Minor Project

Enhancing Road Safety : Real-time Driver Drowsiness Detection System

National Institute of Technology Patna

Project Leadership

Dr. Prabhat Kumar

Professor, CSE, Prof-in-charge (IT   Services) and Chairman, Computer and IT Committee at National Institute of Technology , Patna

https://www.linkedin.com/in/dr-prabhat-kumar-7977b554

Presented By

Name: Manisha

https://www.linkedin.com/in/manisha-a4a01022b

Name: Khushboo Kumari

https://www.linkedin.com/in/khushboo-kumari-563a8822a

Name: Sweety Kumari

https://www.linkedin.com/in/sweety-kumari-492b14229

Contents

Project Overview and Scope

- The Driver Drowsiness Detection System aims to enhance road safety by detecting signs of driver fatigue or drowsiness in real-time.
- Using advanced computer vision and machine learning techniques, the system monitors the driver's behavior and alerts them if they show indications of drowsiness.
- This project used to prevent accidents caused by sleepy or fatigued drivers, thereby saving lives and reducing road accidents
- Scope of this project is assessing the human behaviour using drowsiness detection.

Project Goals

Objectives

- ❑To detect drowsiness and issuing timely alerts to prevent potential accidents.

Constraints

- ❑Image quality and resolution of videos impact performance of the systems.
- ❑Privacy concerns

Technical Stacks

- **Python** is a general-purpose programming language.We used it for coding and used it’s libraries like numpy , matplotlib for Model Building and Visualization.

- **Jupyter** is an integrated development environment (IDE) for scientific programming in the Python language.

❑**Pytorch** is an open source machine learning framework based on the Python programming language and the Torch library, used for creating deep neural networks.

❑**OpenCV** is a library for image processing and performing computer vision tasks.

❑**YOLOv5** is a state-of-the-art object detection algorithm developed by Ultralytics, designed to be faster, more accurate and easier to use, widely used in computer vision tasks particularly in real-time object detection senerios.

