IJNRD.ORG ISSN : 2456-4184
INTERNATIONAL JOURNAL OF NOVEL RESEARCH AND DEVELOPMENT (IJNRD) | IJNRD.ORG
An International Open Access, Peer-reviewed, Refereed Journal

DRIVER DROWSINESS DETECTION SYSTEM

[1]Leelamani Sahu
[2]Domendra Kumar Sahu
[1]B. Tech Student, Department of Computer Science and Engineering, Krishna Engineering College, Bhilai, Chhattisgarh, India
[2]B. Tech Student, Department of Computer Science and Engineering, Krishna Engineering College, Bhilai, Chhattisgarh, India

Abstract: The drowsiness detection system using Python is a software application that monitors a driver's behavior and detects signs of drowsiness or fatigue. This system is built using Python programming language and computer vision techniques. The system uses a camera to capture images of the driver's face and analyze the data to detect signs of fatigue. The drowsiness detection system works by analyzing the driver's eye movements and facial expressions. The system uses computer vision techniques to detect when the driver's eyes are closed for an extended period or when they exhibit drooping eyelids. Additionally, the system also tracks the driver's head movements and monitors their facial expressions to detect any signs of fatigue or drowsiness.

Index Terms – **Driver Drowsiness Detection, Machine learning, Human-Computer interaction, Python, OpenCV, PyCharm, Real-Time face detection**

INTRODUCTION

1.1 Introduction of project

Car accident is the major cause of death in which around 1.3 million people die every year. Majority of these accidents are caused because of distraction or the drowsiness of driver. The countless number of people drives for long distance every day and night on the highway. Drowsiness appears in situations of stress and fatigue in an unexpected and inopportune way, and it may be produced by sleep disorders, certain type of medications, and even, boredom situations, for example, driving for a long time. In this way, drowsiness produces danger situations and increases the probability that an accident occurs.

To prevent such accidents, our team has come up with a solution for this. In this system, a camera is used to record user's visual characteristics. We use face detection and CNN techniques and try to detect the drowsiness of driver, if he/she is drowsy then alarm will be generated. So that the driver will get cautious and take preventive measures. Driver drowsiness detection contributes to the decrease in the number of deaths occurring in traffic accident.

Python, being a versatile programming language, allows for efficient implementation of the system's functionalities. OpenCV, an open-source computer vision library, provides a rich set of tools for image and video processing. It offers functions for capturing video frames, applying image processing techniques, and detecting objects in real-time.

By monitoring the driver's eye state, including eye closure or prolonged periods of inactivity, the system can determine the level of drowsiness. When signs of drowsiness are detected, the system can generate alerts or warnings to the driver, such as audio alarms or visual notifications, to prompt them to take necessary actions and prevent accidents caused by drowsy driving. Overall, the driver drowsiness detection system utilizing Python, OpenCV, and the Haar algorithm combines the power of computer vision techniques with the flexibility of Python programming to enhance road safety by monitoring driver drowsiness and alerting them in critical situations..

1.2 Problem Statement

Traffic accidents due to human errors cause many deaths and injuries around the world. The major cause of these accidents is drowsiness of the driver due to sleeplessness or long driving hours. There is need for a system developed with the technologies that are available today which can overcome this situation. The aim of this system is to reduce the number of accidents by developing a model which can generate an alert if the driver is feeling drowsy so that the driver can become aware and take necessary actions.

1.3 Requirements

- **Software Requirements:**

These are the Software Configurations that are required.

  - Operating System: Linux, Windows, MacOS
  - Language: Python 3.8
  - Libraries: OpenCV, Tkinter, NumPy
  - IDE: PyCharm

- **Hardware Requirements:**

These are the Hardware Configurations that are required.

  - Processor: Any processor above 500 MHz
  - RAM: 4GB and above
  - Hard disk: 40GB and above
  - Input device: Webcam
  - Output device: Monitor

THEORY AND LITRATURE REVIEW

2.1 Theory

The drowsiness detection system is based on the idea that a person's facial expressions and eye movements can be used to detect signs of drowsiness or fatigue. The system uses computer vision techniques and machine learning algorithms to analyze and interpret data from a camera mounted on the driver's dashboard.

