TECHNICAL REPORT STANDARD PAGE

1. Title and Subtitle

High-Fidelity Fatigue, Drowsiness, and Drunk Drivers Detection (FD^4) System

2. Author(s)

Yasser Ismail

3. Performing Organization Name and Address

Electrical Engineering Department
College of Sciences and Engineering,
Southern University and A&M College,
Pinchback Hall, Room 424
Baton Rouge, LA 70804-9245

4. Sponsoring Agency Name and Address

Louisiana Department of Transportation and Development
P.O. Box 94245
Baton Rouge, LA 70804-9245

5. Report No.

FHWA/LA.17/000

6. Report Date

June 2022

7. Performing Organization Code

LTRC Project Number: 22-2TIRE
SIO Number: DOTLT1000415

8. Type of Report and Period Covered

Final report
6/2021 – 6/2022

9. No. of Pages

57

10. Supplementary Notes

Conducted in Cooperation with the U.S. Department of Transportation, Federal Highway Administration

11. Distribution Statement

Unrestricted. This document is available through the National Technical Information Service, Springfield, VA 21161.

12. Key Words

Drowsiness Detection; Machine Learning (ML); Traffic Safety

13. Abstract

Drowsiness, fatigue, and alcohol are major factors that deteriorate driving performance, threaten road safety, and cause severe injuries, deaths, and economical losses in the United States. This study improves the accuracy of drowsiness detection by using the Dlib algorithm for face detection. Dlib shape predictors extract the facial features that aid in calculating essential parameters: the Eye Aspect Ratio (EAR) and Mouth Opening/Aspect Ratio (MOR/MAR). These values are the primary indicators of driver drowsiness. The parameters are evaluated to assess the level of drowsiness/alertness of the driver. Three methods are proposed: fixed thresholding, adaptive thresholding, and the method of Machine Learning modeling and classification. The fixed threshold approach involves alerting the driver of drowsiness once the EAR or/and MOR value meets the criterion. The adaptive thresholding involves a counter to track a set number of consecutive frames that meet the criterion before sending a warning. For the third method, three Machine Learning (ML) classifiers were used: Support Vector Machine (SVM), Naïve Bayes (NB), and Random Forest (RF). A dataset of 1448 images was used for training and testing these classifiers: 70% for training and 30% for testing. The accuracy and sensitivity in detection using fixed thresholding are 89.4% and 96.5%, respectively. The accuracy and sensitivity in detection for the adaptive thresholding method are 93.4% and 89%, respectively. SVM classifier gives the best accuracy of 90.34% compared to Random Forest (89.86) and Naïve Bayes (65.29).

Project Review Committee

Each research project will have an advisory committee appointed by the LTRC Director. The Project Review Committee is responsible for assisting the LTRC Administrator or Manager in the development of acceptable research problem statements, requests for proposals, review of research proposals, oversight of approved research projects, and implementation of findings.

LTRC appreciates the dedication of the following Project Review Committee Members in guiding this research study to fruition.

LTRC Administrator/Manager

Sam Cooper

Research Manager

Vijaya Gopu

Members

Enter member's names, one per line

Directorate Implementation Sponsor

Christopher P. Knotts, P.E.

DOTD Chief Engineer

High-Fidelity Fatigue, Drowsiness, and Drunk Drivers Detection (FD4) System

By

Yasser Ismail

Co-principal Investigator

Department of Electrical Engineering, College of Sciences and Engineering,

Southern University and A&M College

Pinchback Hall, Room 424

Baton Rouge, LA, 70813

LTRC Project No. 22-2TIRE

SIO No. DOTLT1000415

conducted for

Louisiana Department of Transportation and Development

Louisiana Transportation Research Center

June 2022

Abstract

Drowsiness, fatigue, and alcohol are major factors that deteriorate driving performance, threaten road safety, and cause severe injuries, deaths, and economical losses in the United States. Lack of alertness, generated by drowsiness, fatigue, and alcohol, leads to several severe road accidents. The United States National Highway Traffic Safety Administration (NHTSA) reports that drowsy driving results in almost 100,000 road accidents and more than 1,500 deaths per year. Thirty (30) people in the United States die from drunk driving every day. This study improves the accuracy of drowsiness detection by using the Dlib algorithm for face detection. Dlib shape predictors extract the facial features that aid in calculating essential parameters: the Eye Aspect Ratio (EAR) and Mouth Opening/Aspect Ratio (MOR/MAR). These values are the primary indicators of driver drowsiness. The parameters are evaluated to assess the level of drowsiness/alertness of the driver. Three methods are proposed: fixed thresholding, adaptive thresholding, and the method of Machine Learning modeling and classification. The fixed threshold approach involves alerting the driver of drowsiness once the EAR or/and MOR value meets the criterion. The adaptive thresholding involves a counter to track a set number of consecutive frames that meet the criterion before sending a warning. For the third method, three Machine Learning (ML) classifiers were used: Support Vector Machine (SVM), Naïve Bayes (NB), and Random Forest (RF). A dataset of 1448 images was used for training and testing these classifiers: 70% for training and 30% for testing. The accuracy and sensitivity in detection using fixed thresholding are 89.4% and 96.5%, respectively. The accuracy and sensitivity in detection for the adaptive thresholding method are 93.4% and 89%, respectively. SVM classifier gives the best accuracy of 90.34% compared to Random Forest (89.86) and Naïve Bayes (65.29). The proposed methods can be embedded into automobiles, thus enhancing road safety as nearby police are alerted for quick response. The results of this study will be used as preliminary results to secure a Federal grant (NSF) which will help the PI to attract more graduate students to work with him for better research development and career improvement.

Acknowledgments

Thank you to the Louisiana Transportation Research Center (LTRC) for the financial support to finalize this project and to the chair of the Electrical Engineering Department at Southern University and A&M College, Fred Lacy, Ph.D., for his support with graduate students who help in implementing different models for Fatigue, Drowsiness, and Drunk Drivers Detection. The PI would like to especially thank Mr. Ebenezer Essel for his great efforts in implementing the algorithms and writing the final report. A special thanks to the Project manager Dr. Vijaya Gopu and the Project Committee (PRC) members for their valuable feedback. Finally, thank you to all LTRC members who are always supporting Southern University and its faculty and graduate students.

Implementation Statement

The results of this project are directly applicable for implementation by DOTD as well as for local government entities throughout Louisiana and beyond who are interested in Traffic Safety. The project provides three implemented prototypes for Fatigue, Drowsiness, and Drunk Drivers detection. The first prototype utilizes a fixed thresholding method using the Dlib algorithm for face detection. The Dlib shape predictors extract the facial features that aid in calculating essential parameters: the Eye Aspect Ratio (EAR) and Mouth Opening/Aspect Ratio (MOR/MAR). These values are the primary indicators of driver drowsiness, fatigue, and drunk situations and are evaluated to assess the level of alertness of the driver. The adaptive thresholding uses a counter to track a set number of consecutive frames that meet the criterion before sending a warning. The adaptive thresholding makes it possible to track and detect any type of fatigue online and allows fast detection and fast alert process. The third prototype utilizes different Machine Learning (ML) methods. Support Vector Machine (SVM), Naïve Bayes (NB), and Random Forest (RF) are the used three ML methods. The performance of the implemented prototypes is measured using different parameters such as precision, recall, F-score, confusion matrix, and loss and accuracy curves. The research team endeavors to present and publish the findings, which contribute to the overall literature in this field or may be of interest to practitioners in journals with a national audience to facilitate the transfer of research more broadly. The research team will also use the results from this project as base results for securing a Federal grant (NSF).

Table of Contents

List of Tables

List of Figures

Introduction

Drowsiness, fatigue, and alcohol are major factors that deteriorate driving performance, threaten road safety, and cause severe injuries, deaths, and economical losses in the United States. Lack of alertness, generated by drowsiness, fatigue, and alcohol, leads to several severe road accidents. The United States National Highway Traffic Safety Administration (NHTSA) reports that drowsy driving resulted in almost 100,000 road accidents and more than 1,500 deaths per year. Thirty (30) people in the United States die from drunk driving every day. Drunk driving is estimated to be responsible for 40% of all traffic accidents [1]. Public safety and the reduction of drowsiness, fatigue, and drunk driving accidents are essential in the development of a smart city. A better understanding of the drivers' behavior and a good expectation of their attention level ultimately affects serious threats to road safety of Louisiana streets (especially Baton Rouge streets). Road ranges are extremely high and cause much stress on drivers. Thus, it is recommended to embed automatic drowsiness, fatigue, and drunk driver detection system into vehicles that can accurately detect the drivers' attention level and alert the driver and the police station before the arrival of any serious threat to road safety.

The drivers' behavior and attention level detection system is a real-time system that investigates the driver's physical and mental condition based on driver face images. Such detection systems make their decision based on four (4) categoric methods: (1) methods based on drivers' physiological signals, such as electroencephalograph (EEG), electrocardiograph (ECG), and electrooculogram (EOG); (2) methods based on drivers' operation behavior, such as steering wheel operation; (3) methods based on vehicle states, such as the trail of vehicle and lane departure information; and (4) methods based on visual features, such as yawning, eye blink rate, and percentage eyelid closure [2].

In developing such drivers' behavior and attention level detection systems, the main challenge is how to define drowsiness or fatigue or drunkenness and how to measure it. There is a relationship between drowsiness or fatigue or drunkenness and some symptoms such as yawning, eye blink rate, percentage eyelid closure, body temperature, the electrical resistance of the skin, eye movement, etc. The most crucial symptom of drowsiness or fatigue or drunkenness appears in the eye region. There is a relationship

between the Psychomotor Vigilance Task (PVT) and the percentage of eyelid closure over time (PERCLOS). PVT shows the response speed of a person to visual stimulation. Another challenge is measuring the driver's concentration on the road. Concentration on the road is measured from the driver's head orientation, driver nodding, and head direction angle [3].

The drivers' behavior and attention level detection system are divided into face detection, eye detection, symptom extraction, yawning detection, and driver behavior state estimation. There are two methods in the face and eye detection. The first method is based on applying heuristic rules on features, and the second method is based on using statistical learning methods and machine learning algorithms, such as adaptive boosting algorithm, HAAR Cascade [4]. The detection system's crucial symptoms are symptoms related to the eye, mouth, and head region. Symptoms related to the eye region are PERCLOS, eyelid distance, eye-blink speed, eye-blink rate, and gaze direction. A symptom related to the mouth region is yawning. Yawning is extracted by template matching (training and testing samples). Symptoms related to the head region are head orientation, driver nodding, and angle of head direction. After symptom extraction, a machine learning algorithm, such as the Bayesian network and nave dynamic Bayesian network, can be used to classify the drunk driver state. Most of the available research for detecting drowsiness, fatigue, and alcoholic drivers are treated from the algorithmic point of view only. Such available algorithms use recorded videos for evaluating their accuracy. They are not evaluated for real-time videos (under different conditions such as light intensity and darkness).