System Requirements

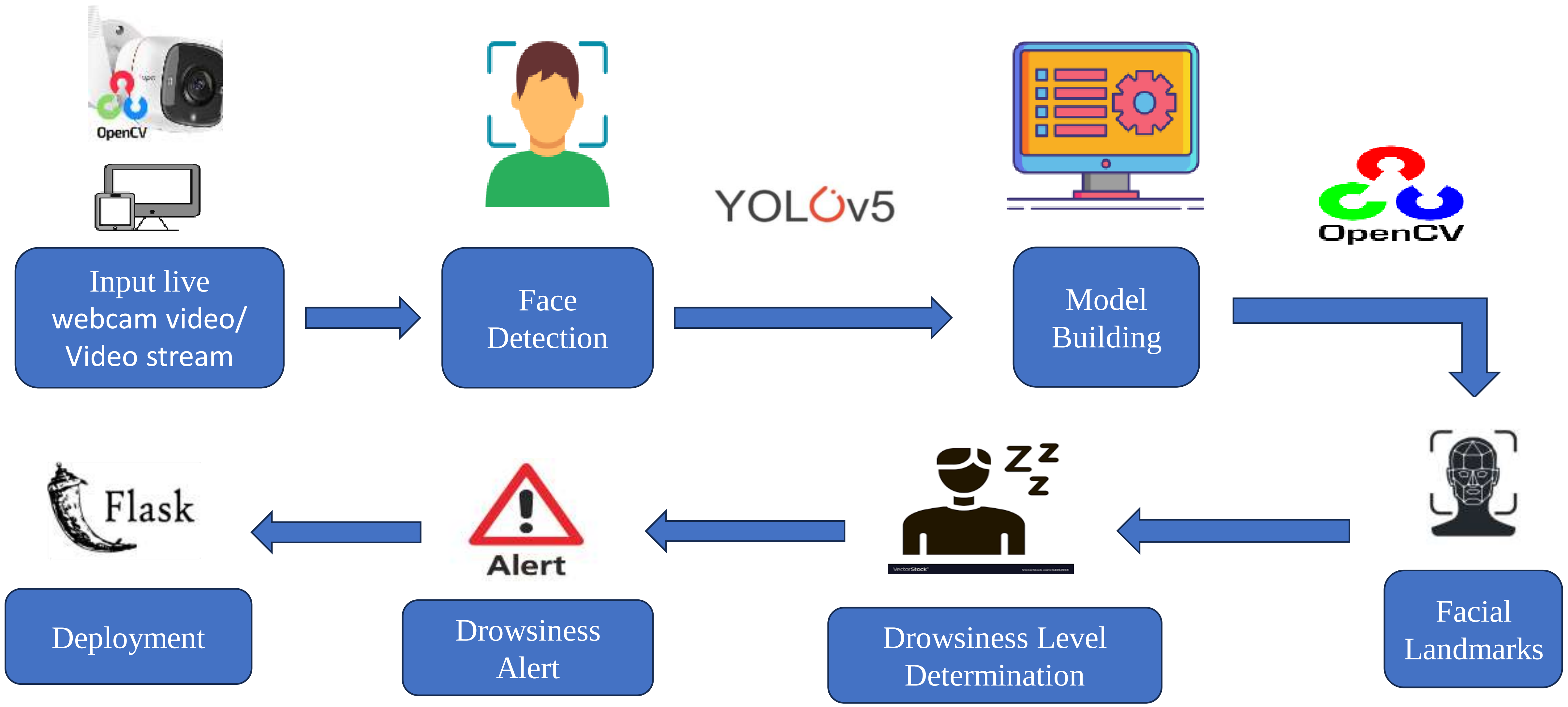
System Requirements

- ❑Memory: 8GB RAM
- ❑CPU: Intel Core i3 and above
- ❑OS: Windows 10 and above

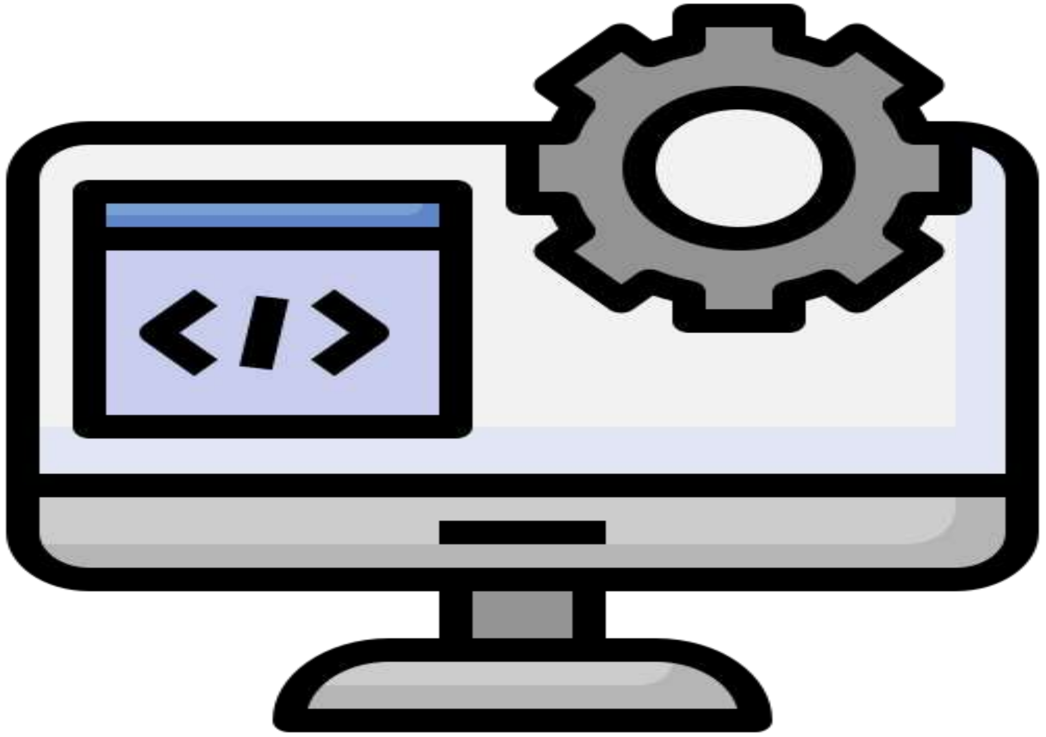
Prepare Environment

- ❑Install python version 3.11.4
- ❑Install Anaconda software to launch IDE platform like jupyter.
- ❑Create a new Environment.
- ❑Install torch Library.
- ❑Git clone yolov5.
- ❑Install OpenCV.

Project Architecture



Model Building



Drowsiness Detection System

- **OpenCV** is used for gathering the images from a webcam and feeding them into a Deep Learning model which will classify whether the person’s eyes are ‘Open’ or ‘Closed’ , person’s mouth ‘Open’ or ‘Closed’, person is nodding or not.

