Week-15

Aim:

To implement a real time object detection system that can identify and localize multiple objects in an image

Algorithm:

* Input Processing, Resize input image to fixed dimension
* Normalize Pixel values to [0,1] range
* Feature Extraction, Pass image through convolutional backbone network
* Divide input image into SxS grid
* Each grid cell is responsible for detecting objects whose center falls within it
* Each grid cell Predicts B bounding boxes
* Each box Contains: (x, y, w, h, confidence
* Each grid all Predicts C class Probabilities
* Conditional Probability for each class given an object is Present
* Apply confidence thresholding

Observation:

* Yolo detects multiple objects with boundary boxes
* Inference is fast and efficient
* Model Accurately detects

Yolo Architecture model

Pseudo code:

```
CLASS YOLO:
METHODS:
    CONSTRUCTOR (num_classes, anchors):
        self.backbone = DarkNetBackbone()
        self.detection_heads = []
        self.anchors = anchors
        self.num_classes = num_classes
    FORWARD (image_tensor):
        features = self.backbone(image_tensor)
        detection = []
        FOR each feature_level IN features:
            detection = self.detection_heads[feature_level]
                (features[feature_level])
            detections.APPEND (detection)
        RETURN detections
    PREDICT (image):
        Processed_image, original_dims = PREPROCESS (image)
        raw_detections = FORWARD (Processed_image)
FUNCTION DETECT_OBJECTS (image_Path, model):
    image = LOAD_IMAGE (image_Path)
    boxes, scores, classes = model.PREDICT(image)
    result_imge = DRAW_BOXES (image, boxes, scores, classes)
RETURN result_image, boxes, scores, classes
```

Result

1. The yolo model successfully Implemented

```
from PIL import Image as PILImage
from IPython.display import Image, display
from ultralytics import YOLO

# Step 1: Load YOLO model (YOLOv8 nano for speed)
model = YOLO('yolov8n.pt')

# Step 2: Convert uploaded image (if .webp)
input_image = "p020rgbg.jpg.webp"   # change if your uploaded filename is different
converted_image = "converted_test.jpg"

# Convert webp → jpg
img = PILImage.open(input_image).convert("RGB")
img.save(converted_image, "JPEG")

# Step 3: Run YOLO detection
results = model(converted_image)

# Step 4: Save detection result
results[0].save(filename="result_detected.jpg")

# Step 5: Display result inline
display(Image(filename="result_detected.jpg"))
```

```
image 1/1 /content/converted_test.jpg: 384x640 34 cars, 10.9ms
Speed: 1.7ms preprocess, 10.9ms inference, 18.9ms postprocess per image at shape (1, 3, 384, 640)
```