IJTRET "Voice of Innovative Minds"

International Journal of Trendy Research in Engineering and Technology
Volume 4 Issue 7 December 2020
ISSN NO 2582-0958

DETECTION OF DROWSINESS AND DISTRACTION OF DRIVERS USING CNN

Gaurav Kanojia, Vijay Kanojiya, Glenn Mendonza

Final Year UG Students, Department of Computer Engineering,
Xavier Institute of Engineering, Mumbai University, Mumbai, India

Abstract— **There are several car accidents taking place on a regular basis where the driver including the passengers are left dead or severely injured. Taking into account the ever increasing population, in India alone there occurs about a million auto-mobile accidents according to analysis and this number only keeps increasing with time. According to various studies conducted, almost 50% of these accidents are caused due to drivers being highly distracted or tired and drowsy. With this project we aim to create a working piece of technology that could help drivers prevent getting distracted or drowsy on the steering wheel and act in time to prevent mishaps by alerting them in time. For this we will make use of machine learning and use Keras package to build a CNN model that will be able to classify if the driver is drowsy or distracted from the input images or video and ring an alert tone or bell if detected correctly giving the driver an early warning and some time to make corrections.**

Keywords — **:Machine Learning, CNN (Convolutional Neural Network), Open CV Haar Cascade Classifier, Classification model.**

I. Introduction

Accidents occur due to various reasons namely: Driver being ill, intoxicated driver, intoxicated passengers, lack of necessary breaks or rest, driver with mental trauma or an unaware driver, etc. After properly analysing the above causes, we can boil down the reason for driver accidents to two general things: 1. A drowsy or fatigued driver 2. A distracted driver. Drowsiness can cause several negative effects on a driver such as Loss of hand-eye coordination, miss judgement of signals or turns and delay in reaction time. All these attributesare very important for safe driving but loss of even one of these attributes can cause major problems and may even lead to an accident. Similarly, Distraction could completely cause the driver to lose sight of the road or lose focus of the controls which again could lead to catastrophic failure. The problem here is that the driver by law and driving ethics, while drowsy or distracted, is meant to halt and rest or get rid of the cause of distraction but most of the drivers ignore this cardinal rule and end up in a disaster.

A.Aims and Objectives

As mentioned above, drivers are prone to ignore the call to halt and take a rest which leads to problems and there is no other means to correct this issue. Hence, the aim of this project is to create a system which alerts the driver in such situations. The alert tone is rung when the driver is detected to be drowsy or distracted. A drowsy driver can become alert by the tone and take necessary actions in time similarly a distracted driver can regain his focus on the road and controls in time. The classification model will be built using CNN which will detect the drivers face and from it detect the eyelids. Depending upon two conditions the output alarm will ring. If the eyelid is closed continuously for more than 7-9 seconds it will be considered as drowsy driver and the alarm will ring. If the eyelids or entire face is not detected for more than certain seconds it will be considered as distracted driver and the alarm will ring. Once the alarm is ringing the driver would need to tap a button on the screen to turn the alarm off. For face and eye-lid detection, OpenCV will be used.

B. Scope of the Project

This project would make use of the android app for GUI purpose and to start and end a session. With this the driver only needs to attach the smartphone on the dashboard in front of him where his face would be clearly detected by the phones camera and start the app. The backend of the app will contain a trained neural network able to classify from the input of the driver images or videos if the driver is drowsy/distracted or not. Considering the android application approach of this project, the final product could not only be easily implemented in cars but with minor modifications could also be implemented into trucks, trains, motorcycles, etc. As several taxi services such as Uber, Ola, etc. already have regulations for the drivers to maintain an on board smart phone for maps, we could make use of this convenience. The driver would only require to download our app and give permission to the phone camera and set the phone on the dashboard in such a way that he would be visible in the camera frame. Such organizations who try to guarantee the safety of passengers could also mandate use of such an application for the drivers and could also monitor the drivers driving ethics. If the application detects drowsy/distracted driver for a particular driver more than a certain number of times, the organization in charge could be automatically notified.