The system analyzes various parameters to detect signs of drowsiness, such as eye closure duration, eyelid movement, and head position. The system uses image processing techniques to extract relevant features from the video feed, such as the position of the eyes, eyebrows, and mouth. The system then analyzes these features to detect any signs of fatigue or drowsiness.

One of the critical parameters that the system uses to detect drowsiness is the duration of eye closure. When a person is drowsy, their eyes tend to close for more extended periods. The system uses an eye-tracking algorithm to monitor the driver's eye movements and detect the duration of eye closure. When the system detects that the driver's eyes are closed for an extended period, it triggers an alarm or warning signal to alert the driver to take a break or stop driving.

Another parameter that the system uses to detect drowsiness is eyelid movement. When a person is drowsy, their eyelids tend to droop. The system uses image processing techniques to monitor the driver's eyelid movement and detect any signs of drooping. If the system detects any signs of drooping eyelids, it triggers an alarm or warning signal to alert the driver to take a break or stop driving.

2.2 Research of Studies

2.2.1 Driver Drowsiness Detection System

One of the major causes of traffic accident is Driver ‘s drowsiness. It is a serious highway safety problem. If drivers could be warned before they became too drowsy to drive safely, some of these crashes could be prevented. In order to reliably detect the drowsiness, it depends on the presentation of timely warnings of drowsiness. To date, the effectiveness of drowsiness detection methods has been limited by their failure consider individual differences. Based on the type of data used, drowsiness detection can be conveniently separated into the two categories of intrusive and non-intrusive methods. During the survey, non-intrusive methods detect drowsiness by measuring driving behaviour and sometimes eye features, through which camera-based detection system is the best method and so are useful for real world driving situations. This paper presents the review of existed drowsiness detection techniques that will be used in this system like Circular Hough Transform, FCM, Lab Colour Space etc

2.2.2 Driver Drowsiness Recognition Based on Computer Vision Technology

In June, 2012, A. Cheng et. al. described 'Driver Drowsiness Recognition Based on Computer Vision Technology’. They presented a nonintrusive drowsiness recognition method using eye-tracking and image processing. A robust eye detection algorithm is introduced to address the problems caused by changes in illumination and driver posture. Six measures are calculated with percentage of eyelid closure,

maximum closure duration, blink frequency, average opening level of the eyes, opening velocity of the eyes, and closing velocity of the eyes. These measures are combined using Fisher's linear discriminated functions using a stepwise method to reduce the correlations and extract an independent index. Results with six participants in driving simulator experiments demonstrate the feasibility of this video-based drowsiness recognition method that provided 86% accuracy.

2.2.3 Drowsiness Detection with OpenCV (Using Eye Aspect Ratio)

A real-time algorithm to detect eye blinks in a video sequence from a standard camera is proposed. Recent landmark detectors, trained on in-the wild datasets exhibit excellent robustness against a head orientation with respect to a camera, varying illumination and facial expressions. We show that the landmarks are detected precisely enough to reliably estimate the level of the eye opening. The proposed algorithm therefore estimates the landmark positions, extracts a single scalar quantity – eye aspect ratio (EAR) – characterizing the eye opening in each frame. 6 Many methods have been proposed to automatically detect eye blinks in a video sequence. Several methods are based on motion estimation in the eye region. Typically, the face and eyes are detected by a Viola-Jones type detector. Next, motion in the eye area is estimated from optical flow, by sparse tracking, or by frame-to-frame intensity differencing and adaptive thresholding. Finally, a decision is made whether the eyes are or are not covered by eyelids..

METHODOLOGY

3.1 Existing System

The existing system used Support Vector Machine (SVM) for classifying the face (if the eyes ae closed or not) as drowsy or not drowsy. To separate the two classes of data points, there are many possible hyperplanes that could be chosen. The objective of SVM is to find a plane that has the maximum margin, i.e., the maximum distance between data points of both classes.

3.2 Proposed System

In this system, instead of Support Vector Machine (SVM) we use a Classification Model based on Haar Cascade Algorithm. Deep Learning is concerned with algorithms inspired by the structure and function of the brain called artificial neural networks. Haar Cascade Algorithm are a type of Artificial Neural Networks which are widely used for image classification and even multi-Class classification of images.

