Drowsiness Detection System

Using CNN

INTRODUCTION

In today's fast-paced world, road safety remains a paramount concern. Drowsy driving, often underestimated, poses a significant threat to drivers, passengers, and pedestrians alike. The consequences of a drowsy driver's reduced reaction times and impaired alertness can be catastrophic, leading to accidents that claim lives and cause substantial economic losses.

This project focuses on developing a Drowsiness Detection System, an innovative solution aimed at mitigating the risks posed by drowsy driving. By leveraging advancements in artificial intelligence and computer vision, our system aims to accurately identify signs of drowsiness in real-time, enabling timely interventions to prevent accidents.

A drowsiness detection system, originally designed for applications like driver safety,can be adapted and applied to various other domains where alertness and vigilance are crucial. Here are some additional contexts and applications where a drowsiness detection system could be beneficial:

- **Education and Learning**
- **Security and Surveillance**
- **Workplace Safety etc.**

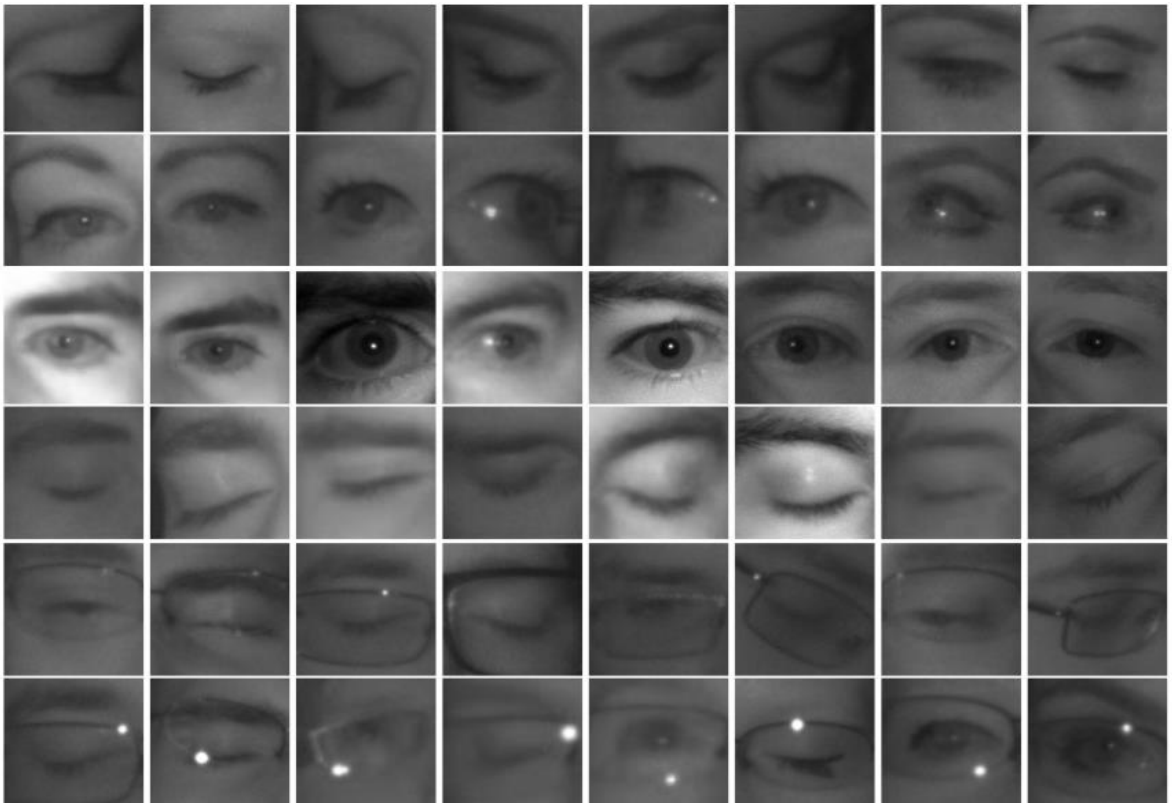
Through this report, we present the development process, methodology, and outcomes of our Drowsiness Detection System. By harnessing the power of AI, we aspire to contribute to a safer driving environment, where accidents caused by drowsiness are minimized, and lives are protected.