The PI with his team implements three prototypes for accurately detecting Fatigue, Drowsiness, and Drunk situations of drivers. The implemented prototypes (if adopted in a complete hardware/software FD^4 system) can alarm both the driver and the police office for better treatment of the situation to prevent futures accidents. The FD^4 system consists of a hardware part that can capture real-time videos of the driver while driving, and the software part that uses machine learning to use the captured videos to estimate the attention level of the driver. Developing such a real-time system will be through a federal grant since it needs more graduate students to be involved. This TIRE proposal helps the

PI to hire graduate students to develop a high-fidelity FD^4 software (algorithms) that will be used for a future federal grant to build the overall FD^4 system. This will greatly reduce the accident rate not only in the United States but also in the overall world. The expected preliminary results from this project are mandatory to be adopted and support a future Federal proposal for the PI. The Federal proposal will use the developed FD^4 algorithms from this project to build a complete software/hardware FD^4 system that could potentially benefit many user groups within DOTD and LTRC. If successful, it will potentially decrease the accident rates in the United States, thereby saving lives. DOTD, MPOs, and other agencies that rely on road safety for various purposes can directly benefit from this research.

Literature Review

Drowsiness detection is a safety technology employed in automobiles to help prevent accidents caused by drowsy drivers. Several drowsiness detection methods exist that check the state of driver drowsiness and alert the driver. Detection techniques can be classified into behavioral, vehicular, and physiological parameter-based methods. Vehicular and behavioral-based techniques are non-invasive; whereas vehicular based is largely dependent on the geometric characteristics of the road which affects the car's movement, such as movement of the steering wheel [1] and drifting of the car from lanes, behavioral-based monitors the behavior of the driver such as the eye closure ratio, eye blinking rate, facial expression, and yawning [5] for detection. Physiological techniques are invasive/intrusive and measure the driver's physical conditions like heartbeat and pulse rate. Vehicular-based techniques are subject to geometric conditions of the road and weather, thus, very unreliable [2]. Physiological-based techniques have high accuracies but present an excessive cost and complexity/difficulty in implementation [2] [1]. Thus, the behavioral-based technique is widely used because it is cost-effective [11][15] and simple/convenient to implement [6] [7]. However, its major setback is designing a model with high efficiency in data processing, face detection, and feature extraction [6] [8].

Anita Jindal et al. [9] researched real-time face and facial landmark points detection using the Haar cascade technique. Image processing and feature extraction were done using Open-Source Computer Vision (OpenCV) library and the Dlib library. Histogram of Oriented Gradients (HOG) detected and tracked the facial landmark points. The Raspberry pi camera captured real-time videos and then frames extracted from the videos in their research. The detected faces from the Haar cascade algorithm were, converted into histograms and then facial features landmarked using facial landmarks predictor. The major findings were that the system was simple and effective, with an accuracy of 98.44% in detecting facial landmark points with or without occlusions on strict frontal faces. However, there were limitations due to the difficulty of detecting subtle changes in features in fatigued faces and missing landmark points for heavy head orientations [9].

N. Kumar et al. [4] designed a system to detect real-time eye blink using the Viola-Jones object detection technique and mouth detection for yawning with the Active Contour

Method. The conducted experiment involved 70 males and 30 female volunteers of different ages and facial characteristics. Participants were categorized as follows: participants with and without glasses, men with and without a beard, men with and without a mustache, women with and without a scarf, different hairstyles, and different clothing. This article assumed that both eyes blinked simultaneously, and the result showed that for eye detection, the proposed system evaluated positive alerts for drivers without eyeglasses with an accuracy of 92% for 18 -25 age drivers in critical time 1. Moreover, for mouth detection, [4] obtained an accuracy of 88% for drivers without mustaches from age 50 to 60 years in critical time 1. Simulation results in [4] showed that the accuracies for the age groups were high in the range of values, 80 – 92.

S. Mehta et al. [10] researched Advanced Driver Drowsiness Detection System (AD3S) using machine learning (ML). The aim was to capture facial landmarks of drivers and calculate Eye Aspect Ratio (EAR), Nose Length Ratio (NLR), and Mouth Opening Ratio (MOR) to detect driver's drowsiness. A dataset of 1200 images was used to train and test the machine learning (ML) classifiers, and thereafter the classifiers extracted facial features. Eye Aspect Ratio (EAR), Nose Length Ratio (NLR), and Mouth Opening Ratio (MOR) were obtained from the facial landmarks. Dlib algorithm captured the facial coordinates of drivers, and then based on the facial coordinates landmarked by the systems: EAR, NLR, and MOR indices were calculated to assess driver drowsiness.

R. Manoharan et al.'s [11] research used an android OpenCV-based monitoring system to check driver distraction and fatigue levels using adaptive template matching and adaptive boosting (Ada boost) and to alert the driver via the android OpenCV application once the drowsiness threshold is exceeded. In this research, captured images were resized, and the Ada boost algorithm extracted features. Haar classifiers from OpenCV detected the face, eye, and mouth. The extracted features were compared to the predefined threshold level for blink rate and yawning to detect drowsiness. The major finding was the system's applicability on android devices. However, the study did not consider the deflected angle of drivers' faces and did not compare the method to other methods.

L. Zhou et al. [12] proposed a system of driver fatigue detection based on machine vision was an improved strategy and practical system designed to detect driver fatigue based on

machine vision and the Ada boost algorithm. Face classifiers are trained using face images from the Massachusetts Institute of Technology (MIT), Yale, and ORL face databases. Ada boost classifier trained from Haar features to detect face and eye and train for open-eyes and closed-eyes classifiers. Fatigue is analyzed with the Percentage of Eye Closure (PERCLOS) index and duration of the closed-eyes state. The approach was to detect faces efficiently using classifiers on both front faces and deflected faces; trained classifiers of opened eyes and closed eyes were used to detect eyes. The index of fatigue was calculated from PERCLOS escalating rate; when the PERCLOS escalating rate increased more than 200%, the driver was considered in a drowsy state. A key finding was that the eye closed-state frequency and duration increased while those of the eye open-state decreased in the drowsiness state. The default face classifier was trained using face images from MIT, Yale, and ORL faces databases. A total of 3000 instances of front faces and 1500 samples of deflected faces were normalized and adopted for training the classifiers. Thereafter, the face classifier was used in detecting target faces. For strict frontal faces, the front face classifier was sufficient for detection, otherwise, the image passed through the deflected face classifiers (both the left and right) as well. Such situations reduced the speed of face detection. Successful face detection moved to the next stage of eye detection. Adaboost classifier was deployed to train the two classes of eye state (eyes-open and eyes-closed) in this system for eye detection. Adaboost is a boosting method that separates relevant features from irrelevant ones by integrating several “weak classifiers” to construct a single “strong classifier”.

R. Jabbar et al. [6] worked on a driver drowsiness detection model using Convolutional Neural Network Techniques for android applications. The aim was to design a real-time driver drowsiness detection system for android devices and embedded systems with high accuracy and ease of use. Dlib extracted facial landmarks from video frames. Drowsiness detection was based on CNN with four layers. The dataset consisted of five (5) different simulated driving scenarios: with glasses, with sunglasses, without glasses, a night with glasses, and a night without glasses. In total, 200 images (30 frames per second) were used and converted to grayscale images. Code box algorithm was used to generate new images from National Tsing Hua University (NHTU) images. The accuracy reported for

the approach was 83.30%. Major findings were an open-source C++ library, Dlib had pre-written functions that could easily extract facial landmarks; microsleeps are short duration where the driver had his eyes closed and could not perceive any visual information thus, drifting of the lane or braking abruptly. A limitation was not evaluating the system in real-time.

Gupta et al. [1] proposed a non-intrusive approach for implementing a driver drowsiness alert system to detect and check the yawning and sleepiness of a driver. An RGB (Red-Green-Blue) camera mounted at the front windows checked the driver's face constantly. Histogram of Oriented Gradients (HOG) and SVM were used for face detection, thereafter, EAR and Mouth Aspect Ratio (MAR) for drowsy recognition. The classifier employed a dynamic frame threshold to compute the aspect ratio of eye and mouth based on facial landmarks that showed drowsiness. The threshold for an eye was 0.15, and that for the mouth was 0.1. The accuracy of the proposed system was 90%. Major findings were HOG used block size of specific dimensions depending on image size with 50% overlap; SVM recognized frames with faces and ignored frames with no faces by analyzing data for classification and regression analysis. The study however did not try to compare the technique to other machine learning approaches.

S. Demidenko et al.'s [13] research targeted designing an eye-tracking system to detect driver drowsiness; the application of the Viola-Jones algorithm and PERCLOS to evaluate the drowsiness index. The level of fatigue was decided from features extracted from real-time videos; then evaluated the drowsiness index against a pre-specified parameter; alerted the driver to high drowsiness index. The accuracy of the system was greater than 95%. The study discovered the possibility of implementing the system on smart devices, making it easily accessible to the public. The limitation was that the research was conducted under only good lighting intensity and face illumination.

The paper by I. Teyeb et al. [3] designed a driver drowsiness detection system using video processing to analyze eye blinking concepts to measure eye closure duration and decide driver vigilance state. The approach was to segment captured videos into frames, detect faces and eyes using the Viola-Jones algorithm; classify eyes by Wavelet Network Classifiers (WNC); then computed eye closure duration. The system implemented gave a

good detection rate with an accuracy of 97.5%. Major findings: No case of false detections due to the Viola-Jones algorithm; correct detections for a driver with or without glasses; greater improvement with the use of a high frame rate camera for blinks characterization. The system's detection was limited to strict frontal faces.

Objective

The primary aims of the project are: (1) Developing a high-fidelity Fatigue, Drowsiness, and Drunk Drivers Detection (FD^4) algorithms. The proposed FD^4 algorithms are capable of accurately detecting and alerting the fatigue level of a driver to prevent any future accident for high-traffic safety and (2) Verify the accuracy of the proposed FD^4 algorithms using standard video footage and comparing its performance to the state-of-the-art fatigue detection algorithms.