C. Existing System

Similar applications to driver drowsiness detection do exist but in only luxury cars with on board computing prowess.There are cars that are becoming automated and in situations where the driver is absent minded and takes his hand off the driving wheel, the car takes over the controls and steers the car back to safety. There are also cars with integrated sensors that monitor the driver's movements and behaviours and in conditions of suspicious behaviour, takes over the controls or stops the car even before an accident occurs.These high end technologies definitely exist but the problem is that it would require the driver to do expensive and technical setup in order to make use of such a system.Such automobiles are very rare and expensive and are hence available to a few. Due to this reason, this safety measure has not been accessible to the common man. Hence, in order to make it accessible to everyone, we plan to make this project into an android application.

II. Related Work

As the aim was to use ML to achieve the classification, several ML algorithms and related papers were researched for our classification model. The first approach that we landed on was by using SVM (Support Vector Machines). In our case SVM could be used to classify open eye or closed eye by giving it identifier objects and training the model. SVM works accurately in classifying such labelled data. But after researching few papers on SVM, few limitations surfaced. Once an SVM model was trained using a particular dataset, that model could later recognize and correctly predict/classify eyes of only those individuals whose pictures were present in the dataset. It would largely fluctuate upon an anonymous user. Another limitation that was discovered during SVM model training was that the accuracy and learning rate of the model drastically reduced after increasing the dataset size. As our dataset consists of 10,000 images of open and closed eyes and each image would be tilted, zoomed and mirrored to form 3 different images, this dataset proved too large for SVM model to handle. Hence, we could not use SVM for our classification model. Random Forest and Logistic regression algorithms were also considered but again they proved inappropriate for our model as when raw data was given as input to these models, there was significant loss in accuracy. Another limitation which was also seen in SVM, was recurring in these algorithms as well. It failed to correctly classify images or data that were not included already in the dataset. This is a very important factor for our model as the trained model would need to correctly classify images or raw data of any user. Therefore, CNN was found to be a perfect fit for our classification model. It could handle raw data well as it would do the feature extraction on its own and convert them into weights and save those weights. Once trained, it would use these weights/ features to find similarities in raw data and classify any user based on this. This was an important deciding factor. Also CNN can handle large datasets well and its accuracy was found to increase with increase in dataset. CNN is also easy to code and debug in case of errors. Training CNN model on a large dataset would definitely consume a lot of time but once trained, the model works very quickly and precisely to classify and can also be used in real time. This was also very important for our final goals.

During research, we also had to look into the human behaviour while and before being drowsy. A person when tired feels drowsy but does not immediately fall asleep. He ends up dozing a few times before a proper nap. How do we program this into the classification model? How do we program the difference between a doze and a nap? After some research, we came to know that an eye blink lasts for about a tenth of a second, if we detect the human eye closed in 2 or 3 consecutive frames we can classify that as a doze. Whenever the user dozes off, a counter increments and if this counter crosses value 3 we can safely say that the person is drowsy and could soon fall asleep and can ring the alert tone. In this way we can alert the driver even before he nods off.

III. Description

A. Analysis

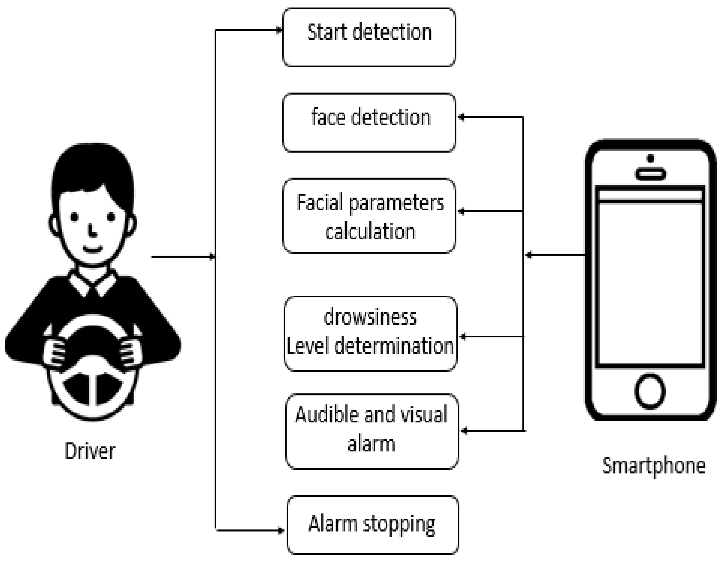


Fig. 1 Basic Architecture of the Project

This is the basic structure that we plan to achieve where the user or driver only interacts with the application by starting the app and ends the session by closing the app and can shut the alert tune in the case it rings. In this way the only thing visible to the user is the Application interface. The back end, which is the classification model, runs independently without the knowledge of the user. It performs all the tasks such as face detection and from it the eye region detection and gives it as input to the classification model and gets output. In this way we have achieved Abstraction. The advantages of this system is that the user needs no training or knowledge to use the App.

