Drowsiness Detection System

IoT Project Report

Team

Rajendra Sahu (731008796)

Swarna Prabhu (932003046)

Meghna Ghole (132006015)

Date: 12/12/2021

1 Executive summary

Our project mainly aimed at applying IoT concepts in detecting and alerting a driver who drives the vehicle in drowsy state. Since IoT applications are well versed in each domain the intent here was to apply it to enhance and support safe driving practice. As per the statistics from a recent finding from NHTSA it was observed that 100,1000 police reported crashed and approximately 1500 people died owing to drowsy driving. Hence there is a significant need to detect and alert the driver as well as his/her family in case the driver seems to drive in drowsy state. The goal and objective of the project has been to analyze the condition of driver using face detection and take appropriate measures to prevent him from sleep driving. This was designed by calculating eye aspect ratio and mouth size (if yawning) and send an alert if the values go beyond a certain threshold. Email alert was sent to the concerned family using the IFTTT applet. The real time video of the driver is captured and hosted on flask server. The alert instances are pushed to influxdb and can be visualized on grafana dashboard. The email alert can be used to call or inform the driver to be safe or stop driving at the same instance since data is captured, processed, and analyzed in real time.

2 Introduction

2.1 Problem background

It is estimated that every year in the US, more than 100,000 crashes and around 1600 deaths are caused due to vehicle crashes and accidents with drowsy driving being the sole reason. This figure was published recently by the National Safety Council. This is especially common among the youths who are involved in overnight work or shifts and among people who need to frequently travel long distances. Drowsiness generally results in improper judgement and sleep that can trigger the accidents.

2.2 Needs statement

There is hence a need to address the above problem using techniques and tools that provide real time analysis of the driver's state and provide necessary information in the form of alerts that could possibly prevent an accident and save our life. There is a need to even store and record the driving pattern of the driver which might help driving authorities or families of the driver to make better decisions regarding the driving capability of the person.

2.3 Goal and objectives

The goal of the project work is to implement an IoT based solution to implement a drowsiness detection system for the person driving and integrate an alert system to prevent the driver from causing any violation of the driving rules.

The objective of the study is to initially perform eye detection on real time video representing the driver by extracting the facial landmarks cornered around eyes and mouth. Conclusion on drowsiness is arrived based on 2 parameters namely: eye aspect ratio and mouth size (when the driver is yawning). We expect that an alert is triggered indicating the driver is drowsy when the eye aspect ratio is less than 2.5 or when the mouth aspect ratio is greater than 0.8 validating the drowsiness state exhibited by the driver. The objective of the project also includes using a database system to capture and store the instances when one of the above drowsiness conditions is met and visualize on Grafana the trend of the drowsiness state based on time as function.

2.4 Design constraints and feasibility

It is not possible to conclude that the driver is in an actual drowsy state where he is unable to make proper judgement only based on blinking of the eye and yawning. This is because every individual has their own tendency to yawn and blink eyes and to use a generic threshold value for comparison may not work in all the cases. For comparison the above 2 parameters do not provide 100% accurate result and we might want to include a system that also tracks or monitors the pulse rate, heart rate which will require the integration

of hardware and sensors and combine with the image processing techniques used for eye , mouth and facial analysis detection

3 Literature and technical survey

There have been several developments and approaches to design a feasible solution that accurately detects the drowsiness state of person driving ranging from image processing to sensor-based hardware solutions to artificial neural network-based methods.

1. Steering Wheel Movement method was used where sensors and hardware is integrated in the vehicle to determine the steering angle. Special sensors named steering angle are added near the steering portion of the car and the change in the steering pattern is used to infer the drowsiness state of the driver. Those drivers who made very less steering rehearsal were supposed to be sleep deprived. But this approach is not too feasible as this will now demand the users to upgrade their vehicles to those models that come with this feature. It is not economical for everybody to buy this model only for one feature. There is also a disadvantage with the hardware approach that on a frequent basis it should be maintained and upgraded. This method is also subjected to the environment and weather conditions where the car is being driven and hence doesn't guarantee a precise outcome.
2. EEG based physiological method- This is also a conventional model for sleep detection which is based completely on the physiological parameters where brain signals are studied for fatigue detection. The driver will have to give the EEG signal and the signal levels are studied to chalk out a pattern for fatigue detection. These methods are not feasible for common people who drive on a frequent basis as the user will have to plug the sensor on the scalp and in the long run the associated health risks with this is yet to be researched.

