Drowsiness Detection Using Haar and CNN Algorithm

Submitted for partial fulfillment of the requirements
for the award of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE & ENGINEERING

by

M.SAKETH - 19BQ1A05B7

K. SAITEJA - 19BQ1A0577

K. SUMASRI - 19BQ1A0575

K. SHRAVANI - 19BQ1A0587

Under the guidance of

Mr.P. R. Krishna Prasad

Associate Professor

VASIREDDY VENKATADRI
INSTITUTE OF TECHNOLOGY

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

(B. Tech Program is Accredited by NBA)

VASIREDDY VENKATADRI INSTITUTE OF TECHNOLOGY

Permanently Affiliated to JNTU Kakinada, Approved by AICTE

Accredited by NAAC with ‘A’ Grade, ISO 9001:2008 Certified

NAMBUR (V), PEDAKAKANI (M), GUNTUR – 522 508

Tel no: 0863-2118036, url: www.vvitguntur.com

April 2023

DECLARATION

We, Mr. M. Saketh, Mr. K. Sai Teja, Mrs. K. Suma Sri, Mrs. K. Shravani, hereby declare that the Project Report entitled **"Drowsiness Detection Using Haar and CNN Algorithm"** done by us under the guidance of Mr. P. R. Krishna Prasad, Associate Professor, CSE at Vasireddy Venkatadri Institute of Technology is submitted for partial fulfillment of the requirements for the award of Bachelor of Technology in Computer Science & Engineering. The results embodied in this report have not been submitted to any other University for the award of any degree.

DATE :

PLACE :

SIGNATURE OF THE CANDIDATE (S)

ACKNOWLEDGEMENT

We take this opportunity to express my deepest gratitude and appreciation to all those people who made this project work easier with words of encouragement, motivation, discipline, and faith by offering different places to look to expand my ideas and helped me towards the successful completion of this project work.

First and foremost, we express my deep gratitude to **Mr. Vasireddy Vidya Sagar**, Chairman, Vasireddy Venkatadri Institute of Technology for providing necessary facilities throughout the B.Tech programme.

We express my sincere thanks to **Dr. Y. Mallikarjuna Reddy**, Principal, Vasireddy Venkatadri Institute of Technology for his constant support and cooperation throughout the B.Tech programme.

We express my sincere gratitude to **Dr. V. Ramachandran**, Professor & HOD, Computer Science & Engineering, Vasireddy Venkatadri Institute of Technology for his constant encouragement, motivation and faith by offering different places to look to expand my ideas.

We would like to express my sincere gratefulness to our Guide **Mr. P. R. Krishna Prasad**, Associate Professor, CSE for his insightful advice, motivating suggestions, invaluable guidance, help and support in successful completion of this project.

We would like to express our sincere heartfelt thanks to our Project Coordinator **Mr. P. R. Krishna Prasad**, Associate Professor, CSE for his valuable advices, motivating suggestions, moral support, help and coordination among us in successful completion of this project.

We would like to take this opportunity to express my thanks to the **Teaching and Non-Teaching** Staff in the Department of Computer Science & Engineering, VVIT for their invaluable help and support.

Name (s) of Students

M.SAKETH(19BQ1A05B7)
K.SAI TEJA(19BQ1A0577)
K.SUMASRI(19BQ1A0575)
K.SHRAVANI(19BQ1A0587)

VASIREDDY VENKATADRI INSTITUTE OF TECHNOLOGY

VASIREDDY VENKATADRI INSTITUTE OF TECHNOLOGY

Permanently Affiliated to JNTU Kakinada, Approved by AICTE

Accredited by NAAC with 'A' Grade, ISO 9001:2008 Certified

Nambur, Pedakakani (M), Guntur (Dt) - 522508

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

B.Tech Program is Accredited by NBA

CERTIFICATE

This is to certify that this Socially Relevant Project Report is the bonafide work of **M. SAKETH, K.SAITEJA, K.SUMASRI, K.SHRAVANI** bearing **Reg.Nos 19BQ1A05B7, 19BQ1A0577, 19BQ1A0575, 19BQ1A0587** who had carried out the project entitled "**Drowsiness Detection Using Haar and CNN Algorithm**" under our supervision.

Project Guide	**Head of the Department**
(Mr. P. R. Krishna Prasad, Associate Professor)	(Dr. V. Ramachandran, Professor)

Submitted for Viva voce Examination held on ____________________

Internal Examiner	**External Examiner**

TABLE OF CONTENTS

LIST OF FIGURES

NOMENCLATURE

ML	Machine Learning
DL	Deep Learning
RTP	Real Time Processing
CNN	Convolutional Neural Network
ANN	Artificial Neural Network

ABSTRACT

The objective of this project is to create a drowsiness detection system that can recognize when someone's eyes are closed for a brief period of time. When sleepiness is detected, this system will give the user a warning. When someone is falling asleep, an alarm buzzes to wake them up. Making the model platform independent, computationally efficient, and affordable for the low-end spec platform is the main goal of this project. Furthermore, to boost the detection's face-sensing accuracy, a mixture of two improved algorithms is applied. The existing system occasionally generates false positive results, which results in erroneous drowsiness detections. These systems might not function properly in various lighting scenarios or with different facial expressions. The proposed system is made with the intention of reducing accident rates and advancing technology in order to reduce the number of deaths and injuries brought on by traffic accidents.

■ -

CHAPTER 1

INTRODUCTION

1.1. . What is Drowsiness Detection?

One of the most prevalent causes of accidents is driver sleepiness and weariness. Each year, the deaths of individuals in such incidents rises across the world. The intent of this research is to reduce the frequency of accidents caused by driver sleepiness and exhaustion. As a result, transportation safety will improve. Driver drowsiness detection is a technology that can help avoid accidents and save drivers' lives when they become drowsy. This research use computer vision to identify driver drowsiness. This research focuses on creating an efficient and cost-effective drowsiness detection system. The approach required in the current circumstance identifies tiredness using geometric aspects of the eyes and lips. This research aims to accomplish the same goal by constructing a sleepiness detection system to monitor and avoid a negative consequence from tiredness neglect.

1.2.What is Object Detection?

Object detection is a computer vision technique used to identify and locate objects within an image or a video. It involves detecting the presence of an object in an image or a video and drawing a bounding box around it to indicate its location.

Object detection algorithms use machine learning techniques, including deep learning and computer vision, to analyze an image or video frame by frame and identify the objects within it. These algorithms typically involve two main stages: object localization and object classification.

Object localization involves identifying the location of an object within an image or a video by drawing a bounding box around it. This is typically done using techniques such as sliding windows, selective search, or region proposal networks.

Object classification involves identifying the type of object within the bounding box. This is typically done using machine learning algorithms such as convolutional neural networks (CNNs) or support vector machines (SVMs).

Object detection has many applications, including surveillance, self-driving cars, robotics, and medical imaging. By accurately identifying and locating objects within an image or video, object detection algorithms can enable a wide range of automated tasks, from tracking moving objects to recognizing specific individuals or objects within a scene.

1.3.Object Detection Models

There are several object detection models that can be used for drowsiness detection, including:

Haar Cascade Classifier: This is a classical object detection model that uses machine learning algorithms to detect objects in images or videos. It can be trained to detect certain facial features associated with drowsiness, such as droopy eyelids or changes in eye shape.

YOLO (You Only Look Once): This is a real-time object detection system that can detect objects in images and videos with high accuracy and speed. It can be trained to detect drowsiness by analyzing facial features and changes in eye movement.

Faster R-CNN: This is a popular object detection model that can detect objects in images and videos with high accuracy. It uses a region proposal network (RPN) to identify regions of interest and then performs classification and bounding box regression on those regions. It can be trained to detect drowsiness by analyzing facial features and eye movement.

SSD (Single Shot MultiBox Detector): This is another real-time object detection model that can detect objects in images and videos with high accuracy and speed. It can be trained to detect drowsiness by analyzing facial features and eye movement.

RetinaNet: This is an object detection model that is designed to address the problem of imbalanced data in object detection tasks. It can be trained to detect drowsiness by analyzing facial features and eye movement.

These object detection models can be trained using labeled datasets of images and videos of drowsy and non-drowsy individuals, allowing them to accurately detect drowsiness in real-time.

1.4.General Applications of Drowsiness Detection

Drowsiness detection has several applications in various industries, including:

Transportation: Drowsiness detection can be used to alert drivers, pilots, and train operators when they are becoming drowsy, thereby preventing accidents caused by fatigue.

Healthcare: Drowsiness detection can be used to monitor patients in hospitals or nursing homes, ensuring they are not becoming excessively drowsy or sedated, which can be harmful to their health.

Workplace safety: Drowsiness detection can be used in workplaces, such as factories or construction sites, to ensure that employees are alert and not at risk of accidents caused by fatigue.