The results of this study can have the following long-term impacts (objectives) and benefits:

1. Saving lives, money, and efforts for DOTD, MPOs, and other agencies who are interested in achieving high traffic safety.
2. Providing a better understanding of drivers' behavior distribution, will help in better understanding and classifying the reasons for accidents in Louisiana and especially in the Baton Rouge area.
3. Using a real-time automatic FD^4 system is an important source of information for planners and policymakers when dictating transportation planning, traffic safety, and infrastructure spending.
4. Improve the accuracy of streets and highway safety by providing timely and detailed traffic fatigue, drowsiness, and drunk driver cases information.
5. The results from this project will provide base results for a Federal grant (NSF). The Federal grant will help the PI as a new tenure-track assistant professor, to attract more graduate students to work with him for better research development and career improvement.

Scope

In this study, the research team implemented three methods to detect the fatigue, drowsiness, and drunk situations of drivers. The three implemented methods are: fixed thresholding approach, adaptive thresholding approach, and Machine Learning modeling and classification. The fixed threshold approach involves alerting the driver of drowsiness once the EAR or/and MOR value meets the criterion. An improvement in the detection accuracy is obtained by using the Dlib algorithm for face detection. Dlib shape predictors extract the facial features that aid in calculating essential parameters: the Eye Aspect Ratio (EAR) and Mouth Opening/Aspect Ratio (MOR/MAR). These values are the primary indicators of driver drowsiness. The parameters are evaluated to assess the level of drowsiness/alertness of the driver. The main challenge arises with fixed thresholding is the false alarming of drowsiness, therefore, the second method (the adaptive thresholding) is implemented in this study to reduce the false alarming situation. The third approach uses different ML methods (Support Vector Machine (SVM), Naïve Bayes (NB), and Random Forest (RF)). A dataset of 1448 images was used for training and testing these ML methods: 70% for training and 30% for testing. The metrics used to judge the viability of the classifiers included accuracy, precision, recall, and F-score. The following tasks were followed to achieve the overall scope and objectives of the project:

Task 1: Performing a Wide Study

The research team performs a comprehensive search for the-state-of-the-art fatigue, drowsiness, and drunk drivers' algorithms. Algorithms that solve face detection, eye detection, symptom extraction, yawning detection, and driver drunk state estimation will be considered.

Task 2: Developing a High-Fidelity FD^4 Algorithm

Developing a High-Fidelity FD^4 Algorithms: Three methods were proposed: the method of fixed thresholding, adaptive thresholding, and the method of Machine Learning modeling and classification. The fixed threshold approach involves alerting the driver of drowsiness once the EAR or/and MOR value meets the criterion. The adaptive thresholding is implemented in this study to avoid the false alarming situation results

from the fixed thresholding method. For the third method, three ML classifiers were used: Support Vector Machine (SVM), Naïve Bayes (NB), and Random Forest (RF).

Task 3: Refinement of the Proposed FD^4 Algorithm

Refinement of the Proposed FD^4 Algorithms: All previous algorithms in Task 2 are customized for maximum performance. Anaconda-based python software is used for implementation. A dataset of 1448 images is used for training and testing the implemented ML models: 70% for training and 30% for testing.

Task 4: Data Analysis and Performance Measurements

Data Analysis and Performance Measurements: The performance of the implemented prototypes is measured using different parameters such as precision, recall, F-score, confusion matrix, and loss and accuracy curves.

Task 5: Project Dissemination, Conclusions, and Reporting

The project report is prepared based on the LTRC guidelines. The report will include project results, challenges, dissemination plan, and overall assessment of the degree to which goals are achieved.

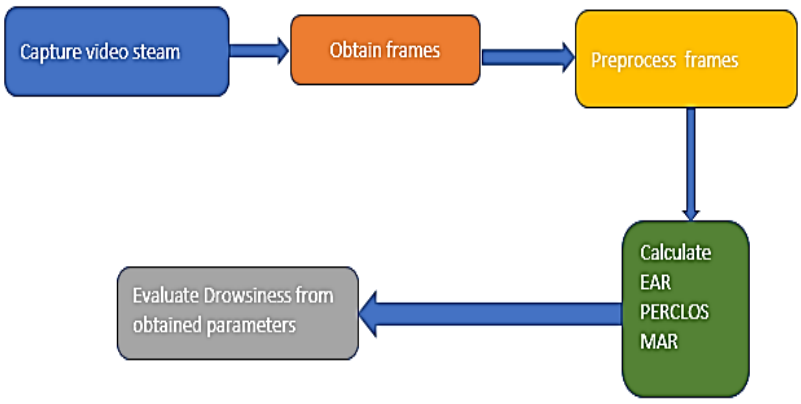
Methodology

1. Fixed Thresholding Algorithm

Proposed system

In the fixed thresholding algorithm, the threshold for checking drowsiness is fixed for both yawning and drooping eyes. The flowchart of the fixed thresholding algorithm as shown in Figure 1. consists of five blocks;(1) video stream, (2) frames, (3) Preprocessing of frames, (4) Calculate EAR, PERCLOS, and MOR, and (5) Evaluate drowsiness.

Figure 1. Flowchart of fixed threshold algorithm



A. Video stream

The first step in the fixed thresholding algorithm is to capture a video of the driver. A camera is often positioned in front or at an angle where the portrait of the driver is captured. Prototype models use the web camera [14] of the computer. However, some cameras can be used for this including the Raspberry Pi camera and the night vision camera [15]. Most times the Raspberry Pi camera [15] works effectively during the day and poorly for night vision. [15] recommend the use of a night vision camera that works for both day and night and even conditions with low illumination and brightness.

In fixed thresholding algorithm, a Logitech HD 1080p web camera was used for video streaming. The frame rate of the camera was 30fps. It is cheap, specifically designed, and optimized for professional-quality video streaming, full HD glass lens, and premium

autofocus. It also records clear videos even in dim or poor backlit settings with automatic light correction. This camera was used on Intel® Core (TM) i3-2328M CPU @ 2.20Ghz, 4.00GB RAM, 64-bit Operating System.

The video stream is processed using Python programming language. Python has the library 'OpenCV' designed for computer vision projects. OpenCV is an open-source library that provides an interface for capturing live streams with the webcam. A 'VideoCapture' object from OpenCV is used to capture the video. cv2.VideoCapture() is used to get a capture object for the camera. Setup an infinite 'while' loop to keep streaming the video until an exit call is started. cv2.imshow() is used to display the video stream while the cv2.imwrite() is used to store frames from the continuous stream.

B. Frame extraction

Machine learning algorithms are designed to work on static frames; they cannot run on continuous streams. After the video is recorded, it is split into frames which can be fed to the algorithm. The number of frames obtained per second (fps) depends on the frame rate of the camera. In our proposed system, the inbuilt image processing functions wrapped in a python package by the name Imutils is deployed for easy manipulation of the stored frames. Imutils enables basic image processing actions including translation, rotation, resizing, and many more to be performed on the stored frames. The webcam has a frame rate of thirty (30) frames per second, but stores one frame every second and then delayed another second before storing the next frame to avoid overwhelming the computer. The 'imutils' resize function' resized the stored frames to a size of 320 x 240.

C. Preprocess of frames

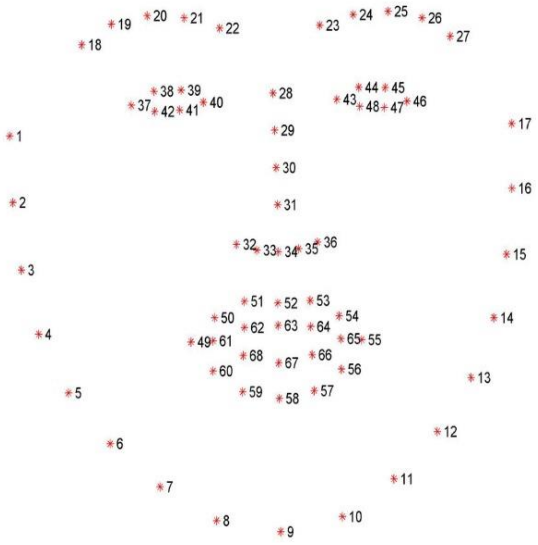
The quality of frames is affected by several factors such as the illumination of the environment, the position of the camera, and noise. Preprocessing is a method of improving/enhancing the quality of frames by removing/minimizing noise, adjusting/regulating the brightness and contrast of light in the frame, and reducing the size of the frame. This is done to increase the detection accuracies. Some of the preprocessing methods include a histogram of equalization [12] to enhance the brightness

and contrast, using image tools to reduce the size, and controlling the illumination level [15]. Extraction of the face with Infrared (IR) illuminators was suggested by R. Manoharan and S. Chandrakala in [11].

D. Estimate Metrics

In this study, Dlib is used to detect drowsiness. There is an open-source C++ Dlib library that was developed to enhance computer vision applications in detecting drowsy states [6]. This library has pre-written functions that find a human face and localize key facial points. Dlib library has a shape predictor, a 68 facial landmark point detector used to estimate the location of 68 coordinates (x, y) that map the facial points on a face.

Figure 2. Facial landmark points **[6] [16]**



In the fixed thresholding algorithm, 68 face landmark shape predictor, an open-source data file was downloaded to mask the landmark point onto the face. Dlib library was used to detect face objects and landmark face region in the frames. The shape predictor finds coordinates/points around the key features extracted. The 68 points are localized around the face.

E. Eye Aspect Ratio – EAR

EAR is a metric that is computed using the coordinates around the eyes. The region around each eye has six (6) points. In figure 10, the left eye has coordinates with values (37 – 42) and the right eye has values (43 – 48). EAR is the proportion of the vertical distance between the upper (coordinates facial landmark points 44 and 45, or 38 and 39) and lower (coordinates 47 and 48, or 41 and 42) eyelids to twice the distance between the horizontal distance at the corners of the eye (coordinates 43 and 46, or 37 and 40). Mathematically computed as:

$$EAR = \frac{Y_2 - Y_1}{2*(X_2 - X_1)} \qquad (1)$$

Where, Y_2 is the coordinates for the upper eyelid, Y_1 is the coordinates for the lower eyelid, and X_2 and X_1 are the coordinates for the corners of the eyes. In our fixed thresholding algorithm, the threshold was set to the value 0.15. When the EAR value exceeds the threshold, a visual message is displayed as being 'DROWSY' else displays 'ALERT'.