The method we implement provides a simple software based iot solution in open cv , flask and influxdb that provides real time data to the user whenever drowsiness is detected for a certain time.

4 Proposed work

4.1 Evaluation of alternative solutions

To make our system IoT compatible, our first choice was NodeRed. It's a flow-based development platform for visual programming. It helps the developer connect various IoT endpoints, APIs, hardware devices and create flows with the most convenience. It has support for some sophisticated frameworks like Google TensorFlow, MQTT etc in the form of nodes.

Our first attempt in coming up with a solution was to figure out an image processing node that can process our webcam video feed and detect for drowsy faces and then forward those time instances to the alert system. So, our next step was to find a NodeRed appropriate image processing node or any free image processing API.

We came up with these solutions

- NodeRed using OpenCV node solution
- NodeRed using Google Vision API solution

4.1.1 NodeRed using OpenCV node solution

OpenCV has an image processing node which is based on the popular open-source computer vision package OpenCV. The node is named similarly. Nonetheless, we couldn't fully adopt this solution. We faced issues in installation of the node itself. Installing cli nodes on NodeRed usually is troublesome. On further investigation we came to know that this node is no longer supported by the NodeRed developer community.

The developer of this particular node got back to us saying that we might need to develop our own image processing node based on OpenCV. Hence, we had to drop this solution.

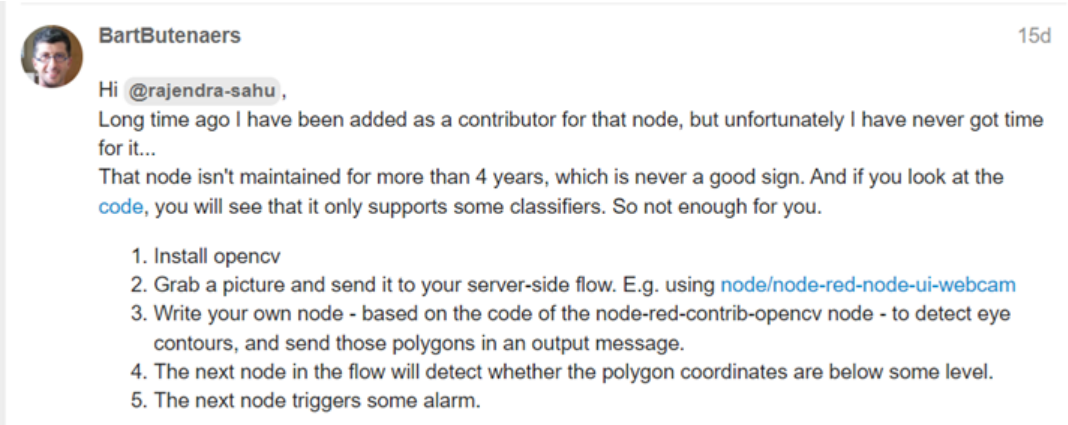

Figure 1. Response from the OpenCV node developer

4.1.2 NodeRed using Google Vision API solution

We also tried to use the Google Vision API to send our captured frames for image processing. This is a paid service to get the API results in the form of json. The json response gives out generic results like facial landmarks, emotion likelihood etc. However, we couldn't use this solution either. We tried to calculate the probability of the eyes being closed. We compared this probability in both types of sample images i.e. open & closed eyes and the difference was not significant enough. Our calculation was based on the EAR(Eye Aspect Ratio) explained later in the next section. The inaccuracy could be probably because of the discrepancy between the facial landmarks the API provides and the landmarks the calculation needs as input.

Please note that the above solutions worked on the assumption that image inputs were provided to them. The sampling of images from a video capture still exists for which another node should be used. Considering the lack of support on the NodeRed side we decided to discard NodeRed from our design and proceed with a flask-based solution.

4.2 Design specifications

We adopted a new approach in which we moved our image processing framework onto the flask server. This step comes in after we decided to discard NodeRed (reasons explained in the above section). But we still want to make our system IoT compatible. Hence, we adopted the Flask approach. We host our image processing framework & alert system on Flask. The Flask server also runs the webcam & a mini-UI. We also have added features like email alert, database update & data visualization. This is intended to make the system complete from a user perspective. The data logging should be used to infer dozing patterns of the driver and establish some driving criteria.