- **YOLO (You Only Look Once)** is a popular object detection algorithm that has revolutionized the field of computer vision. It is fast and efficient, making it an excellent choice for real-time object detection tasks.

Why YOLO ?

Reasons that make YOLO popular for Object Detection :

❑**Speed :** YOLO is extremely fast because it does not deal with complex pipelines. It can process images at 45 Frames Per Second (FPS).

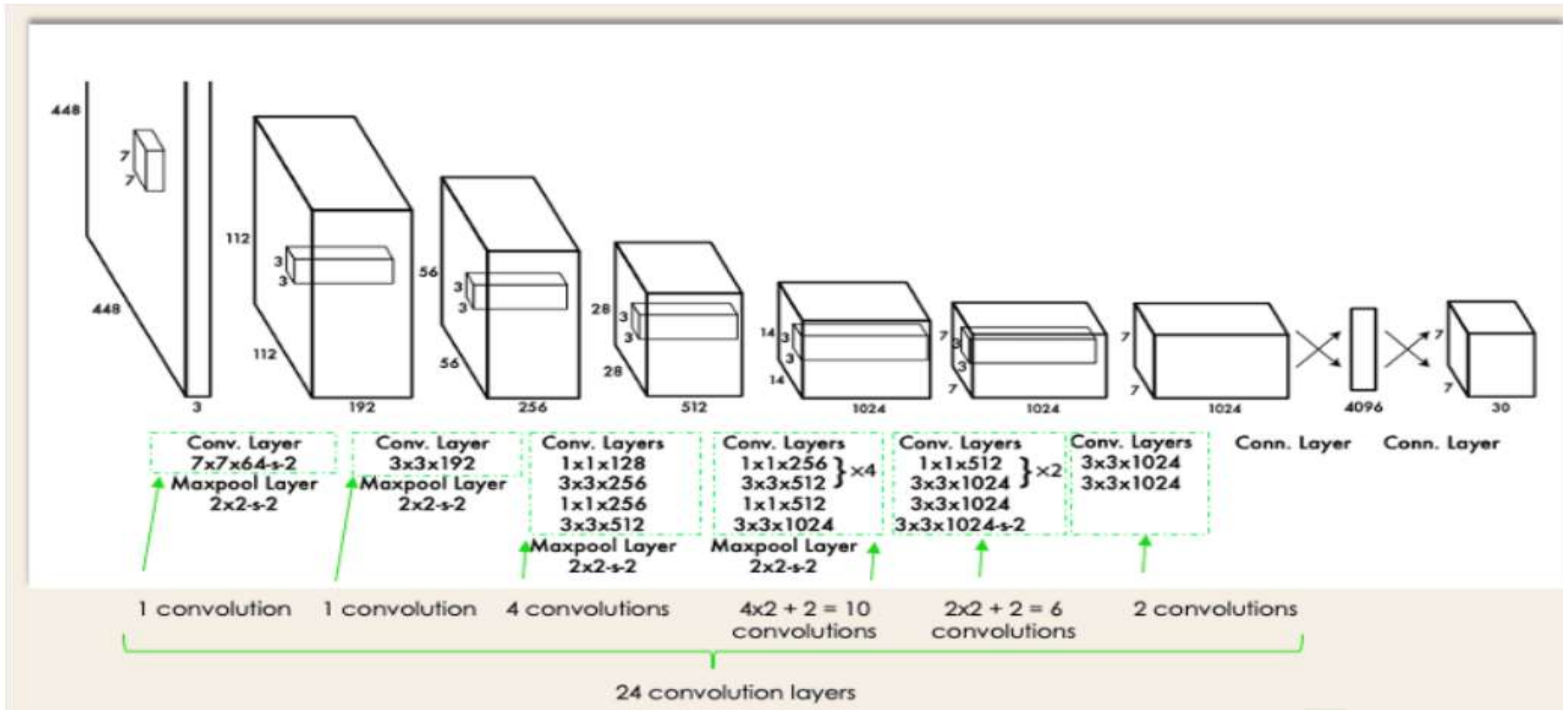
❑**Detection accuracy:** YOLO is far beyond other state-of-the-art models in accuracy with very few background errors.