Figure 3. Driver fully alert

Error! Reference source not found. shows the EAR values that indicate the person is fully alert. The recorded EAR values are 0.35, 0.36, and 0.45 which all satisfy the drowsiness alert condition.

Figure 4. Driver in a Drowsy state

Error! Reference source not found. shows the EAR values of 0.08 and 0.14 which fall below the threshold of 0.15. As stated earlier, any value below the threshold gets indicated as drowsy.

F. Mouth Opening/Aspect Ratio - MOR/MAR

Yawning is an important factor for drowsy detection. MOR is a metric that is computed using the coordinates around the mouth. The region around the mouth has twenty (20) points. In **Error! Reference source not found.**, the mouth has coordinates with values (49 – 68). The contour of the mouth makes it easy to decide whether a person is yawning

or not by checking the size of the mouth. For all the points on the contour, get the largest and smallest Y-coordinate values, then take the difference between them to get the height of the mouth [4]. The height is divided by the horizontal distance between the ends of the mouth: MOR is the proportion of the vertical distance between the upper (coordinates 50-54) and lower (coordinates 56-60) contours of the mouth to twice or thrice the distance between the horizontal distance at the corners of the mouth (coordinates 43 and 46, or 37 and 40) [14] [17] [10]. Mathematically computed as:

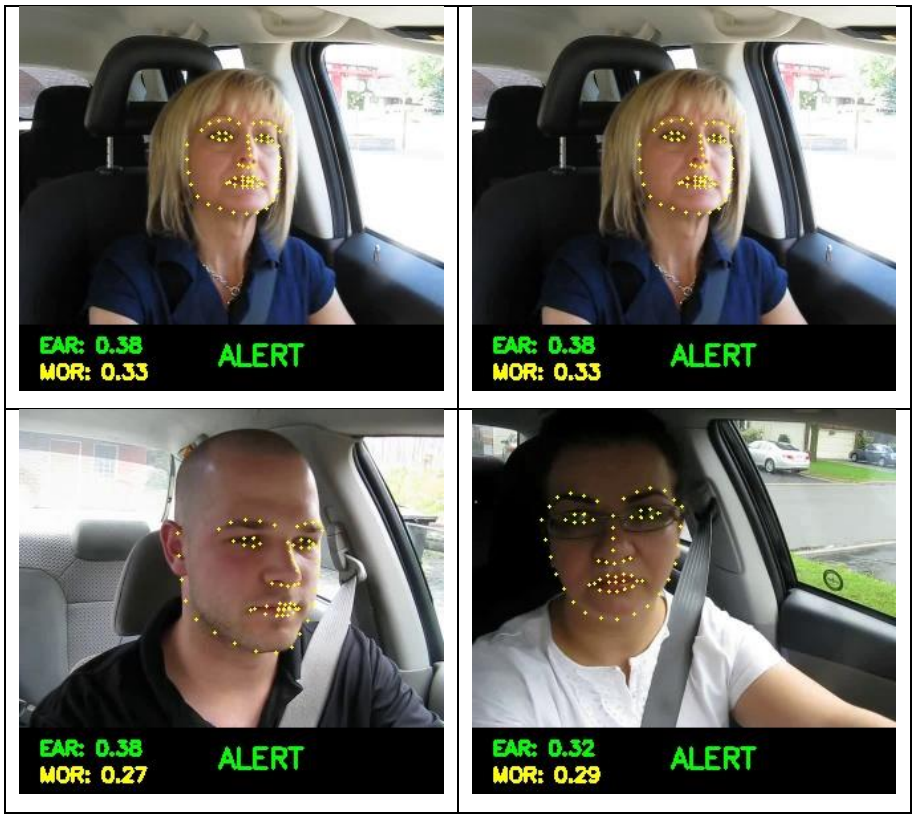
$$MOR = \frac{Z_2 - Z_1}{2*(W_2 - W_1)} \; or \; \frac{Z_2 - Z_1}{3*(W_2 - W_1)} \qquad (2)$$

Where, Z_2 is the coordinates for upper mouth contour, Z_1, is the coordinates for lower mouth contour, W_1 and W_2 are the coordinates for the corners of the mouth. [1] proposed a threshold value of 0.1 for MOR. In our fixed thresholding algorithm, the set threshold was 0.4 as a lower threshold gives a high number of false positives. When the MOR value exceeds the threshold, a visual message is displayed as 'DROWSY' else display 'ALERT'.

Figure 5. Drowsy driver showing different MOR and EAR values

In **Error! Reference source not found.**, MOR values (0.61, 0.73, 0.85, and 1.04) are above the set threshold of 0.4. In the above images, drivers are alerted as being 'DROWSY'.

Figure 6. Alert driver showing different MOR and EAR values



Error! Reference source not found. show MOR values that indicate the alertness of the driver. The MOR values (0.27, 0.29, 0.33) fall below the threshold value of 0.4.

G. Assess Drowsiness

Generally, a person is alerted of drowsiness via visual, audio, or both messages. If MOR or EAR meet the threshold, either states, that is, 'DROWSY' or 'ALERT'.

Figure 7. Drowsy driver with both closed eyes and yawning expressions

Error! Reference source not found. has EAR values below the threshold and MOR values above the threshold. In these images, the drivers' eyes were dim (closed) and yawning simultaneously. In the first image we have the pair of values (EAR: 0.08, MOR: 0.65), the second with values (EAR: 0.14, MOR: 0.57), and the last image (EAR: 0.14, MOR: 0.91).

Fixed Threshold Algorithm - Summary

The fixed threshold algorithm can be summarized as shown in **Error! Reference source not found.** as follows:

a. Capture video stream for real-time detection.
b. Obtain frames from the video.
c. Preprocess frames to reduce processing time by resizing.
d. Dlib pre-trained face detectors and shape predictors were used to detect faces and find landmark points. The eye and mouth regions are landmarked.
e. EAR, MOR are computed using equations (1) and (2).
f. The driver is alerted if any of the two or both metrics meet respective thresholds.

Referring to **Error! Reference source not found.**, a Logitech camera with a frame rate of 30 frames per second can be used to capture real-time videos. Then, the above steps from 1 through 6 will be implemented. The following steps are followed to assess fatigue, drowsiness, and drunk alert situations:

1. Determine Eye Aspect Ratio (EAR)
    a. Initialize a variable, to store the EAR values.
    b. A ‘for’ loop was created to assign values to the coordinates around the eyes.
    c. The coordinates around the eyes were passed into an array using NumPy. The coordinates have values from 37 to 48 and used to compute EAR.
    d. The landmark coordinates are assigned in a dictionary.
    e. EAR is calculated using the equation mentioned earlier in the theory, that is, by estimating the proportion of the maximum vertical distance between the upper (coordinates 44 and 45) and lower (coordinates 47 and 48) eyelids to twice the horizontal distance between the corners of the eye (coordinates 43 and 46).
    f. The threshold for EAR was 0.15. Values below the threshold were categorized as ‘ALERT’ and vice versa.

2. Determine Mouth Open Ratio (MOR)
    a. Initialize a variable, to store the MOR values.
    b. A ‘for’ loop was created to assign values to the coordinates around the mouth.
    c. The coordinates around the mouth were passed into an array using NumPy. The coordinates for the mouth range from 49 to 68.
    d. The landmark coordinates are assigned in a dictionary.
    e. MOR is determined from the equation in theory, that is, by estimating the proportion of the maximum vertical distance between the upper contours (coordinates 51,52 and 53) and lower contours (coordinates 57,58 and 59) of the mouth to thrice the horizontal distance between the corners of the mouth (coordinates 49 and 55).
    f. The MOR threshold setting was 0.4. Below 0.4, label frames as ‘ALERT’ else categorize them under ‘DROWSY’

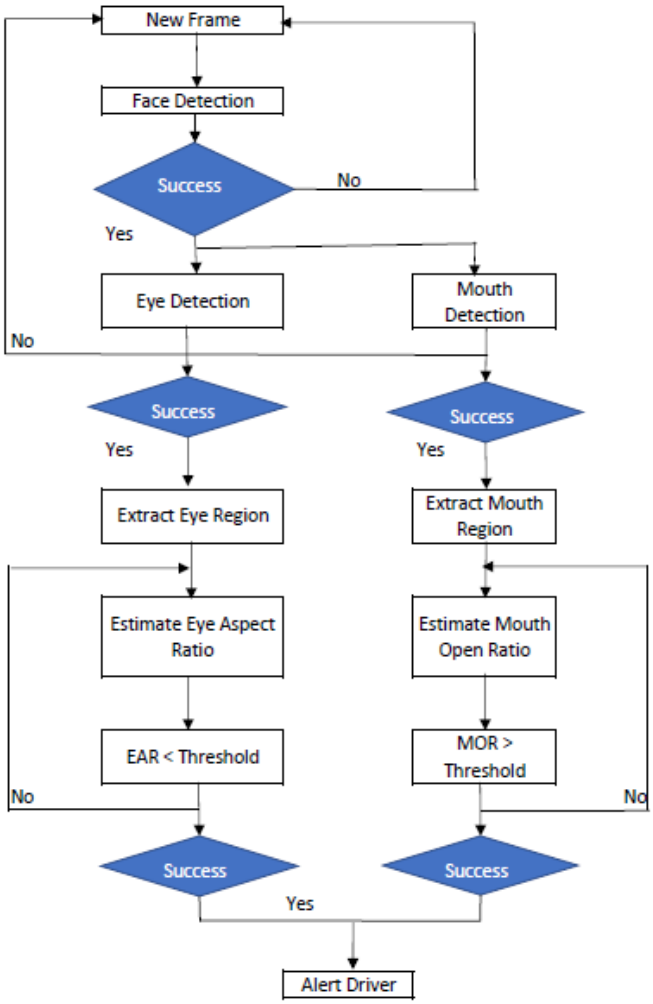
3. Assessing fatigue, drowsiness, and drunk drivers alert situations

   A person is alerted of drowsiness via visual messages if any of the parameters, that is, EAR or MOR or both meet the criteria for drowsiness.

The used Tools, Packages, and Libraries are:

a. Python programming language.
b. Tensor Flow environment in python; Tensor flow is an open-source platform for machine learning.
c. Dlib library consists of the Histogram of Gradients and Linear SVM for face and feature extraction. Dlib is an open-source library for implementing a variety of machine learning algorithms.
d. Imutils for image resizing.
e. NumPy for storing images as arrays for easy manipulation.
f. Matplotlib for data visualization and graphical plotting in Python.
g. Sci-kit learn for predictive data analysis.
h. Logitech HD 1080p web camera.
i. Intel® Core (TM) i3-2328M CPU @ 2.20Ghz, 4.00GB RAM, 64-bit Operating System.

