Intel Unnati Industrial Training Program 2024

PROBLEM STATEMENT

STATEMENT NO : 7

INNOVATIVE MONITORING SYSTEM FOR TELEICU PATIENTS USING VIDEO PROCESSING AND DEEP LEARNING.

GROUP MEMBERS :

1. P. Vaishnavi
2. K. Gouthami
3. Ayyapu Reddy dande

INNOVATIVE MONITORING SYSTEM FOR TELEICU PATIENTS USING VIDEO PROCESSING AND DEEP LEARNING

The problem statement "Innovative Monitoring System for TeleICU Patients Using Video Processing and Deep Learning" outlines the focus and objective of a research or development project aimed at enhancing intensive care unit (ICU) monitoring through advanced technologies.

Here's a breakdown of what each component of the problem statement implies:

1. **Innovative Monitoring System**: This suggests the development of a new or significantly improved method for monitoring patients. Traditional ICU monitoring typically involves sensors and direct observation by healthcare professionals. An innovative system could imply a more automated, precise, or efficient approach.
2. **TeleICU Patients**: TeleICU refers to telemedicine for remote monitoring of ICU patients. These systems allow specialists located elsewhere to monitor and manage ICU patients remotely, which can improve access to critical care expertise.
3. **Using Video Processing**: Video processing involves analyzing and interpreting data from video feeds. In this context, it likely refers to using cameras in ICU rooms to capture patient information such as movement, vital signs, and other relevant indicators.
4. **Deep Learning**: Deep learning is a subset of artificial intelligence (AI) that involves algorithms capable of learning from data. In the context of ICU monitoring, deep learning can be used to analyze complex patterns in patient data extracted from video feeds, potentially identifying early signs of deterioration or abnormalities.

Overall Objective:

The main goal of this project is to develop and implement a system that leverages video processing techniques and deep learning algorithms to monitor ICU patients remotely (TeleICU). This system aims to enhance the accuracy, timeliness, and efficiency of ICU patient monitoring, potentially leading to better outcomes through early detection of critical changes in patient condition.

TECHNOLOGIES USED :

<u>1. YOLOv8 (You Only Look Once version 8)</u>
Description: YOLOv8 is a state-of-the-art real-time object detection model. It is known for its speed and accuracy.
Usage: Training an object detection model using labelled images. YOLOv8 uses a specific format for labels and images, requiring careful preparation of datasets.

<u>2. Google Drive</u>

Description: A cloud storage service by Google.

Usage: Storing and sharing datasets (images and labels) for your project. You've provided links to files stored on Google Drive, which include images and possibly label files.

3. Label Files (YOLO Format)

Description: Text files that contain annotation data for object detection. Each line in a file corresponds to one object, with the format: <class_id> <x_center> <y_center> <width> <height>

Usage: These files are necessary for training the YOLOv8 model, and they need to be correctly formatted and placed in the directories.

4. Python Scripts

Description: Custom scripts to verify and preprocess datasets.

Usage: Checking the existence and format of label files, normalizing coordinates, and ensuring the directory structure is correct.

5 . CVAT (Computer Vision Annotation Tool)

Description: CVAT is an open-source tool designed for annotating video and image data for computer vision tasks.

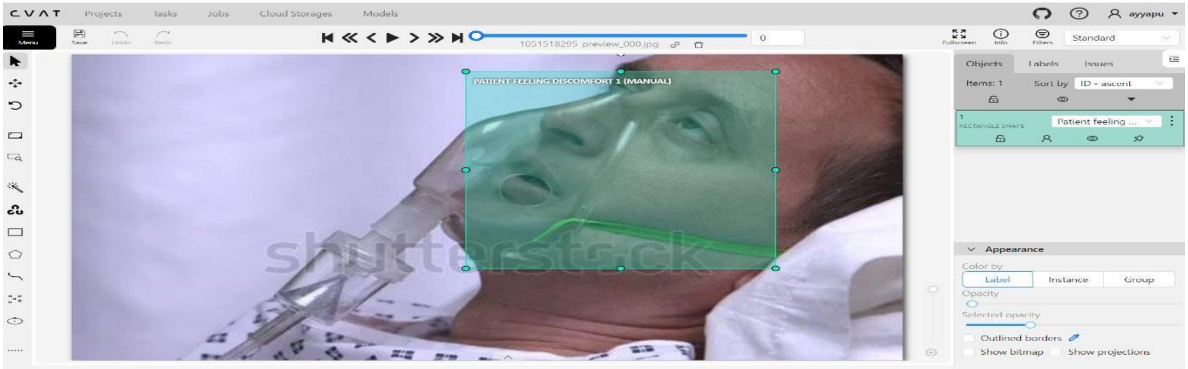
Usage: It is used to create, manage, and export accurate and detailed annotations (bounding boxes, polygons, points, etc.) for training machine learning models in various computer vision applications.

