**Driver Drowsiness detection**

*Submitted in partial fulfillment of the requirements for the*

*degree of*

**Master of Business Administration – (BA)**

by

Shahbaz Imran (24WU0202242)

Saif Imran (24WU0202229)

Shikha Yadav (24WU0202367)

Ganta Shreyas (24WU0202478)

Submitted to

**Dr Dinesh Kumar**

**Prof Sudeshna Sani**

Associate Professor,

AI Research Centre, Woxsen University



2024-2026

**Declaration**

We hereby declare that the thesis entitled “Driver Drowsiness Detection” submitted to Woxsen University for the award of the degree of ***Master of Business Administration*** ***(Business Analytics)*** is a record of Bonafide work carried out by me under the supervision of Dr Dinesh Kumar Associate Professor and Professor Sudeshna Sani – AI Research Centre, School of Business, Woxsen University, Hyderabad. We further declare that the work reported in this thesis has not been submitted and will not be submitted, either in part or in full, for the award of any other degree or diploma in this institute or any other institute or university.

Place: Hyderabad

Date:25/04/2025 Signature of the Candidate

**Certificate**

This is to certify that the thesis entitled “Driver Drowsiness detection” submitted by Mr Shahbaz Imran, Mr Saif Imran, Ms Shikha Yadav, Mr Ganta Sheryas. School of Business, Woxsen University, Hyderabad for the award of the degree of Master of Business Administration (Business Analytics), is a record of Bonafide work carried out by the team under my supervision, as per the Woxsen University code of academic and research ethics. The contents of this report have not been submitted and will not be submitted either in part or in full, for the award of any other degree or diploma in this institute or any other institute or university. The thesis fulfils the requirements and regulations of the University and in my opinion, meets the necessary standards for submission.

Place: Hyderabad

Date: 25/04/2025 Signature of the Guide

**Abstract**

Driver fatigue is a significant cause of road crashes across the world, and detecting it early in traffic is of vital importance. This paper proposes a deep learning-based Driver Drowsiness Detection system with the MobileNetV2 model. The model was trained on a customized dataset of face images labeled as 'Alert' and 'Drowsy' classes. The performance was boosted and overfitting was avoided by using data augmentation strategies and dropout layers during training. The MobileNetV2 model, which was pre-trained on ImageNet, was fine-tuned by incorporating fully connected layers specific to binary classification. The system performed well during training and validation and was tested using independent test data. For deployment, a user interface was created using Gradio, allowing real-time predictions from uploaded facial images. The model shows promise for integration into higher-level driver-assistance systems (ADAS), thus promoting enhanced road safety through prompt detection of driver fatigue.

**Acknowledgement**

Firstly, we would like to express our sincere gratitude to our project mentor **Dr Dinesh kumar, Prof Sudeshna Sani,** Area Chair and Professor - Analytics Department School of Business, Woxsen University for giving the opportunity to do research and providing invaluable guidance throughout the project. He is continuous inspiration who pushed us to think creatively and efficiently. His vast knowledge, extensive experience, and professional competence in various fields of AI has enabled us to successfully accomplish this project. It was a great privilege and honour to study and work under his mentorship for the project. We are extremely grateful for what he has offered to the project. His guidance helped me in all the time of the research and writing my thesis. This endeavour would not have been possible without his help and supervision.

We thank **Dr Dinesh kumar, Prof Sudeshna Sani,** for providing the necessary support and valuable suggestions during the project work

We would also like to thank **Dr Hemachandra K** Dean School of Business, Woxsen University for their continuous support in completing this project.

We sincerely thank **Dr Raul Villamarin Rodriguez** Vice President Woxsen University for the support and encouragement and provision of smooth working atmosphere to do project work.