Figure 8. Fixed threshold algorithm



2. Adaptive Thresholding Using Online Video Datasets

Two categories of video datasets were acquired from different articles [18] [19]. The first dataset consists of videos from the University of Texas, Arlington called 'University of Texas, Arlington Real-life Drowsiness Detection' (UTA - RLDD) [18] specifically recorded to assess drowsiness by checking for closed eyes. The videos could not be used for yawning conditions as many of the individual recordings had closed-mouth expressions. Thus, there was a need to get a video that could also check for the yawning conditions. The second dataset is called 'Yawning Detection Dataset' or YawDD [20]. Variation of the Eye Aspect Ratio (EAR) continuous frame threshold was done on the UTA - RLDD datasets. Each video file from that dataset was filmed for ten (10) minutes.

The frame rate for each video varied ranging from 15 fps to 30 fps. In merging the videos as a single stream, the frames per second was set to 15 frames per second. Fifteen (15) frames per second because a comparative analysis was to be conducted between this study and that of Gupta et al [1].

The video was two (2) hours long after merging. So, working with 12 hours [21], by proportion and considering the video data that is 2 hours long, then every 1 hour is proportional to 6 hours in the standard hourly time set by Uber [21]. As stated earlier, because a comparative analysis was to be made with the article by Gupta et al, it was expedient to set the same parameters for the analysis to be viable. Therefore, the time to change frame threshold was set to every three (3) hours [14]. Appropriating the proportion to the 2-hour video, every thirty (30) minutes in the 2-hour video represents/proportional to every 3 hours in 12 hours. Every thirty (30) minutes long video was a merge of three 10-min videos.

For eye closure, the number of consecutive frames considered ranged from 15 frames to 105 frames.

$$0.5\ seconds\ x\ 15\ fps = 7.5\ frames \approx 8\ frames \tag{3}$$

$$3.5\ seconds\ x\ 15\ fps = 52.5\ frames \approx 53\ frames \tag{4}$$

Had a full 12-hour video been used, the number of frames, that is, 53 – 8 frames would have been considered. However, because it was only a 2-hour long video segmented into four sessions, each thirty (30) minutes long, the range will vary from (53/6 = 9) to (8/6 = 2) for the entire duration. Now the frame values from 9 - 2 will determine the continuous frame threshold change after every 30 minutes.

The slope is given as:

$$slope = \frac{\Delta frames}{\Delta time} = \frac{9-2}{0-2} = -3.5 \tag{5}$$

The equation for checking the continuous frame threshold for the eyes is therefore given as:

$$CE = 9 - \frac{3.5(TIME)}{3600} \tag{6}$$

where CE (Consecutive Eye) stands for the continuous frame threshold for eyes and TIME is the number of minutes elapsed (in seconds).

Table 1. Time against CE Threshold

TIME(MIN)	CE VALUE
0 – 30	9
30 – 60	7
60 – 90	6
90 – 120	4
120	2

The dimension for each video file used was 640 x 480 pixels but in storing, the dimension of the captured frames was reduced to 320 x 290 pixels.

In checking continuous frame threshold for the eyes, the algorithm was set such that while the number of seconds elapsed is less than the duration allowed:

If EAR < EAR_Threshold (that is, 0.15), initiate the counter, count until the threshold for the continuous frame is reached before sending message displaying ‘CLOSED EYES!!!’, please see Figure 9.

Figure 9. Closed Eye state

o

If EAR > EAR_threshold (that is, 0.15), keep informing the driver of his alertness by displaying an ‘ALERT’ or ‘AWAKE’ message, please see Figure 10.

Figure 10. Alert message

3. Machine Learning (ML) Methods

Three ML methods are implemented in this study to detect High-Fidelity Fatigue, Drowsiness, and Drunk Drivers Detection (FD4) System: Support Vector Machine (SVM), Naïve Bayes (NB), and Random Forest (RF). Followings are detailed explanation for these methods.

A. SVM Classifier

Explain in here

SVM is one of the early machine learning classification methods dating to the early 1990s. SVM is a generalized form of the ‘maximal margin classifier’. Support Vector Classifier was initially limited to cases with linear boundary. However, improvement has seen its application to a wide range of datasets. Majorly, SVM were intended for binary classification but can now be extended to classifying datasets with many classes. SVM uses the concept of a hyperplane. In n-dimensional space, a hyperplane is a flat affine subspace of dimension (n-1). In two dimensional (2D), the hyperplane is a line whereas

in three dimensional (3D), the hyperplane is a plane. In 2D, a hyperplane is defined by the equation:

$$ax_1 + bx_2 + c = 0 \; or \; ax + b = 0 \qquad (7)$$

If the position of training point lies on this equation, then the point is directly on the hyperplane. However, often, the training observations lies on either side of the hyperplane, which positionally satisfies the equations:

$$ax_1 + bx_2 + c < 0 \; or \; ax_1 + bx_2 + c > 0 \qquad (8)$$

The goal is to construct a hyperplane that separates the training sets perfectly according to their class labels. Then a test data is classified based on the side of the plane it lies. If the data can be perfectly separated using a hyperplane, then there will exist an infinite number of such hyperplanes. To construct a classifier based upon a separating hyperplane, there must be a reasonable way to decide from the infinite possible separating hyperplanes. The obvious choice is to choose the hyperplane which gives the maximum margin away from the training points particularly termed the optimal separating hyperplane. This is done by computing the (perpendicular) distance from each training point to a given separating hyperplane. The smallest distance is the minimal distance of a training point away from the hyperplane known as the margin. We can then classify a test observation based on which side of the maximal margin hyperplane it lies. Support vectors are training points closest to the hyperplane. These support vectors determine how far the margin can be set. In other words, they regulate the magnitude of the margin away from the hyperplane. In cases, we do not have a separating hyperplane, the concept of soft margin is introduced. This concept deals with allowing some of the training data to be on the incorrect side just so the classifier will be good in classifying other observations. This tradeoff is to make the classifier robust to different observations and better classify the training data. This optimization problem is affected by the regularization parameter, C, which is a nonnegative tuning parameter that defines the limit of the margins. When C is small, narrow margins are developed which indicates the classifier is highly fitted to the data. Thus, having a low bias and a high variance with few

support vectors. On the other hand, when C is large, margin is wider and subject to more wrong positioning. There is less hard fitting, high bias and low variance with many support vectors. In non-linear class boundaries, the feature space is enlarged using higher order polynomial functions of the predictors.

B. Random Forest

Random Forest stems from the use of several decision trees. The key advantages of decision trees include: the ease of explanation, and how they closely mirror human decision making. However, a small change in the data can cause a large change in the final estimated tree. Random Forest is an aggregation of many decision trees; Aggregation is a procedure to reduce high variance by totaling the result of several decision trees by majority vote in classification thereby increasing the prediction accuracy. The process involves: making several sub-training sets, building separate prediction models using the training sets, and then averaging the prediction results by majority vote. Mathematically, if the results for the individual prediction models are given as:

$$f^1(x), f^2(x), f^3(x), \dots, f^n(x) \quad (9)$$

The average of these will be given by:

$$f_{avg}(x) = \frac{1}{n}\sum_{n=1}^{n} f^n(x) \quad (10)$$

Due to the difficulty of obtaining large datasets, the concept of bootstrapping aggregation (repeated sampling with replacement) popularly called bagging is utilized. Noteworthy is the parameter, n, that is the number of estimators. A value of hundred (100) is sufficient to achieve good performance. A good parameter to estimate the test error is the Out-of-Bag (OOB) error. It is proven that on average, each bagged tree utilizes two-thirds of the observations. The remaining one-third not used for fitting is termed the Out-of-Bag observations. Predictions can be made for each ith observation in OOB observations and then the majority vote taken for classification. Finally, OOB approach for estimating the

test error is particularly convenient when performing bagging on large datasets for which cross-validation is tough.

C. Naïve Bayes

The Naïve Bayes classes utilizes three principles, that is, the use of probability, Bayes' theorem, and feature independence assumption. Probability is the likelihood of event occurrence. The Naïve uses the conditional probability equation given as:

$$P(A|B) = \frac{P(A,B)}{P(B)} \qquad (11)$$

Where P (A, B) is the intersection between A and B, and P (B) is the probability of B. P(A|B) is referred as the probability of A occurring given that B has already occurred.

The relation between P (A|B) and P (B|A) can be represented through Bayes' theorem which is stated as:

$$P(B|A) = \frac{P(A|B)*P(B)}{P(A)} \qquad (12)$$

Naïve Bayes classifier relates input features to class based on probability. Given a set of features Y = {Y_1, Y2, Y3… Yn} and the goal is to predict the class C, then look for C that gives the highest P (C|Y). It is difficult and complex to run through all features for each class thus the ideal approach is to resort to the next principle that is the Bayes' theorem. It can be defined for this case as:

$$P(C|Y) = \frac{P(Y|C)*P(C)}{P(Y)} \qquad (13)$$

P(Y) is same for all classes as it does not depend on C; it is a constant. The focus is mainly to determine the value for P (Y|C), and P(C) which can be estimated from the data. P(C) obtained from the training data is defined as:

$$P(C) = \frac{Number\ of\ samples\ labeled\ C}{Total\ Number\ of\ samples} \qquad (14)$$

For P (Y|C), we use the independence assumption that describes how the individual features are independent of each other. It is given thus as:

$$P(X_1, X_2, \dots, X_n|C) = P(X_1|C) * P(X_2|C) * \dots * P(X_n|C) \qquad (15)$$

The advantages of this classifier include: it is fast and simple to implement; it scales well, that is do not need several parameters, the feature probability can be calculated in parallel since they are independent of each other. However, the major disadvantages include: the independence assumption may not always hold true, and there are no model interactions between the features. Thus, limits classification power.

Discussion of Results

Performance Metrics

Different metrics have been developed for statistical analysis; these metrics considered include accuracy, precision, recall, and F-score.

Accuracy is often the most widely used performance measure in machine learning, especially for even, unbiased class distribution. It defines the classifier best at evaluating or analyzing and finding the relationship, patterns, and variabilities between parameters in a dataset. Accuracy is often based on training or input data and how the classifier can use the learned features to make better predictions on unseen data. It is given mathematically as:

$$Accuracy = \frac{\sum TP + TN}{\sum TP + FP + TN + FN} \tag{1}$$

Where: *TP* is the True Positive, *TN* is True Negative, *FP* is the False Positive, and *FN* is the False Negative.