STEPS FOLLOWED :

Step 1: Annotation Using CVAT

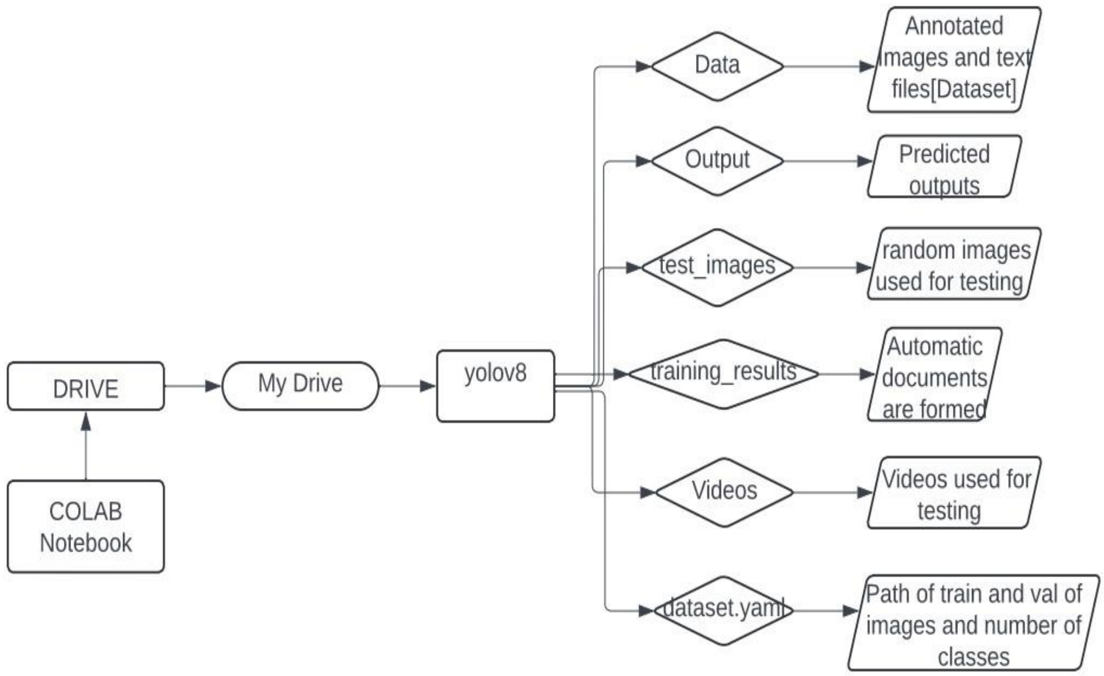
To prepare the dataset for training, I utilized the Computer Vision Annotation Tool (CVAT) to annotate images. The annotations were categorized into several key classes:

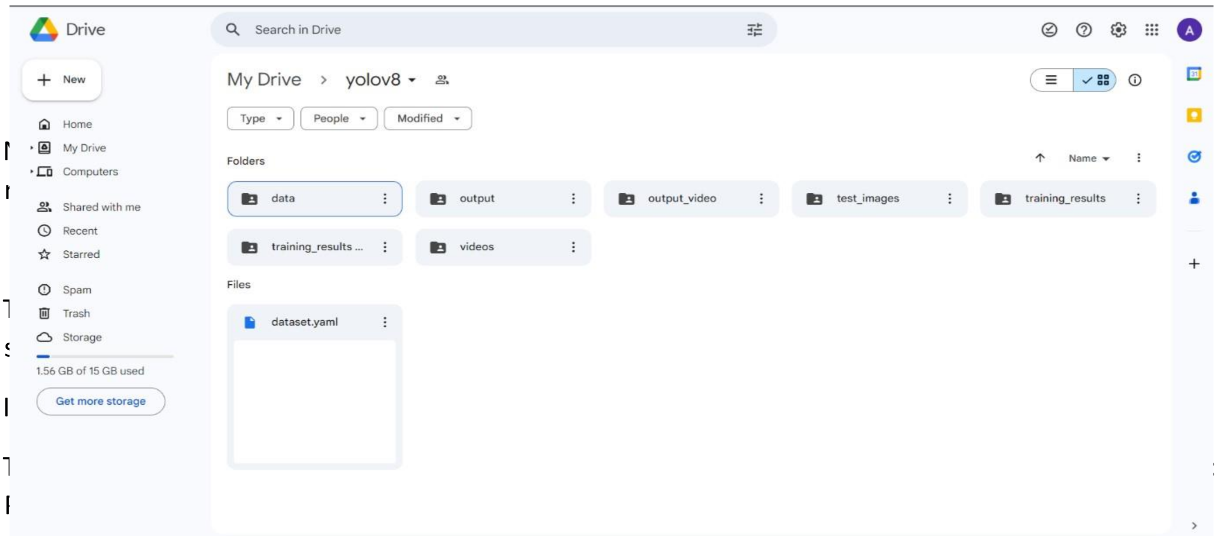
- Doctor: Annotations for healthcare professionals attending to the patient.
- Patient: Annotations indicating the presence and location of the patient.
- Intensivist: Annotations specific to intensivists, specialists overseeing critical care.
- Nurse: Annotations representing nursing staff attending to the patient.
- Patient Feeling Discomfort: Annotations identifying moments when the patient displays signs of discomfort or distress.



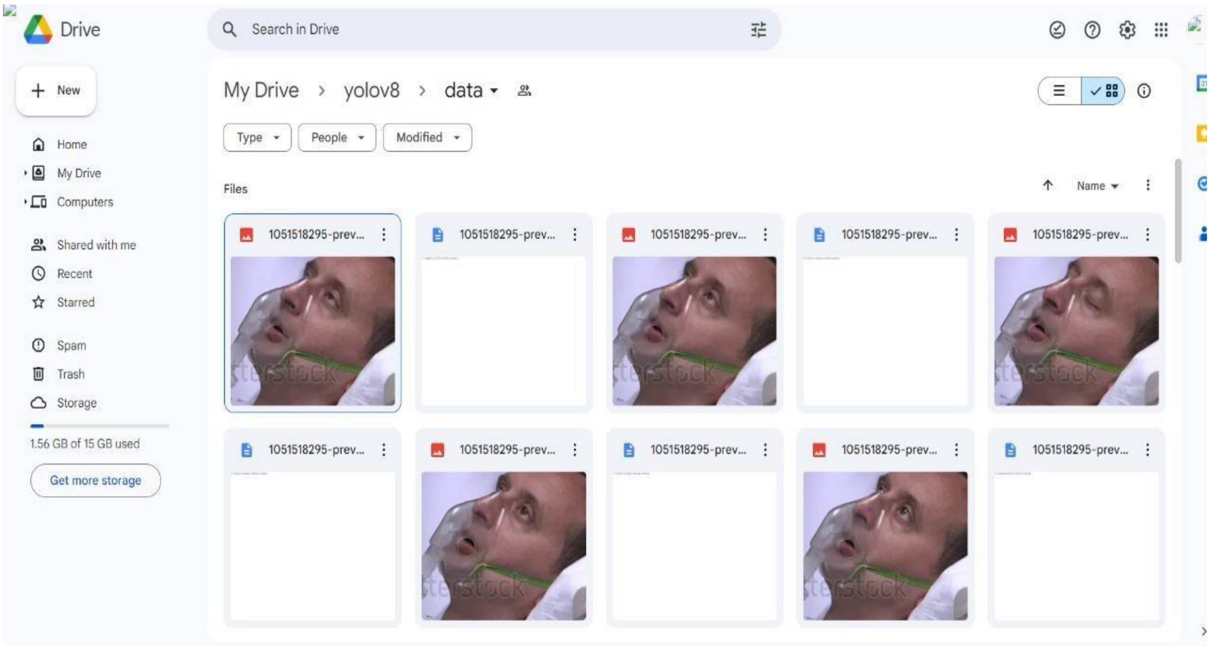
STEP 2 : Setting up google drive

This is the setup we have created in google drive.