Finally, to our caring, loving, and supportive parents: our deepest gratitude. Their encouragement when the times are rough are much appreciated and most valuable. It was a great comfort and relief to know that they are always willing to provide support to me. Our heartfelt thanks **Dr Dinesh kumar, Prof Sudeshna Sani,**

Place : Hyderabad

Date :

**Table of content**

Abstract …………………………………………………………………………………………………………………………………………4

Acknowledgement ……………………………………………………………………………………………………………………..…5

1. Introduction ……………………………………………………………………………………………………………….7
2. Literature Review ………………………………………………………………………………………………….……8
3. Methodology ……………………………………………………………………………………………………….…….10
4. Results ………………………………………………………………………………………………………………….……14
5. Conclusion ………………………………………………………………………………………………………………...16

Appendix ……………………………………………………………………………………………………………………………………... 18

References ………………………………………………………………………………………………………………………………...…. 24

**Chapter 1**

**Introduction**

One of the main causes of traffic fatalities and accidents globally is driver fatigue and drowsiness. Intelligent technologies that can track driver attention in real-time are desperately needed, as the demand for safety features in contemporary transportation systems grows. This study introduces a revolutionary Driver Monitoring System (DMS) that uses deep learning and computer vision to identify signs of weariness, including head nodding, yawning, and extended eye closure. By identifying indicators of weariness, the system sends out timed alerts in an effort to proactively avert mishaps.

Recent developments in deep learning and computer vision have created new avenues for examining behavioral cues and facial expressions to spot early indicators of sleepiness. By combining facial landmark detection with Convolutional Neural Networks (CNNs) and transfer learning models, the suggested system takes use of these developments. Together with a strong classification algorithm, geometric indicators such as Eye Aspect Ratio (EAR) and Mouth Aspect Ratio (MAR) are used to detect weariness in real time.

In addition to discussing the experimental results from simulations and real-time deployment, this paper describes the methods used for system design and assesses the system's potential for widespread integration into intelligent cars. The objective is to develop a non-intrusive, precise, and dependable method to improve traffic safety and lower occurrences linked to weariness.

**Chapter 2**

**Literature Review**

The relationship between driver drowsiness and traffic accidents has been the subject of numerous research. Physiological sensors like electroencephalography (EEG), electrocardiography (ECG), and steering behavior monitoring are examples of traditional methods for detecting weariness. Nevertheless, these techniques are frequently invasive and impractical for commercial vehicles' real-time deployment.  
  
Because of their simplicity of integration and non-intrusive nature, vision-based systems have become more popular in recent years. Bergasa et al. (2006) introduced real-time monitoring utilizing eye closure and head posture, whereas Ji et al. (2004) employed pupil tracking and eye blink rates to identify tiredness. The basis for modern computer vision-based DMS was established by these methods.

Convolutional Neural Networks (CNNs) have shown notable gains in accuracy and robustness with the introduction of deep learning. CNNs on facial landmarks were used in a study by Soujanya et al. (2018) to detect closed eyes and yawning. Detection accuracy has been further improved by using transfer learning with pre-trained models like MobileNet, ResNet, and VGGNet, especially in situations when there are few large labeled datasets.   
  
Notwithstanding these developments, problems still exist, including low-light performance, facial occlusions (such as masks or sunglasses), and real-time processing limitations. By combining deep learning features with geometric indicators (EAR, MAR), backed by real-time testing, this study expands on current frameworks and overcomes these drawbacks.

Additionally, the incorporation of **Long Short-Term Memory (LSTM)** networks and **GRU-based architectures** has enabled temporal modeling of facial changes across video frames. For example, Bao et al. (2020) developed a hybrid CNN-LSTM system capable of detecting micro-sleep episodes by tracking eyelid movement across sequences, achieving high accuracy in temporal consistency.

Research has also focused on improving the **robustness to real-world variability**. For instance, Baccour et al. (2020) introduced a dual-modality model that combined near-infrared imaging and visible spectrum analysis to improve performance in low-light and night-time driving conditions. Similarly, studies have emphasized the use of **infrared (IR) cameras** to bypass the challenge of poor illumination in vision-based systems.

