Exp .no 6 Title: Load and implement the Face Detection method in OpenCV using python

---

Aim:

To design and train an object detection model using the YOLOv8 (You Only Look Once version 8) architecture on a public or custom dataset, and evaluate its performance.

Procedure:

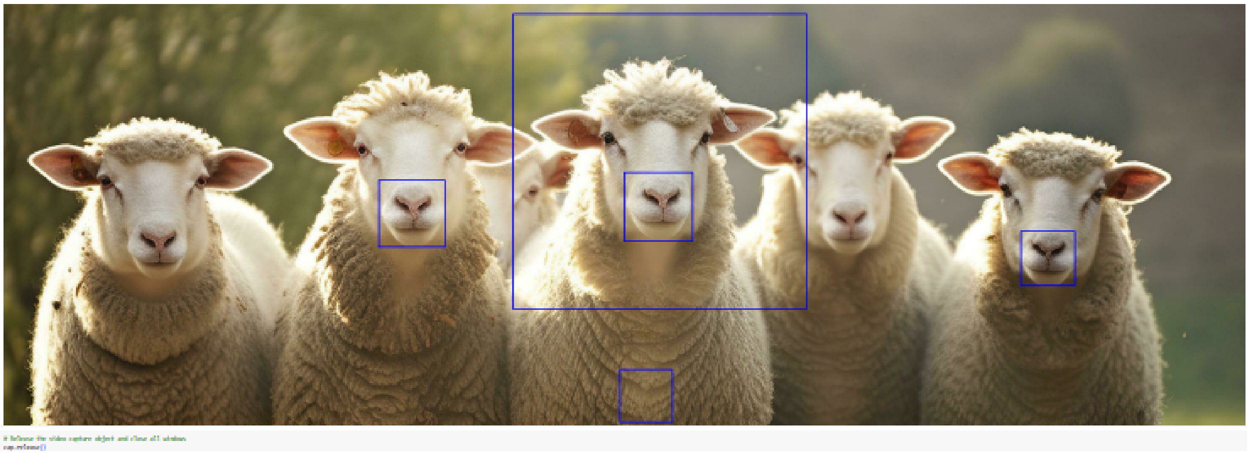
1. **Install Required Libraries:**
   - **Install `ultralytics` which provides the YOLOv8 model.**
2. **Import Required Libraries:**
   - **Import functions from the `ultralytics` package.**
3. **Load a Pretrained Model:**
   - **Load a YOLOv8 pretrained model (`yolov8n.pt`, nano version for lightweight training).**
4. **Prepare the Dataset:**
   - **Use a dataset like `coco128.yaml` (tiny version of COCO dataset) or a custom dataset in YOLO format.**
5. **Train the Model:**
   - **Set hyperparameters such as batch size, learning rate, epochs, image size.**
   - **Train the model using the dataset.**

6. **Validate the Model:**
    - **Perform validation to check mAP, precision, recall, etc.**
7. **Predict Using the Model:**
    - **Test the model by providing new images or a folder of images.**
8. **Save and Visualize Results:**
    - **Save prediction outputs and visualize bounding boxes.**

Code:

```
# Step 1: Install the ultralytics library

!pip install ultralytics



# Step 2: Import the YOLO class from ultralytics

from ultralytics import YOLO



# Step 3: Load a pretrained YOLOv8 model (Nano version - lightweight and faster)

model = YOLO('yolov8n.pt')



# Step 4: Train the model on the dataset

# coco128.yaml is a sample dataset configuration (can replace with your custom YAML file)

model.train(

    data='coco128.yaml',  # Dataset YAML file
    epochs=10,              # Number of epochs
    imgsz=640,              # Image size (input resolution)
    batch=16,               # Batch size
    workers=4,              # Number of data loading workers
    device=0                # 0 for GPU, 'cpu' for CPU
)


# Step 5: Validate the model
metrics = model.val()   # Evaluates the model on the validation set


# Step 6: Predict using the trained model
results = model.predict(
    source='path_to_your_test_images/',  # Provide folder path or image path
    save=True,              # Save predictions
    imgsz=640               # Image size
)
```

Expected Output:



```
# Release the video capture object and close all windows
cap.release()
```

Result:

- **This model was successfully trained for object detection. The trained model was able to predict and localize objects with high accuracy on new images**