Sports: Drowsiness detection can be used in sports to monitor athletes' fatigue levels and prevent injuries caused by fatigue.

Gaming: Drowsiness detection can be used in gaming to prevent players from becoming excessively fatigued and potentially developing health problems.

Education: Drowsiness detection can be used in educational settings to monitor students' attention levels and ensure they are alert and engaged in the learning process.

But in this project, we focused mainly on driver drowsiness detection and it can be implemented as shown in the below figure.

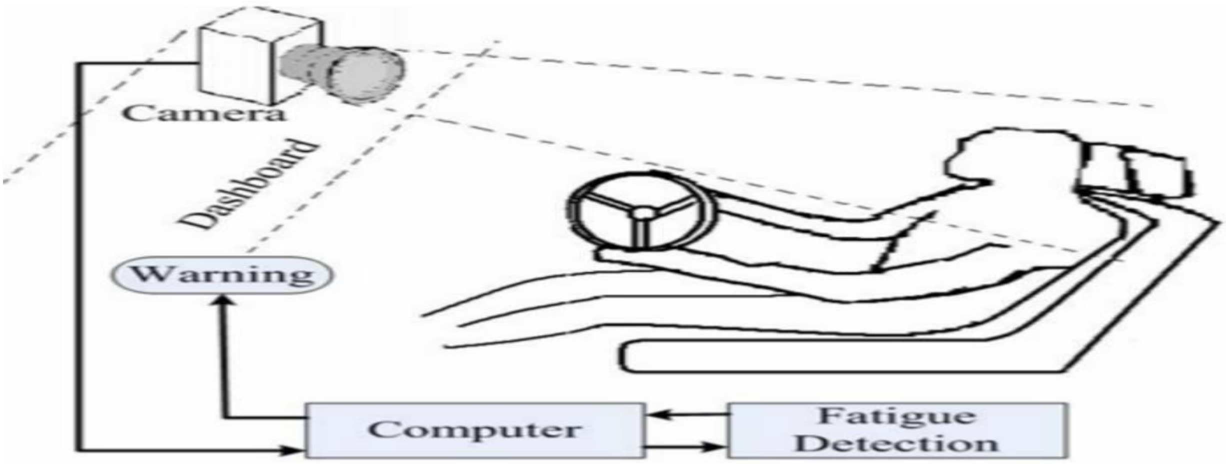

Fig.1.4.

1.5.What is Haar cascade classifier?

It is an Object Detection Algorithm used to identify faces in an image or a real time video. The algorithm uses edge or line detection features proposed by Viola and Jones in their research paper “Rapid Object Detection using a Boosted Cascade of Simple Features” published in 2001. The algorithm is given a lot of positive images consisting of faces, and a lot of negative images not consisting of any face to train on them.

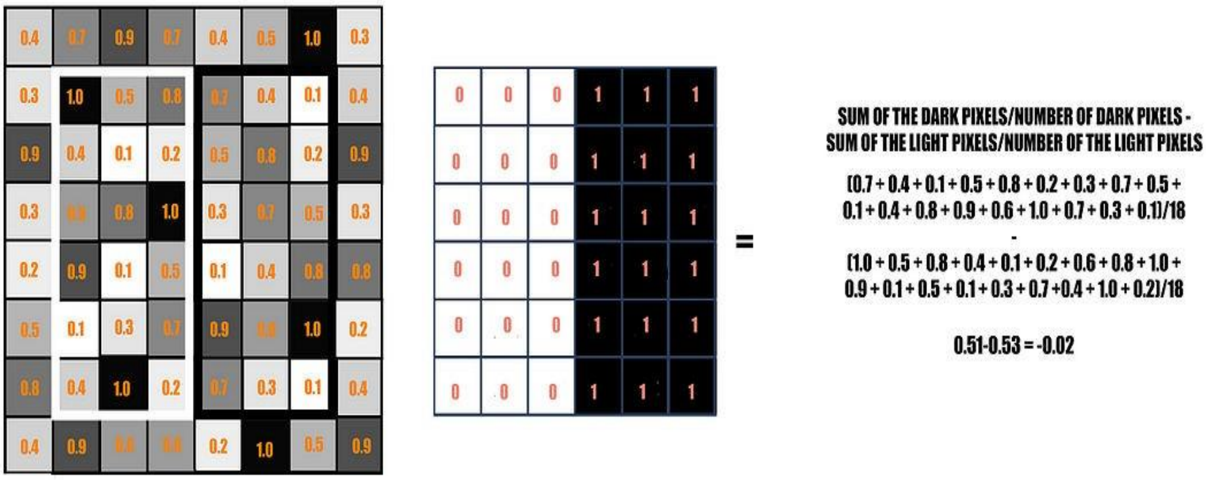

Fig.1.5.

The rectangle on the left is a sample representation of an image with pixel values 0.0 to 1.0. The rectangle at the center is a haar kernel which has all the light pixels on the left and all the dark pixels on the right. The haar calculation is done by finding out the difference of the average of the pixel values at the darker region and the average of the pixel values at the lighter region. If the difference is close to 1, then there is an edge detected by the haar feature.

A sample calculation of Haar value from a rectangular image section has been shown here. The darker areas in the haar feature are pixels with values 1, and the lighter areas are pixels with values 0. Each of these is responsible for finding out one particular feature in the image. Such as an edge, a line or any structure in the image where there is a sudden change of intensities. For ex. in the image above, the haar feature can detect a vertical edge with darker pixels at its right and lighter pixels at its left.

The objective here is to find out the sum of all the image pixels lying in the darker area of the haar feature and the sum of all the image pixels lying in the lighter area of the haar feature. And then find out their difference. Now if the image has an edge separating dark pixels on the right and light pixels on the left, then the haar value will be closer to 1. That means, we say that there is an edge detected if the haar value is closer to 1. In the example above, there is no edge as the haar value is far from 1.

1.6. Haar Algorithm

The Haar algorithm is a well-known image processing algorithm that is used for object detection and facial recognition. The Haar algorithm can be used in the context of drowsiness detection to identify characteristics on the face such as the eyes and mouth, which can subsequently be used to assess whether the individual is drowsy.

The Haar algorithm works by analysing the contrast between different regions of an image. It uses a set of pre-defined Haar features that are trained to recognize certain patterns in the image. These features are typically rectangular and vary in size and orientation.

To detect sleepiness, the algorithm would first take a picture of the person's face with a camera. Then it would use the Haar algorithm to identify the eyes and mouth in the image. The algorithm would then analyse the position and movement of these features over time to determine if the person is drowsy.

For example, if the algorithm detects that the person's eyes are closed or drooping, or that the mouth is open for an extended period, this may indicate that the person is sleepy and needs rest.

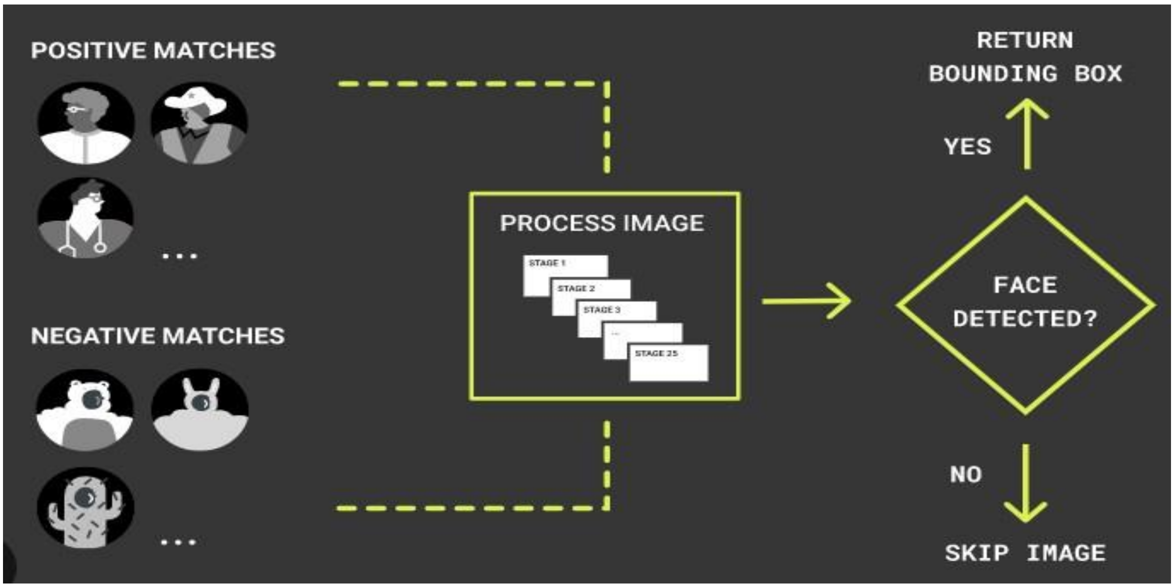

Fig.1.6. (a)

Overall, Haar edge and line detectors can be used to detect various facial features that can be used to determine whether the driver is drowsy. By combining multiple features, it is possible to create a more accurate drowsiness detection system.

