**YOLO (You Only Look Once) Object Detection Tutorial**

**Introduction**

YOLO (You Only Look Once) is a real-time object detection algorithm that is widely used for detecting objects in images and videos. In this tutorial, we will cover:

* Installing YOLO and dependencies
* Running YOLO on images and videos
* Understanding the YOLO model structure
* Customizing YOLO for your own dataset

**1. Installing Dependencies**

Before using YOLO, ensure you have the necessary dependencies installed.

**Install OpenCV and YOLO Requirements**

pip install opencv-python numpy torch torchvision

**Download Pre-trained YOLO Model**

YOLO models can be installed from the official repository:

git clone https://github.com/ultralytics/yolov5.git

cd yolov5

pip install -r requirements.txt

**2. Running YOLO on Images**

Once YOLO is installed, you can run object detection on an image.

python detect.py --weights yolov5s.pt --img 640 --conf 0.25 --source image.jpg

* --weights yolov5s.pt: Use the small YOLOv5 model.
* --img 640: Set image size to 640 pixels.
* --conf 0.25: Confidence threshold for detection.
* --source image.jpg: Specify the image file.

The output will display detected objects and save the results in the runs/detect/ directory.

**3. Running YOLO on Videos or Webcam**

You can also use YOLO for real-time detection with a video file or webcam.

**Detect Objects in a Video File**

python detect.py --weights yolov5s.pt --source video.mp4

**Run YOLO on a Webcam**

python detect.py --weights yolov5s.pt --source 0

* --source 0 uses the default webcam.

**4. Understanding YOLO Model Structure**

YOLO uses a neural network to divide an image into grids, predicting bounding boxes and class probabilities for objects in each grid.

* **Backbone**: Extracts features (e.g., CSPDarknet in YOLOv5).
* **Neck**: Enhances feature extraction (e.g., PANet in YOLOv5).
* **Head**: Predicts bounding boxes, objectness scores, and class probabilities.

**5. Custom Training with YOLO**

To train YOLO on your own dataset:

**Step 1: Prepare the Dataset**

Your dataset should be in the YOLO format:

/images

- train

- val

/labels

- train

- val

Each label file should contain:

<class\_id> <x\_center> <y\_center> <width> <height>

All values are normalized (0-1 range).

**Step 2: Modify the Data Config**

Create a .yaml file (e.g., custom\_data.yaml):

train: /path/to/train/images

val: /path/to/val/images

nc: 2

names: ['class1', 'class2']

* nc: Number of classes.
* names: List of class names.

**Step 3: Train the Model**

python train.py --img 640 --batch 16 --epochs 50 --data custom\_data.yaml --weights yolov5s.pt

* --img 640: Image size.
* --batch 16: Batch size.
* --epochs 50: Number of training epochs.
* --data custom\_data.yaml: Path to dataset config.
* --weights yolov5s.pt: Start training from a pre-trained model.

**Step 4: Test the Model**

After training, test your model:

python detect.py --weights runs/train/exp/weights/best.pt --source test.jpg

**Conclusion**

This tutorial covered:

* Setting up YOLOv5
* Running YOLO on images and videos
* Understanding YOLO’s architecture
* Custom training with YOLO

With this knowledge, you can now customize YOLO for your own object detection tasks. Experiment with different configurations to improve performance!