B. *Feasibility Study*

1) Economic Feasibility: The cost of the project depends upon the training of the model and system requirements for that purpose such as CPU RAM, GPU and disk space. A computer with about 8 GB ram and basic GPU card was used in unison with shared CPU and GPU processing offered by Google Colab. The disk space offered by the same was also used. The same system was used for the GUI and Android App construction. Hence, by cost-benefit analysis we can conclude that benefit to cost ratio is high.

2) Technical Feasibility: As mentioned earlier, the task of coding and debugging was made easier by using CNN for the model. As only smartphone sensors such as camera and speakers are required and much GUI is not required, the app construction coding was also not a very difficult task. As every segment will be coded individually and separately, the risk assessment for each segment is easy.

3) Operational Feasibility: As illustrated in the use-case diagram, the user only controls the initiation of the App and stopping of the alarm. Hence, the controls are simple and basic. Efficiency of the project is based upon few factors, namely, the model and its individual accuracy, the front end and its latency, the inter-connectivity. All these components can again be controlled individually and hence accuracy of the project as a whole can be easily controlled.

C. *Design*

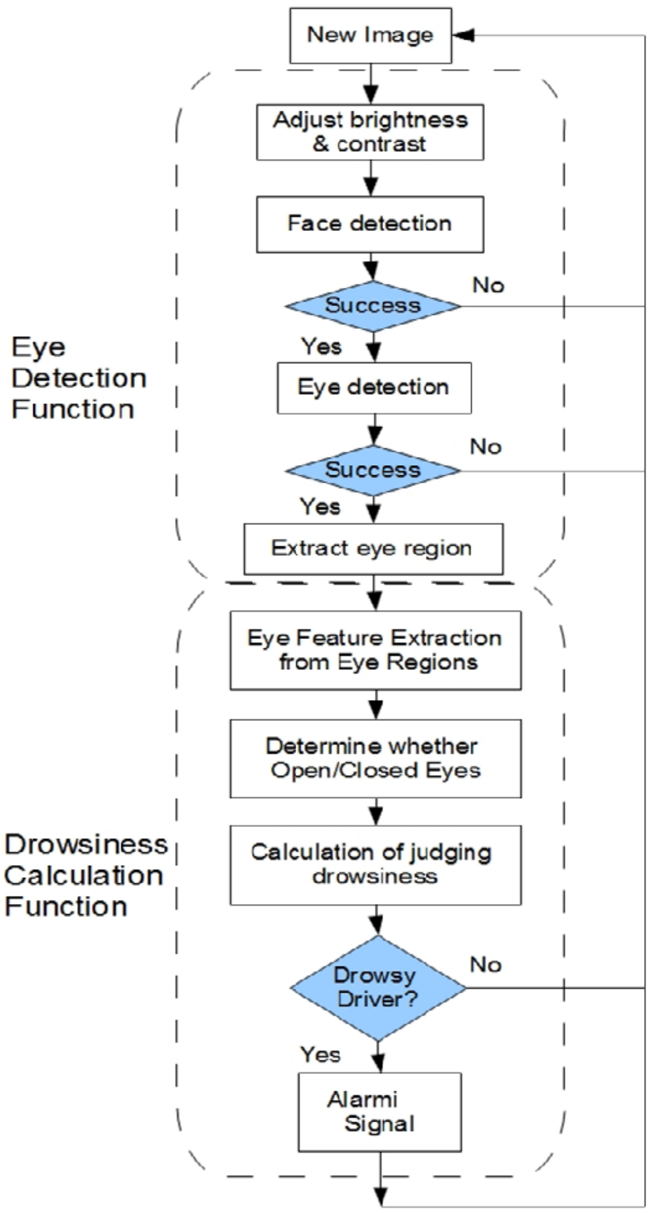

New Image
Adjust brightness & contrast
Face detection
Success
No
Yes
Eye detection
Success
No
Yes
Extract eye region
Eye Detection Function
Eye Feature Extraction from Eye Regions
Determine whether Open/Closed Eyes
Calculation of judging drowsiness
Drowsiness Calculation Function
Drowsy Driver?
No
Yes
Alarmi Signal

