DROWSINESS DETECTION AND ALERTING SYSTEM

A Mini Project Progress Report

Submitted in partial fulfilment for the degree of

Bachelor of Technology

By

V. S S S Vara Prasad	N180545
Sk. Baji Baba	N181148
T. Trivendra	N180501
G. Nandini Arundhathi	N180559
R. Monika Ravalisri	N180033

Under the Guidance of

Mr. Dara Jagadeesh

Assistant Professor

Department of CSE

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

Rajiv Gandhi University of Knowledge Technologies

(Nuzvid, Eluru, Andhra Pradesh - 521202)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

Rajiv Gandhi University of Knowledge Technologies

(Nuzvid, Eluru, Andhra Pradesh - 521202)

CERTIFICATE OF COMPLETION

This is to certify that the mini project report entitled "**DROWSINESS DETECTION AND ALERTING SYSTEM**", Submitted by **V. Sai Siva Sankara Vara Prasad, Sk. Baji Baba, T. Trivendra, G. Nandini Arundhathi and R. Monika Ravalisri** has been carried out under the guidance of **Mr. Dara Jagadeesh (Assistant Prof. Department of CSE)**, Computer Science and Engineering, Rajiv Gandhi University of Knowledge Technologies - Nuzvid, Andhra Pradesh. The project is approved for submission requirement for Mini Project in the 6^{th} Semester in Computer Science and Engineering from Rajiv Gandhi University of Knowledge Technologies

The results contained in our project thesis have not been submitted in part or in fully to any other University or Institute for the award of any degree or diploma to the best of our knowledge.

. . . .

Mr. Dara Jagadeesh
Assistant Professor,
Department of CSE,
RGUKT - Nuzvid.

. . . .

Mr. Chiranjeevi Sadu
Head of the Department,
Department of CSE,
RGUKT - Nuzvid.

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

Rajiv Gandhi University of Knowledge Technologies

(Nuzvid, Eluru, Andhra Pradesh - 521202)

CERTIFICATE OF EXAMINATION

This is to certify that the work entitled, “**DROWSINESS DETECTION AND ALERTING SYSTEM**”, and here by accord our approval of it as a study carried out and presented in a manner required for its acceptance in 3rd year Mini Project of **Bachelor of Technology** for which it has been submitted. This approval does not necessarily endorse or accept every statement made, opinion expressed or conclusion drawn, as a recorded in this thesis.

It only signifies the acceptance of this thesis for the purpose for which it has been submitted.

. . . .

Mr. Dara Jagadeesh

Assistant Professor,

Department of CSE,

RGUKT - Nuzvid.

. . . .

Project Examiner

Department of CSE,

RGUKT - Nuzvid.

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

Rajiv Gandhi University of Knowledge Technologies

(Nuzvid, Eluru, Andhra Pradesh - 521202)

CERTIFICATE OF DECLARATION

We hereby declare that the project entitled "**DROWSINESS DETECTION AND ALERTING SYSTEM**", submitted by **V. Sai Siva Sankara Vara Prasad, Sk. Baji Baba, T. Trivendra, G. Nandini Arundhathi and R. Monika Ravalisri**, done by us under the guidance of **Mr. Dara Jagadeesh**, Assistant Professor, is submitted for the fulfillment of mini project during the academic session **February, 2023 – June, 2023** at RGUKT-Nuzvid. We also declare that this project is a result of our own effort and has not been copied or imitated from any source. Citations from any websites are mentioned in the references. The results embodied in this project report have not been submitted to any other university or institute for the award of any degree or diploma.

……………………………

Internal Examiner:	V. S S S Vara Prasad	N180545
Date:	Sk. Baji Baba	N181148
Place: Nuzvid	T. Trivendra	N180501
	G. Nandini Arundhathi	N180559
	R. Monika Ravalisri	N180033

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

Rajiv Gandhi University of Knowledge Technologies

(Nuzvid, Eluru, Andhra Pradesh - 521202)

ACKNOWLEDGEMENTS

We would like to express our profound gratitude and deep regards to our guide **M**r. **Dara Jagadeesh** for his exemplary guidance, monitoring and constant encouragement to us throughout the B-Tech course. We shall always cherish the time spent with him during the course of this work due to the invaluable knowledge gained in the field of reliability engineering. We are extremely grateful for the confidence bestowed in us and entrusting our project entitled "**DROWSINESS DETECTION AND ALERTING SYSTEM**".

We express our gratitude to **Mr. Chiranjeevi Sadu (HoD, Dept. of CSE)** and other faculty Members for being source of inspiration and constant encouragement which helped us in completing the project successfully.

Finally, yet importantly, we would like to express our heartfelt thanks to our beloved parents, faculty, and God for their blessings. We are also grateful to our friends for their help and wishes for the successful completion of this project

ABSTRACT

Computer based Drowsiness Detection and Alerting System is a novel approach which is aimed to improve the productivity of a person or organization. By leveraging the Deep Neural Network (DNN) method, a model is trained using a carefully collected dataset to accurately identify signs of drowsiness. The system incorporates Haar Cascades to detect faces and eyes through a webcam, while also offering a face authentication feature for enhanced security. Real-time data is analyzed using the trained model, enabling timely detection of drowsiness. To ensure immediate action, the system includes an alarm and phone call feature for alerting the individual. The implementation of this system aims to improve productivity and prevent accidents caused by drowsiness in various settings, including offices, schools, and homes. This application can be deployed as standalone software or integrated into existing productivity tools, providing an additional layer of safety and awareness for users.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER-1

INTRODUCTION

1.1 Introduction

The development of a computer-based system for the detection of drowsiness in individuals has become increasingly important, particularly in office settings during working hours. This system focuses on leveraging the Deep Neural Network (DNN) working in front of their laptops. The system incorporates Haar Cascades to detect faces and eyes through the laptop's webcam, enabling real-time monitoring of the individual's alertness level.