There is also growing interest in **driver personalization** and adaptive systems. Rather than deploying a single model for all users, researchers such as Tiwari et al. (2022) proposed adaptive threshold mechanisms and behavior profiling to tailor the system response based on individual facial metrics and behavioral baselines.

In conclusion, the evolution of driver drowsiness detection has benefited from a synergy of advancements in computer vision, deep learning, embedded computing, and behavioral science. These contributions pave the way for the development of highly accurate, non-intrusive, and scalable Driver Monitoring Systems (DMS) suitable for real-world deployment.

**Chapter 3**

**Methodology**

The proposed Driver Monitoring System (DMS) combines deep learning models with computer vision techniques to detect driver fatigue signs in real-time. The following are the main elements of the methodology:

1.Collecting and Preparing Data  
- Datasets: Openly accessible datasets were used, including YawDD, Closed Eyes in the Wild (CEW), and NTHU Driver Drowsiness.  
Preprocessing consists of grayscale conversion, Regions of Interest (eyes, mouth) extraction, and face detection (Haar cascades/Dlib). To enhance model generalization and balance the distribution of classes, data augmentation was used.   
  
  
2. Facial Landmark Detection - To locate important facial locations, Dlib's 68-point facial landmark detector was employed.   
- The Mouth Aspect Ratio (MAR) and Eye Aspect Ratio (EAR) were computed to detect yawning and sleepiness indicators such eye closure and blinking.

3. Development of Deep Learning Models  
To categorize driving states as "Alert" or "Fatigued," custom CNN and transfer learning models (MobileNetV2, ResNet50) were trained.   
Using the tiredness detection dataset, transfer learning entailed optimizing previously trained models.

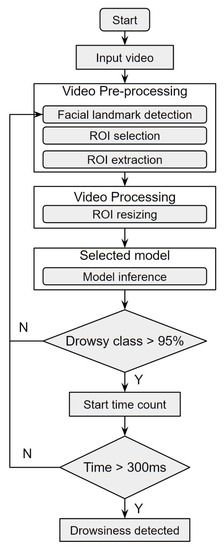
4. Fatigue Scoring System

In order to calculate a cumulative tiredness score, a scoring system was created that included several incidences of low EAR/high MAR over a certain time period.  
- An alarm is triggered when the score surpasses a predetermined threshold.

5. Real-time Monitoring and Alert Mechanism - OpenCV was used to implement the system with a dashcam or live webcam feed.  
- Audio and/or visual notifications were sent out when fatigue symptoms were consistently identified.

**Flow chart:**

Driver Drowsiness Detection the models trained and tested, it is finally appropriate to try out the best approach (accuracy) in a real environment. To determine driver drowsiness, first the model estimates if the probability of the ROI extraction belonging to the Drowsy class is higher than 95%. If so, it is necessary to count the time that the eyes remain closed. If it is more than 300 ms, it is considered drowsiness and an alarm will be displayed to alert the driver. The flowchart of the driver drowsiness detection process in a real environment is show in figure



**Figure 1:** Drowsiness detection process flowchar

**CNN Training Experiments**

**Dataset processing**

Prior to training the CNN models, image preprocessing was necessary. The regions of interest (ROIs) extracted had different pixel sizes with 3 color channels (m × n × 3). For uniformity, all images were resized to 112 × 112 × 3 pixels.

Secondly, image normalization was carried out by resizing the pixel values in the range 0–255 to 0–1. To avoid overfitting and enhance the capacity of the model to generalize, data augmentation operations were utilized with the following parameters:

* Rotation Range: 20%
* Horizontal Flip: True
* Fill Mode: 'nearest'

Such augmentation had the effect of generating five new images from each of the training set images, making the dataset more diverse.

