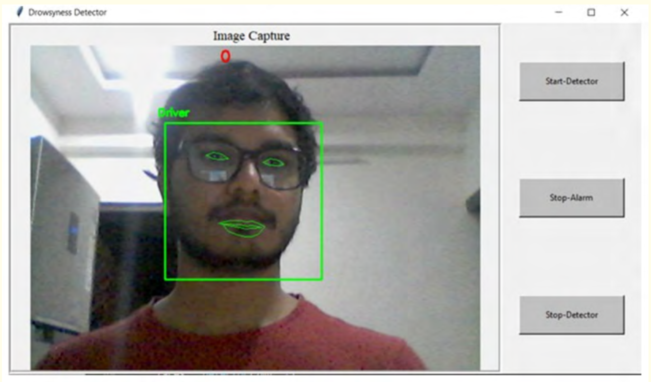

Figure 5: Application detecting Single face.

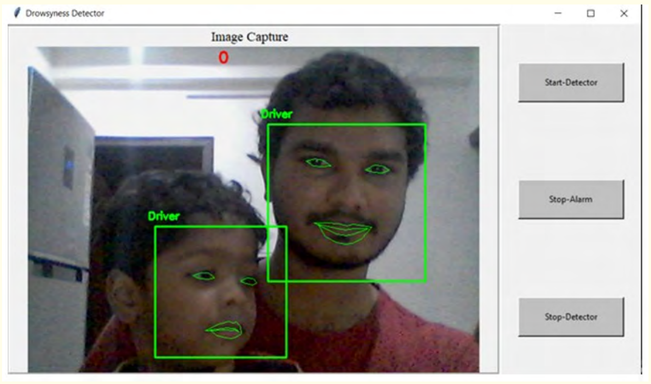

Figure 6: Application detecting Multiple faces.

The integrated backend was tested on a testing subset of the data. The training and testingaccuracy that we achieved was 93.2% and 92.5% respectively. In addition to this, the application was tested with live feed and it correctly predicted if the person was drowsy or not.

Conclusion

Our model can detect multiple faces. It is able to detect facial features such as eyes and mouth, which are useful to determine whether the person is drowsy or not. Previous attempts to solve the problem of drowsy driving require the use of costly sensors and from an economic perspective, that may not be viable. The model is lightweight and it has a good accuracy, and thisenables us to use this model as a mobile application which would be fast and won't lag, meaningthat there is no need for costly sensors or any other device.

Bibliography

1. Wilbert O Galitz. "The Essential Guide To User Interface Design" (2007).
2. David Amos. "Python GUI Programming with Tkinter" (2012).
3. Primoz Podrzaj. "A brief demonstration of some Python GUI libraries" (2019).
4. Guilherme Polo. The Python Papers 2.1.
5. I Culjak., *et al*. "A brief introduction to OpenCV". 2012 Proceedings of the 35th International Convention MIPRO, Opatija, Croatia (2012): 1725-1730.
6. Grant Zhong. "Drowsiness Detection with Machine Learning" (2019).
7. M Ngxande., *et al*. "Driver drowsiness detection using behavioral measures and machine learning techniques: A review of state-of-art techniques". 2017 Pattern Recognition Association of South Africa and Robotics and Mechatronics (PRASA-RobMech), Bloemfontein, South Africa (2017): 156-161.
8. Phakawat Pattarapongsin. "Real-time Drowsiness and Distraction Detection using Computer Vision and Deep Learning" (2020).
9. H Garg. "Drowsiness Detection of a Driver using Conventional Computer Vision Application". 2020 International Conference on Power Electronics and IoT Applications in Renewable Energy and its Control (PARC) (2020): 50-53.

10. Gauri Thakre. “Drowsy Driver Detection and Alert System” (2018).

11. C Sagonas., *et al*. “300 faces In-the-wild challenge: Database and results”. Image and Vision Computing (IMAVIS) (2016).

12. C Sagonas., *et al*. “300 Faces in-the-Wild Challenge: The first facial landmark localization”. Challenge (2013).

13. Tashakori M., *et al*. “Driver drowsiness detection using facial thermal imaging in a driving simulator”. *Proceedings of the Institution of Mechanical Engineers, Part H: Journal of Engineering in Medicine* 236.1 (2022): 43-55.

14. Babu A., *et al*. “Driver’s Drowsiness Detection System Using Dlib HOG”. In: Karuppusamy P., Perikos I., García Márquez F.P. (eds) Ubiquitous Intelligent Systems. Smart Innovation, Systems and Technologies, 243 (2022).

15. Ajinkya Rajkar., *et al*. “Driver Drowsiness Detection Using Deep Learning” (2021).

16. Mohit Dua., *et al*. “Deep CNN models-based ensemble approach to driver drowsiness detection”. *Neural Computing and Applications* 33 (2021): 3155-3168.

17. Mahmoodi M and Nahvi A. “Driver drowsiness detection based on classification of surface electromyography features in a driving simulator”. *Proceedings of the Institution of Mechanical Engineers, Part H: Journal of Engineering in Medicine* 233.4 (2019): 395-406.

18. Houshmand S., *et al*. “A novel convolutional neural network method for subject-independent driver drowsiness detection based on single-channel data and EEG alpha spindles”. *Proceedings of the Institution of Mechanical Engineers, Part H: Journal of Engineering in Medicine* 235.9 (2021): 1069-1078.

19. Emashharawi MJS., *et al*. “Computer Vision Based Driver Assistance Drowsiness Detection”. In: Isa K. *et al*. (eds) Proceedings of the 12th National Technical Seminar on Unmanned System Technology 2020. Lecture Notes in Electrical Engineering 770 (2022).

20. Bhatia U., *et al*. “Drowsiness Image Detection Using Computer Vision”. In: Sharma T.K., Ahn C.W., Verma O.P., Panigrahi B.K. (eds) Soft Computing: Theories and Applications. Advances in Intelligent Systems and Computing 1380 (2022).

21. Rahman NAA., *et al*. “EMG Signal Segmentation to Predict Driver’s Vigilance State”. In: Hassan M.H.A. *et al*. (eds) Human-Centered Technology for a Better Tomorrow. “Lecture Notes in Mechanical Engineering”. Springer, Singapore (2022).

22. NHTSA.

23. https://ibug.doc.ic.ac.uk/download/annotations/ibug.zip/