Precision measures how the model found correct positive responses ('True Positive') to the total number of positive responses. It is mathematically expressed as:

$$Precision = \frac{\sum TP}{\sum TP + FP} \tag{2}$$

Recall in machine learning is a measure of how correctly the model predicted or found correct positive responses (termed 'True Positive') against the total number of expected correct responses. It is mathematically expressed as:

$$Recall = \frac{\sum TP}{\sum TP + FN} \tag{3}$$

F-score is the weighted average of precision and recall. For even class distribution, accuracy is an ideal measure of performance while for uneven class distribution, an F-score is the best measure of a system's performance. F-score is also another measure of a test's accuracy given mathematically as:

$$F1 = 2 * \frac{precision * recall}{precision + recall} \tag{4}$$

a. Learning Curve and Cross-validation

A learning curve displays an estimator's validation and training scores for various training sample counts. It is a tool to determine how much more training data might help and whether the estimator is more susceptible to bias or variance errors. Each estimator has benefits and disadvantages. Bias and variance can be used to break down the generalization error. An estimator's bias is represented by its average error across many training sets. An estimator's variance reveals how responsive it is to various training sets. Ideally, dataset is grouped into three: training dataset, validation dataset, and testing dataset. Training takes place on the training set, followed by evaluation on the validation set, and when it appears that the experiment has been successful, a final evaluation on the

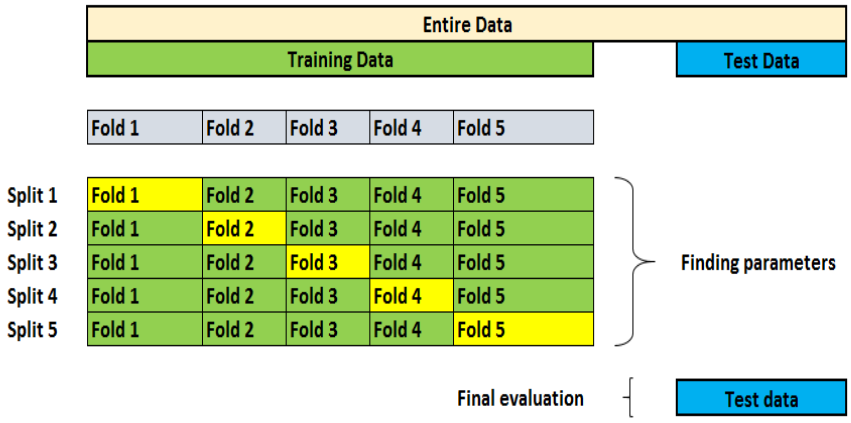
test set may be conducted. However, by dividing the available data into three sets, we dramatically cut down on the number of samples that can be used to train the model, and the outcomes can vary depending on the randomization of the pair of (train, validation) sets.

Cross-validation is a technique that can be used to solve this issue (CV for short). When doing CV, the validation set is no longer required, but a test set should still be kept aside for final assessment. The training set is divided into k smaller sets in the fundamental strategy, known as k-fold CV. For each of the k "folds," the procedure is as follows:

- A model is trained using k-1 of the folds as training data.
- The resulting model is validated on the remaining part of the data (that is, used as a test set to compute performance).

The average of the results calculated in the loop is then the performance indicator supplied by k-fold cross-validation. Although this method can be computationally expensive, it does not waste a lot of data (unlike fixing a random validation set), which is a significant benefit when there are few samples.

Figure 11. k-fold Cross Validation



b. Model Scalability

Scalability describes how the model can learn, that is, the rate it is able to fit the model to a training data size. Often the goal is to have a model which can learn fast without consuming much of memory irrespective of the training data size.

1. Fixed Threshold

Figure 12 shows samples obtained using the fixed thresholding when the driver was drowsy.

Figure 12. Fixed threshold examples

Table 2. Confusion Matrix of fixed thresholding algorithm

N = 1000 test samples	Positive	Negative
True	605	289
False	84	22

Table 2 illustrates the confusion matrix of the fixed thresholding algorithm. The total positive samples, that is, images labeled ‘drowsy’ were 627: Out of that, the classifier correctly predicted (605) as truly ‘drowsy’ and misclassified (22) as ‘awake’. On the other hand, the total negative samples, that is, images labeled as ‘awake/alert’ were (373): The classifier correctly predicted (289) as truly ‘awake/alert’ and misclassified seven (84) as ‘drowsy’. Performance metrics were drawn out of the confusion matrix. Metrics considered were accuracy, and sensitivity.

The sensitivity value of fixed thresholding method is measured as 96.50%, and the accuracy is 89.40%. For real-time detection, our emphasis was more on sensitivity and accuracy. Sensitivity formula determines drowsiness when it is present. It measures how well the classifier was able to identify drivers as truly ‘drowsy’. A value of 96.40% means taking a sample of only one hundred (100) ‘drowsy’ samples, it was able to predict approximately 96 of the images as being ‘drowsy’ correctly and misclassified four of them. Moreover, the accuracy was good because taking a total of one hundred (100) samples consisting of both ‘drowsy’ and ‘alert’ images, the classifier correctly predicted approximately eighty-nine (89.40) out of the (100) as truly as either ‘drowsy’ or ‘awake’.

2. Adaptive Thresholding

Figure 13 shows samples obtained using the adaptive method when the driver’s eyes are closed and when the driver is detected as yawning, respectively.

Figure 13. Closed Eye and Yawning states of a driver

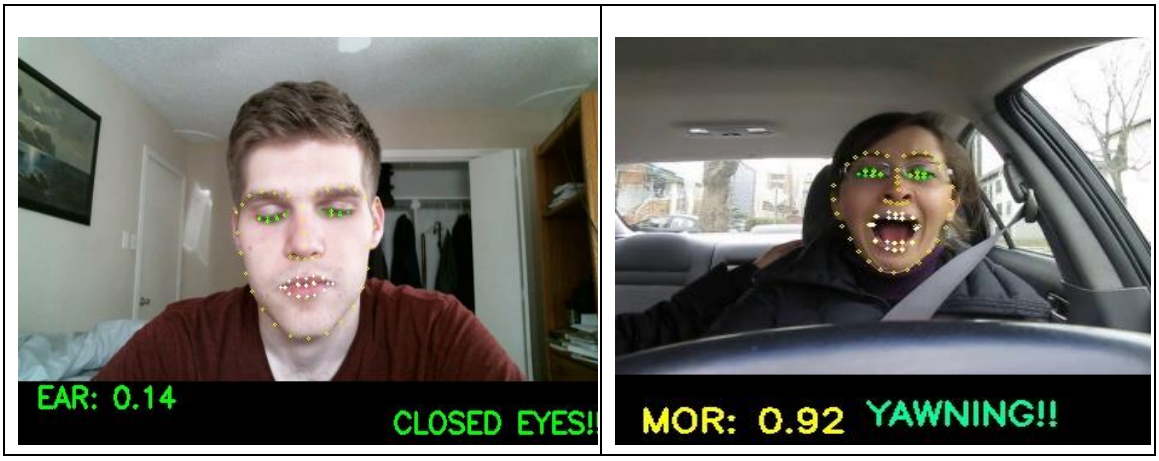


Table 3. Confusion Matrix of adaptive thresholding algorithm

N = 686 test samples	Positive	Negative
True	16	625
False	43	2

From Table 3, the total positive samples, that is, images labeled ‘drowsy’ were 18: Out of that, the classifier correctly predicted (16) as truly ‘drowsy’ and misclassified (2) as ‘awake/alert’. On the other hand, the total negative samples, that is, images labeled as ‘awake/alert’ were (668): The classifier correctly predicted (625) as truly ‘awake’ and misclassified seven (43) as ‘drowsy’. Performance metrics were drawn out of the confusion matrix. Metrics considered were accuracy, and sensitivity based on [22].
The sensitivity and the accuracy obtained by the adaptive threshold algorithm are 89%, and 93.4%, respectively.

3. Machine Learning Modeling and Classification

Three (3) ML classifiers were used: SVM, RF, and Naïve Bayes. First, the models had to be fitted with the training images and labels, and then allowed to make predictions on unseen images and labels (test images and labels).

Figure 14. Learning Curves for Classifiers

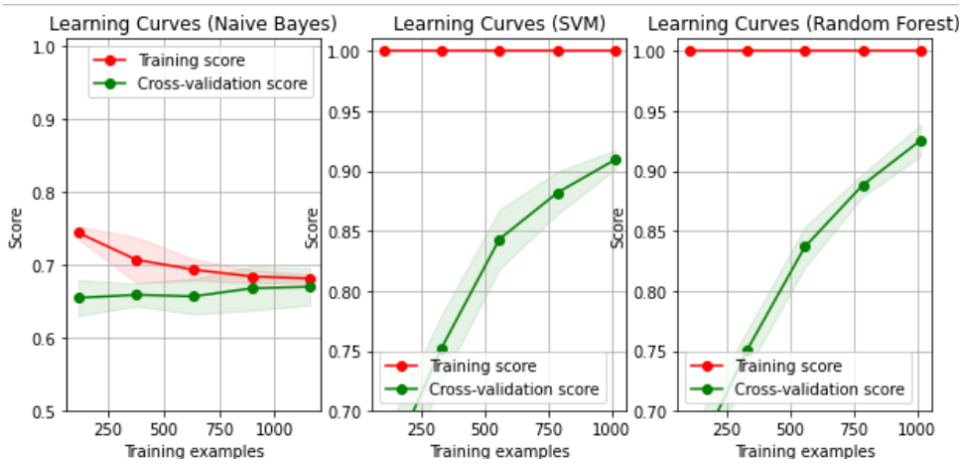


Figure 14 is a plot of the learning curves for the classifiers. For Naïve Bayes, the training score declined as the number of samples increased whiles the cross-validation score was approximately constant. For the SVM, the training score was constant, an accuracy score of 100, while the cross-validation score increased with increase in the number of training examples. For Random Forest, the training score stayed constant as well, an accuracy score of 100, while the cross-validation score increased with increase in the number of training examples.