We use OpenCV and relevant computer vision python packages in our image processing framework. The programming flow is described below

1. Sample a frame from the video capture
2. Process the frame
3. Detect drowsiness on two parameters: EAR(Eye Aspect Ratio) & mouth size (yawning)
4. If drowsy (based on thresholds) keep accumulating such frames
5. If such frames exceed a threshold frame count, then

a. send email alert by sending a POST request to IFTTT registered address
b. write time instance and EAR value to InfluxDB

6. Keep running the flow
7. Visualize the InfluxDB data on Grafana to make inferences about the dozing pattern of the driver

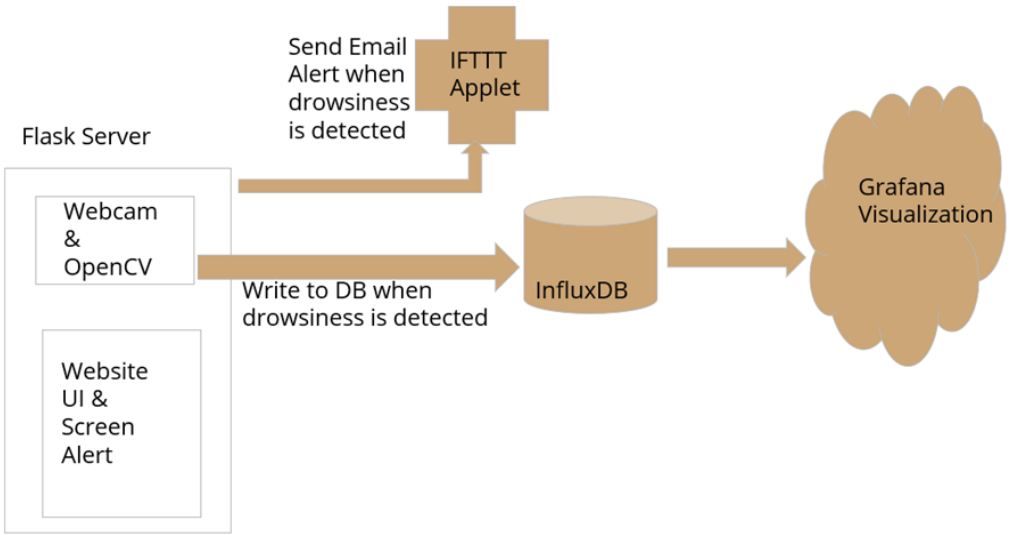

Figure 2. Overall Design

4.2.1 Image Processing Algorithm

The algorithm has a popular standard 68 facial landmarks plot maintained by imutils(python package) as its foundation. OpenCV with this landmark reference predicts the coordinates of the same points on the frame image using another adjacent python package named dlib. We use these projected coordinates to calculate the EAR(eye aspect ratio) and mouth size.

EAR is calculated as per the below formula. This concept has been derived by reference paper 1 which says that EAR for closed eyes is very low, around 0. We use this result to trigger an alarm when the EAR falls below 0.25 (dozing means partially closed). In a similar way we calculate mouth size. On manual calibration we find the mouth size shouldn't exceed 0.8.

The UI alerts are constructed using the contours function of OpenCV.

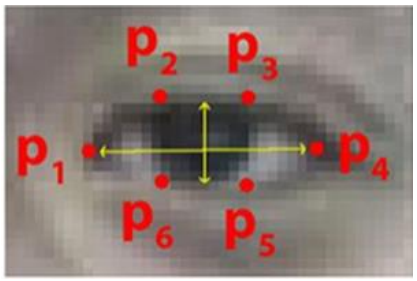

Figure 3. Eye landmark map

Eye Aspect Ratio(EAR) :