By utilizing a carefully collected dataset, the model becomes proficient in detecting drowsiness with high precision. The system's primary goal is to prevent accidents and enhance productivity in office environments. When the system detects signs of drowsiness, it immediately alerts the individual to take necessary actions and maintain a safe working environment.

Real-time data analysis plays a crucial role in the timely detection of drowsiness. The trained model continuously analyzes the individual's eye movements to identify potential signs of drowsiness. To ensure immediate action is taken, the system is equipped with an alarm and notification feature that alerts the person when drowsiness is detected. This prompt notification serves as a reminder for the individual to take a break, engage in rejuvenating activities, or seek assistance to ensure their well-being and productivity.

The implementation of this system in office settings holds immense potential. By proactively addressing drowsiness and fatigue, potential accidents and reduced productivity can be mitigated. The system can be seamlessly integrated into existing office setups, providing an additional layer of safety and awareness for employees during their working hours.

This Application documentation provides a comprehensive overview of the system's development process, including the setup and installation of the Integrated Development Environment (IDE), the collection of the dataset, the training and testing of the model using the DNN method, the integration of Haar Cascades for face and eye detection, the implementation of drowsiness prediction, and the inclusion of alarm and notification functionalities. The documentation also outlines the system implementation within office environments and highlights its potential impact on improving safety and productivity by effectively addressing drowsiness-related risks.

1.2 Problem Statement

The problem addressed by this system is the detection of drowsiness in individuals. Drowsiness can significantly impact productivity and can also lead to accidents, particularly in settings where individuals need to remain alert and focused, such as offices, schools, and homes. Existing methods for detecting drowsiness, such as manual observation or wearable devices, may have limitations in terms of accuracy, real-time monitoring, and user comfort. Therefore, there is a need to develop a computer-based system that can effectively and accurately detect drowsiness to mitigate its negative consequences.

1.3 Need of the Study

The need for this study arises from the potential risks associated with drowsiness while using laptops or PCs. Accidents caused by drowsiness can lead to a decrease in productivity, errors in work, and potential damage to the computer system. By developing a reliable system for drowsiness detection, it becomes possible to prevent these issues and improve overall productivity and safety. This study aims to address the limitations of existing methods and provide an accurate, real-time, and non-intrusive solution for detecting drowsiness while using laptops or PCs. The implementation of this system enhances its usability and effectiveness in alerting users and promoting a safer and more efficient computing experience.

1.4 Study Objective

The main objective of this study is to develop a computer-based system for the detection of drowsiness in individuals. The specific objectives include:

1. Training a Deep Neural Network (DNN) model using a carefully collected dataset to accurately identify signs of drowsiness.
2. Incorporating Haar Cascades to detect faces and eyes through a webcam.
3. Implementing a face authentication feature for enhanced security.
4. Analysing real-time data using the trained model to enable timely detection of drowsiness.
5. Integrating an alarm and phone call feature to alert individuals and ensure immediate action.
6. Evaluating the performance and effectiveness of the system in detecting drowsiness.
7. Investigating the potential integration of the developed system into existing productivity tools.

1.5 Summary

This introduction outlined the focus and objectives of the system, which is the development of a computer-based system for drowsiness detection using a Deep Neural Network (DNN) approach. The system incorporates Haar Cascades for face and eye detection, as well as a face authentication feature for enhanced security. Real-time data analysis enables timely detection of drowsiness, and an alarm and phone call feature ensure immediate action.

The implementation of this system aims to improve productivity and prevent accidents caused by drowsiness in various settings. The dissertation is organized into chapters that cover different aspects of the system, including literature review, methodology, results, and discussion.

CHAPTER-2

LITERATURE REVIEW

2.1 Introduction

This chapter provides a literature review related to the development of computer-based systems for drowsiness detection. It explores existing research and technologies in the field of drowsiness detection, computer vision, and deep learning methods. The literature review serves as a foundation for understanding the current state of the art, identifying key approaches and techniques, and highlighting any research gaps that the present study aims to address.

2.2 Literature Review

The literature review reveals several studies and approaches in the field of drowsiness detection. Various techniques have been explored, including physiological measurements, behavioral analysis, and computer vision-based methods

[3]Physiological measurements, such as Electro Encephalo Gram (EEG) and Electro Oculo Graphy (EOG), have been used to detect drowsiness by monitoring brainwave patterns and eye movements. However, these methods often require specialized equipment and can be invasive or uncomfortable for the users.

[4]Behavioural analysis techniques focus on analysing behavioural cues associated with drowsiness, such as eye closure duration, head movements, yawning, and blink frequency. These methods often rely on wearable devices or sensors, but they may suffer from limitations in terms of accuracy and real-time monitoring.

[2]Eye Aspect Ratio (EAR) has considered as a factor for predicting the drowsiness by using the dlib library to detect landmarks on the face. However, this method may face sufferings when using webcam to detect landmarks which are crucial in this process.

[1]Haar Cascades, a popular feature detection technique, has been widely used in computer vision-based drowsiness detection systems for face and eye detection. This technique uses a trained classifier to identify specific features, such as eyes or faces, within an image or video stream. It provides a robust and efficient solution for detecting faces and eyes, which are important cues for drowsiness detection.

While existing research has made significant progress in drowsiness detection, there are still some research gaps that need to be addressed. These gaps include the need for improved accuracy and reliability of drowsiness detection algorithms, real-time monitoring capabilities, integration into existing systems or tools, and usability in various environments and contexts.