Fig. 2 Flowchart of the Project

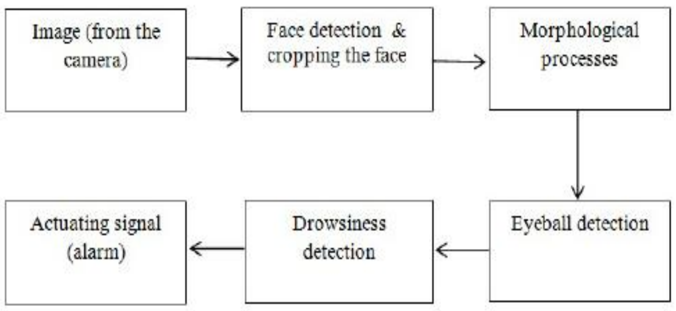


Fig. 3 Block Diagram

IV. Implementation

Firstly the dataset used in this project is the mrlFace dataset available on Kaggle. This dataset consists of 10,000 images of several people's faces with open and closed eyes. The first problem we came across was that this dataset was more of a directory with a dump of images in one place. There was no labelling or order to it. This dataset needed to be arranged and labelled properly and we needed only images of the eyes so this needed to be extracted. For this we used Open CV Haar-Cascade classifier. This is a feature based cascade classifier. Using its inbuilt functions to detect face and from it to detect eyes region, we ran this process over all the 10,000 images in the dataset and generated eyes dataset.[8]

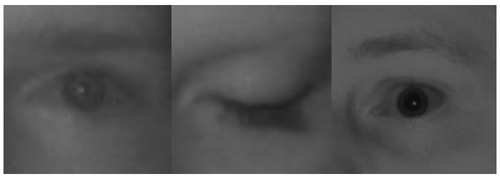


Fig. 4 Eye Dataset

After this the eye pictures were segregated into labelled folders as open and closed appropriately. This concluded the dataset acquisition step.

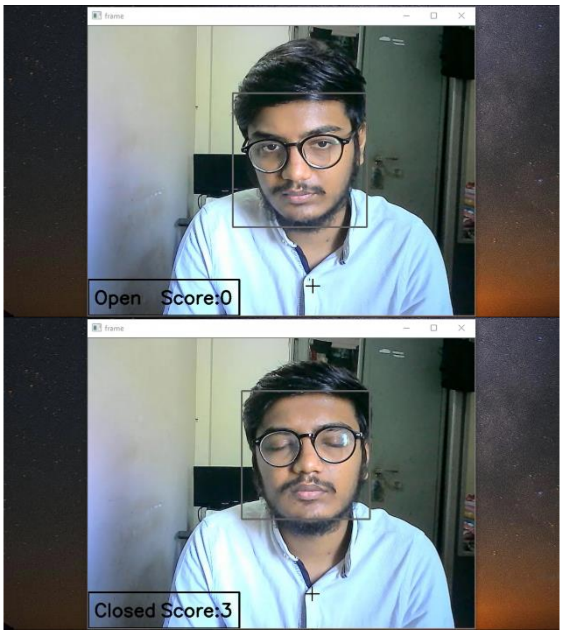
Next we began working on the CNN classification model. As mentioned earlier CNN or Convolutional Neural Network is a classification algorithm that works best for categorical data/images. It identifies and does the feature extraction work on its own by using pixel edge detection method, saving us the work to code extraction of individual features. CNN generally consists of 3 layers, Convolution layer, pooling layer and fully connected layer. This structure provides us with a basic blueprint where we only have to choose certain activation functions, values for inbuilt variables, etc. depending upon the desired output.
We used a sequential model with Relu (Rectified Linear Unit) activation. We then converted the 3 channel images (RGB images) into grayscale images with 32 feature detector. In the pooling layer we are using MaxPooling to reduce theConvolution Matrix. We then add 3 hidden layers with 32, 32 and 64 neurons respectively with MaxPooling. These values and number of layers were agreed upon after several trial and error runs. Next we use the dropout function to reduce the chances of over fitting. Now we flatten the output matrix and use dense layer to create fully connected layer. Sigmoid function along with Softmax activation is used to output values in binary format such as 0 or 1 for Open or Closed respectively. Next we compile the model with Adam optimizer and set value for epochs and fit the model with train and test data. We trained the model for 20 – 30 epochs. This model gave us an accuracy of about 91.3% against the validation dataset.[1][2]