Figure 15. Scalability of Classifiers

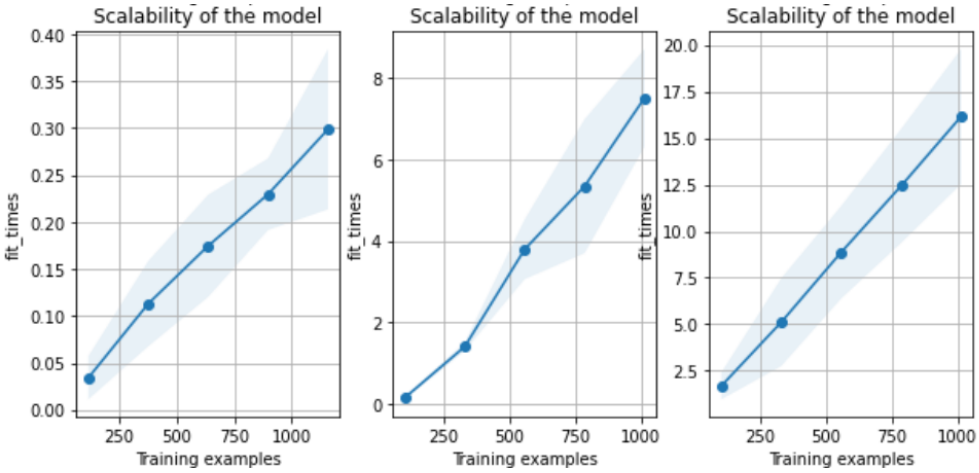


Figure 15 shows the scalability of the classifiers, a plot of the training examples against the fit_times. Naïve bayes took less time in fitting the model to the training samples (few milliseconds), followed by SVM (0 – 7.8 seconds). It took a longer time (2.5 – 16 seconds) for Random Forest to fit all the training samples to the model.

Figure 16. Performance of Classifiers

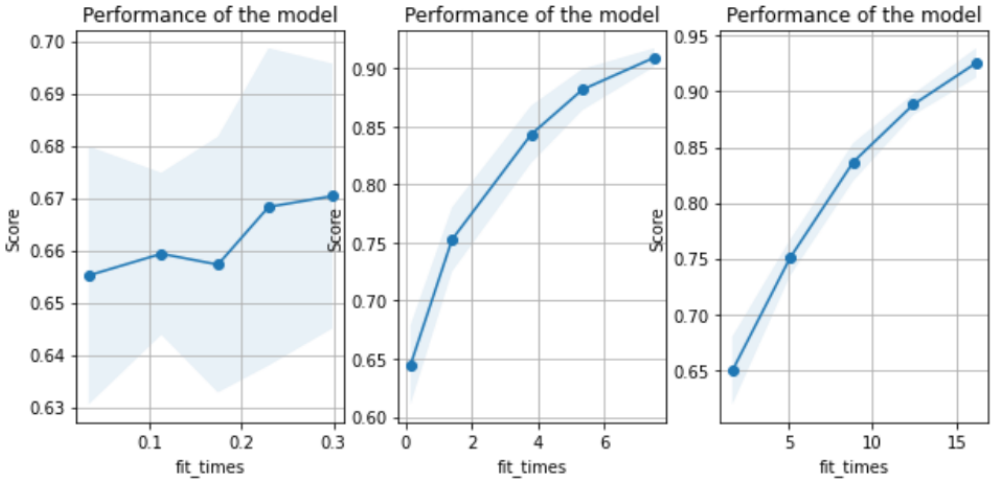


Figure 16 shows the performance of the three models, a plot of the test score against the fit_time. For the Naïve Bayes, the accuracy score increases from a little above 65 (0-100 milliseconds) to 66 and sharply declined between 100 – 200 milliseconds. The accuracy score increased to about 67 at the end of the fit_time (300 milliseconds). For SVM, the accuracy score increased from 65 to a little above 90 during its fit time. Similarly, the accuracy score for RF increased from 65 to above 90 during the fit time.

Table 4 shows the values obtained for three classifiers, that is, Naïve Bayes, Support Vector Machine, and Random Forest after training and testing the raw images on them. From the table, SVM recorded the highest accuracy (90.34), followed by Random Forest (89.86) and lastly Naïve Bayes classifier (65.29). SVM had (87.24) and Random Forest had (85.06) precision values; that of Naïve Bayes was 65.29. Of the three classifiers, the largest recall was by Random Forest and the lowest by Naïve Bayes. In this research, even class distribution was used and hence accuracy was a good measure of performance. However, F-score, a similar and relevant performance metric was obtained. In this experiment, SVM had the highest F-score (90.50) and the lowest (72.19) by Naïve Bayes.

Table 4. Table of Performance Metrics on Different Classifiers

No.	Classifier	Accuracy	Precision	Recall	F-score
1.	Naïve Bayes	65.29	59.39	92.02	72.19
2.	SVM	90.34	87.34	93.89	90.50
3.	Random Forest	89.86	85.06	96.24	90.31

Figure 17. The plot of Accuracy against Classifier

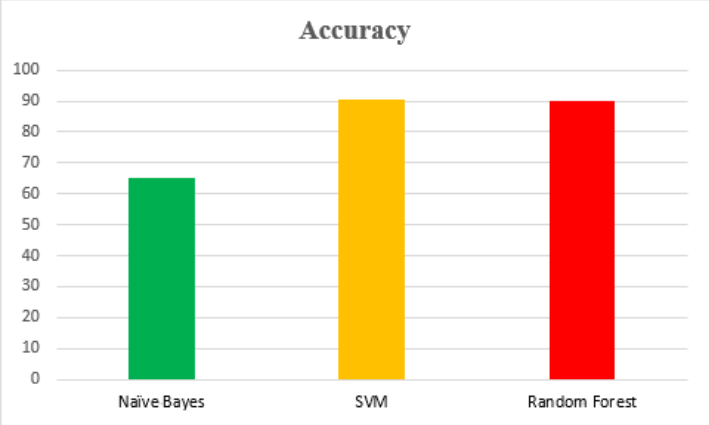


Figure **17** is a histogram showing the accuracy values of the three classifiers. From the plot, it is seen that Naïve Bayes has the lowest accuracy between 60 and 70 while SVM has the highest accuracy slightly better than Random Forest.

Figure 18. The plot of Precision against Classifier

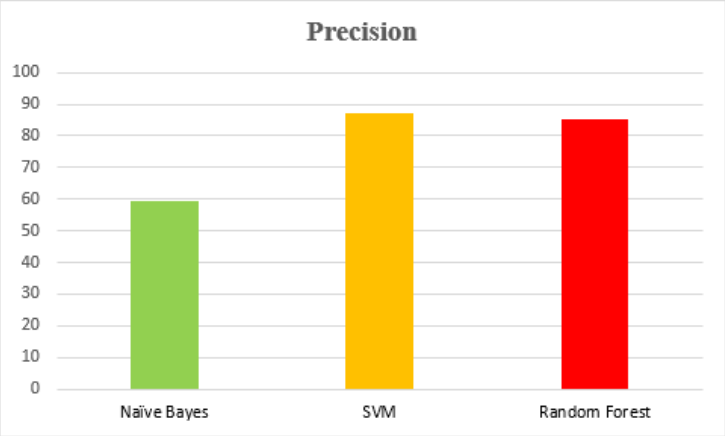


Figure 18 is a histogram showing the precision values of the three classifiers. From the plot, it is seen that Naïve Bayes has the lowest precision; a little below 60 whiles SVM and Random Forest had the highest precision. This means, given 100 test images, SVM and RF correctly predicted above 80 of the positive label/class (here 'Drowsy'). Figure 19 shows the performance metrics of the implemented ML classifiers.

Figure 19. The plot of Performance Metrics of different ML Classifiers

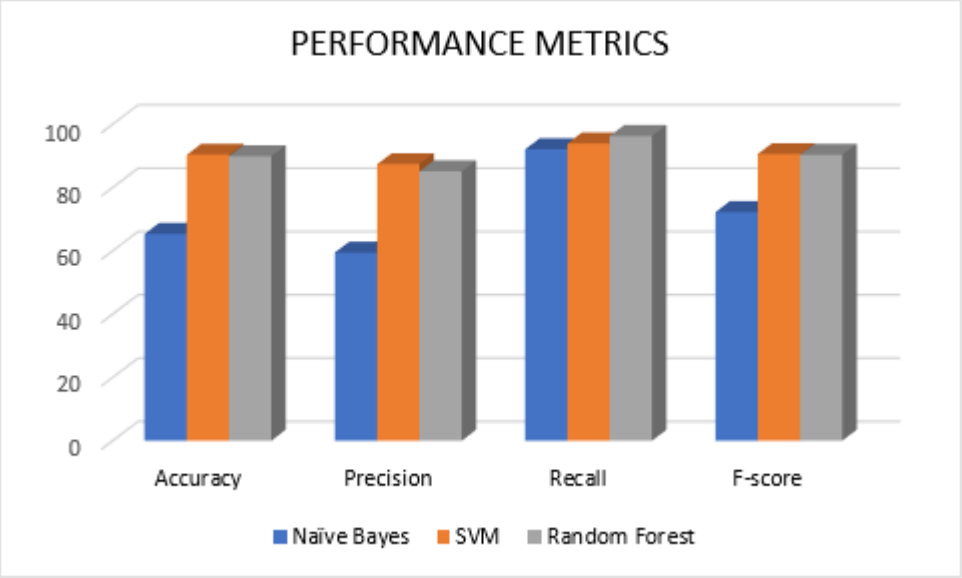


Table 5. Confusion Matrix of Classifiers

No.	Classifier	True Positive	True Negative	False Positive	False Negative
1.	**Naïve Bayes**	88	196	134	17
2.	**SVM**	193	200	29	13
3.	**Random Forest**	186	205	36	8

The confusion matrix describes the prediction success of the classifiers. Moreover, it is from the matrix that we decide the overall performance metrics such as the accuracy, precision, recall, and f-score and thus can conclude on the overall performance of a proposed system. Out of 1448 images, 70%, that is, 1013 images were used for training the model; 30%, that is, 435 images were used for testing.

Table 5 shows the confusion matrix for the three classifiers. A total of 435 images were used for testing. Various terms have been introduced here; we have ‘True Positive,’ ‘True Negative,’ ‘False Positive,’ and ‘False Negative.’

True Positive refers to the number of images with an expected positive label and correctly predicted with a positive label. True Negative also refers to the number of images with an expected negative label and correctly predicted with a negative label. However, False Positive refers to the number of images assigned a positive label rather than the correct expected negative label. Also, False Negative refers to the number of images assigned a negative label rather than the correct expected positive label.