3.3 Proposed Techniques

3.3.1 Machine Learning

Machine learning is an application of artificial intelligence (AI) that provides systems the ability to automatically learn and improve from experience without being explicitly programmed. Machine Learning is defined as the study of computer programs that leverage algorithms and statistical models to learn through inference and patterns without being explicitly programmed. Machine Learning field has undergone significant developments in the last decade. Machine learning focuses on the development of computer programs that can access data and use it to learn for themselves. There are also some types of machine learning algorithms that are used in very specific use-cases, but three main methods used today are:

Supervised Machine Learning Algorithm

Unsupervised Machine Learning Algorithm

Reinforcement Machine Learning Algorithm

Among these, the type of machine learning algorithm we used in our system is supervised machine learning algorithm.

- **Supervised Machine Learning**

This can apply what has been learned in the past to new data using labelled examples to predict future events. Starting from the analysis of a known training dataset, the learning algorithm produces an inferred function to make predictions about the output values. The system is able to provide targets for any new input after sufficient training. The learning algorithm can also compare its output with the correct, intended output and find errors in order to modify the model accordingly.

- **Unsupervised Machine Learning**

It holds the advantage of being able to work with unlabeled data. This means that human labor is not required to make the dataset machine-readable, allowing much larger datasets to be worked on by the program. In supervised learning, the labels allow the algorithm to find the exact nature of the relationship between any two data points. However, unsupervised learning does not have labels to work off of, resulting in the creation of hidden structures. Relationships between data points are perceived by the algorithm in an abstract manner, with no input required

from human beings. The creation of these hidden structures is what makes unsupervised learning algorithms versatile. Instead of a defined and set problem statement, unsupervised learning algorithms can adapt to the data by dynamically changing hidden structures. This offers more post deployment development than supervised learning algorithms.

- **Reinforcement Machine Learning**

It directly takes inspiration from how human beings learn from data in their lives. It features an algorithm that improves upon itself and learns from new situations using a trial-and-error method. Favorable outputs are encouraged or 'reinforced', and non-favorable outputs are discouraged or 'punished'. Based on the psychological concept of conditioning, reinforcement learning works by putting the algorithm in a work environment with an interpreter and a reward system. In every iteration of the algorithm, the output result is given to the interpreter, which decides whether the outcome is favorable or not. In case of the program finding the correct solution, the interpreter reinforces the solution by providing a reward to the algorithm. If the outcome is not favorable, the algorithm is forced to reiterate until it finds a better result. In most cases, the reward system is directly tied to the effectiveness of the result. In typical reinforcement learning use-cases, such as finding the shortest route between two points on a map, the solution is not an absolute value. Instead, it takes on a score of effectiveness, expressed in a percentage value. Thus, the program is trained to give the best possible solution for the best possible reward.

3.3.2 Python

Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. It's high-level built in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components together. Python's 18 simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance. Python supports modules and packages, which encourages program modularity and code reuse. The Python interpreter and the extensive standard library are available in source or binary form without charge for all major platforms, and can be freely distributed.

Since there is no compilation step, the edit-test-debug cycle is incredibly fast. Debugging Python programs is easy: a bug or bad input will never cause a segmentation fault. Instead, when the interpreter discovers an error, it raises an exception. When the program doesn't catch the exception, the interpreter prints a stack trace. A source level debugger allows inspection of local and global variables, evaluation of arbitrary expressions, setting breakpoints, stepping through the code a line at a time, and so on. The debugger is written in Python itself, testifying to Python's introspective power. On the other hand, often the quickest way to debug a program is to add a few print statements to the source: the fast edit-test-debug cycle makes this simple approach very effective. It's easy to learn syntax and portability capability makes it popular these days

3.3.3 PyCharm

PyCharm is a hybrid platform developed by JetBrains as an IDE for Python. It is commonly used for Python application development. Some of the unicorn organizations such as Twitter, Facebook, Amazon, and Pinterest use PyCharm as their Python IDE!.