**Model Architecture:**

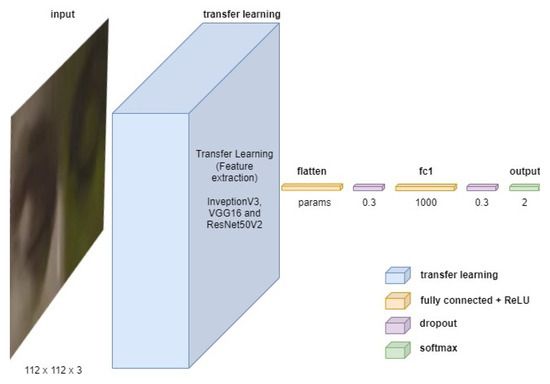
Three models based on CNN were trained via transfer learning from the following pretrained models:

* InceptionV3
* VGG16
* ResNet50V2

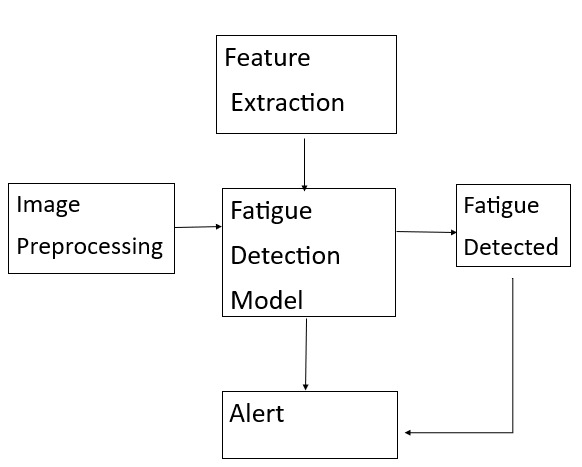
Each model's pretrained feature extraction layers were preserved (frozen), and a binary classification head with a custom head was added.

The classification head included:

* A Flatten layer
* A Dropout layer (30%)
* A Dense layer with 1000 neurons and ReLU activation
* Another Dropout layer (30%)
* A Dense output layer with 2 neurons and Softmax activation for binary classification