2.3 Research Gaps

Based on the literature review, the following research gaps have been identified

1. The need for improved Accuracy and Reliability: While computer vision-based methods show promise in drowsiness detection, there is still a need to enhance the accuracy and reliability of these algorithms to minimize false positives and false negatives.
2. Real-Time Monitoring Capabilities: Ensuring real-time monitoring is crucial for timely detection of drowsiness and prompt action. Research is needed to develop efficient algorithms that can process data in real-time and provide immediate alerts when drowsiness is detected.
3. Integration into Existing Systems or Tools: To maximize the impact of drowsiness detection systems, it is important to explore ways to integrate them into existing productivity tools or systems commonly used in various settings, such as offices, schools, and homes.
4. Usability in Different Environments and Contexts: Drowsiness detection systems should be adaptable to different environments and contexts, considering factors such as lighting conditions, camera angles, and user variability. Further research is needed to address these variations and ensure the effectiveness of the system in different scenarios.

2.4 Summary

The literature review chapter provides an overview of existing research and technologies in the field of drowsiness detection. It covers various approaches, including physiological measurements, behavioural analysis, and computer vision-based methods. Computer vision-based methods, particularly those utilizing deep learning techniques and Haar Cascades, offer non-intrusive and real-time monitoring capabilities. However, there are research gaps that need to be addressed, including the need for improved accuracy, real-time monitoring, integration into existing systems, and usability in different environments. The findings from the literature review provide a foundation for the development of the proposed computer-based system for drowsiness detection in the present study.

CHAPTER-3

PROPOSED MODEL

3.1 Proposed Model:

Deep learning models are suitable to analyze sequential data. The data would be labeled and preprocessed. The preprocessed data would be divided into 80% training and 20% testing data sets. The model would be trained with the training data set. Based on the knowledge gained during the training phase, the process will be analyzed.

Deep Neural Network: It is a kind of network architecture for deep learning algorithms and is specifically used for image recognition and tasks that involve the processing of pixel data. At its simplest, a neural network with some level of complexity, usually at least two layers, qualifies as a Deep Neural Network (DNN), or deep net for short. Deep nets process data in complex ways by employing sophisticated math modeling

3.2 Diagram

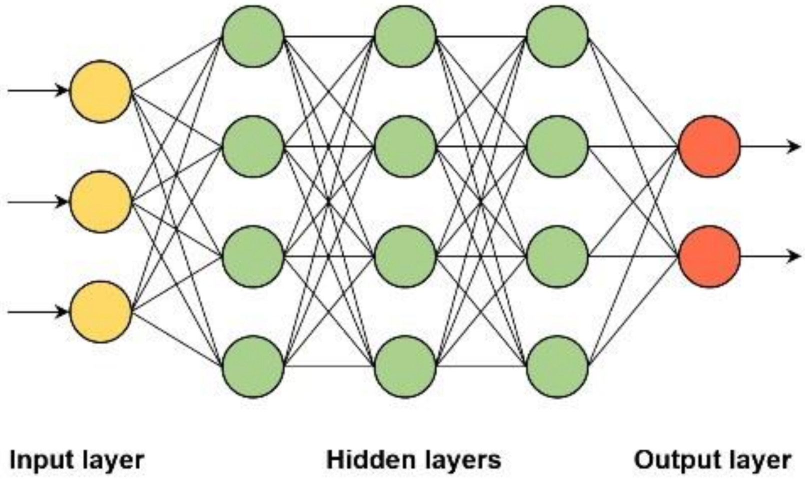
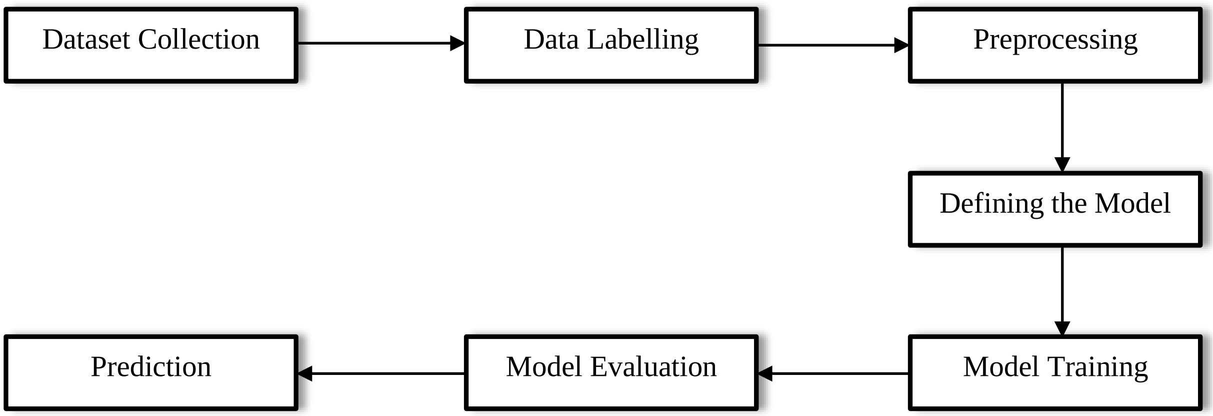


Figure 1: Deep Neural Network

3.3 Work Flow



Flow chart

Dataset Collection: The datasets are collected from Kaggle, an open-source platform that provides a wide range of large datasets across various categories. In this case, we are specifically collecting datasets for open eyes, closed eyes, right eye, and left eye. These datasets will be used for further analysis or training purposes.

Data Labelling: The data is labeled to facilitate the training of a model. This process involves assigning labels to specific data points, such as images and videos. In this case, we create separate folders for open eyes and closed eyes to organize the labeled data.

Preprocessing: During the data preprocessing step, we perform several operations on the collected dataset. First, we load the dataset into two separate lists. The first list contains the images, which are then converted to grayscale and resized to a resolution of 64x64 pixels. The second list contains corresponding values of 1's and 0's, indicating whether the images in the first list represent open or closed eyes. To accomplish this, we import the necessary Python libraries: os, cv2, and NumPy. These libraries provide the required functionalities for the preprocessing tasks.