EXPLORE THE DATASET

The Dataset Used here is taken from the **MRL Eye Dataset** . the large-scale dataset of human eye images. This dataset contains infrared images in low and high resolution, all captured in various lightning conditions and by different devices. The dataset is suitable for testing several features or trainable classifiers. this dataset consist of **84,898 images** of 37 different persons (33 men and 4 women).

But here in this project only 20551 images are used for Training the Model and 3129 images are used as testing Data.

The example images from this dataset of open and closed eyes are shown below.



TRAINING THE MODEL

While training the model different different approaches are used for better accuracy scores. some pretrained models used like DensNet , InceptionV3 , VGG16 and also trained the model from skretch. and the model which gives best score is used in the project.

Baseline Model : While training the model from skretch , different numbers of layers and different numbers of kernals are taken and accuracy score is measured. The best structure of them is choosen which consist of three **Convolutional layer** and Maxpooling layers attached with them.

- **First Layer** : A convolutional layer with 16 kernels each of size 3 and **"Relu"** activation function is used with no padding. A max pooling layer with window 2x2 Is attached.

- **Second Layer**: A convolutional layer with 32 kernels each of size 3 and **"Relu"** activation function is used with no padding. A max pooling layer with window 2x2 Is attached.
- **Third Layer**: A convolutional layer with 64 kernels each of size 3 and **"Relu"** activation function is used with no padding. A max pooling layer with window 2x2 Is attached.

After these three layer a flatten layer is added for transform the multidimensional output of the convolutional and pooling layers into a one-dimensional vector. And then two dense layers are added with dropout layers of 0.5 dropout rate.

In the last, output layer is added with "sigmoid" activation.

Result : As a result we got here **0.9088 accuracy** on training data and **0.8109 accuracy** on validation data.

Retraining with VGG16: VGG16 is a well-established convolutional neural network (CNN) model that has been pretrained on extensive datasets, making it an ideal candidate for various computer vision tasks. Our primary objective was to tailor this model for a specific task by incorporating custom-designed fully connected layers with specific configurations.

We introduced two distinctive fully connected layers with kernel sizes of 64 and 32,respectively. These fully connected layers, To adapt the VGG16 model for our task, also known as dense layers, are essential elements in deep neural networks, facilitating the extraction of intricate relationships within the data. Our choice of kernel sizes was intended to provide the layers with appropriate capacity to capture nuanced patterns present in our dataset.

For the activation function of our custom layers, we employed the Rectified Linear Unit (ReLU), a widely-used non-linear activation function in neural networks. By introducing non-linearity, ReLU enables the model to learn complex mappings between inputs and outputs, thus enhancing its representation capabilities. The ReLU activation function replaces negative input values with zero while maintaining positive values, which fosters the model's ability to understand intricate relationships and mitigate issues like vanishing gradients.

Result : As a result for this we got **accuracy: 0.8750** on training data and **Val_accuracy: 0.8041** on validation data.

Retraining with InceptionV3 : In this project, we undertook the task of retraining a machine learning model using the InceptionV3 architecture as a foundation.

InceptionV3 is a widely recognized convolutional neural network (CNN) model that has been pretrained on a large dataset, making it an excellent starting point for various computer vision tasks. The objective was to repurpose this model for a specific task by adding a custom fully connected layer with certain specifications.

To adapt the InceptionV3 model to our task, we introduced a novel fully connected layer with 64 kernels. A fully connected layer, also known as a dense layer, is a fundamental component in deep neural networks that helps in learning complex relationships between features. Each kernel in the layer performs a set of transformations on the input data. We chose a kernel size of 64 to provide the layer with a suitable capacity to capture intricate patterns within the data.