Next we plan to merge this machine learning model into an android application which will enable the driver to use his smartphone as a tool which will facilitate this system. The camera of the smartphone will be used for inputting image or video while the classification model works in the background to detect the driver's condition. Again Haar-Cascade classifier will be used upon the captured images or videos to capture the face and from it the eye region. This data will be given to the model for classification. Based upon a pre-set conditional program, if the eyes were found to be closed or not detected for a particular number of consecutive frames then the alarm should ring. The phones speakers can be used to alert the driver by playing the alert tune.[4]

V. RESULTS

As we see in the following figure, the entire frame captures the face of the user. On the bottom left of the frame, we can see the eye state, Open or Closed and a score or counter.

The score keeps track of number of consecutive frames with same outcome. If the closed eye state is detected and the score for closed eye exceeds 10 in this demonstration, the alert tone rings and the borders of the frame turns red for visual cues.



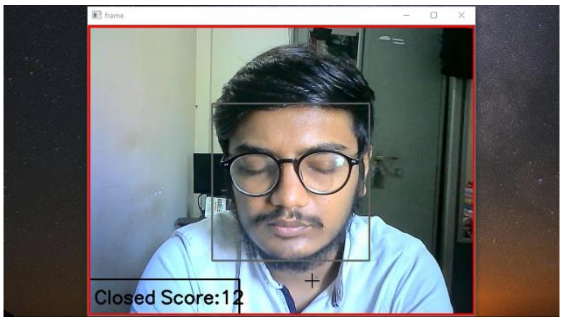


Fig. 5 Results

VI. PLANS FOR FUTURE

As per our plans, we have completed with the back end or the classification model part of the project.
Next we aim to finish the front end or the Android part of the project. Considerable amount of research yet needs to be done in this field as we need to take latency into consideration. We have to decide whether to deploy the model on the app itself or to deploy the model on a common cloud server which will communicate with the android front end. Again the option with least amount of latency will be chosen. We also plan to research into Firebase and Tensor Flow Lite as these platforms support ML model linking with Android App. We will also work on enhancing the model itself if possible to better perform in real time. The final aim is to reduce the project response time in real life as even a seconds delay could cause an accident to occur.

VII. CONCLUSION

There is one thing to bear in mind while approaching this topic of driver safety and warning system, Accidents can and will occur in a split second. It is very difficult for even the most complex system to look into the situation and always predict the right outcome and give an early warning. We need to understand that even after everything is done, some situations may be too sudden and inevitable. Our project only aims to be a preliminary system that helps in any way possible.
Some limitations our project would face are:
1. Dark Environment 2. Dark Sunglasses 3. Fault in Camera, etc.
One way to resolve most of these issues would be to make a dedicated unit that solely functions for the purpose of our project. An Infrared camera, an alarm, an IOT chip would be sufficient to run these processes seamlessly. Light enhancement algorithms can also be used to reduce the effects of dark environment.
We have learned a lot of new things and have achieved the planned output in our project.After several testing and presentation rounds, suggestions were noted and improvements were done. We plan to deploy a fully functioning Application with the goal of keeping the user alert and safe. We believe that an alert driver equals safe passengers and safe pedestrians.

REFERENCES

[1] https://arxiv.org/ftp/arxiv/papers/1811/1811.01627.pdf
[2] https://www.researchgate.net/publication/333627508
[3]https://www.kaggle.com/varanr/
[4] https://towardsdatascience.com/drowsiness-detection-with-machine-learning765a16ca208a