# REAL TIME DRIVER DROWSINESS

**DETECTION**

## Summary:

## This project aims to develop a driver drowsiness detection model using Python, OpenCV, Keras, and TensorFlow. The methodology involved collecting a diverse dataset of drowsy and awake states, preprocessing the images, developing a drowsiness detection model using CNNs in Keras, training and evaluating the model, and implementing real-time drowsiness detection using OpenCV. The system analyzes video feeds to detect signs of drowsiness, such as closed eyes, and promptly alerts the driver while sending an email notification to a designated person and store the image in database.

## Introduction:

The problem at hand is the need to develop a driver drowsiness detection model using Python, OpenCV, Keras, and TensorFlow.The primary objective is to create a system that can analyse real-time video feeds and accurately detect signs of driver drowsiness or fatigue and alert the concern people (authorities or loved ones) and alert the driver as well through some sound signal.

## Methodology:

The project consists of the following key steps:

### Step 1: Data Collection

Collecting a diverse and representative dataset is crucial for training an effective drowsiness detection model. The dataset collected includes a variety of images depicting both drowsy and awake states to ensure accurate model training.

### Step 2: Preprocessing:

The collected images need to undergo preprocessing to enhance their quality and extract relevant features. This process involves techniques such as resizing, normalization, and grayscale conversion. Preprocessing ensures that the input data is in a suitable format for training the model.

### Step 3: Model Development:

The drowsiness detection model is built using Keras, a high-level deep learning library that operates on top of TensorFlow. Convolutional neural networks (CNNs) are employed to extract meaningful features from the input images. The model architecture also consist of multiple convolutional and pooling layers, followed by fully connected layers.

### 

### Step 4: Training and Evaluation:

The dataset is divided into training and testing sets and the model is trained on the training set, optimizing its parameters to minimize the loss function. The model's performance is evaluated using the testing set, measuring metrics such as accuracy, precision, recall, and F1-score.

### Step 5: Real-time Drowsiness Detection:

Once the model is trained and evaluated, it is used to detect drowsiness in real-time video feeds. OpenCV, a widely-used computer vision library is employed for video frame capture and processing. The trained model will analyse each frame to identify signs of drowsiness, such as closed eyes. When drowsiness is detected, an alert will be issued to notify the driver and an email is sent to concerned person (relative or loved ones) and stores the image in database.

## Technique and Implementation:

The implementation of the driver drowsiness detection model will involve the utilization of various Python libraries and tools, including:

* + **OpenCV**: For video capture and processing.
  + **Keras**: For model development and training.
  + **TensorFlow**: As the backend for Keras, providing efficient computations.
  + **Pygame:** To alert the driver with alarm
  + **Pymongo:** To connect to mongodb

The code will be structured into modules or functions to enhance readability and maintainability. Implementation will adhere to best practices, including code documentation, error handling, and appropriate utilization of libraries.

A diagram of a computer flowchart

Description automatically generated

## Results:

The performance of the drowsiness detection model will be evaluated based on metric such as accuracy. This metric will provide insights into the model's ability to accurately identify instances of drowsiness. The real time detection is done using the laptop camera which detects the drowsiness along with alarm beep, sends email notification and stores image in MongoDB. However, there is some latency in video capturing and it is important to consider that video quality and individual variability can influence the accuracy of the model.

## Challenges Faced:

Throughout the project, several challenges were encountered, including:

* **Data Collection:** Acquiring a diverse and representative dataset is challenging.
* **Preprocessing:** Addressing varying lighting conditions, camera angles, and facial orientations during preprocessing.
* **Model Selection:**Determining the optimal architecture and hyperparameters for the model.
* **Training Time and Resources:** Dealing with the computational intensity of training deep learning models.
* **Real-Time Performance:** Achieving real-time performance in drowsiness detection.(fro example:detection of face without latency using laptop camera.)

## Limitations:

There are certain limitations to consider:

* **Dependence on Video Quality:** The accuracy of the drowsiness detection model may rely on the quality of the video feed.
* **Limited Generalization:** The model may not perform well on drivers or conditions significantly different from the training data.

**8. Tutorial to run the project:**

**🡺 Environment Setup:** Initially you need Python installed on your system. If not, download the latest version of Python from the official Python website (https://www.python.org). You also need to install the required libraries OpenCV, Keras, and TensorFlow. We used VSCode editor and MongoDB compass.

**Below are the commands to be executed in a command prompt or a terminal to install these libraries**:

pip install opencv-python

pip install keras

pip install tensorflow

pip install pymongo

pip install pygame

* **Download and Extract the Project Files:** You need to Download the project files and extract them into a dedicated folder on your computer.
* **Note:** If you want to just run and check the detection you can skip the dataset collection and model training and just run the **detect.py** file as we have used the pre-trained model in the detection.
* **Dataset Collection:** Once you extract the files, You can Collect a dataset or download the dataset from: <http://mrl.cs.vsb.cz/eyedataset>, this dataset is mainly used for the detection of eyes and their parts, gaze estimation, and eye-blinking frequency which are important tasks in computer vision. It comprises of images representing both drowsy and awake states as open eyes and close eyes which ensures that the dataset is diverse and accurately reflects the target scenarios.
* **Extract the zip file in the same folder as the project files and then run the Python script file:** The python script file **data\_preparation.py** will separate the data into open and close eyes folder. Now create two folders (train and test) and then create open and close eyes folder in both train and test. Finally cut and paste about 20% of images from both close and open eyes folder into test data’s close and open eyes, and the rest of the images in train folder.
* **Model Training:** After separating data into open and close eyes folder, you must open the Python script file **model\_training.py** in a suitable code editor or integrated development environment (IDE). Initiate the training of the drowsiness detection model by executing the following command: **python model\_training.py** which allows the training process to complete. The script will provide progress updates and display metrics related to the training. Finally, a model would be created and stored in the model’s folder.
* **Running the Drowsiness Detection Program:** Once the model is trained, you can employ it to detect drowsiness in real-time video feeds. Connect a webcam or other suitable video source to your computer. Launch the program by executing the following command in the command prompt or terminal: **python detect.py.**

**Note before running the program**:(update the program with your mongodb server and also for the email it requires sender and receiver's gmail address the sender has to have Auth2 enabled in his account and has to create a new app password for a test app through google account webpage)

* The program will start capturing video frames from the connected source in real-time.
* The drowsiness detection model will analyze each frame, detecting signs of drowsiness.
* In the event of drowsiness detection, the program will issue an alert to notify the driver ,send an email notification to concerned people and captured image is also pushed to the database.
* To exit the program, press the 'Q' key on your keyboard.
* **Viewing image from DB:** To view the image that is stored in the DB run the ViewImageFromDB.py program it shows the image.

By following the above step-by-step instructions, you can successfully run the driver drowsiness detection progam.