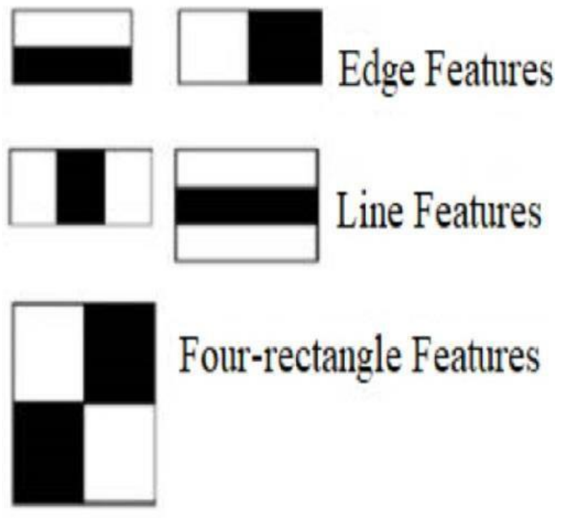

Fig.1.6. (b).

The Haar method may be used to identify the edges of the driver's eyes. It is possible to tell if the driver's eyes are closed or partially closed by measuring the aspect ratio of the observed eyes, which is an indicative of drowsiness.

Role of Haar Algorithm:

1.Collect Data: Collect a dataset of images or videos of individuals in different states of drowsiness.

2.Preprocess Data: Pre-process the data by converting the images or videos into a format that can be processed by the Haar algorithm. This may involve normalization, resizing, and cropping.

3.Face Detection: Apply the Haar algorithm to detect the faces in the images or videos.

4.Feature Detection: Apply a cascade of Haar classifiers to detect specific facial features such as eyes and mouth.

5.Feature Tracking: Track the movement of the detected features over time to identify changes that may indicate drowsiness.

6.Decision Making: Use the tracked features to determine if an individual is exhibiting signs of drowsiness.

Haar-like features formula: The Haar-like features are used to detect specific features such as the eyes and mouth of a driver. The formula for Haar-like features is:

$f(x) = \Sigma w(i) * p(i)$

where $f(x)$ is the value of the Haar-like feature at location x, $w(i)$ is the weight of the ith rectangle, and $p(i)$ is the sum of pixel intensities within the ith rectangle.

1.7. CNN Algorithm

Convolutional Neural Networks (CNNs) can be used in drowsiness detection systems to automatically extract relevant features from images or videos of a person's face or eyes.

The CNN model extracts characteristics that are critical for discriminating between these two states from a huge dataset of annotated images, where each image is classified as drowsy or non-drowsy. These characteristics might include drooping eyelids, alterations in the structure of the eyes, and the degree of eye closure.

Once trained, the CNN model may be used to categorize new images or video frames as drowsy or non-drowsy. The CNN model analyses an image or video frame through a series of convolutional layers that learn to detect patterns and characteristics in the input image. The convolutional layer output is then input into fully connected layers, which learn to map these characteristics to a final classification of drowsy or non-drowsy.

In summary, CNNs are used to detect drowsiness by automatically extracting key data from images or video frames and using them to identify the state of drowsiness. This can aid in the development of real-time systems that can inform drivers or operators if they are growing drowsy and are at risk of an accident.

Role of CNN Algorithm:

1.Collect Data: Collect a dataset of labeled images that have been preprocessed and feature extracted using the Haar algorithm.

2.Split Data: Split the dataset into training and validation sets.

3.Build Model: Build a deep neural network using a machine learning framework such as TensorFlow or Keras.

4.Train Model: Use the training dataset to train the model and the validation dataset to validate it.

5.Model Evaluation: Evaluate the trained model's performance using measures like as accuracy, precision, and recall.

6.Use Model: Use the trained model to classify new images as either drowsy or not drowsy in real-time scenarios.

CNN is a deep learning algorithm that is commonly used for drowsiness detection. The formula for CNN is:

$$y = f(W * x + b)$$

where y is the output of the CNN, W is the weight matrix, x is the input vector, b is the bias vector, and f is the activation function such as ReLU, sigmoid, or softmax .

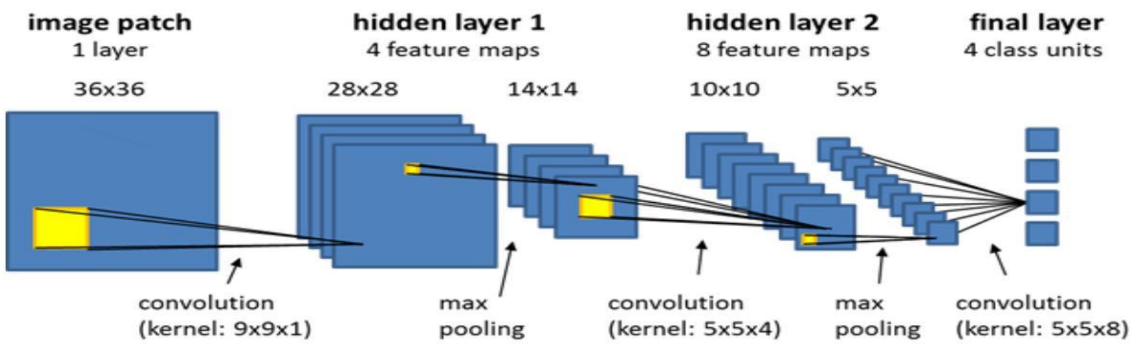

Fig.1.7.

1.8.CNN Architecture:

We created the model using Keras and Convolutional Neural Networks (CNN). A convolutional neural network is a sort of deep neural network that works very well for image categorization. A CNN is made up of three layers: an input layer, an output layer, and a hidden layer with numerous layers. Convolution is conducted on these layers with the help of a filter that conducts 2D matrix multiplication on the layer and filter.

The CNN architecture is trained using a dataset of tagged video frames, with each frame categorized as sleepy or not. The model is taught to recognize drowsiness-related patterns and characteristics in video frames, such as drooping eyelids and sluggish head motions.

Once trained, the model may be used to identify further video frames in real-time, enabling for the early identification of driver sleepiness and the provision of alerts or actions to avert accidents.

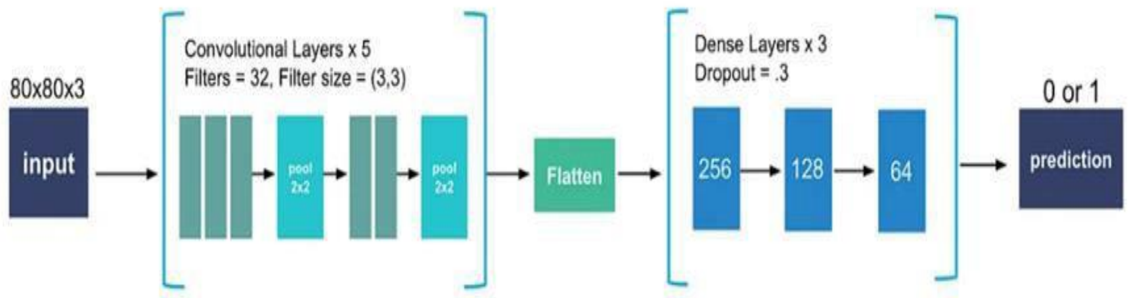

Fig.1.8.

1.9. . DEEP LEARNING:

Deep learning is a subfield of machine learning that involves training artificial neural networks with multiple layers to learn from complex data. It is a type of artificial intelligence (AI) that has revolutionized the field of computer vision, natural language processing, and speech recognition, among others.

In deep learning, a neural network is typically composed of multiple layers of interconnected nodes, or artificial neurons, that perform various mathematical computations on input data. The output of one layer is then passed as input to the next layer, allowing the network to learn increasingly complex representations of the input data.

Deep learning models are typically trained using large datasets of labeled examples, allowing them to learn complex patterns and relationships within the data. During training, the network adjusts its weights and biases to minimize the difference between its predicted output and the actual output.

One of the key advantages of deep learning is its ability to learn hierarchical representations of data. By learning multiple layers of abstraction, deep learning models can automatically extract relevant features from raw data, enabling them to solve complex tasks such as image recognition, natural language processing, and speech recognition.

Some popular deep learning architectures include Convolutional Neural Networks (CNNs) for image recognition, Recurrent Neural Networks (RNNs) for sequential data, and Generative Adversarial Networks (GANs) for generating new data.