By combining the capabilities of these libraries, our preprocessing code becomes capable of effectively traversing through the directory structure using ‘os.walk()’. We can then construct valid file paths using ‘os.path.join()’, which helps handle the differences in path separators between operating systems. The 'cv2.cvtColor()' function allows us to convert the images into grayscale, while 'cv2.resize()' resizes the images to the desired resolution. Finally, we can create NumPy arrays to store the images and their corresponding values, enabling efficient manipulation and analysis of the data.

Overall, this preprocessing approach enables us to efficiently process data stored in various directories, allowing us to perform operations on individual files within those directories.

i) Splitting the Dataset: The dataset is divided into training and testing sets using the train_test_split() function from the sklearn.model_selection module in the scikit-learn (sklearn) library. This step ensures that we have separate data for training and evaluating the model's performance.
ii) One-Hot Encoding: The target variable is encoded using the to_categorical() function from the tensorflow.keras.utils module, which is part of the TensorFlow library. This function converts the target variable into a one-hot encoded representation, enabling the model to understand and predict categorical classes accurately.
iii) Reshaping the Input Data: The input data is reshaped to match the expected input shape of the model. This step ensures that the data is in the appropriate format for training and inference. It may involve modifying the dimensions or structure of the input data.

Defining the Model: The model architecture is defined by creating an instance of the 'Sequential' class from the 'keras.models' module. This class allows for a sequential arrangement of layers in the neural network. Using the instance of the 'Sequential' class, layers are added to the model using the 'add()' method. The model is then compiled using the 'compile()' method, which specifies the loss function, optimizer, and metrics to be used during training.

Model Training: The dataset is split into training data and testing data, with a ratio of 80% for training and 20% for testing. The model is trained using the training data, and then it is evaluated and tested using the testing data.

The model is trained using the 'fit()' method. This involves feeding the training data to the model, iterating over a specified number of epochs (iterations), and optimizing the model's parameters using the chosen optimizer and loss function. The model learns from the training data and adjusts its internal parameters to minimize the loss function.

Model Evaluation: After training, the model's performance is evaluated using the 'evaluate()' method. The model is provided with the testing data, and metrics such as accuracy, loss, or other evaluation metrics are computed to assess the model's performance on unseen data. This step helps measure the effectiveness and generalization capability of the trained model.

Prediction: The Trained and tested model is used to predict the data. The model predicts drowsiness based on whether the user's eyes are open or closed and Provides a Prediction Class if eyes closed 1, eyes opened 0

CHAPTER-4

REQUIREMENTS AND ANALYSIS

4.1 Hardware Components

- Computer or Device
- CPU: Intel Core i3 or higher
- RAM: 4GB or higher
- Camera: Built-in camera with a resolution of at least 720p
- Speakers: Built-in or external speakers for audio alerts

4.2 Software Components

1. Python: Install the latest version of Python, which includes the Python interpreter and necessary libraries for running Python programs.
2. OpenCV: Install the OpenCV library for Python. OpenCV provides a wide range of image and video processing functions, including Haar Cascades for face and eye detection.
3. TensorFlow: Choose a deep learning framework based on your preference and requirements. Install TensorFlow to train and deploy the drowsiness detection model.
4. NumPy: NumPy is a fundamental library for scientific computing in Python. It provides support for handling multidimensional arrays and performing numerical operations. Install NumPy to process image and video data efficiently.
5. Keras: If you choose to work with TensorFlow, you can also install Keras, a high-level deep learning library that provides an easy-to-use interface for building and training neural networks.
6. Tkinter: Tkinter is the standard Python interface for creating Graphical User Interfaces (GUIs). It allows you to design and implement the user interface for your drowsiness detection system. Tkinter is included in the standard Python installation for Windows.
7. Wave: The wave module in Python's standard library is an easy interface to the audio WAV format. The functions in this module can write audio data in raw format to a file like object and read the attributes of a WAV file.

8. PYAudio: PYAudio provides Python bindings for PortAudio v19, the cross-platform audio I/O library. With PYAudio, you can easily use Python to play and record audio

9. Twilio: To implement the phone call feature for alerts, you can use the Twilio Python library to integrate with the Twilio API and initiate phone calls.

10. Requests: The requests module allows you to send HTTP requests using Python.

CHAPTER-5

DATA COLLECTION

5.1 Introduction

This section provides an overview of the application and its focus on developing a computer-based system for drowsiness detection. It highlights the significance of the application in improving productivity and preventing accidents caused by drowsiness. The introduction sets the context for the subsequent sections.

5.2 Data Collection

The data collection phase involves gathering 2726 images of opened eyes and 2726 images of closed eyes from Google. However, it's important to note that collecting images from Google raises concerns regarding copyright and intellectual property rights. It is crucial to ensure that proper permissions and licenses are obtained for using the images in the system. Ethical considerations and privacy regulations should also be taken into account when collecting images of individuals' eyes.

Figure 2: Data Set

5.3 Summary

The data collection phase involves gathering a specific number of images of opened and closed eyes from Kaggle. However, it is essential to handle this process carefully, ensuring compliance with copyright laws, obtaining necessary permissions, and respecting privacy regulations. Proper attribution and diversity of the dataset should also be considered.

CHAPTER-6

IMPLEMENTATION

6.1 Introduction

The drowsiness detection system is implemented using a Deep Neural Network (DNN) method. A trained model, stored as model.h5, is utilized for predicting whether a person is drowsy or not.

The implementation involves continuously checking for drowsiness in the person for a duration of 30 seconds. If the person is identified as drowsy within this timeframe, an alert is triggered to notify them.

Furthermore, the system includes a functionality to monitor if the person is not in front of the laptop. If there is no response detected within 30 seconds, an alert is generated to ensure attention.