❑**Good generalization**

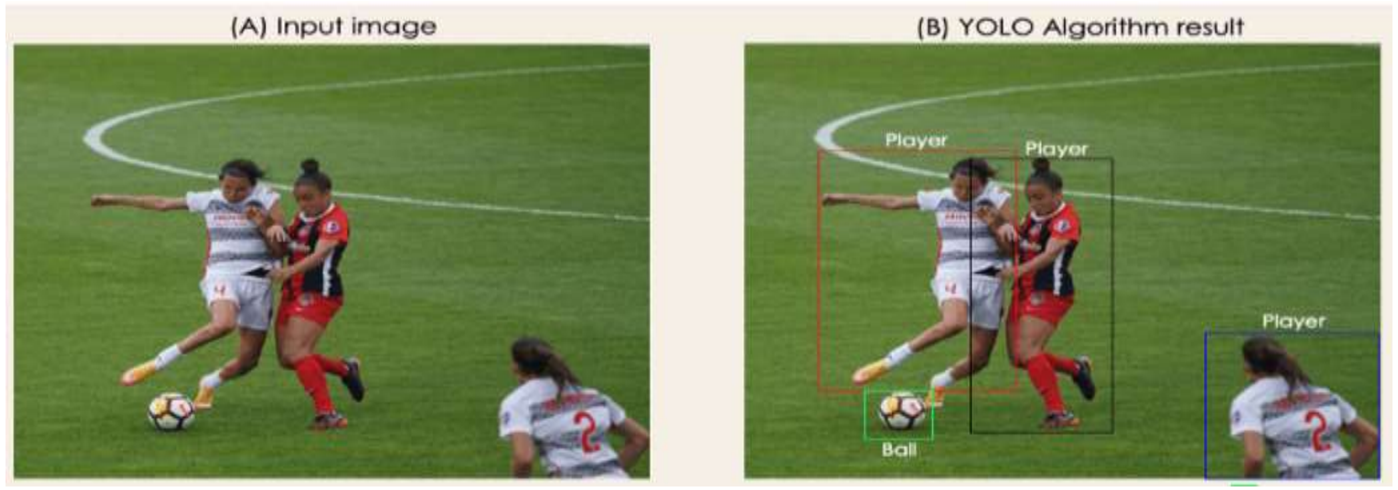
❑**Open-source**

YOLO Architecture

- YOLO architecture is similar to GoogleNet. It has overall 24 convolutional layers, four max-pooling layers, and two fully connected layers.



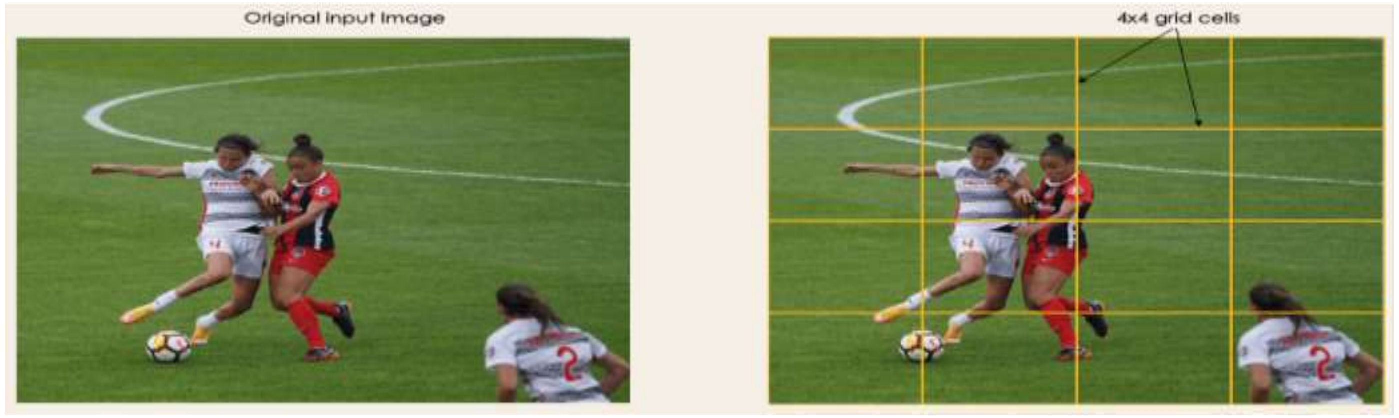
How Does YOLO Object Detection Work?