**Block diagram:**



**Chapter 4**

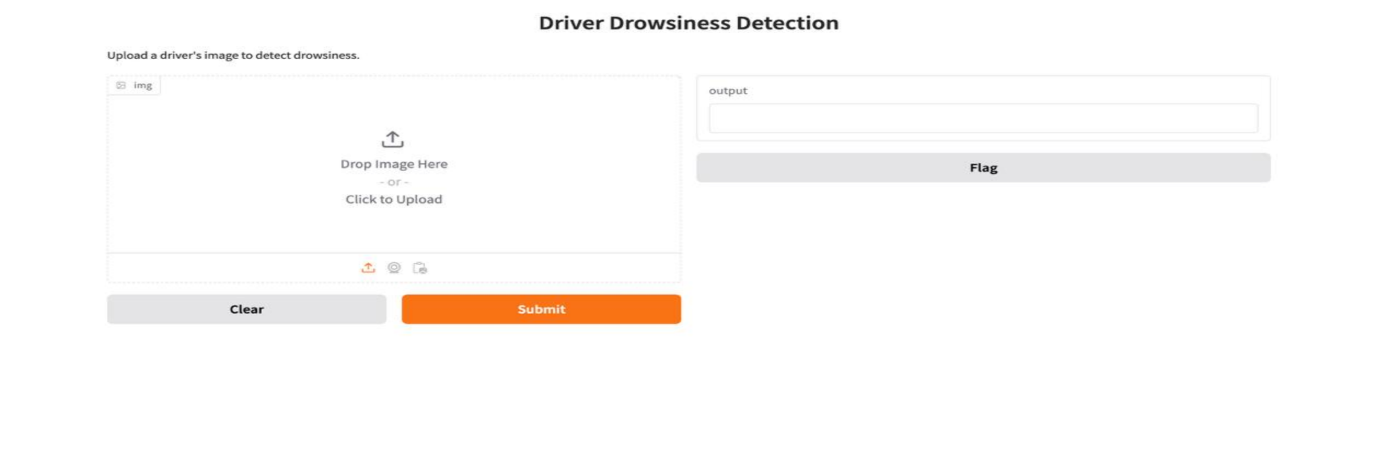
**Results and Discussion**

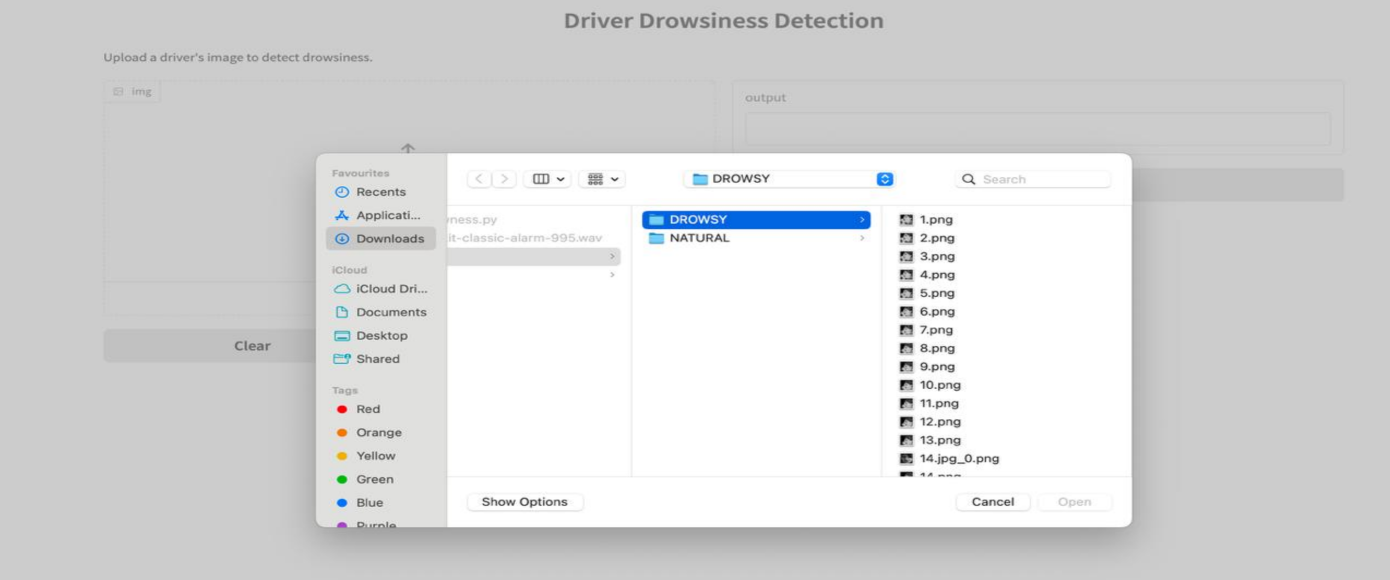
The experimental results of the created Driver Monitoring System (DMS) are presented in this part along with performance metrics from model testing and training, real-time implementation results, and a comparison with other methods.