For the activation function of our custom layer, we opted for Rectified Linear Unit (ReLU), a non-linear activation function commonly used in neural networks. ReLU introduces non-linearity to the model, enabling it to learn and represent a wider range of functions. The ReLU activation function replaces negative values with zero while leaving positive values unchanged, which facilitates the model's ability to capture complex relationships without suffering from vanishing gradient problems.

Result : As a result for this we got **accuracy: 0.9245** on training data and **Val_accuracy: 0.8389** on validation data.

I have also did fine tuning with InceptionV3 and got result **0.9346** and **0.8333** on training and validation data respectively.

Retraining with DenseNet : DenseNet is a cutting-edge convolutional neural network (CNN) model that has undergone pretraining on extensive datasets, making it a compelling candidate for various computer vision tasks. Our core objective was to fine-tune this model for a specific task by integrating custom-designed fully connected layers featuring specific configurations.

We incorporated two distinct fully connected layers with kernel sizes of 64 and 16, respectively. These fully connected layers, also referred to as dense layers, play a pivotal role in deep neural networks, enabling the extraction of intricate relationships embedded within the data. Our selection of kernel sizes was made strategically to ensure that these layers possess adequate capacity to capture subtle patterns intrinsic to our dataset.

For the activation function of our custom layers, we employed the Rectified Linear Unit (ReLU), a widely-utilized non-linear activation function in neural networks.
By introducing non-linearity, ReLU empowers the model to learn intricate mappings between inputs and outputs, thereby enhancing its ability to represent complex relationships within the data. The ReLU activation function replaces negative input

values with zero while retaining positive values, mitigating challenges such as vanishing gradients and fostering the model's capacity to comprehend intricate patterns.

Our model architecture harmoniously combines the innate feature extraction capabilities of the DenseNet pretrained model with our bespoke fully connected layers incorporating kernel sizes of 64 and 16, both integrated with the ReLU activation function.

Result : As a result for this we got **accuracy: 0.9062** on training data and **Val_accuracy: 0.8464** on validation data.

After Training on all these type of models. We got the best Score from DenseNet Model. Also did Model Ensembeling but that too not give a better score than the DenseNet.

OUTPUT

The developed drowsiness detection system employs a CNN with a pretrained Dense The developed drowsiness detection system employs a Convolutional Neural Network (CNN) with a pretrained DenseNet model to achieve an accuracy of approximately 85%.The system is integrated with the OpenCV library and utilizes two pretrained HaarCascade classifiers: **haarcascade_eye_tree_eyeglasses** and **haarcascade_frontalface_default**. A specific logic is implemented to trigger an alarm sound when the detected eye closure exceeds a threshold of 0.5 for a predefined duration.

Key Components:

Pretrained DenseNet Model: The core of the system, this model is used for feature extraction and classification. It has been fine-tuned to recognize drowsiness based on input eye images.

OpenCV Integration: The system benefits from OpenCV's robust image processing capabilities, enabling tasks such as image capture, face detection, and eye detection.

Haar Cascade Classifiers:

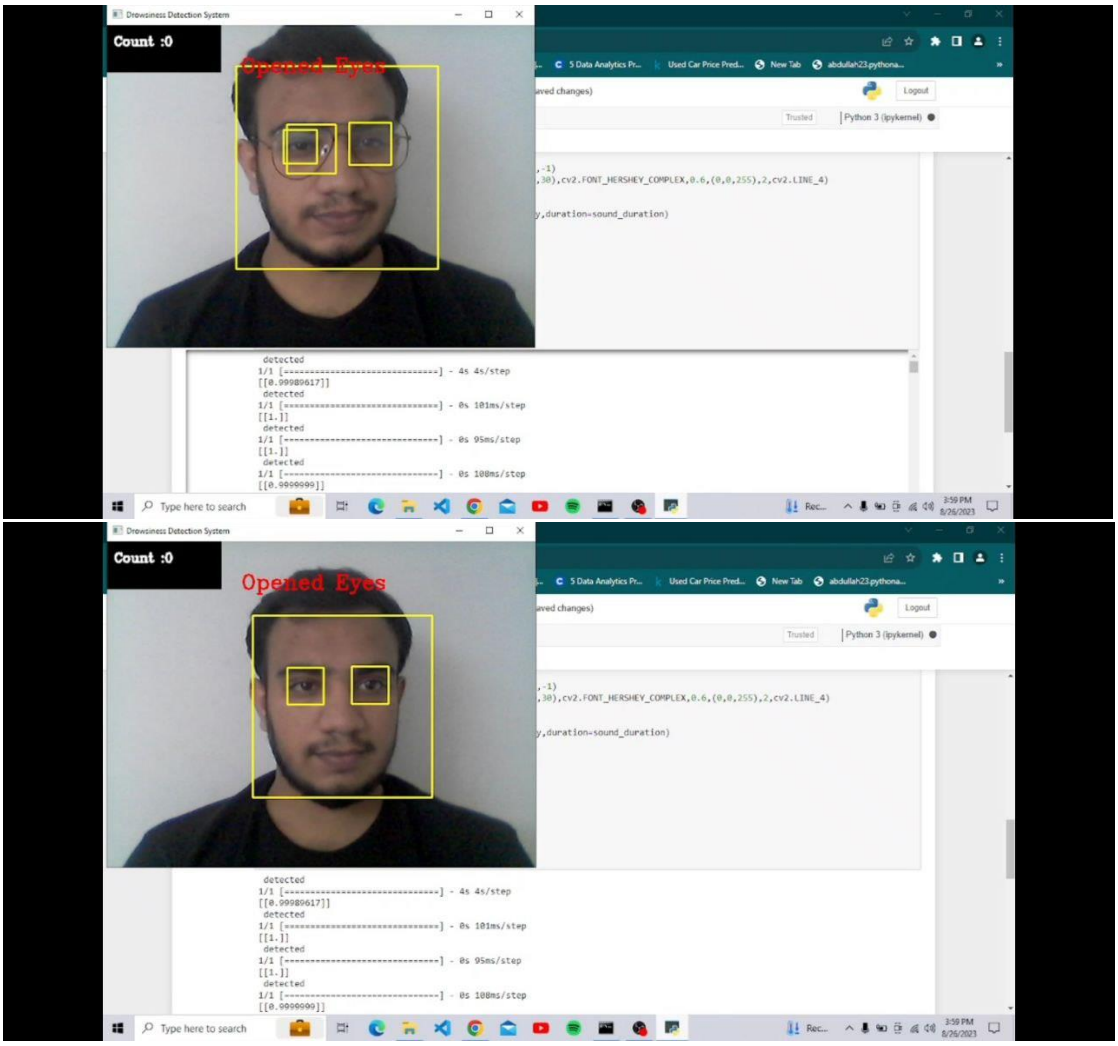
- **haarcascade_eye_tree_eyeglasses**: Detects eyes in the input image.
- **haarcascade_frontalface_default**: Detects faces in the input image, allowing for precise eye region identification.

Drowsiness Detection Logic:

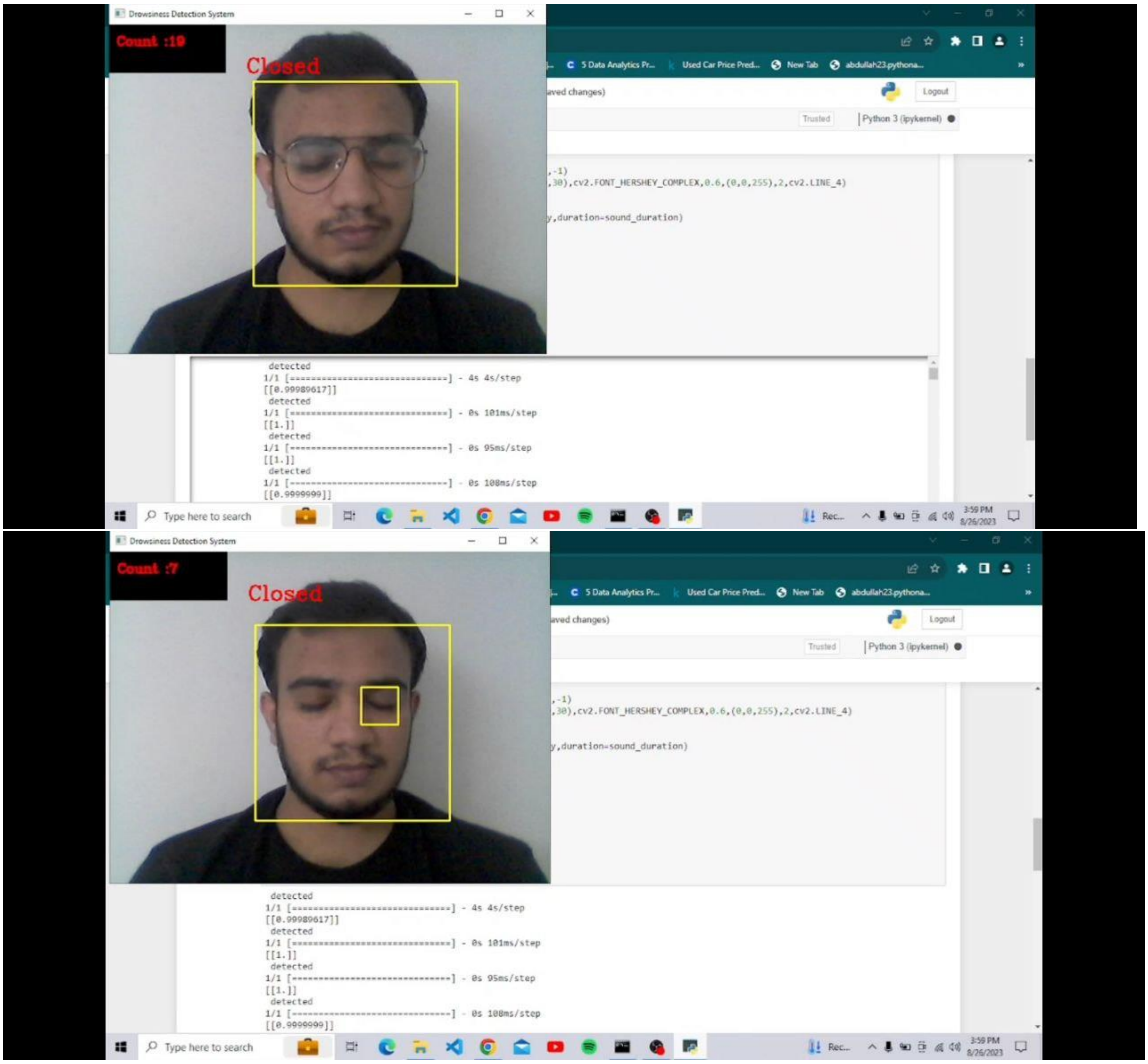
- The system continuously analyzes image frames to monitor eye states.
- When the model's prediction indicates eye closure (with a confidence level exceeding 0.5), a counter is incremented.
- If the counter reaches a predefined duration (representing a specific period of continuous eye closure), an alarm sound is activated.

Screenshots :

Opened Eyes:



Closed :



When eyes goes closed for more than a specific time (here 20 Second) . An Alarm is triggered for alerting the Driver. And counter is set to 0 again.

Future Enhancements: Mention that, in future work, the system can be further improved by exploring additional datasets, incorporating data augmentation

techniques, optimizing for real-time processing, and considering ethical and legal aspects in its deployment.

Conclusion: Conclude by summarizing the system's significance in addressing driver fatigue and enhancing road safety, underscoring its potential as a valuable addition to the field of driver assistance systems.