We can run PyCharm on Windows, Linux, or Mac OS. Additionally, it contains modules and packages that help programmers develop software using Python in less time and with minimal effort. Further, it can also be customized according to the requirements of developers

3.4 LIBRARIES USED

3.4.1 OpenCV (Open-Source Computer Vision Library)

OpenCV (Open-Source Computer Vision Library) is an open-source computer vision and machine learning software library. It provides a comprehensive set of tools and functions that enable developers to process, analyze, and manipulate visual data, such as images and videos.

OpenCV supports a wide range of programming languages, including Python, C++, Java, and more. It has bindings for these languages, allowing developers to utilize its functionality within their preferred programming environment.

OpenCV is widely used in various domains, including robotics, augmented reality, autonomous vehicles, surveillance systems, medical

imaging, and more. It offers a vast collection of functions and algorithms, making it a powerful tool for computer vision tasks.

3.4.2 NumPy

NumPy stands for Numerical Python. NumPy was created in 2005 by Travis Oliphant. NumPy is the fundamental package for scientific computing in Python. It is a Python library that provides a multidimensional array object, various derived objects (such as masked arrays and matrices), and an assortment of routines for fast operations on arrays, including mathematical, logical, shape manipulation, sorting, selecting, I/O, discrete Fourier transforms, basic linear algebra, basic statistical operations, random simulation and much more. At the core of the NumPy package, is the ndarray object. It provides a lot of supporting functions that make working with ndarray very easy. Arrays are very frequently used in data science, where speed and resources are very important. NumPy arrays are stored at one continuous place in memory unlike lists, so processes can access and manipulate them very efficiently. This behaviour is called locality of reference in computer science. This is the main reason why NumPy is faster than lists. It is optimized to work with latest CPU architectures. NumPy is a Python library and is written partially in Python, but most of the parts that require fast computation are written in C or C++.

3.4.3 Tkinter

Tkinter is a standard Python library used for creating graphical user interfaces (GUIs). It provides a set of modules and classes that allow developers to build interactive and visually appealing applications. Tkinter is based on the Tk GUI toolkit, which originated from the Tcl programming language.

Tkinter is known for its simplicity, ease of use, and integration with Python. It provides a convenient way to create desktop applications with graphical interfaces, ranging from simple scripts to complex software applications.