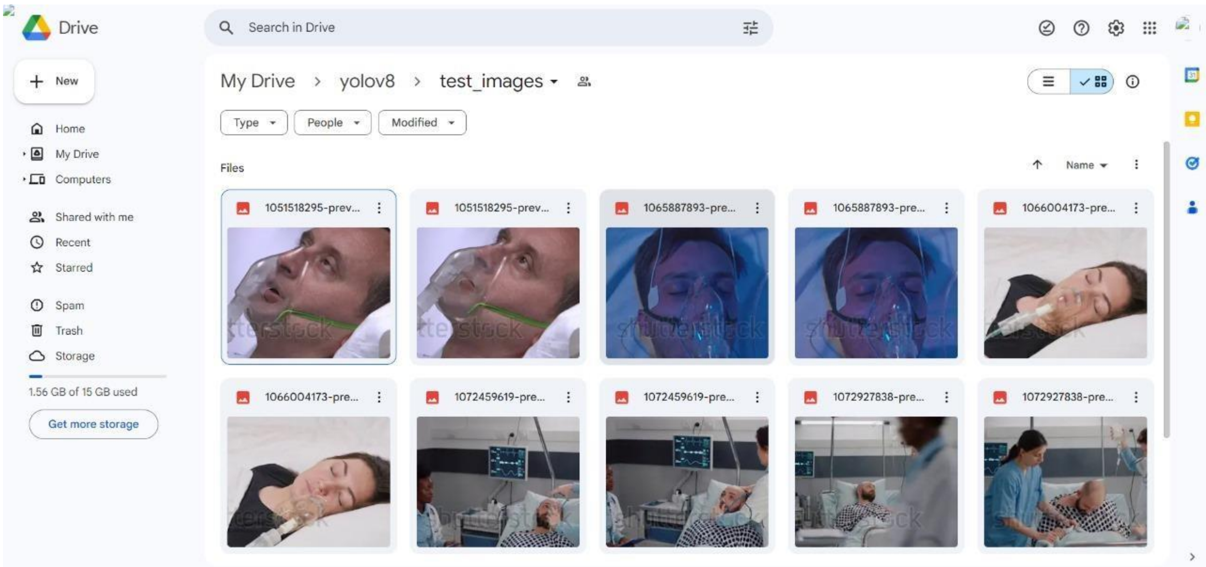
The repetitive presence of the same image and associated text files suggests that these files are prepared for use in a machine learning , specifically for training an object detection model using YOLOv8. The images and their annotations will be used to teach the model to recognize specific features or objects within similar images.



2. test_images : The test_images folder contains multiple images depicting various individuals in medical settings, such as wearing oxygen masks or lying in hospital beds.
Description: These images show different scenarios involving medical care, including patients with oxygen masks, patients in hospital beds, and medical staff attending to patients.

- Purpose: These images are used for testing the object detection model trained with YOLOv8 to ensure it can correctly identify and analyze different medical situations.

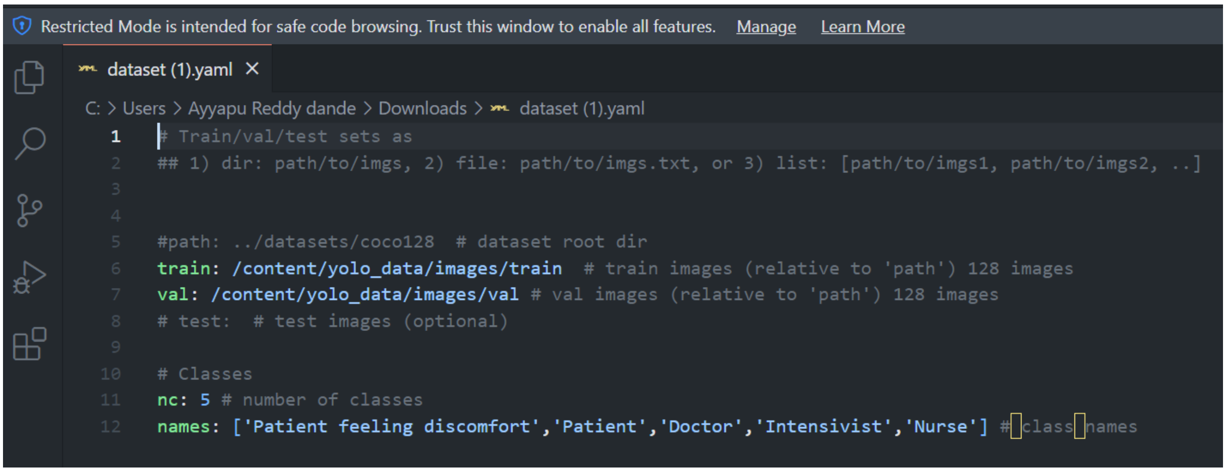
The diverse set of images in the test_images folder provides a comprehensive test dataset for evaluating the performance of the YOLOv8 model. The images cover various medical scenarios, helping to validate the model's ability to accurately detect and classify objects in different contexts.