Deep learning has enabled significant advances in many fields, including self-driving cars, medical diagnosis, and natural language translation. Its ability to learn from large amounts of data has made it a powerful tool for solving complex problems that were previously thought to be impossible.

There are several types of deep learning architectures, each designed to solve specific types of problems. Some of the most commonly used types of deep learning include:

Convolutional Neural Networks (CNNs): CNNs are a type of deep learning architecture that is particularly well-suited for image and video recognition tasks. They use convolutional layers to extract features from the input data and pooling layers to reduce the dimensionality of the data.

Recurrent Neural Networks (RNNs): RNNs are designed for sequential data, such as text or speech. They use feedback connections to allow information to flow from one step to the next, allowing them to capture dependencies between inputs.

Long Short-Term Memory Networks (LSTMs): LSTMs are a type of RNN that are designed to handle long-term dependencies. They use specialized memory cells to store information for long periods of time, allowing them to remember previous inputs and use that information to make predictions.

Generative Adversarial Networks (GANs): GANs are a type of deep learning architecture used for generating new data. They consist of two networks: a generator network that generates new data samples, and a discriminator network that tries to distinguish between the generated samples and real samples.

Autoencoders: Autoencoders are a type of neural network used for unsupervised learning. They learn to encode the input data into a lower-dimensional representation and then decode it back into the original input.

Deep Belief Networks (DBNs): DBNs are a type of neural network that consists of multiple layers of restricted Boltzmann machines. They are used for unsupervised learning and are particularly well-suited for tasks such as image and speech recognition.

Each type of deep learning architecture has its own strengths and weaknesses, and the choice of architecture depends on the specific problem being solved.

In the context of drowsiness detection, deep learning models can be trained to detect patterns and features associated with drowsiness, such as drooping eyelids, slow eye movements, and head nods. There are several deep learning architectures that can be used for drowsiness detection, including Haar cascades and Convolutional Neural Networks (CNNs).

Haar cascades are a type of object detection algorithm that uses features such as edges, lines, and corners to detect objects in an image or video. Haar cascades can be trained to detect specific facial features associated with drowsiness, such as the position and shape of the eyes, eyebrows, and mouth.

On the other hand, CNNs are a type of neural network that are particularly well-suited for image recognition tasks. CNNs are composed of multiple layers of convolutions, pooling, and fully connected layers, which can learn hierarchical representations of images. CNNs can be trained to detect patterns and features associated with drowsiness in images or videos, such as changes in eye shape and movement.

To train deep learning models for drowsiness detection, large datasets of labeled images or videos of drowsy and non-drowsy individuals are required. Once trained, these models can be used to detect drowsiness in real-time by analyzing video feeds from cameras or sensors. The output of the model can then be used to trigger an alert, such as a sound or vibration, to alert the driver or operator of a potential drowsiness-related hazard.

1.10. . AIM:

The aim of drowsiness detection using Haar and CNN algorithm is to develop a system that can accurately identify when a person is becoming drowsy or sleepy, particularly while driving or operating heavy machinery, by using a combination of Haar cascades and Convolutional Neural Networks (CNNs).

1.11. PROCESS:

Haar cascades are a type of object detection algorithm that can be used to detect specific features in images, such as the eyes and mouth of a person's face. By using Haar cascades, we can accurately locate the eyes of a person and track their movements, which can be used as a key indicator of drowsiness.

On the other hand, CNNs are a type of machine learning algorithm that are particularly effective in image recognition tasks. By training a CNN on a large dataset of images of both drowsy and alert individuals, the system can learn to accurately distinguish between the two states and make predictions about a person's level of alertness.

By combining these two algorithms, we can develop a drowsiness detection system that is both accurate and robust, even in varying lighting conditions and with different individuals. The system can provide real-time alerts to the driver or operator, potentially preventing accidents caused by fatigue.

1.12. FEATURES:

The features of a drowsiness detector using Haar and CNN algorithm include:

1.Eye detection using Haar cascades: The system uses Haar cascades to detect and track the person's eyes, which are a key indicator of drowsiness.

2.Eye aspect ratio (EAR) calculation: The EAR is calculated based on the distance between the person's eye landmarks, which can indicate if the eyes are closing or drooping.

3.Convolutional Neural Network (CNN) model: A CNN model is trained on a large dataset of images of both drowsy and alert individuals, which allows the system to accurately distinguish between the two states.

4.Real-time detection: The system can provide real-time alerts to the driver or operator, potentially preventing accidents caused by fatigue.

5.Robustness: The system can perform well under different lighting conditions and with different individuals.

6.Low computational complexity: The system is designed to have a low computational complexity, making it suitable for implementation on embedded systems or mobile devices.

7.User-friendly interface: The system can have a user-friendly interface, displaying the driver or operator's level of alertness and providing alerts when necessary.

1.13. EXISTING SYSTEM:

There are several existing methods for drowsiness detection, including:

- Electroencephalography (EEG) based methods: This method measures the electrical activity of the brain through electrodes attached to the scalp. EEG-based methods can detect changes in brain waves associated with drowsiness. However, this method is expensive and requires specialized equipment and expertise.
- Eye-tracking methods: These methods use cameras to track eye movements and can

detect changes in eye behavior associated with drowsiness. However, this method is limited to detecting only visual drowsiness, and may not detect other forms of drowsiness.

- Facial recognition methods: These methods use cameras to detect changes in facial expressions and movements associated with drowsiness. However, this method may be limited by changes in lighting and facial occlusion.
- The existing system is a real time system. It uses image processing for eye and face detection. HAAR based cascade classifier is used for face detection. An algorithm to track objects is used to track the eyes continuously. In order to identify the drowsy state of the driver, the PERCLOS algorithm issued.

Limitations:

1. False Positives: Haar-based classifiers can sometimes produce false positive results, leading to false detections of drowsiness.
2. Limited Robustness: These systems may not work well under different lighting conditions or with various face orientations.
3. Reliance on Haar Features: The system's accuracy is directly tied to the Haar features used in the classifier, and their ability to accurately detect drowsiness may vary.
4. Inability to detect Cognitive Fatigue: Haar-based classifiers primarily focus on detecting physical signs of drowsiness, such as eye closure, and may not be effective in detecting cognitive fatigue.

1.14. PROPOSED SYSTEM

Drowsiness detection using Haar and CNN algorithm involves detecting whether a person's eyes are closed or not, which is an indication of drowsiness. The Haar algorithm is used to detect the face and eyes in an image or video stream, while the CNN algorithm is used to classify the state of the eyes as open or closed. The Haar algorithm uses a series of rectangular filters to detect features in an image. The algorithm detects the face in an image by searching for rectangular patterns of pixel intensities that match a pre-defined set of Haar features. Once the face is detected, the algorithm can then locate the eyes using a similar approach.

Once the eyes are located, a CNN algorithm can be trained to classify the state of the eyes as open or closed. The CNN algorithm takes the eye image as input and outputs a probability score indicating the likelihood that the eyes are closed. The CNN is typically trained on a large dataset of eye images labeled with their corresponding state (open or closed) and uses a deep learning architecture to learn features that are relevant for drowsiness detection.

The combination of the Haar and CNN algorithms provides an effective approach for drowsiness detection in real-time applications such as driver monitoring system.

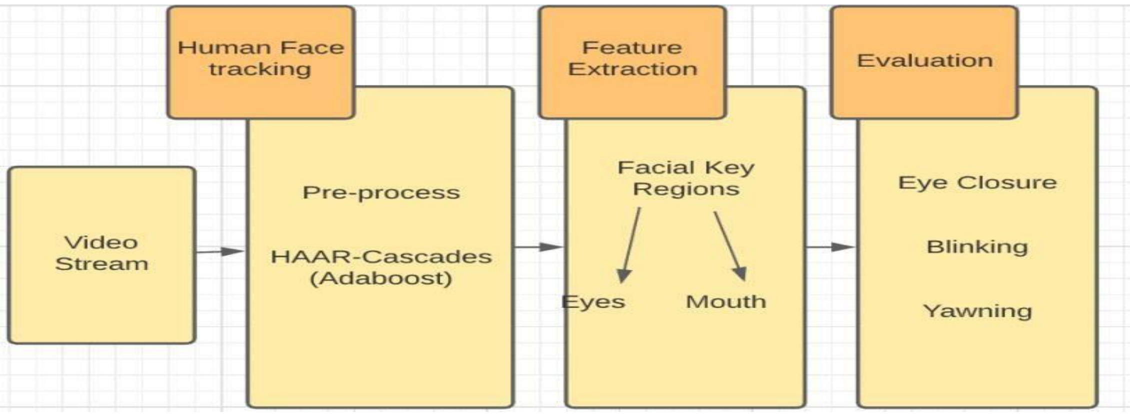

Fig.1.14.