3.5 Flow Chart of System

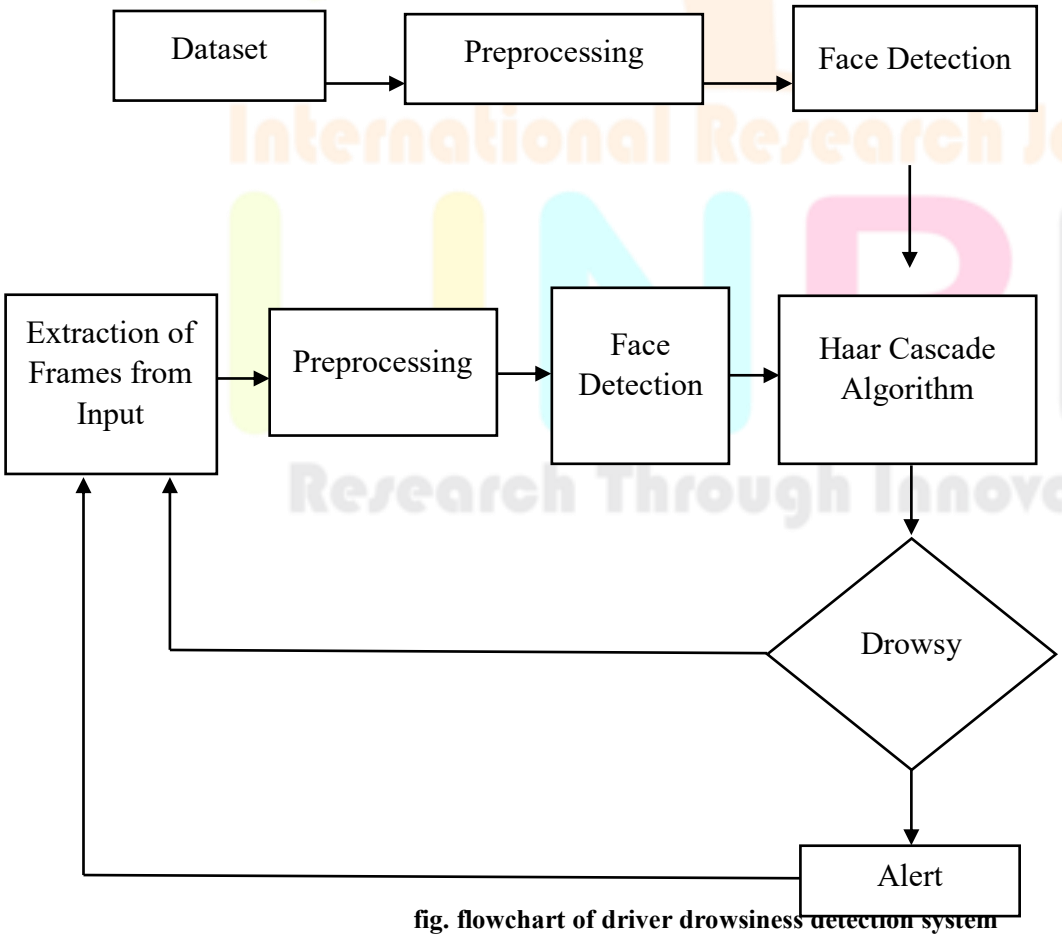

fig. flowchart of driver drowsiness detection system

The system utilizes the Tkinter library to create a graphical user interface (GUI) for the application. The GUI consists of a main window with various buttons for different functionalities.

The system imports the necessary libraries such as NumPy, pygame. mixer, time, cv2 (OpenCV), and Tkinter.
The GUI is created using the Tkinter library, including a window with a title and a frame to hold the components.
Menu options are added to the GUI, including "Tools" and "About" menus with relevant commands.
The system defines functions for different actions:
web(): Opens the webcam feed using OpenCV and displays the video stream in a window.

webrec(): Opens the webcam feed and records the video stream to a file.

webdet(): Detects faces and eyes in the webcam feed using Haar cascade classifiers and highlights them with rectangles.

webdetRec(): Detects faces and eyes in the webcam feed, records the video stream, and highlights the detected regions.

alert(): Plays an alert sound using pygame.mixer to notify the driver.

blink(): Detects eye blinks in the webcam feed using Haar cascade classifiers and alerts the driver.

Buttons are created in the GUI to trigger the respective functions when clicked.
The system enters the main event loop using root.mainloop() to handle user interactions and keep the GUI responsive.
The code provides several functionalities for driver drowsiness detection, including live webcam feed, video recording, face and eye detection, eye blink detection, and alerting the driver when a blink is detected.

RESULT

In this drowsiness detection system we learn how to build a machine learning based an alert system and this is a valuable knowledge we gathered in this project this project has some important feature and also has some limitation.

Driver drowsiness detection can provide following features:

Open Camera: The code allows users to open the camera feed, which enables real-time monitoring of the driver's face.

Record Video: It provides an option to record the camera feed as a video file. This feature can be useful for capturing instances of drowsiness or for further analysis and review.

Detect Faces: The code utilizes the Haar cascade algorithm to detect and locate faces in the camera feed. It marks the detected faces with rectangles for visual identification.

Detect Eye Blink: Using the Haar cascade algorithm for eyes, the code can identify and track the driver's eyes. This feature allows for detecting eye blinks, which are indicative of drowsiness.

Sound Alert: When an eye blink is detected, the code triggers a sound alert using the mixer module from Pygame. The sound alert aims to notify the driver of their drowsiness and potentially prevent accidents.

Cascade Classification: The code employs Haar cascade classifiers for face and eye detection. These pre-trained classifiers can accurately identify specific patterns in the camera feed, enabling efficient detection of faces and eyes.

Graphical User Interface: The code presents a user-friendly GUI using the Tkinter library. The GUI allows users to interact with the system easily and access various functionalities.

Overall, this code provides a foundation for a driver drowsiness detection system. By detecting faces, tracking eyes, and analyzing eye blinks, it aims to identify signs of drowsiness in real-time. The inclusion of the GUI and the ability to record videos enhance usability and facilitate further analysis and research. However, the performance and accuracy of the system would depend on the quality of the Haar cascade classifiers, the training data, and the implementation details, which would require further evaluation and testing.