$$EAR = \frac{\|p_2 - p_6\| + \|p_3 - p_5\|}{2\|p_1 - p_4\|}$$

Figure 4. EAR Formula

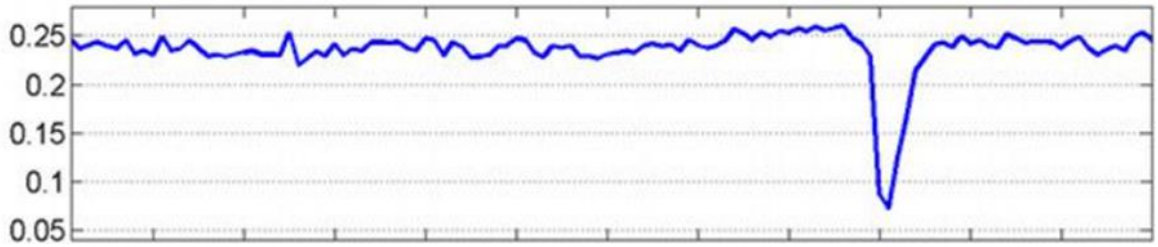

Figure 5. Plotting EAR over time. The dip in the EAR indicates a blink (Figure 1 of Soukupová and Čech)

4.2.2 Packages, Dependencies and Tech Stack

We use several frameworks and technologies to build our system. Apart from a webcam every technology is purely software. All the software technologies are open source. We rely heavily on Python based packages. Image processing is seamless on Python. The following is the list of packages and dependencies.

1. Python
2. Flask: The server framework to host our application
3. OpenCV: To capture the video feed and sample frames
4. dlib: Python package that has a face predictor
5. numpy: Python package to convert the frames to numpy array for easier processing
6. imutils: Python package that has the 68 facial landmark
7. scipy: Provides a distance module to calculate the 2D distance between points
8. influxDB: To store the data
9. requests: Package to send http requests
10. Grafana: Data visualization framework
11. IFTTT: Email alert framework

4.3 Approach for design validation

This project provides an opportunity of simplistic, manual testing. And this testing is bound to give us the highest accuracy. We test our application with live video capture from the webcam. We continuously test it for different faces and multiple iterations over a long period of time. That's how we calibrate the EAR and mouth size threshold to be 0.25 and 0.8 respectively. For a detailed demonstration please refer to the video attached with this report. As a future scope, machine learning models could be incorporated to tune and calibrate these thresholds.