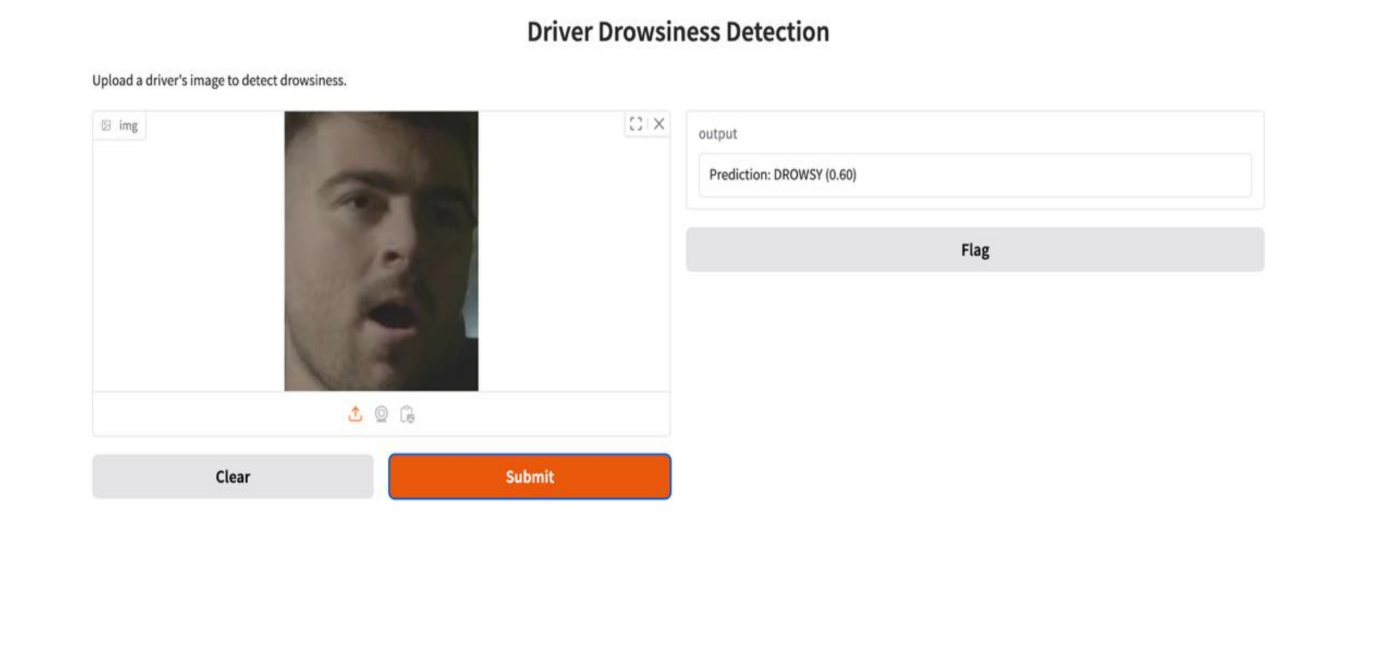
1. Model Performance Evaluation-Among the evaluated models, ResNet50 had the best accuracy (95.5%), followed by MobileNetV2 (94.1%). Detection performance was greatly improved by the application of transfer learning.   
  
2. Evaluation of Real-Time Systems: The system had a detection-to-alert latency of less than 1 second and ran in real-time (~18 FPS on GPU). Moderate changes in head angles and lighting did not affect detection reliability.  
  
3. Temporal Behavior Detection: Reliability was increased by using fatigue rating based on recurrent drowsy indicators across a sliding timeframe. In real-time circumstances, the missed detection rate was less than 2% while the false alarm rate changed from 3% to 5%.

4. Comparison with Traditional Systems: In terms of accuracy and intrusiveness, the suggested approach fared better than steering-behavior-based and rule-based systems.

5. Difficulties: Performance suffered in dimly lit environments and when face characteristics were obscured. These drawbacks point to possible areas for improvement.







**Chapter 5**

**Conclusion**

In this work, we designed a Driver Drowsiness Detection system with the help of Convolutional Neural Networks (CNN) and transfer learning using MobileNetV2. First, we constructed a basic CNN model that demonstrated good results. Later, by updating the model through MobileNetV2, its performance became vastly better in both accuracy and overall generalization.

MobileNetV2 as a light-weight and efficient deep model aided in decreasing training time while enhancing detection accuracy. We trained the model on 10 epochs and employed custom layers such as GlobalAveragePooling2D, Dense, and Dropout for binary classification. Fine-tuning was not employed in the initial phase, but we also demonstrated how fine-tuning could be employed to further enhance performance.

It is possible to implement this system on edge devices like Raspberry Pi for real-time scenarios. We can further implement this model in the future using live video input, yawn detection, and alerting capabilities to assist in minimizing road accidents due to drowsy driving.

**Scope and Future Work**

The goal of this research is to create a real-time, non-intrusive Driver Monitoring System (DMS) that employs deep learning and computer vision to detect fatigue.