For the Naïve Bayes classifier, 88 images were identified as True Positive, 196 images identified as True Negative, 134 images as False Positive, and 17 predicted as False Negative. For the SVM classifier, 193 images were predicted as True Positive, 200 images predicted as True Negative, 29 False Positives, and 13 False Negative. Finally, 186 predicted True Positive, 205 predicted True Negative, 36 False Positives, and 8 False Negative using the Random Forest classifier.

Figure 20. Histogram showing confusion matrix of classifiers

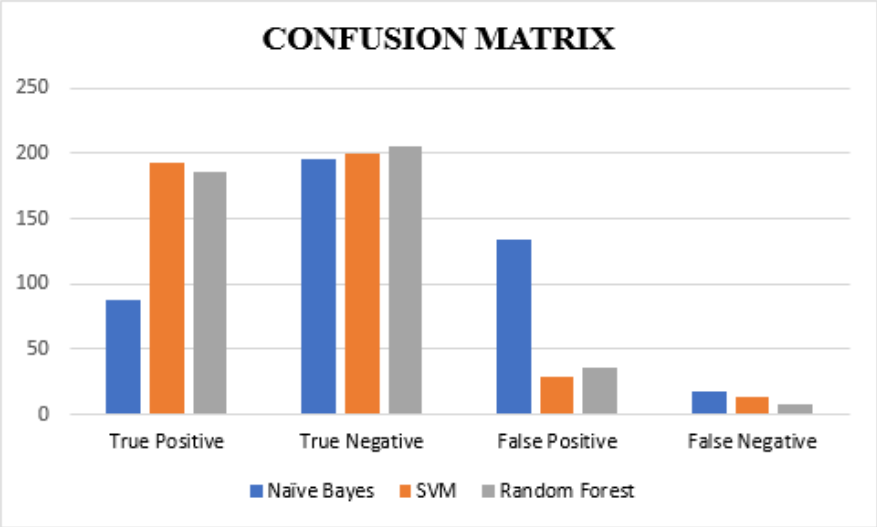


Figure 20 is a graphical representation of the confusion matrix for each of the classifiers. Different measurement parameters were drawn against the other according to the classifiers. According to the plot: SVM has the highest True Positive value; Random Forest has the highest True Negative; Naïve Bayes with the highest False Positives and False Negatives.

Conclusions

In this study, three methods have been proposed in fatigue, drowsiness, and drunk Drivers detection. Fixed Thresholding, adaptive Thresholding, and machine learning classification have been implemented and evaluated. The primary contributions of this study are: (1) A drowsiness detection approach involving two parameters, that is, the Eye Aspect Ratio (EAR), and Mouth Opening Ratio (MOR), (2) Generating equations that can be used to vary continuous frame threshold as time elapses to reduce false triggering, and (3) A Machine Learning approach which involves training three (3) classifiers and testing on unseen data as an alternative way for drowsiness detection. For both fixed Thresholding and adaptive Thresholding, there is no training involved and thus fast to compute. Also, the EAR and MOR continue to vary, hence able to obtain real-time computations. Machine learning classifiers are integrated to also predict the state of drivers with high accuracies. All discussed methods are non-intrusive and cost-effective as no external sensors and detection devices are needed. These algorithms can be embedded in devices as well as online applications for easy detection.

The PI will use the results from this study to submit a Federal grant (NSF). The Federal grant will help the PI to attract more graduate students to work with him for better research development and career improvement. A complete online detection system will be implemented. Different Deep-Learning models will be implemented. The research team will also increase the number of image data that can be used for training and validating the classifiers. Live training and validating sets will be used for more generalization of the implemented models.

Recommendations

The proposed algorithms in this study are: (1) A drowsiness detection approach involving two parameters [i.e., the Eye Aspect Ratio (EAR) and Mouth Opening Ratio (MOR)], (2) Generating equations that can be used to vary continuous frame threshold (adaptive thresholding method) as time elapses to reduce false triggering, and (3) A Machine Learning approach which involves training three (3) classifiers and testing on unseen data as an alternative way for drowsiness detection. The research team will use the results from this study as preliminary results to secure a Federal grant (NSF) which will help the PI to attract more graduate students to work with him for better research development and career improvement.

Acronyms, Abbreviations, and Symbols

Term	Description
NHTSA	National Highway Traffic Safety Administration
EAR	Eye Aspect Ratio
PERCLOS	Percentage of Eye Closure
MOR	Mouth Opening Ratio
MAR	Mouth Aspect Ratio
OpenCV	Open-source Computer Vision
HOG	Histogram Oriented Gradients
SVM	Support Vector Machine
AD3S	Advanced Driver Drowsiness Detection System
ML	Machine Learning
CNN	Convolutional Neural Network
MIT	Massachusetts Institute of Technology
NHTU	National Tsing Hua University
WNC	Wavelet Network Classifiers
TP	True Positive
FP	False Positive
TN	True Negative
FN	False Negative
RF	Random Forest
NB	Naïve Bayes

Term	Description
CV	Cross-Validation

References

[1] I. Gupta, N. Garg, A. Aggarwal, N. Nepalia and a. B. Verma, "Real-Time Driver's Drowsiness Monitoring Based on Dynamically Varying Threshold," *International Conference on Contemporary Computing (IC3),* p. pp. 1–6, Aug. 2018.

[2] V. E. Machaca Arceda, J. P. Cahuana Nina and K. M. & Fernandez Fabian, "A survey on drowsiness detection techniques," *CEUR Workshop Proceedings,* vol. 2747, p. 152–161, 2020.

[3] I. Teyeb, O. Jemai, M. Zaied and C. B. Amar, "A novel approach for drowsy driver detection using head posture estimation and eyes recognition system based on wavelet network," *IISA 2014 - 5th Int. Conf. Information, Intell. Syst. Appl.,* p. 379–384, 2014.

[4] N. Kumar and N. C. Barwar, "Analysis of Real-Time Driver Fatigue Detection Based on Eye and Yawning," *Int. J. Comput. Sci. Inf. Technol.,* vol. 5, no. 4, p. 7821–7826, 2014.

[5] V. E. Machaca Arceda, C. N. J. P. and K. M. Fernandez Fabian, "A survey on drowsiness detection techniques?," *CEUR Workshop Proceedings,* vol. 2747, p. 152–161, 2020.

[6] R. Jabbar, M. Shinoy, M. Kharbeche, K. Al-Khalifa, M. Krichen and K. Barkaoui, "Driver Drowsiness Detection Model Using Convolutional Neural Networks Techniques for Android Application," *IEEE International Conference on Informatics, IoT, and Enabling Technologies (ICIoT),* p. 237–242, 2020.

[7] W. Kong, L. Zhou, Y. Wang, J. Zhang, J. Liu and S. Gao, "A system of driving fatigue detection based on machine vision and its application on smart device," *Journal of Sensors,* 2015.

[8] A. Adouani, W. M. Ben Henia and Z. Lachiri, "Comparison of Haar-like, HOG, and LBP approaches for face detection in video sequences," *16th International Multi-Conference on Systems, Signals, and Devices, SSD 2019,* p. 266–271, 2019.

[9] A. Jindal and R. Priya, "Landmark Points Detection in Case of Human Facial Tracking and Detection," *Int. J. Eng. Adv. Technol.,* vol. 9, no. 2, p. 3769–3776, 2019.

[10] S. Mehta, P. Mishra, A. J. Bhatt and P. Agarwal, "AD3S: Advanced Driver Drowsiness Detection System using Machine Learning," *2019 Fifth International Conference on Image Information Processing (ICIIP),* p. 108–113, 2019.

[11] R. Manoharan and S. Chandrakala, "Android OpenCV based effective driver fatigue and distraction monitoring system," *International Conference on Computing and Communications Technologies (ICCCT),* p. 262–266, 2015.

[12] W. Kong, L. Zhou, Y. Wang, J. Zhang, J. Liu and S. Gao, "A system of driving fatigue detection based on machine vision and its application on smart device," *Journal of Sensors,* 2015.

[13] T. P. Nguyen, M. T. Chew and a. S. Demidenko, "Eye tracking system to detect driver drowsiness," *ICARA 2015 – Proc. 2015 6th Int. Conf. Autom. Robot. Appl.,* p. 472–477, 2015.

[14] A. Kumar and R. Patra, "Driver drowsiness monitoring system using visual behavior and machine learning," *IEEE Symposium on Computer Applications & Industrial Electronics (ISCAIE),* p. 339–344, 2018.

[15] R. A. Bhope, "Computer Vision-based drowsiness detection for motorized vehicles with Web Push Notifications," *4th International Conference on Internet of Things: Smart Innovation and Usages (IoT-SIU),* pp. 1-4, 2019.

[16] C. Sagonas, G. Tzimiropoulos, S. Zafeiriou and M. Pantic, 300 Faces in-the-Wild Challenge: The first facial landmark localization Challenge, Proceedings of IEEE Int'l Conf. on Computer Vision (ICCV-W), 2013.

[17] R. A. Bhope, "Computer Vision-based drowsiness detection for motorized vehicles with Web Push Notifications," *4th International Conference on Internet of Things: Smart Innovation and Usages (IoT-SIU),* pp. 1-4, 2019.

[18] S. Abtahi, M. Omidyeganeh, S. Shirmohammadi and B. Hariri, "YawDD: A Yawning Detection Dataset," *Proc. ACM Multimedia Systems,* pp. 24-28, 2014.

[19] R. Ghoddoosian, M. Galib and V. Athitsos, A realistic dataset and baseline temporal model for early drowsiness detection, IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, 2019.

[20] S. Abtahi, M. Omidyeganeh, S. Shirmohammadi and a. B. Hariri, "YawDD: A Yawning Detection Dataset," *Proc. ACM Multimedia Systems, Singapore,* pp. 24-28, 2014.

[21] I. Source, 2022. [Online]. Available: https://techcrunch.com/2018/02/12/uber-to-require-a-6-hour-break-for-every-12-hours-of-driving-in-the-u-s/.

[22] I. Gupta, N. Garg, A. Aggarwal, N. Nepalia and B. Verma, "Real-Time Driver's Drowsiness Monitoring Based on Dynamically Varying Threshold," *Eleventh International Conference on Contemporary Computing (IC3),* p. 1–6, 2018.

[23] C. Sagonas, G. Tzimiropoulos, S. Zafeiriou and M. Pantic, "300 Faces in-the-Wild Challenge: The first facial landmark localization Challenge," *Proceedings of IEEE Int'l Conf. on Computer Vision (ICCV-W),* 2013.

Appendix

Click or tap here to enter text.