Limitation of our sign language detection system are followings:

Limited Drowsiness Detection Methods: The code primarily relies on Haar cascade classifiers for face, eye, and blink detection. While Haar cascades can be effective, they may not capture more subtle signs of drowsiness, such as changes in head position or eye movement patterns. Therefore, the code's ability to accurately detect drowsiness may be limited.

Single-User Detection: The code does not support multi-user detection, as it focuses on detecting drowsiness from the webcam feed of a single user. In scenarios where multiple individuals are present, the code may not be able to differentiate between different users or track their drowsiness independently

Limited Training and Customization: The code does not include provisions for training a custom drowsiness detection model. It relies on pre-trained Haar cascade classifiers, which may not be optimized for specific environments or individuals. Customization options, such as retraining the model with new data, are not available.

Performance and Real-Time Constraints: Depending on the hardware capabilities of the system, the code's performance may vary. Running multiple detection algorithms in real-time can be computationally intensive, leading to potential latency or reduced frame rates in the webcam feed. This limitation may impact the system's bility to provide timely and accurate drowsiness detection.

CONCLUSION

Driver drowsiness detection systems are an important tool for enhancing road safety. These systems use various sensors and algorithms to detect signs of drowsiness in drivers, such as changes in facial expression, eye movements, and vehicle behavior. Research has shown that drowsy driving can be as dangerous as driving under the influence of alcohol or drugs. Driver drowsiness detection systems can alert drivers to take a break or other actions to prevent accidents caused by drowsiness.

Driver drowsiness detection systems are a valuable technology that can enhance road safety by alerting drivers to take action to prevent accidents caused by drowsiness. However, they should be used in conjunction with other strategies, such as sufficient rest and regular breaks, to promote safe driving habits.

Road Safety Improvement : Drowsy driving is a significant cause of accidents worldwide. By detecting and alerting drivers when they are drowsy, the system helps prevent accidents and potentially saves lives.

Early Warning System : The drowsiness detection system utilizes various sensors and algorithms to identify signs of driver fatigue, such as eyelid movements, head positioning, and steering patterns. It provides an early warning to drivers, allowing them to take necessary actions and avoid potential dangers.

Real-Time Monitoring : The system continuously monitors the driver's behavior and can detect changes in drowsiness levels over time. This real-time monitoring allows for proactive intervention and provides valuable data for analysis and research purposes.

Driver Awareness : The implementation of a drowsiness detection system raises awareness among drivers about the risks and consequences of driving while fatigued. It encourages responsible driving habits and emphasizes the importance of taking breaks and getting sufficient rest.

FUTURE SCOPE

- The system can be made more accurate using various other parameters such as State of the Car, Detecting Foreign Substances on Face etc.
- An application can be developed where it can alert or prevent the user from sleeping.
- It can be used to develop an IOT device that can be installed in the car to detect driver’s drowsiness.
- Similar models and techniques can be used for various other uses such as Netflix, Hotstar and other streaming service platforms can detect whether the person is sleeping and stop the video accordingly.
- Similar it will use in online classes to notify teacher.

REFERENCE

[1] Armingol, J.M., de la Escalera, A., Hilario, C., Collado, J., Carrasco, J., Flores, M., Pastor, J.Rodriguez, F.: IVVI: intelligent vehicle based on visual information. Robot. Auton. Syst. 55,904–916 (2007). doi:10.1016/j.robot.2007.09.004..

[2] Mahek Jain, Bhavya Bhagerathi, Sowmyarani C N,“Driver Drowsiness Detection System Using Computer Vision”, International Journal of Engineering and Advanced Technology (IJEAT), ISSN: 2249-8958 (Online), Volume – 11 Issue – 1, October 2021.

[3] Chris Schwarz, John Gaspar, Thomas Miller, RezaYousefian, “The Detection of Drowsiness using a DriverMonitoring System (2019)”, Traffic Injury Prevention, 2019, Volume 20, No. SUP1, S157-S161.

[4] https://github.com/opencv/opencv/tree/master/data/haarcascades

[5] https://opencv.org/