In case there is still no response even after the alert, the system initiates a phone call to the login associated with the user. If there is no response from the user even after the phone call, the system enters a sleep mode after the third alert.

The implementation aims to effectively detect and address drowsiness in individuals, promoting safety and preventing potential accidents caused by drowsiness.

6.2 Models and Algorithms

Deep Neural Network (DNN) Algorithm

1. Input Layer
    - The DNN algorithm begins with an input layer that takes in the pre-processed eye images as input data.
    - Each image is represented as a set of pixel values, forming the input vector.

2. Hidden Layers
    - The input layer connects to one or more hidden layers, which are responsible for learning and extracting features from the input data.
    - Each hidden layer consists of multiple nodes or neurons, and the connections between nodes have associated weights and biases.

3. Activation Function
    - After each hidden layer, an activation function is applied to introduce non-linearity into the network.
    - The activation function helps the network to learn complex patterns and make the model more expressive.
    - Common activation functions used in DNNs include ReLU (Rectified Linear Unit), sigmoid, or tanh.

4. Output Layer
    - The final hidden layer connects to the output layer, which produces the prediction or classification results.
    - In the case of drowsiness detection, the output layer typically has two nodes representing the two classes: drowsy and non-drowsy.

5. Training the Model
    - To train the DNN model, a labelled dataset is used, where each eye image is labelled as drowsy or non-drowsy.
    - The training process involves iteratively adjusting the weights and biases in the network to minimize the difference between the predicted outputs and the true labels.
    - This optimization is typically done using backpropagation, where the gradients of the loss function with respect to the network parameters are calculated and used to update the weights and biases.

6. Loss Function and Optimization
    - The loss function measures the discrepancy between the predicted outputs and the true labels.
    - Common loss functions used for binary classification in DNNs include binary cross-entropy or mean squared error.
    - The optimization algorithm, such as stochastic gradient descent (SGD) or Adam, is used to update the weights and biases during training to minimize the loss.

7. Prediction
    - Once the DNN model is trained, it can be used to make predictions on new, unseen eye images.
    - The input eye image is passed through the network, and the final output layer produces the predicted class probabilities.
    - A threshold can be applied to interpret the probabilities and classify the input as drowsy or non-drowsy.

The DNN algorithm is a powerful approach in deep learning, allowing the model to learn complex patterns and make accurate predictions in tasks such as drowsiness detection. The iterative training process and activation functions enable the model to capture and represent intricate relationships in the data, leading to effective classification results.

6.3 Functionalities

In the drowsiness detection system, users register and login to access the system. After successful login, the system predicts whether the person is drowsy or not using the DNN model. If drowsiness is identified, the system checks for 30 seconds to confirm the state. If the person remains drowsy, alerts are generated. The process continues until the user terminates the application. Additionally, the system monitors if the person is not in front of the laptop, triggering alerts. If there is no response after 30 seconds, a phone call is made to the login associated with the user. If there is still no response, after the third alert, the system enters sleep mode.

6.3.1 Video Capturing

Video capturing involves accessing the webcam or camera to obtain live video footage for further processing and analysis. It typically involves using a library like OpenCV to establish a connection with the camera and retrieve frames from the video stream.

To detect faces and eyes using Haar cascade files, you need to import the required libraries, load the cascade XML files, preprocess the image or video frame, perform face detection using the cascade file, define a region of interest around the detected face, and then perform eye detection within that region. Finally, you can draw rectangles or markers to highlight the detected regions and display the results

Figure 3: Detect Face and Eyes

6.3.2 Registration

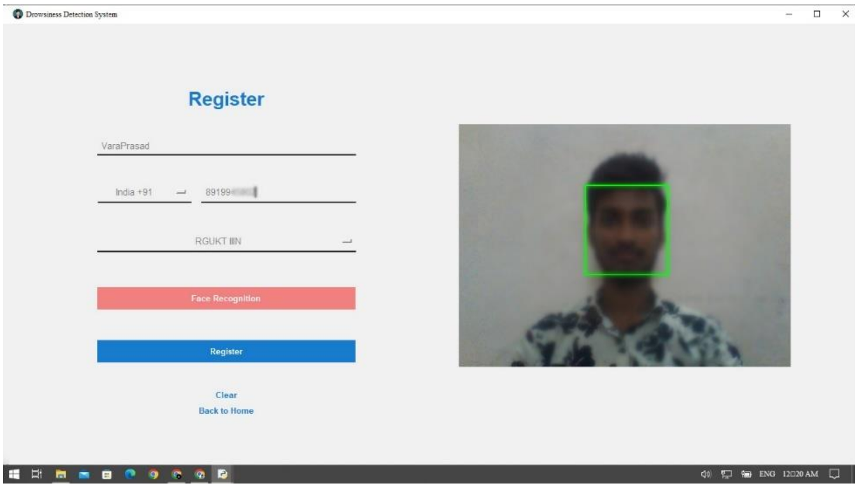


Figure 4: Registration

1. User Details
    - The user provides their details, including a username, phone code, mobile number, and company name. These details are necessary for user identification and communication purposes.

2. Face Encodings
    - The system incorporates face recognition functionality. To capture face encodings, the user clicks the face recognition button, which activates the webcam.
    - The system captures live video footage and uses computer vision techniques to extract facial features and encode them into a numerical representation called face encodings.
    - These face encodings serve as a unique representation of the user's face and are used for face verification and authentication.

3. Registration Submission
    - After filling in all the required details and capturing the face encodings, the user clicks the register button to submit their information.
    - The registration data, including the user details and face encodings, is compiled into a request to be sent to a hosted website.

4. Sending Data to Hosted Website
   - The requests library in Python is used to send a POST request to the hosted website, located on PythonAnywhere.com.
   - The registration data, structured in a JSON or form-encoded format, is included in the body of the POST request.
   - The request is sent to the appropriate endpoint or URL on the hosted website.