1.15. FEASIBILITY STUDY

The feasibility study of drowsiness detection using Haar and CNN algorithm involves assessing whether the proposed solution is practical, technically feasible, and economically viable.

Technical Feasibility:

The technical feasibility of drowsiness detection using Haar and CNN algorithm involves evaluating whether the necessary hardware and software components are available and whether the algorithm can be implemented effectively. The algorithm requires a camera, computer or embedded system, and software for image processing, object detection, and deep learning. The required components are generally available, and the algorithm has been implemented effectively in many research studies.

Operational Feasibility:

The operational feasibility of drowsiness detection using Haar and CNN algorithm involves evaluating whether the solution can be integrated with existing systems and whether it can be operated effectively by the intended users. The algorithm can be integrated with existing driver monitoring systems and can be operated by drivers or other users with minimal training.

Economic Feasibility:

The economic feasibility of drowsiness detection using Haar and CNN algorithm involves evaluating whether the benefits of the solution outweigh the costs. The cost of implementing the algorithm depends on the hardware and software components required, the complexity of the system, and the level of accuracy required. However, the benefits of the solution, such as improved safety in transportation and reduced accidents, can outweigh the costs.

In summary, the feasibility study of drowsiness detection using Haar and CNN algorithm suggests that the solution is technically feasible, economically viable, and operationally feasible. However, further research and testing may be required to validate the performance and effectiveness of the algorithm in different environments and conditions.

CHAPTER 2

REVIEW OF LIERATURE

V B Navya Kiran, Dr. Nagamani N P, Raksha R, Varsha K N, Anisoor Rahman [1] proposed a comparative study on papers related to driver drowsiness detection and alert system. In order to provide a solution to the problem of detecting the state of drowsiness, an arithmetic-based method is used. This system uses eye movement in order to detect fatigue. Eye movement is detected using a camera. This is done to recognize the symptoms of fatigue in order to avoid accidents. The proposed system detects drowsiness if the eyes have been closed for a period of four or more frames. The detection system differentiates the normal eye blink from drowsiness. The developed system is a non-invasive system. This system uses the concept of video processing. It also mentions certain disadvantages of the proposed system and methods to overcome those disadvantages.

This paper proposed by **Mahek Jain, Bhavya Bhagerathi, Sowmyarani C N [2]** is focused on designing a drowsiness detection system that functions efficiently and is affordable. The method that is needed in the present scenario detects drowsiness based on the eyes and mouth geometric features. This paper proposes to achieve the same, by developing a drowsiness detection system to monitor and prevent a destructive outcome from the negligence of fatigue. There are an increasing number of accidents on roads today and driver drowsiness is an important factor and this has been accepted extensively. The whole project is designed to decrease the rate of accidents and to contribute to the technology with the goal to prevent fatalities caused due to road accidents. Once the eyes are localized, zooming in automatically will help increase the accuracy. The accuracy of detection of eyes and mouth reduces when the driver is not facing the camera.

In the research: "A smartphone-based driver safety monitoring system using data fusion. Sensors", **Lee and Chung [3]** proposed a method to monitor driver safety levels using a data fusion approach such as: eye characteristics, variation of biological signals, temperature inside the vehicle and vehicle speed. This system is developed as an application for an Android-based smartphone, where measuring security-related data that does not require additional costs or additional equipment. The presented results indicate the efficiency of the system that is of higher level and even better than other systems referred to in this work. The lower levels of accuracy occur when users include elements that do not allow to correctly identify the face gestures, but despite that, their levels are satisfactory.

The work presented by **Elena Magan, M. Paz Sesmero ,Juan Manuel Alonso-Weber and Araceli Sanchis [4]** the development of an ADAS (advanced driving assistance system) focused on driver drowsiness detection, whose objective is to alert drivers of their drowsy state to avoid road traffic accidents. In a driving environment, it is necessary that fatigue detection is performed in a non-intrusive way, and that the driver is not bothered with alarms when he or she is not drowsy. To detect whether the driver shows symptoms of drowsiness or not, two alternative solutions are developed, focusing on the minimization of false positives. The first alternative uses a recurrent and convolutional neural network, while the second one uses deep learning techniques to extract numeric features from images, which are introduced into a fuzzy logic-based system afterwards. The accuracy obtained by both systems is similar: around 65% accuracy over training data, and 60% accuracy on test data. However, the fuzzy logic-based system stands out because it avoids raising false alarms. In this paper, two different implementations for a driver drowsiness detection system are proposed, where deep learning plays an important role. These systems use images of the driver to identify fatigue symptoms, but instead of predicting whether a driver is tired or not from a single image, in this work, a full sequence of 60 s is used to determine whether the driver is tired or not over the last minute. Although the obtained results do not achieve very satisfactory rates, the proposals presented in this work are promising and can be considered a solid baseline for future works.

The paper presented by **T. Vesselenyi, S. Moca , A. Rus , T. Mitran , B. Tataru [5]** a study regarding the possibility to develop a drowsiness detection system for car drivers based on three types of methods: EEG and EOG signal processing and driver image analysis. In previous works the authors have described the researches on the first two methods. In this paper the authors have studied the possibility to detect the drowsy or alert state of the driver based on the images taken during driving and by analyzing the state of the driver’s eyes: opened, half-opened and closed. For this purpose, two kinds of ANNs were employed: a 1 hidden layer network and an autoencoder network. For the classification of the driver’s drowsy or alert state, ANNs were used. ANNs are extensively used for the image classification in the last decades. A new paradigm named DL has been developed in and in many other researches. In future works the authors intend to study the possibility to apply neural network classifiers on more different images, in different positions and lighting conditions. Also, the elimination of the cropping and down sampling of the image processing will be a goal for future research.

CHAPTER 3

PROPOSED SOLUTION

3.1 OVERVIEW:

The drowsiness detection system is typically designed to monitor a person's level of alertness or drowsiness in real-time, often with the goal of preventing accidents caused by drowsy driving or other safety-critical tasks. This system typically utilizes a combination of Haar Cascade and Convolutional Neural Network (CNN) algorithms for effective detection of drowsiness. Drowsiness detection using Haar and CNN algorithm involves detecting whether a person's eyes are closed or not, which is an indication of drowsiness. The Haar algorithm is used to detect the face and eyes in an image or video stream, while the CNN algorithm is used to classify the state of the eyes as open or closed.

3.2 DATASET DESCRIPTION:

The Drowsiness Detection Dataset is obtained from Kaggle. Kaggle is a popular platform for finding and sharing datasets for machine learning and computer vision tasks. It is generated using MRL and Closed Eyes in Wild (CEW) dataset, as well as our own unique dataset. This large-scale dataset comprising of both closed and open human eye images can be majorly used for eye detection and further be extended for detecting the drowsiness in an individual. Images from the dataset were taken under a variety of conditions, including diverse lighting conditions, distance, resolution, face angle, and eye angle. These parameters aid in obtaining good outcomes with minimal chances of getting low accuracy.

Images from the dataset were taken under a variety of conditions, including diverse lighting conditions, distance, resolution, face angle, and eye angle. These parameters aid in obtaining good outcomes with minimal chances of getting low accuracy.

There are various versions of this dataset, with Version 1 having 10,000 images split into 5,000 images for closed eyes and 5,000 images for open eyes. The 5,000 images in Version 2 are split into 2,500 images for closed eyes and 2,500 images for open eyes. Another collection of 10,000 images, divided into 5,000 images for each of the closed and open eyes, may be found in Version 3. 2,000 images of closed eyes and 2,000 images of open eyes make up the 4th version of the dataset i.e. Version 4 with 4,000 images in total.

The dataset also includes both normal and yawning images. Dataset is divided into training and testing, which is used in the project for training and testing respectively.