1. 4.3.1 Output

Figure 6. Closed Eyes Alert

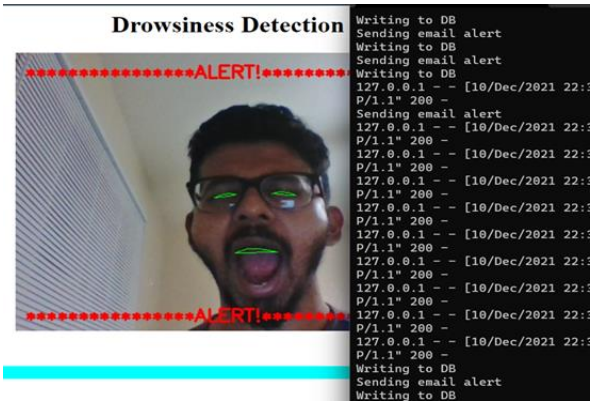

Figure 7. Yawning Alert

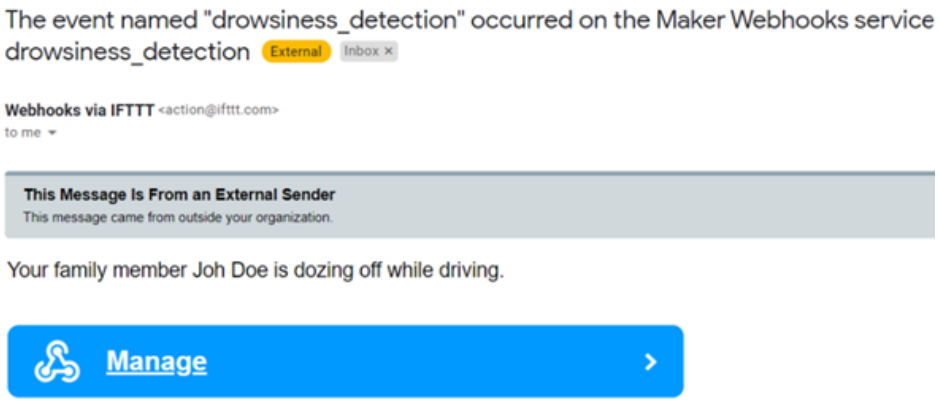

Figure 8. Email Alert

```
> select * from driverdata
name: driverdata
time                EAR                 Name
----                ---                 ----
1623540150000000000 0.17381261151178956 John Doe
1623540151000000000 0.0916801288245464  John Doe
1623540152000000000 0.134389444106081   John Doe
1623540153000000000 0.1501696489539891  John Doe
1623540158000000000 0.09576137932557255 John Doe
1623540159000000000 0.21965380911781981 John Doe
1623540163000000000 0.15451851375062436 John Doe
1623540164000000000 0.19045949971952852 John Doe
1623540168000000000 0.20317849144832634 John Doe
1634076056000000000 0.16731156142353232 John Doe
1634076057000000000 0.2104905253259104  John Doe
1634076058000000000 0.228937317757293   John Doe
1634076065000000000 0.2028016408064566  John Doe
```

Figure 9. InfluxDB Update

Figure 10. Grafana EAR visualization

5 Engineering standards

5.1 Project management

Raj : Analytical ability, Leadership, Problem solving

Swarna : Critical thinking, Attention to detail, Organization

Meghna : Logical thinking, Creativity

Table 1. Task division.

Members	Tasks	Characteristics
Raj	Team Leader, Testing	• Create team vision, assignment of individual tasks • Emphasis on providing needed dependencies • Incorporation and testing the outcomes one by one • Implementation and combination of every part together
Swarna	System Design	• Consider all opportunities and viewpoints. • Researching and providing workable modules and components needed.

Meghna	Software Design	• Solving problems and errors faced during compilation and debug. • Planning and providing requisite parts of the code.

Table 2. Task matrix.

		Team members		
Task	Description	Raj	Swarna	Meghna
1	Team Leader	X		
2	System design		X	
3	Software design			X
4	Testing	X		

5.2 Weekly schedule of tasks, Pert and Gantt charts

- Week 1 : Finalizing the topic for the project, individually ideas and solutions to the problem statements were in process.
- Week 2 : First stage was to incorporate different ideas and find a better solution. Then dividing sub parts among us. Raj worked on opencv inclusion with nodered and Swarna and Meghna were responsible for working on API. After the failure of the first trial, we moved on to a new idea.
- Week 3 : Continuing with a new approach, Raj incorporated opencv, dlib, flask nodes and also worked on code and Swarna and Meghna progressed ahead with coding for needed results. Compiled each part together to create a workable and expected output.
- Week 4 : After successful completion, implemented and tested the project for accuracy and adequacy. In parallel, included results and shortcomings in report and presentation.