5. Hosted Website
   - The hosted website, running on PythonAnywhere.com receives the POST request containing the registration data.
   - The website's backend server, implemented using a Python web framework Django, handles the incoming request.
   - The server parses the registration data, extracting the user details, and saves them into a database for future reference and authentication.

6. Database Storage
   - The hosted website's backend server utilizes a database management system, Django default Database to store and manage user data.
   - The server saves the user details, including the username, phone code, mobile number, company name, and associated face encodings in the database.
   - This data can be used for subsequent login and authentication processes, where the user's face encodings are compared against the stored encodings for verification.

By capturing user details and face encodings during the registration process and storing them in a database on a hosted website, the system establishes a reliable and secure means of user identification and authentication.

6.3.3 Authentication

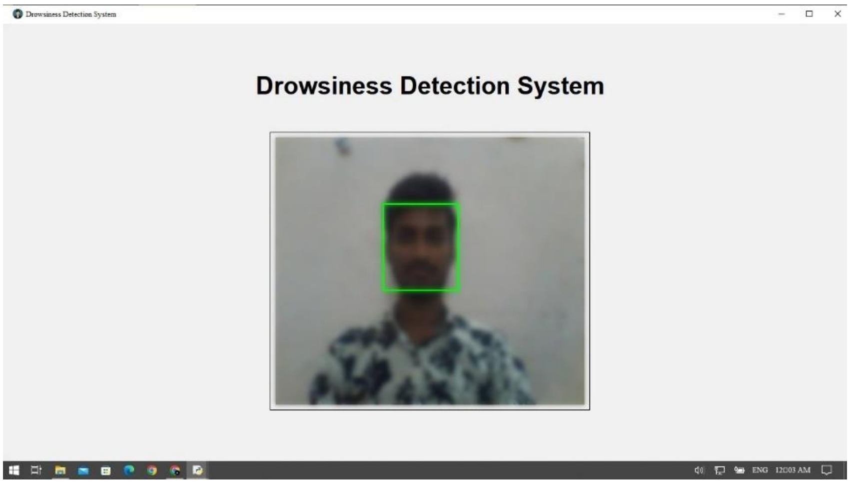


Figure 5: Authentication

During the authentication process, when a user clicks the login button on the home page, the system accesses the camera to read frames and perform face recognition.

1. User Clicks Login Button
    - The user initiates the login process by clicking the login button on the home page.

2. Camera Access
    - The system requests access to the camera and activates it to capture live video frames.

3. Frame Reading and Encoding
    - The system reads the frames from the camera feed and extracts facial features from each frame using face detection and recognition algorithms.
    - These facial features are then encoded into numerical representations called face encodings.

4. Retrieving Known Face Encodings
    - The system interacts with the hosted website's API to retrieve the known face encodings of registered users from the database.
    - The API endpoint provides the necessary data, including the face encodings associated with each registered user.

5. Comparing Face Encodings
    - The system compares the face encodings obtained from the live video frames with the known face encodings of registered users.
    - It calculates the distance or similarity between the live face encodings and the known face encodings using a suitable distance metric.
    - The minimum distance among the calculated distances is determined.

6. Authentication Decision
    - The minimum distance between a live face and a registered user's face is compared with a predefined threshold by the system, and if the minimum distance is below the threshold, indicating a close match between the two faces, the user is considered authenticated. In such cases, the system grants access to the detection interface or the intended functionality.
    - If the minimum distance is more than the threshold, indicating a mismatch or inadequate similarity, the system shows a "login failed, try again" notice and asks the user to try again with their login credentials.

This authentication process ensures that only registered users with sufficiently similar facial features can access the system and its associated functionalities.

6.3.4 Face detection, Eyes detection and Prediction

Figure 6: User Controls

The drowsiness detection system includes start and pause buttons for user control. Clicking the start button initiates the prediction process, detecting faces, eyes, and monitoring drowsiness. If drowsiness is detected, alerts are triggered. Clicking the pause button temporarily halts the prediction process and suspends alerts. Resuming by clicking start resumes the prediction process. This user control enables active engagement or breaks while monitoring drowsiness

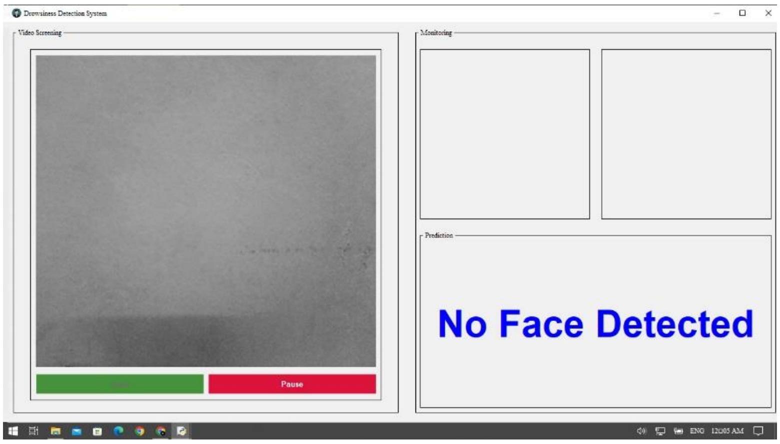


Figure 7: No Face Detected

If no face is detected in the drowsiness detection system, the system starts a timer counting up to 30 seconds. Even if no response is detected during this time, the system triggers alerts, which can be repeated up to three times. If there is no response from the user even after the three alerts, the system enters sleep mode. Sleep mode may involve conserving resources or temporarily disabling the monitoring and alerting functions until the user interacts with the system again