3.3PROCESS:

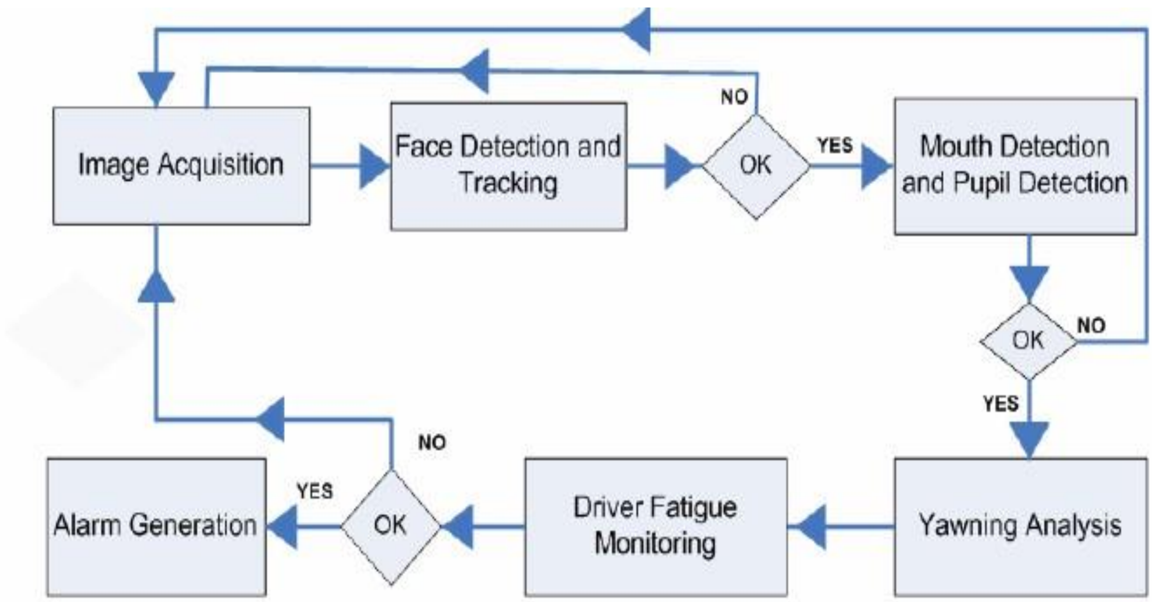

Fig.3.3.

Let us take a step-by-step look at how our algorithm works.

1. Using a camera, we will collect pictures as input. Thus, in order to access the webcam, we created an infinite loop that captures each frame. Each frame is read and the picture is saved in a frame variable.

2.Construct an Region of Interest based on the detection of a face in an image.

To find the face in the image, we must first convert it to grayscale because the method for object identification only accepts grayscale images as input. Color information is not required to detect the items. To detect faces, we will employ the Haar cascade classifier. The detection is then carried out. It produces an array of detections with x,y coordinates and height, which is the width of the object's border box. We can now iterate through the faces, drawing boundary boxes for each one.

3.Using the ROI, detect the eyeballs and give them to the classifier.

The same method that is used to detect faces is utilized to detect eyes. We first configure the cascade classifier for eyes in left eye and right eye. We must now extract only the eyes data from the entire picture. This may be accomplished by extracting the eye's border box and then using this code to extract the eye image from the frame.
Left eye just holds the eye's picture data. This information will be sent into our CNN classifier, which will predict whether the eyes are open or closed. Similarly, the right eye will be extracted into right eye.

4. The Classifier will determine if the eyes are open or closed. The CNN classifier is being used to forecast the eye state. To input our image into the model, we must first execute certain operations because the model need the proper dimensions to begin with.
To begin, we must transform the color image to grayscale. The image is then resized. We then standardize our data to improve convergence. We prepared our model. Now we use our model to forecast each eye. If the value of prediction variable is 1, the eyes are open; if the value of prediction variable is 0, the eyes are closed.

5. Compute the score to see if the person is drowsy. We're drawing the result on the screen with a function that displays the person's current state in real time.

If the score surpasses a certain limit, it shows that the person's eyelids have been closed for a prolonged period of time. The alarm goes off at this point.

Overall, the Haar method is used to detect and track features, whilst the CNN algorithm is employed to classify images. Drowsiness detection systems can identify indicators of sleepiness in real-world circumstances by integrating the strengths of both methods.

The Pseudo code can be represented as shown below:

- The modules required are imported first as shown below:

```
import cv2
import os
from keras.models import load_model
import numpy as np
from pygame import mixer
import time
```

- We use the method provided by OpenCV, cv2.VideoCapture(0) to access the camera and set the capture object (cap). cap.read() will read each frame and we store the image in a frame variable.

- Detect Face in the Image and Create a Region of Interest (ROI).

```
face = cv2.CascadeClassifier('haar cascade
files\haarcascade_frontalface_alt.xml')
for (x,y,w,h) in faces:
        cv2.rectangle(frame, (x,y), (x+w, y+h), (100,100,100), 1 )
```

- Detect the eyes from ROI and feed it to the classifier.

```
leye = cv2.CascadeClassifier('haar cascade
files\haarcascade_lefteye_2splits.xml') reye = cv2.CascadeClassifier('haar
cascade files\haarcascade_righteye_2splits.xml')
l_eye = frame[ y : y+h, x : x+w ]
```

- l_eye only contains the image data of the eye. This will be fed into our CNN classifier which will predict if eyes are open or closed. Similarly, we will be extracting the right eye into r_eye**.**

    - Classifier will Categorize whether Eyes are Open or Closed.

```
model = load_model('models/cnncat2.h5')
path = os.getcwd()
cap = cv2.VideoCapture(0)
font = cv2.FONT_HERSHEY_COMPLEX_SMALL
 count=0
score=0
thicc=2
rpred=[99]
 lpred=[99]
```

```
for (x,y,w,h) in right_eye:
      r_eye=frame[y:y+h,x:x+w]
      count=count+1
      r_eye = cv2.cvtColor(r_eye, cv2.COLOR_BGR2GRAY)
      r_eye = cv2.resize(r_eye,(24,24))
      r_eye= r_eye/255
      r_eye= r_eye.reshape(24,24,-1)
      r_eye = np.expand_dims(r_eye,axis=0)
      rpred = model.predict_classes(r_eye)
      if(rpred[0]==1):
          lbl='Open'
       if(rpred[0]==0):
           lbl='Closed'
       break
```

- Calculate Score to Check whether Person is Drowsy.

```
if(rpred[0]==0 and lpred[0]==0):
        score=score+1
        cv2.putText(frame, “Closed”, (10, height-20), font,
1,(255,255,255),1,cv2.LINE_AA)
 else:
      score=score-1
      cv2.putText(frame, “Open”, (10, height-20), font,
1,(255,255,255),1,cv2.LINE_AA)
```

```
if(score<0):
    score=0
   cv2.putText(frame, ‘Score:'+str(score), (100,height-20), font,
1,(255,255,255),1,cv2.LINE_AA)

if(score>5):
        cv2.imwrite(os.path.Join(path,'image.jpg'),frame)
        try:
              sound.play()
       except:
              pass

if(thicc<16):
     thicc= thicc+2
 else:
    thicc=thicc-2
    if(thicc<2):
       thicc=2
 cv2.rectangle(frame,(0,0),(width,height),(0,0,255),thicc)
 cv2.imshow('frame',frame)
```

CHAPTER 4
IMPLEMENTATION

4.1. Prerequisites

1.Labeled Dataset: A labeled dataset of images or videos is required for training the CNN algorithm. The dataset should contain a representative sample of images or videos of individuals in different states of drowsiness.

2.Haar Cascade Classifiers: The Haar algorithm requires a set of pre-trained classifiers to detect specific facial features such as eyes and mouth. These classifiers are typically available in OpenCV library or can be trained from scratch.

3.Image Pre-processing: Pre-processing of images is necessary to reduce noise and improve the accuracy of feature detection. The pre-processing steps can include image normalization, resizing, and cropping.

4.Machine Learning Framework: A machine learning framework such as TensorFlow or Keras is required for training the CNN algorithm. These frameworks provide a convenient and efficient way to build, train and deploy machine learning models.

5.Hardware Requirements: Drowsiness detection using Haar algorithm and CNN algorithm requires significant computational resources. This includes a powerful CPU or GPU, and sufficient memory to process and store large datasets.

6.Real-Time Data Input: Drowsiness detection using Haar algorithm and CNN algorithm requires real-time input data, such as from a camera or sensor. Therefore, a reliable data input system should be in place.

Overall, drowsiness detection using Haar algorithm and CNN algorithm is a complex process that requires significant expertise in computer vision, machine learning, and software engineering. It is important to carefully consider the prerequisites before beginning to develop a drowsiness detection system.
Drowsiness detection is an important aspect of driver safety, and both Haar and CNN

algorithms can be used for this purpose. Here are some common tools that are used for drowsiness detection using these algorithms:

4.2. TOOLS USED:

4.2.1 Open CV:

OpenCV is an open-source computer vision library that provides several tools for image and video processing. It includes Haar Cascade Classifiers, which can be used for object detection, including detecting eyes and faces in images and video streams. OpenCV can be used with Python, C++, and Java programming languages.

In drowsiness detection using CNN and Haar algorithm, OpenCV plays a crucial role in image processing and analysis. The CNN (Convolutional Neural Network) is used to learn the features of the input image and predict the output. The Haar algorithm is used to detect the face and eyes in the input image.

The role of OpenCV in drowsiness detection can be summarized as follows:

- Image capture: OpenCV can capture the images from the camera and process them in real-time. It can be used to capture the images of the driver's face and eyes.

- Preprocessing: OpenCV can be used to preprocess the images by resizing, cropping, and converting them to grayscale. This helps in reducing the computational complexity of the CNN model.