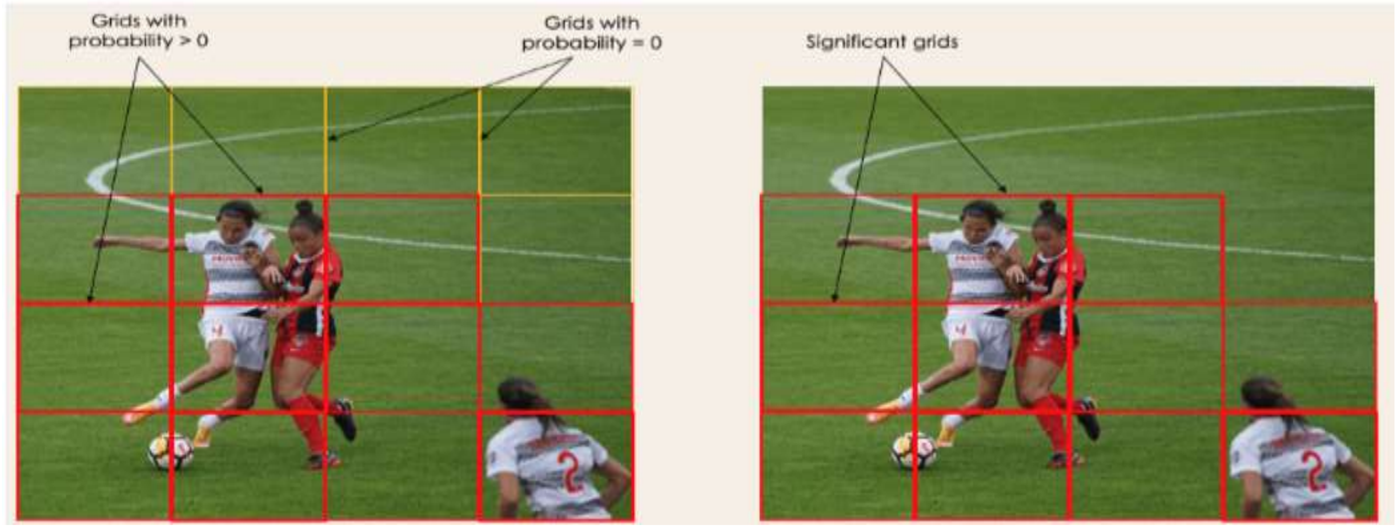
The algorithm works based on the following four approaches:

- Residual blocks
- Bounding box regression
- Intersection Over Unions or IOU for short
- Non-Maximum Suppression.

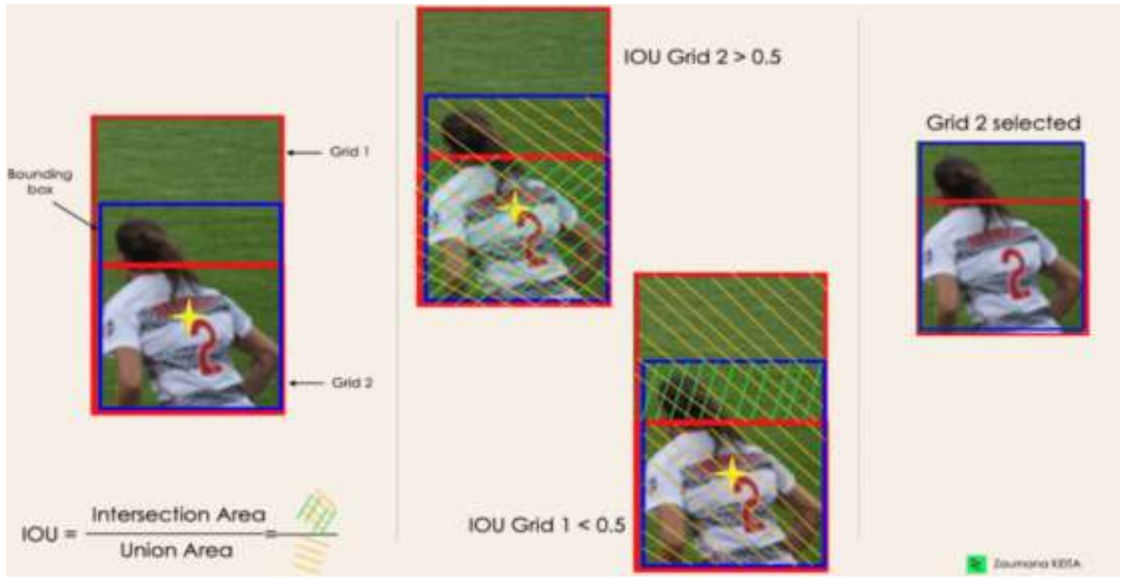
1. Residual Blocks:



2. Bounding Box Regression:



3. Intersection Over Unions or IOU:



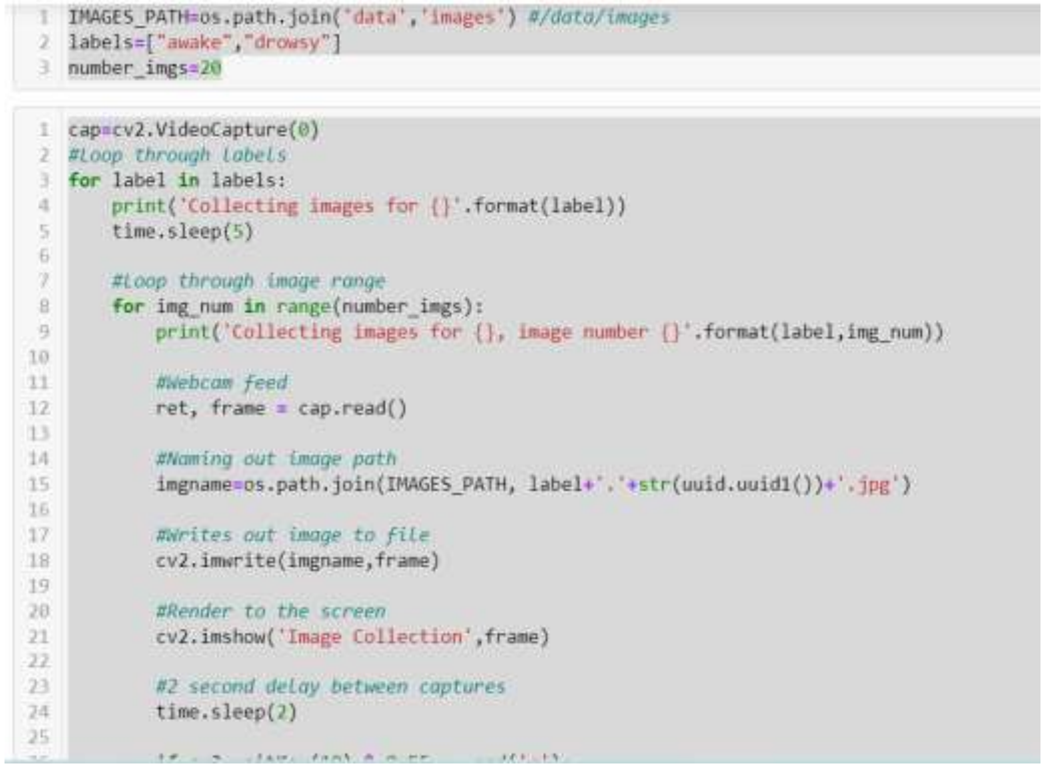
4. Non-Max Suppression or NMS: Setting a threshold for the IOU is not always enough because an object can have multiple boxes with IOU beyond the threshold, and leaving all those boxes might include noise. Here is where we can use NMS to keep only the boxes with the highest probability score of detection.

Steps involved in Drowsiness Detection

- ❑**Data Collection**: Setup a camera that monitors a stream for faces.
- ❑**Data Preprocessing**: Preprocess the collected data to ensure uniformity and compatibility with YOLO, involve resizing images, and labeling the images or frames with appropriate annotations indicating drowsy or alert states.
- ❑**Training YOLO**: Train the YOLO model using the annotated (labelled) dataset.
- ❑**Model Evaluation**: Evaluate the trained YOLO model on a separate validation dataset to assess its performance in detecting drowsiness accurately.