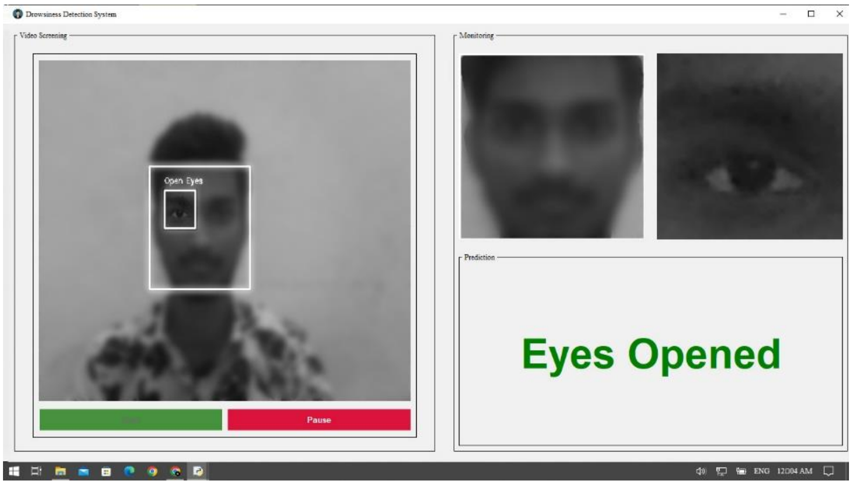


Figure 8: Eyes Opened

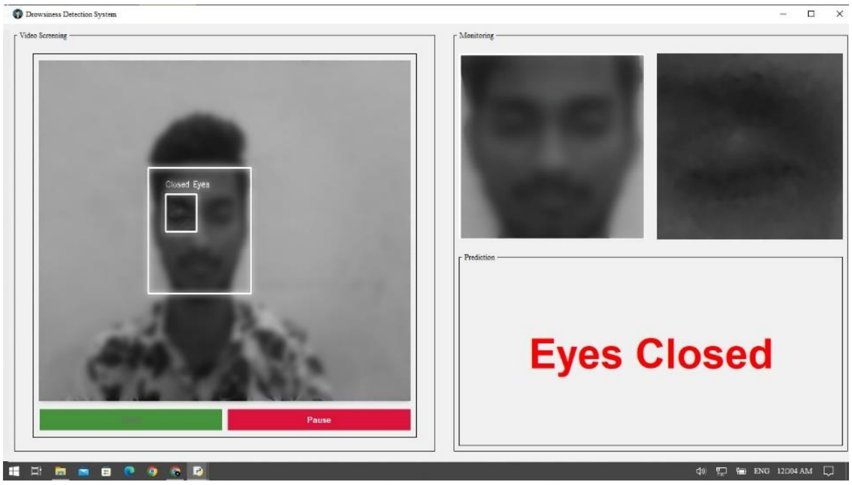


Figure 9: Eyes Closed

In the drowsiness detection system, the following steps are involved in detecting faces, eyes, and predicting drowsiness using OpenCV's Haar cascade classifiers

1. Face Detection
    - Use OpenCV's Haar cascade classifier for face detection.
    - Apply the pre-trained face detection cascade file to the video frames captured from the webcam or camera.
    - If a face is detected in the frame, proceed to the next step. Otherwise, continue monitoring for face detection.

2. Eye Detection
    - Once a face is detected, define a region of interest (ROI) around the detected face.
    - Within the ROI, apply another Haar cascade classifier specifically designed for eye detection.
    - If eyes are detected within the face ROI, proceed to the next step. Otherwise, continue monitoring for eye detection within the face ROI.

3. Drowsiness Prediction
    - During the drowsiness prediction phase, if eyes are detected within the face ROI, the system takes the region of interest (ROI) around the eyes and predicts whether the eyes are closed or open. This prediction is made using a trained DNN model specifically designed for eye state classification Track the duration of eye closures.
    - If the DNN model predicts closed eyes Figure 9: Eyes Closed continuously for 30 seconds, it triggers an alert to indicate drowsiness. However, if the eyes are predicted as open Figure 8: Eyes Opened or a blink is detected within the 30-second interval, the system continues the prediction process without triggering an alert.
    - This continuous monitoring of eye state using the trained DNN model allows the system to detect and alert the user in real-time when prolonged eye closures, indicating drowsiness, are observed.

By using the Haar cascade classifiers for face and eye detection, along with continuous monitoring and alerting mechanisms, the system can effectively detect drowsiness based on eye closures and detect the absence of a face in real-time video feed.

6.3.5 Time Check Function

1. Start Timer
    - When the eyes are predicted as closed, start a timer to track the duration of continuous eye closures.
    - Set the initial time as the current time.

2. Check Timer
    - Periodically check the elapsed time since the timer started.
    - If the elapsed time exceeds or equals 30 seconds, trigger an alert indicating drowsiness.

3. Alert Counter
    - Maintain a counter to keep track of the number of times the alert has been triggered.
    - If the alert is triggered, increment the counter.

4. Sleep Mode Transition
    - If the alert counter reaches three (indicating three consecutive alerts), transition to sleep mode.

By implementing this time check function, you can monitor the duration of continuous eye closures and trigger alerts accordingly. After a certain number of alerts, transitioning to sleep mode ensures a controlled response to persistent drowsiness indications.

6.3.6 Phone Call

In the drowsiness detection system, after two consecutive alerts are triggered due to continuous eye closures, a phone call is initiated to the user using Twilio. If there is no response from the user, a third alert is triggered after 60 sec, and the system enters sleep mode

CHAPTER-7

OBSERVATIONS AND RESULTS

7.1 Observation

Based on the observations from the accuracy metrics of different models, the DNN model achieved the highest accuracy of 0.988 (98.8%) compared to the CNN model's accuracy of 0.962 (96.2%) and the SVM model's accuracy of 0.957 (95.7%).