- Face detection: OpenCV's Haar algorithm can be used to detect the face in the input image. This is important because the drowsiness detection algorithm needs to focus on the eyes of the driver, which are located on the face.

- Eye detection: Once the face is detected, OpenCV can be used to detect the eyes using the Haar algorithm. The eye detection is important because it helps in determining whether the driver is drowsy or not.

- Drowsiness detection: Finally, the CNN model can be trained on the eye images to detect drowsiness. OpenCV can be used to feed the eye images to the CNN model and obtain the predictions.

In summary, OpenCV is an important tool in drowsiness detection using CNN and Haar algorithm because it provides the necessary image processing and analysis tools to detect the face and eyes of the driver, and feed the eye images to the CNN model for drowsiness detection.

4.2.2 TensorFlow:

TensorFlow is a popular open-source machine learning library developed by Google. It provides tools for building and training deep learning models, including CNNs. TensorFlow can be used with Python, C++, and Java programming languages.

In summary, TensorFlow is a critical component in drowsiness detection using CNN and Haar algorithm because it provides the necessary tools and libraries to develop, train, and deploy the CNN model for drowsiness detection. TensorFlow allows for real-time processing of the input eye images and enables accurate and efficient drowsiness detection.

4.2.3 Keras:

Keras is a high-level deep learning library that provides a simple interface for building and training neural networks. It can be used with TensorFlow and other backend engines. Keras supports CNNs and can be used with Python programming language.

The role of Keras in drowsiness detection can be summarized as follows:

- CNN Model Development: Keras provides a high-level API for building deep neural networks. It can be used to develop the CNN model for drowsiness detection. Keras allows for easy construction of complex neural network architectures using simple and intuitive code. The model can be built by stacking various layers such as convolutional layers, pooling layers, and fully connected layers.

- CNN Model Training: Once the CNN model is developed, Keras can be used to train the model on a large dataset of eye images. Keras provides a wide range of optimization algorithms and loss functions that can be used to improve the accuracy and performance of the model. The model is trained by feeding it batches of eye images and adjusting the model weights to minimize the loss function.

- Integration with OpenCV: Keras can be integrated with OpenCV to preprocess the input eye images before feeding them to the CNN model. OpenCV can be used to capture the images from the camera, detect the face and eyes using the Haar algorithm, and preprocess the images to improve the accuracy of the CNN model.

- Real-Time Drowsiness Detection: Once the CNN model is trained, Keras can be used to perform real-time drowsiness detection on the input eye images. The model processes the input images and predicts whether the driver is drowsy or not. The predictions can be displayed on a monitor or used to trigger an alarm to alert the driver.

In summary, Keras is an important tool in drowsiness detection using CNN and Haar algorithm because it provides a high-level API for building and training the CNN model. Keras simplifies the development and training of deep neural networks and enables accurate and efficient drowsiness detection.

4.2.4 PyTorch:

PyTorch is an open-source machine learning library developed by Facebook. It provides tools for building and training deep learning models, including CNNs. PyTorch can be used with Python programming language.

In summary, PyTorch is an important tool in drowsiness detection using CNN and Haar algorithm because it provides a flexible and intuitive interface for building and training the CNN model. PyTorch allows for easy construction of complex neural network architectures and enables accurate and efficient drowsiness detection.

4.2.5 Pygame:

Pygame is a free and open-source Python library that is used for developing 2D video games and multimedia applications. It provides functionality for creating graphical user interfaces, handling input events, playing sounds, and displaying images and animations.

The role of Pygame in drowsiness detection can be summarized as follows:

- GUI Development: Pygame can be used to develop a GUI for the drowsiness detection system. The GUI can display the real-time output of the drowsiness detection system, including the video feed from the camera, the output of the Haar algorithm, and the predictions from the CNN model. The GUI can also include controls for adjusting the sensitivity of the system and for setting the alarm threshold.

- Alarm Triggering: Pygame can be used to trigger an alarm if the drowsiness detection system detects that the driver is drowsy. The alarm can be a sound played through the computer's speakers or a visual alert displayed on the monitor. Pygame can also be used to log the detection events to a file or a database.

- Integration with OpenCV and CNN Model: Pygame can be integrated with OpenCV and the CNN model to create a complete drowsiness detection system. OpenCV can be used to capture the images from the camera, detect the face and eyes using the Haar algorithm, and preprocess the images. The preprocessed images can then be fed into the CNN model, and the output of the model can be used to trigger the alarm.

In summary, while Pygame may not have a direct role in the drowsiness detection process, it can be used to develop a GUI for the system and to trigger an alarm if the driver is drowsy. It can also be used to integrate the various components of the system, including OpenCV and the CNN model, to create a complete drowsiness detection solution.

4.2.6 Scikit-learn:

Scikit-learn is a popular machine learning library that provides tools for data analysis and modeling. It includes several classification algorithms, including SVM and Random Forest, which can be used for drowsiness detection.

4.2.7 OS:

The OS (operating system) module in Python can be used for various tasks related to file and directory handling.

The role of the OS module in drowsiness detection can be summarized as follows:

- File Handling: The os module can be used to access and manipulate files and directories. For example, the module can be used to read and write images, videos, and other files that are used in the drowsiness detection system. It can also be used to create directories to store the data generated by the system.

- Data Management: The os module can be used to manage the data generated by the drowsiness detection system. For example, the module can be used to create and update log files that track the events detected by the system, such as the number of times the alarm was triggered or the duration of the drowsy state. The module can also be used to organize the data generated by the system into folders or subdirectories for easy access and management.

- Integration with Other Modules: The os module can be integrated with other modules used in the drowsiness detection system. For example, it can be used with Pygame to play audio files that alert the driver when drowsiness is detected. It can also be used with scikit-learn to load data for model training or with OpenCV to access the camera and video files.

In summary, the os module plays a vital role in drowsiness detection using CNN and Haar algorithm by providing tools for file and directory handling, data management, and integration with other modules. It can be used to create and update log files, organize the data generated by the system, and integrate with other modules used in the system.

4.2.8 Time:

The time module in Python can be used to track the duration of different events and to control the timing of various actions in the system.

The role of the time module in drowsiness detection can be summarized as follows:

- Time Tracking: The time module can be used to track the duration of different events in the drowsiness detection system. For example, it can be used to measure the time elapsed between blinks, yawns, or changes in head position. This information can be used to determine the level of drowsiness of the driver and trigger an alert if necessary.

- Timing Control: The time module can also be used to control the timing of various actions in the system. For example, it can be used to set the duration of the alarm sound or the interval between consecutive blinks or yawns. It can also be used to set the time interval for data collection and analysis.

- Integration with Other Modules: The time module can be integrated with other modules used in the drowsiness detection system. For example, it can be used with Pygame to control the timing of the audio alerts. It can also be used with OpenCV to control the timing of image capture or video processing.

In summary, the time module plays an important role in drowsiness detection using CNN and Haar algorithm by providing tools for time tracking, timing control, and integration with other modules. It can be used to track the duration of different events, control the timing of various actions, and integrate with other modules used in the system.

4.3. System Implementation

The system implementation of drowsiness detection using Haar algorithm and CNN algorithm typically involves several steps, including data collection, preprocessing, feature extraction, classification, and visualization.

1. Data Collection: The first step in implementing a drowsiness detection system is to collect data. This involves capturing images or videos of individuals in different states of drowsiness, such as awake, sleepy, and falling asleep. The data should be diverse and representative of real-world scenarios.

2. Preprocessing: The next step is to preprocess the data. This involves converting the images or videos into a format that can be processed by the algorithms. It may also involve image normalization, resizing, and cropping to reduce noise and improve accuracy.

3. Feature Extraction: The Haar algorithm is used for feature extraction, which involves detecting specific facial features such as the eyes and mouth. This is accomplished using a cascade of Haar classifiers that are trained to recognize different patterns in the image data.

4. Classification: The CNN algorithm is used for classification, which involves training a deep neural network to recognize patterns in the image data and classify the images as either drowsy or not drowsy. The network is trained using a labeled dataset of images that have been preprocessed and feature extracted using the Haar algorithm.

5. Visualization: Once the system has been trained, it can be used to detect drowsiness in real-time scenarios, such as while driving. The system can generate visual alerts or sounds to warn the driver if they are exhibiting signs of drowsiness.

Overall, the system implementation of drowsiness detection using Haar algorithm and CNN algorithm involves combining the strengths of both algorithms to improve the accuracy and reliability of drowsiness detection. By detecting facial features using the Haar algorithm and classifying the images using the CNN algorithm, the system can accurately detect drowsiness in real-world scenarios.

In local machine the software we use to implement our project in a normal usage and by utilizing the computational power are as follows VS Code is an local machine python environment set up software in which we can execute programs and code for the strong things in the daily tasks. It is an local machine python environment set up software in which we can execute programs and code for the strong things in the daily tasks.