3 . dataset.yaml

The image shows a configuration file in YAML format used for setting up a dataset for training a YOLO (You Only Look Once) model. Here's a breakdown of the components in the file:

1. **Dataset Paths**:
   - train: Path to the directory containing the training images. In the file, it's set to /content/yolo_data/images/train.
   - val: Path to the directory containing the validation images. In the file, it's set to /content/yolo_data/images/val.
   - test: (Optional) Path to the directory containing the test images. This is commented out in the file, indicating it is not currently being used.
2. **Classes**:
   - nc: Number of classes in the dataset. Here, it is set to 5.
   - names: List of class names. The file lists the following classes:
     - Patient feeling discomfort
     - Patient
     - Doctor
     - Intensivist
     - Nurse

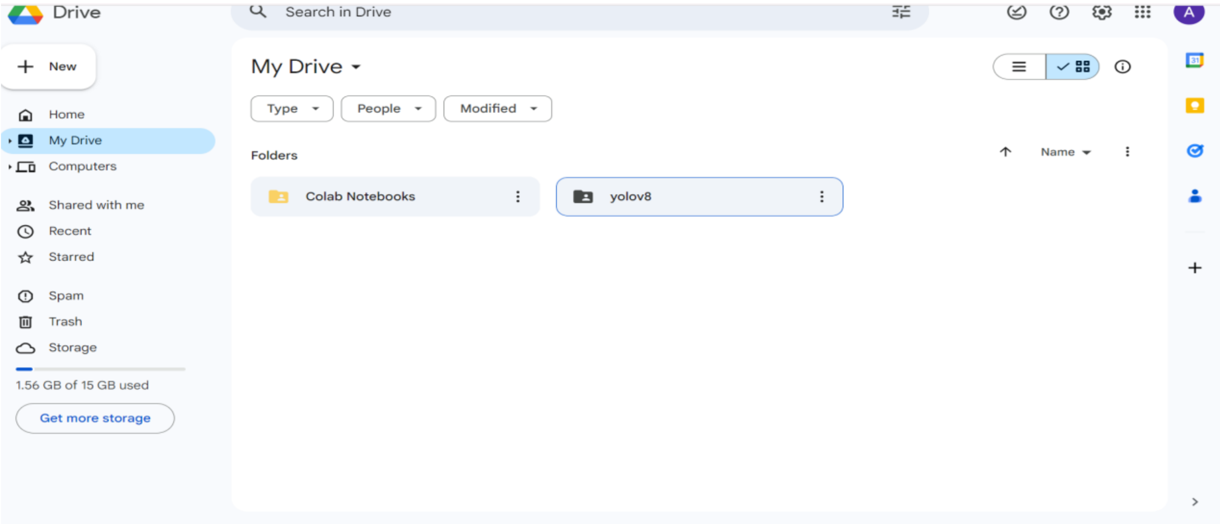



```yaml
# Train/val/test sets as
## 1) dir: path/to/imgs, 2) file: path/to/imgs.txt, or 3) list: [path/to/imgs1, path/to/imgs2, ..]


#path: ../datasets/coco128  # dataset root dir
train: /content/yolo_data/images/train  # train images (relative to 'path') 128 images
val: /content/yolo_data/images/val # val images (relative to 'path') 128 images
# test:  # test images (optional)

# Classes
nc: 5 # number of classes
names: ['Patient feeling discomfort','Patient','Doctor','Intensivist','Nurse'] # class names
```

STEP 3 : Linking Python Scripts to My drive.

Follow the below path for the python script :

https://colab.research.google.com/drive/14HXgpDsO8JttF4drVuurWyF2xQodNOck?usp=sharing



STEP 4 : TRAINING and TESTING the Dataset

TRAINING :

After running the training code, the YOLO model's predicted outputs are stored in the `output` folder that was created. Here's a more detailed breakdown:

1. Output Folder Structure:
   - The predicted outputs are organized in the `output` folder under a subfolder named `predict`.