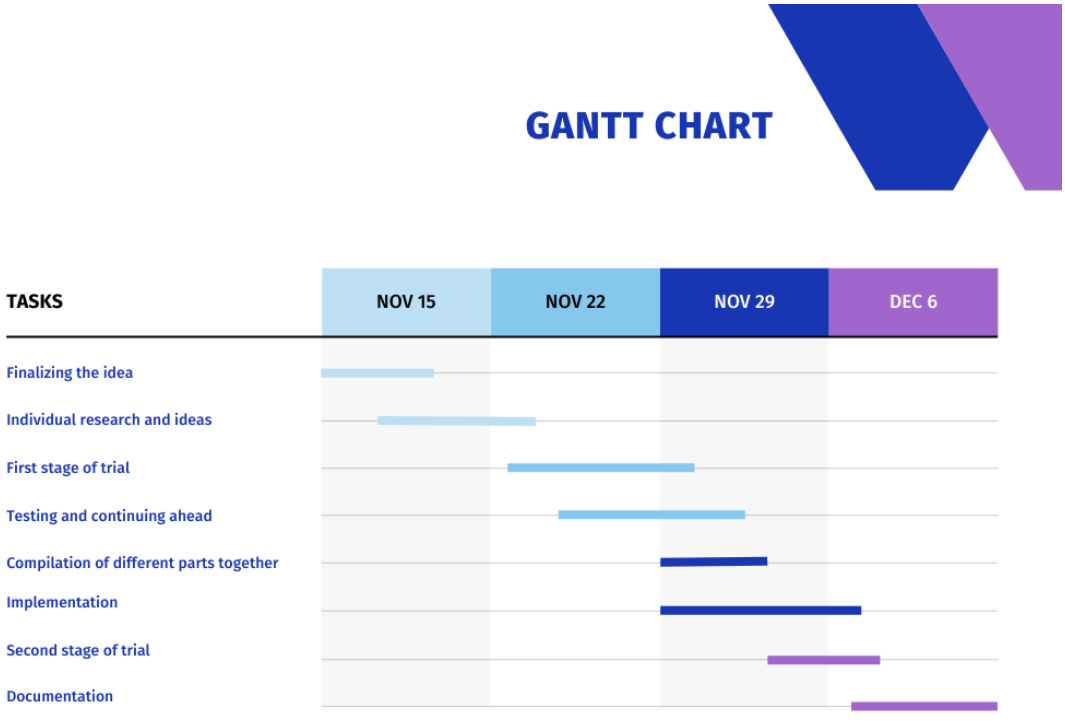

Figure 11. Gantt chart

5.3 Economic analysis

Economic analysis, when the system to become a commercial product:

- Economical viability: Since the project is solely constructed using software products and a web camera, manufacturing the system for mass production would be feasible and cost effective. Installation cost would be less due to high mobility facilities.
- Sustainability: We can make the code available for aspiring enthusiasts who wish to use the system. Due to software inclination, maintenance is reduced significantly. If there are any improvements needed due to compatibility issues, then the system would require support.
- Manufacturability: Tools like flask, influxdb, grafana used in the project have a local server where the resulting output and analysis come about, if the servers are down or under maintenance, this could affect the performance of the product.

5.4 Itemized budget

The project was implemented using software, free and open source resources. Tools/Libraries used viz. Opencv, influxdb,dlib are free and open source platforms. The only hardware used is a web camera for capturing video of the subject.

6 References

[1] T. Soukupová and J. Čech, "Real-Time Eye Blink Detection using Facial Landmarks", presented at *21st Computer Vision Winter Workshop*, Luka Čehovin, Rok Mandeljc, Vitomir Štruc (eds.) Rimske Toplice, Slovenia, February 3–5, 2016.

[2] A. Rosebrock, "Eye blink detection with OpenCV, Python, and dlib", pyimagesearch.com.

https://www.pyimagesearch.com/2017/04/24/eye-blink-detection-opencv-python-dlib/ (Accessed Dec. 4, 2021).

[3] A. Behl, "Video Streaming Using Flask and OpenCV", medium.com

https://medium.datadriveninvestor.com/video-streaming-using-flask-and-opencv-c464bf8473d6 (Accessed Dec. 4, 2021)

[4] N. Pathak, "Yawn Detection using OpenCV and Dlib", medium.com https://medium.com/analytics-vidhya/yawn-detection-using-opencv-and-dlib-e04ba79b9936 (Accessed Dec. 1 2021)

[5] Face recognition using Vision API, cloud.google.com.

https://cloud.google.com/vision/docs/detecting-faces (Accessed Dec. 1 2021)

[6] A. Neves and D. Lopes, "A practical study about the Google Vision API" in *22nd Portuguese Conf. on Pattern Recognition(RECPAD 2016),* Aveiro, Portugal, October 2016, Vol 1, 2 pages.

[7] P. Choudhary, R. Sharma, G. Singh, S. Das, "A Survey Paper On Drowsiness Detection & Alarm System for Drivers", in *International Research Journal of Engineering and Technology (IRJET)* ISSN: 2395 -0056 Volume: 03 Issue: 12 , Dec. 2016, pp. 1433-1437.

7 Appendices

7.1 Bio-sketch

Rajendra: I am the one who came up with the idea of Drowsiness detection. This gives me the confidence that I have a good sense of the real world problems and how technology can be used to solve them. I have knowledge of multiple software frameworks and I was able to suggest alternatives when one solution didn't

work. It was an immense learning experience for me as I got to know that real life problems always come with outliers that your conventional solutions won't be able to solve.

Meghna Ghole : My ability to think and analyze by looking at the current development and data available is prodigious. Relating information to tools we have has always been one of my key strengths. I enjoyed working with my group mates and utilizing the products taught during the course and incorporating them for applied experience.

Swarna: I am good at researching and exploring different ways for implementation of a problem and this gave me the opportunity to compare and implement different approaches one can adapt to solve a problem. I utilized my decision-making skills to suggest what additional features can be enhanced to come up with a n iot compatible solution.