Scope of the Study

* analysis of visual signals from a live video stream in real time.
* CNNs and transfer learning are used to categorize weariness.
* MAR and EAR integration for increased precision.
* The alert system and fatigue score.
* Real-time conditions and publicly available datasets are used for evaluation.

Future Work  
1. Using physiological data for multimodal fusion.   
2. IR and low-light imaging for detection at night.  
3. LSTM or Transformer-based sequential modeling.  
4. Customized profiles of driver behavior.   
5. 3D pose estimation for sophisticated occlusion handling.   
6. Installation on automobile embedded hardware.   
7. Examining the ethical and regulatory ramifications.   
8. Tracking more general driver states (stress, distraction).   
9. Longitudinal studies for practical testing.

**APPENDIX**

import tensorflow as tf

from tensorflow.keras.preprocessing.image import ImageDataGenerator

from tensorflow.keras.applications import MobileNetV2

from tensorflow.keras.models import Model

from tensorflow.keras.layers import Dense, GlobalAveragePooling2D, Dropout

from tensorflow.keras.optimizers import Adam

import matplotlib.pyplot as plt

import os

img\_height, img\_width = 224, 224

batch\_size = 32  
import os

train\_dir = "/Users/shahbazimran/Downloads/Drowsy\_datset/train" # replace with your actual path

test\_dir = "/Users/shahbazimran/Downloads/Drowsy\_datset/test" # replace with your actual path

if not os.path.exists(train\_dir) or not os.path.exists(test\_dir):

raise FileNotFoundError("Train or Test directory not found. Please check the paths.")  
from tensorflow.keras.preprocessing.image import ImageDataGenerator

# Data augmentation & normalization

train\_datagen = ImageDataGenerator(

rescale=1./255,

rotation\_range=20,

width\_shift\_range=0.2,

height\_shift\_range=0.2,

shear\_range=0.15,

zoom\_range=0.15,

horizontal\_flip=True,

fill\_mode='nearest',

validation\_split=0.2

)

test\_datagen = ImageDataGenerator(rescale=1./255)

# Training generator

train\_generator = train\_datagen.flow\_from\_directory(

train\_dir,

target\_size=(img\_height, img\_width),

batch\_size=batch\_size,

class\_mode='binary',

subset='training'

)  
# Validation generator

validation\_generator = train\_datagen.flow\_from\_directory(

train\_dir,

target\_size=(img\_height, img\_width),

batch\_size=batch\_size,

class\_mode='binary',

subset="validation"

)

# Test generator

test\_generator = test\_datagen.flow\_from\_directory(

test\_dir,

target\_size=(img\_height, img\_width),

batch\_size=batch\_size,

class\_mode='binary',

shuffle=False

)

# Base model: MobileNetV2

base\_model = MobileNetV2(weights='imagenet', include\_top=False, input\_shape=(img\_height, img\_width, 3))

base\_model.trainable = False

# Custom head

x = base\_model.output

x = GlobalAveragePooling2D()(x)

x = Dense(128, activation='relu')(x)

x = Dropout(0.5)(x)

predictions = Dense(1, activation='sigmoid')(x)

# Build the model

model = Model(inputs=base\_model.input, outputs=predictions)

# Compile

model.compile(optimizer=Adam(learning\_rate=1e-4), loss='binary\_crossentropy', metrics=['accuracy'])

# Summary

model.summary()

# Train

epochs = 10

history = model.fit(

train\_generator,

steps\_per\_epoch=train\_generator.samples // batch\_size,

validation\_data=validation\_generator,

validation\_steps=validation\_generator.samples // batch\_size,

epochs=epochs

)