Given that the models were trained using the same dataset, it is clear that the DNN method outperforms the other models in accurately predicting eye states. Therefore, it is advisable to choose the DNN method to train the model for eye state prediction. The high accuracy of the DNN model suggests that it will provide more reliable and accurate predictions, making it a suitable choice for the drowsiness detection system

7.2Metrics

1. Support Vector Machines (SVM) Accuracy: 95.7%

```
Accuracy: 0.9572127139364304
```

Figure 10: SVM Accuracy

2. Convolutional Neural Network (CNN) Accuracy: 96.2%

```
Accuracy: 0.962199330329895
```

Figure 11: CNN Accuracy

3. Deep Neural Network (DNN) Accuracy: 98.8%

```
Accuracy: 0.9889975786209106
```

Figure 12: DNN Accuracy

Result

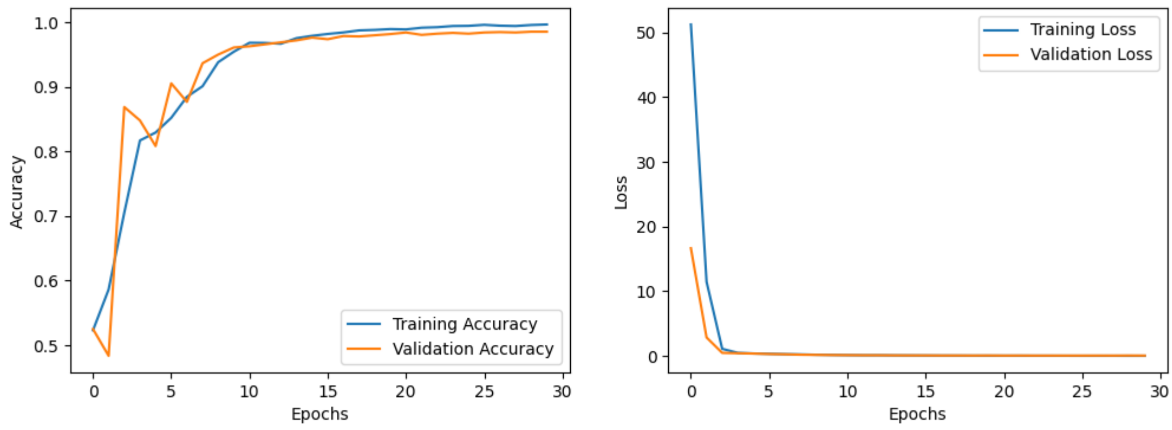


Figure 13: DNN Accuracy Graph

7.3 Summary

The DNN model showed the highest accuracy of 98.8%, making it the recommended choice for accurate eye state prediction in the drowsiness detection system.

CHAPTER-8

FUTURE WORKS

8.1 Future Works

1. Saving Individual Statistics: Implement a feature to save and maintain statistics of individual users, such as the number of drowsiness instances, average duration of drowsiness episodes, and overall drowsiness trends. This data can be useful for analysis and generating personalized insights.

2. Sending Statistics to the Company: Provide an option to securely send the individual statistics to the company or relevant stakeholders for further analysis or monitoring purposes, while ensuring user privacy and data protection.

3. Face Authentication Improvement: Enhance the face authentication mechanism by incorporating additional security measures such as liveness detection, multi-factor authentication, or facial recognition algorithms with higher accuracy and robustness.

4. User-Defined Wait Time: Allow users to configure the duration they need to wait before an alert is triggered. This customizable feature ensures flexibility and accommodates individual preferences.

5. Alert Audio Options: Provide users with a choice to select different alert audio options, such as various sounds or personalized voice messages, to suit their preferences and increase alert effectiveness.

6. Alert Count Tracking: Keep track of the count of alerts triggered for each user. This information can be used for monitoring and providing feedback to individuals regarding their drowsiness patterns and alert response.

7. Phone Call after Specific Alert Count: Implement a feature that triggers a phone call to the user after a certain number of consecutive alerts, indicating a persistent drowsiness issue that requires immediate attention or intervention.

8. Automatically start the application: The application will automatically start when a user opens a company-related site in their browser, providing seamless integration and convenience for users engaging with company-related activities.

By incorporating these additional features, the drowsiness detection system can offer improved user experience, enhanced security, personalized insights, and effective alerting mechanisms to promote safety and well-being in various environments.

CHAPTER-9

CONCLUSION

9.1 Conclusion

This Application focuses on the development of a drowsiness detection system using a Deep Neural Network (DNN) method with 98.8% accuracy. By training a DNN model and leveraging face and eye detection using Haar cascades, the system accurately identifies signs of drowsiness in individuals. Real-time data analysis, along with the incorporation of face authentication and alerting mechanisms, enhances the system's effectiveness in preventing accidents caused by drowsiness. The implementation of the system aims to improve productivity and safety in various settings, such as offices, schools, and homes. The observed high accuracy of the DNN model further validates its suitability for accurate eye state prediction, making it the recommended approach for this application.

REFERENCES

[1] Moulana Mohammed, Addepalli Sri Vaishnavi Devi, Kandlapalli Deepika" Drowsiness Detection using Deep Learning Algorithms", IEEE 2022

[2] Prasath N, Sreemathy J, Vigneswaran P "Driver Drowsiness Detection Using Machine Learning Algorithm", IEEE 2022

[3] V Jothsna, Ibrahim Patel, K. Raghu, P Jahnavi, K. Nagi Reddy, K. Saikumar, "A Fuzzy Expert System for The Drowsiness Detection from Blink Characteristics", IEEE, 2021

[4] Aicha Ghourabi1, Haythem Ghazouani1, and Walid Barhoumi "Driver Drowsiness Detection Based on Joint Monitoring of Yawning, Blinking and Nodding", IEEE, 15 June 2021