Code Snippets

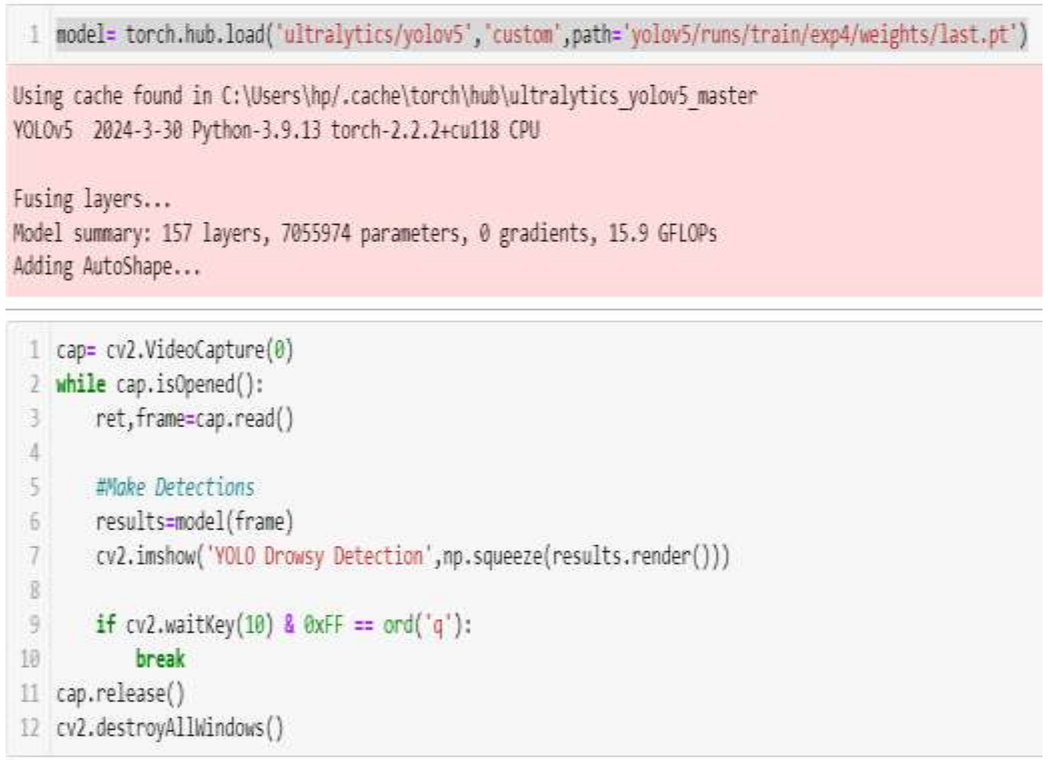
Training Phase



```python
IMAGES_PATH=os.path.join('data','images') #/data/images
labels=["awake","drowsy"]
number_imgs=20
```

```python
cap=cv2.VideoCapture(0)
#Loop through labels
for label in labels:
    print('Collecting images for {}'.format(label))
    time.sleep(5)

    #Loop through image range
    for img_num in range(number_imgs):
        print('Collecting images for {}, image number {}'.format(label,img_num))

        #Webcam feed
        ret, frame = cap.read()

        #Naming out image path
        imgname=os.path.join(IMAGES_PATH, label+'.'+str(uuid.uuid1())+'.jpg')

        #Writes out image to file
        cv2.imwrite(imgname,frame)

        #Render to the screen
        cv2.imshow('Image Collection',frame)

        #2 second delay between captures
        time.sleep(2)

```

Testing Phase



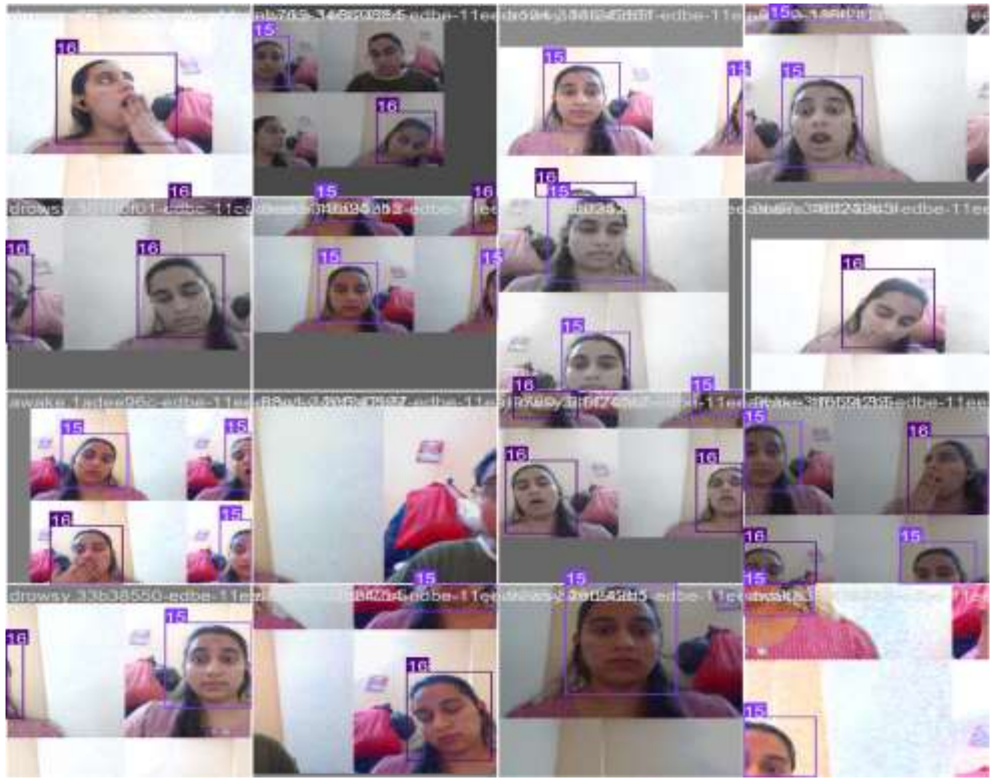
```python
model= torch.hub.load('ultralytics/yolov5','custom',path='yolov5/runs/train/exp4/weights/last.pt')
```

```
Using cache found in C:\Users\hp/.cache\torch\hub\ultralytics_yolov5_master
YOLOv5  2024-3-30 Python-3.9.13 torch-2.2.2+cu118 CPU

Fusing layers...
Model summary: 157 layers, 7055974 parameters, 0 gradients, 15.9 GFLOPs
Adding AutoShape...
```