2. Predicted Images:
   - Each image in the `predict` folder contains the original input image with the model's predictions overlaid.
   - The predictions include bounding boxes around the detected objects, class labels, and confidence scores.

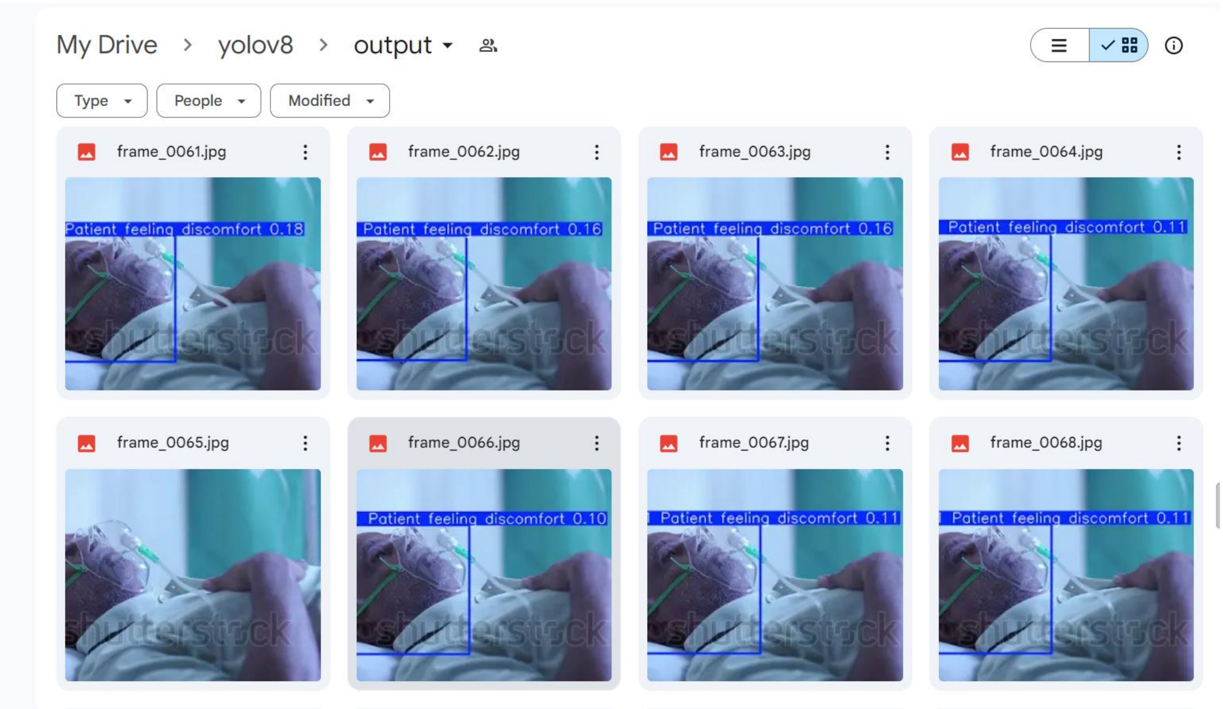
3. Bounding Boxes and Labels:

- The bounding boxes highlight the areas where the model detected objects.
- Class labels, such as "Patient feeling discomfort," are displayed above the bounding boxes.
- Confidence scores, which indicate the model's certainty about its predictions, are also shown next to the class labels.

4. Example:
- As shown in the image, predictions include labels like "Patient feeling discomfort" with a confidence score (e.g., 0.9).
- The filenames, such as "1051518295-preview_000.jpg," help in identifying and organizing the images.

Summary
The `output` folder contains the predicted results from the YOLO model, with each image showing bounding boxes, class labels, and confidence scores, providing a visual representation of the model's performance on the test data.

TEST DATA :



These images show the model's predictions on a test dataset, indicating that the YOLOv8 model has detected and classified patients feeling discomfort. The bounding boxes, class labels, and confidence scores provide a clear visualization of the model's performance.

STEP 5 : Final Tested Video in YOLOv8 Project

Location:

- Google Drive Folder: yolov8/output_video

File Name:

- annotated_video.mp4

Description:

- The video annotated_video.mp4 is the final output from your YOLOv8 model testing. It includes annotations overlaid on the video frames, indicating detected objects or actions along with their confidence scores. For example, the annotation might show "feeling discomfort 0.24," meaning the model detected an instance of discomfort with a confidence score of 24%.

“THANK YOU”