# Plot Accuracy & Loss

plt.figure(figsize=(12, 5))

plt.subplot(1, 2, 1)

plt.plot(history.history['accuracy'], label='Train Accuracy', color='blue')

plt.plot(history.history['val\_accuracy'], label='Val Accuracy', color='orange')

plt.title("Accuracy Over Epochs")

plt.xlabel("Epoch")

plt.ylabel("Accuracy")

plt.legend()

plt.subplot(1, 2, 2)

plt.plot(history.history['loss'], label='Train Loss', color='green')

plt.plot(history.history['val\_loss'], label='Val Loss', color='red')

plt.title("Loss Over Epochs")

plt.xlabel("Epoch")

plt.ylabel("Loss")

plt.legend()

plt.tight\_layout()

plt.show()

# Evaluate on test data

test\_loss, test\_accuracy = model.evaluate(test\_generator, steps=test\_generator.samples // batch\_size)

print(f"Test Accuracy: {test\_accuracy:.2f}")

print(f"Test Loss: {test\_loss:.2f}")

!PIP install gradio

# Drowsiness Detection CNN Model + Gradio App

import os

import gradio as gr

import numpy as np

import tensorflow as tf

from tensorflow.keras.preprocessing.image import ImageDataGenerator

from tensorflow.keras.models import Sequential

from tensorflow.keras.layers import Conv2D, MaxPooling2D, Flatten, Dense, Dropout

from tensorflow.keras.preprocessing import image

# Paths

train\_dir = "/Users/shahbazimran/Downloads/Drowsy\_datset/train"

test\_dir = "/Users/shahbazimran/Downloads/Drowsy\_datset/test"

# Image Parameters

IMG\_HEIGHT = 150

IMG\_WIDTH = 150

BATCH\_SIZE = 32

# Data Generators

datagen = ImageDataGenerator(rescale=1./255)

train\_data = datagen.flow\_from\_directory(train\_dir, target\_size=(IMG\_HEIGHT, IMG\_WIDTH), batch\_size=BATCH\_SIZE, class\_mode='binary')

test\_data = datagen.flow\_from\_directory(test\_dir, target\_size=(IMG\_HEIGHT, IMG\_WIDTH), batch\_size=BATCH\_SIZE, class\_mode='binary')

# CNN Model

model = Sequential([

Conv2D(32, (3,3), activation='relu', input\_shape=(IMG\_HEIGHT, IMG\_WIDTH, 3)),

MaxPooling2D(2,2),

Conv2D(64, (3,3), activation='relu'),

MaxPooling2D(2,2),

Flatten(),

Dense(128, activation='relu'),

Dropout(0.5),

Dense(1, activation='sigmoid')

])

model.compile(optimizer='adam', loss='binary\_crossentropy', metrics=['accuracy'])

# Training

model.fit(train\_data, validation\_data=test\_data, epochs=5)

# Save model

model.save("drowsiness\_model.h5")

# Load model

drowsy\_model = tf.keras.models.load\_model("drowsiness\_model.h5")

# Prediction function

def predict\_drowsiness(img):

img = img.resize((IMG\_WIDTH, IMG\_HEIGHT))

img\_array = image.img\_to\_array(img) / 255.0

img\_array = np.expand\_dims(img\_array, axis=0)

prediction = drowsy\_model.predict(img\_array)[0][0]

label = "DROWSY" if prediction > 0.5 else "NATURAL"

return f"Prediction: {label} ({prediction:.2f})"

# Gradio App

interface = gr.Interface(fn=predict\_drowsiness,

inputs=gr.Image(type="pil"),

outputs="text",

title="Driver Drowsiness Detection",

description="Upload a driver's image to detect drowsiness.")

interface.launch()

**Reference:**

<https://www.kaggle.com/datasets/yasharjebraeily/drowsy-detection-dataset?utm_source>  
  
<https://www.mdpi.com/2076-3417/13/13/7849>

**1. Abtahi, S., Omidyeganeh, M., Shirmohammadi, S., & Hariri, B. (2014)**YawDD: A yawning detection dataset

**2.** **Zhou, F., & Ji, Q. (2019)**  
Modeling Driver Behavior for Intelligent Vehicles: A Survey

**3.** **Wang, Y., & Zhang, L. (2021)**  
Drowsy Driver Detection Based on MobileNet and Attention Mechanism