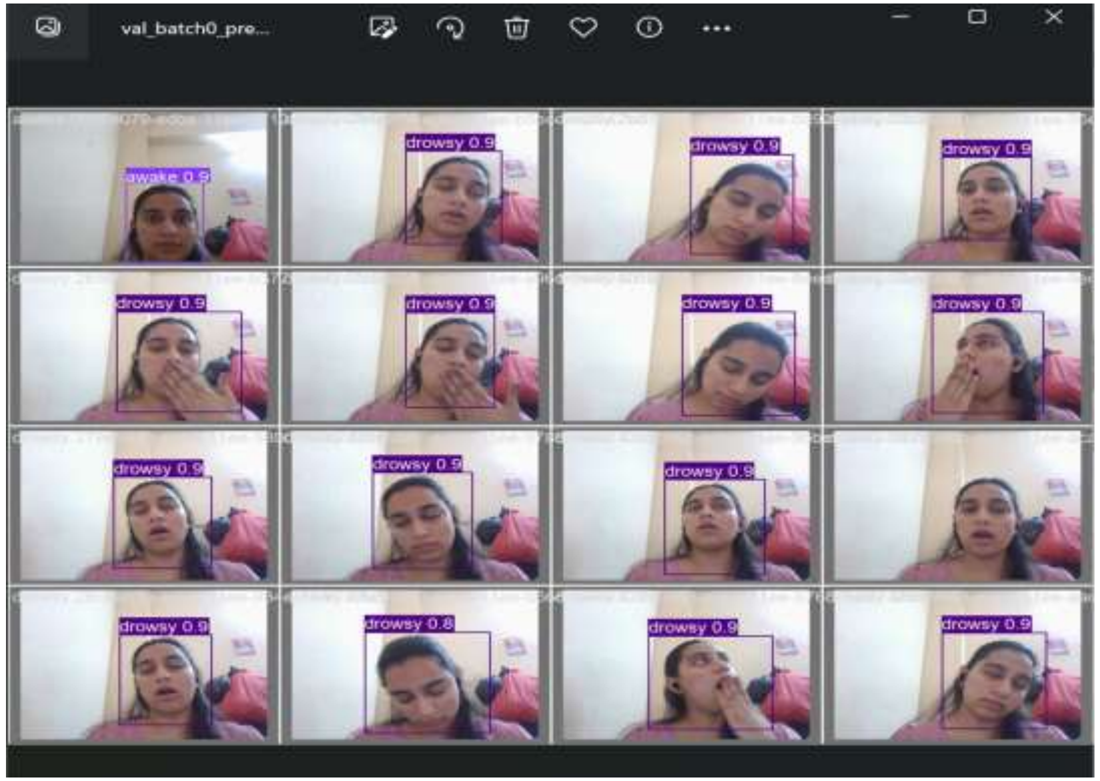
```python
cap= cv2.VideoCapture(0)
while cap.isOpened():
    ret,frame=cap.read()

    #Make Detections
    results=model(frame)
    cv2.imshow('YOLO Drowsy Detection',np.squeeze(results.render()))

    if cv2.waitKey(10) & 0xFF == ord('q'):
        break
cap.release()
cv2.destroyAllWindows()
```

Output of the Model

Training Phase



Testing Phase



Challenges

- ❑Ensuring the system detects drowsiness reliably without false alarms.
- ❑Achieving prompt alerts when drowsiness is detected.
- ❑Handling and processing less quality images is typical.
- ❑Irregular lightning conditions while passing video stream or live video stream as an input.

Thank You