The features that are detected after implementation are mentioned in the table given below:

Eye position	Mouth position	Drowsiness Status
Open	Open	Partially detected
Closed	Closed	Detected
Open	Closed	Not detected
Closed	Open	Detected

CHAPTER 5

RESULTS

- Captured Image with Eyes Closed:

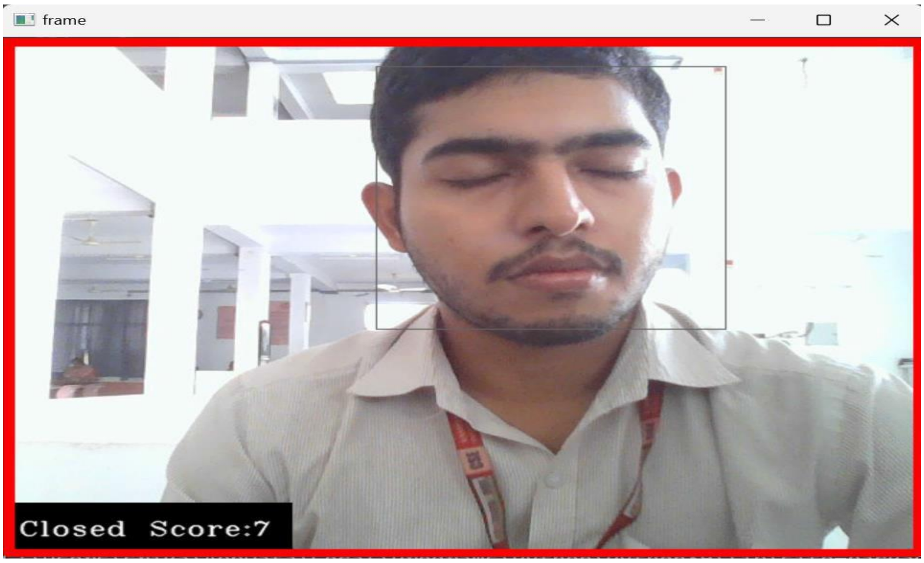

Fig 5. (a).

- Captured Image with Eyes Open:

Fig.5. (b).

- Captured Image with Eyes Open:

Fig.5. (c).

- Resultant Accuracy plot

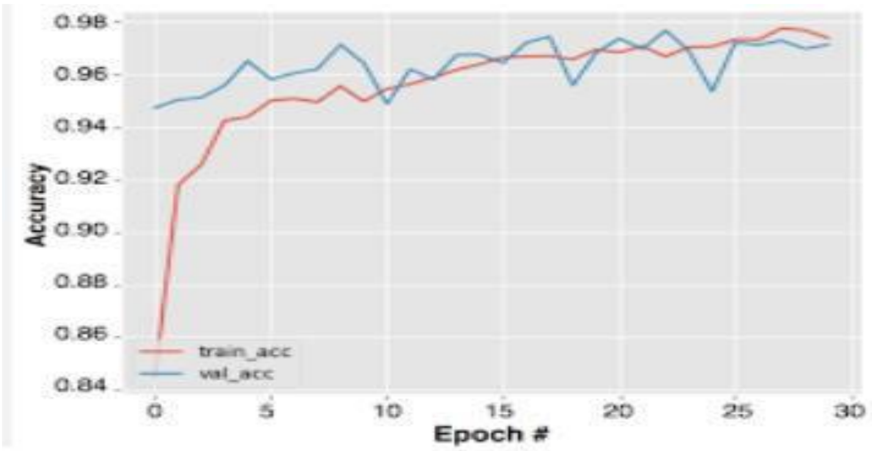

Fig.5. (d).

6. CONCLUSION AND FUTURE SCOPE

We implemented a drowsiness detection system in this paper using the Haar algorithm and the CNN algorithm, which is a combination of computer vision and machine learning techniques. A dataset of images and videos of people in various states of drowsiness is computed. The images or videos are transformed into a format that the algorithms can understand. Normalization, scaling, and cropping are all involved. The Haar algorithm detects certain face characteristics such as the mouth and eyes. This is performed through the use of a cascade of Haar classifiers that have been trained to detect various patterns in image data. The mobility of the identified features is monitored over time in order to detect alterations that may suggest drowsiness. The labelled dataset is created by extracting the characteristics and classifying them as sleepy or not drowsy. CNN Model Training is accomplished by building a deep neural network with TensorFlow and Keras. The prepared dataset is used to train the model, and a validation set is used to validate it. Metrics such as accuracy, precision, and recall are used to assess the trained model's performance. In real-time circumstances, such as when driving, the trained model is used to categorise new images as drowsy or not drowsy. If the driver shows indications of drowsiness, audio warnings are issued to warn them.

LIMITATIONS

- Limited detection accuracy: Although the combined Haar and CNN algorithms can significantly improve the detection accuracy of drowsiness, it may not always be 100% accurate, particularly in situations where there is significant variation in the lighting or user's head position.
- Need for training data: The CNN algorithm used for eye state classification requires a large dataset of labeled eye images to be trained. This can be time-consuming and requires a significant amount of resources.
- Limited applicability: The proposed system may not be applicable to all individuals, particularly those with certain eye conditions that can affect the accuracy of the system.
- Improved detection accuracy: Future research can focus on improving the detection accuracy of the system, particularly in situations with challenging lighting conditions and head positions.
- Enhanced user experience: Future developments can focus on enhancing the user experience of the system, such as by reducing false positives.

- Multi-modal sensing: Combining Haar and CNN algorithms with other sensing modalities such as heart rate monitoring, can help to improve the overall accuracy of the system.

FUTURE SCOPE

- Improved detection accuracy: Future research can focus on improving the detection accuracy of the system, particularly in situations with challenging lighting conditions and head positions.
- Enhanced user experience: Future developments can focus on enhancing the user experience of the system, such as by reducing false positives or by providing more personalized alerts.
- Multi-modal sensing: Combining Haar and CNN algorithms with other sensing modalities such as heart rate monitoring, can help to improve the overall accuracy of the system.

7. REFERENCES

[1] V B Navya Kiran, Raksha R , Anisoor Rahman , Varsha K N "Driver Drowsiness Detection", International Journal of Engineering Research & Technology (IJERT), Special Issue - 2020.

[2] Mahek Jain, Bhavya Bhagerathi, Sowmyarani C N "Real-Time Driver Drowsiness Detection using Computer Vision", International Journal of Engineering and Advanced Technology (IJEAT), Volume-11 Issue-1, October 2021.

[3] Erick Galarza, Fabricio D. Egas, Franklin M. Silva, Paola M. Velasco, Eddie D. Galarza, "Real Time Driver Drowsiness Detection Based on Driver's Face Image Behavior Using a System of Human Computer Interaction", Proceedings of the International Conference on Information Technology & Systems (ICITS 2018) (pp.563-572).

[4] Elena Magan, M. Paz Sesmero ,Juan Manuel Alonso-Weber and Araceli Sanchis ,"Driver Drowsiness Detection by Applying Deep Learning Techniques to Sequences of Images", Appl. Sci. 2022.

[5] T. Vesselenyi, S. Moca , A. Rus , T. Mitran , B. Tataru, "Driver drowsiness detection using ANN image processing", CAR-2017 , IOP Conf. Series: Materials Science and Engineering 252 (2017) 012097.

APPENDIX

Conference Presentation Certificate

Certificate of Presentation

International Conference on

Recent Advances in Science, Technology, Engineering and Management

ICRASTEM-2K23 on 11th April 2023

Organised By

Department of Computer Science and Engineering

VASIREDDY VENKATADRI INSTITUTE OF TECHNOLOGY

(AUTONOMOUS)

Permanently Affiliated to JNTU Kakinada, Approved by AICTE, Accredited by NAAC with 'A' Grade, ISO 9001:2015 Certified,
All eligible B.Tech branches Accredited by NBA, Nambur, Pedakakani (M), Guntur (Dt.) – 522508.

This is to certify that the paper entitled Drowsiness Detection Using Haar and CNN Algorithm with author(s) M.N. Sri Harsha, M.Saketh, K.S. Teja, K.Sumasri, K.Shravani was presented by - in the ICRASTEM - 2K23 International Conference on Recent Advances in Science, Technology, Engineering and Management, held on 11th April 2023 at Vasireddy Venkatadri Institute of Technology, Guntur Dt. in association with SOLETE (Society for Learning Technologies) Vijayawda.

Mr. T. Kranthi Kumar
CEO, SOLETE

Dr. V. Ramachandran
Convener
ICRASTEM-2K23
Professor & HOD CSE, VVIT

Dr. Y. Mallikarjuna Reddy
Principal, VVIT