EX.NO.7 Object Detection using SSD300 with VGG16 Backbone

Aim:

To detect objects in an uploaded image using a pretrained Single Shot Multibox Detector (SSD300) model with a VGG16 backbone.

Procedure:

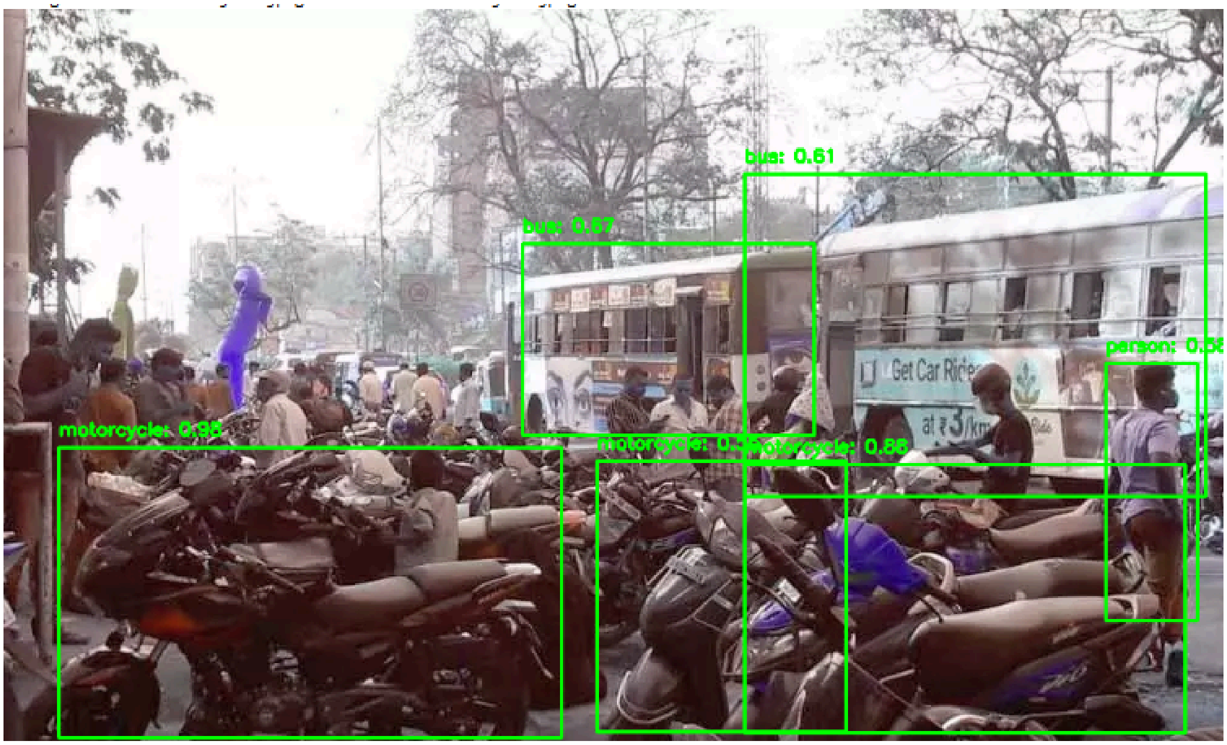
1. **Install Required Packages:**
   - **Install `torch`, `torchvision`, `opencv-python`, and `matplotlib` for deep learning, image processing, and visualization.**
2. **Import Libraries:**
   - **Import PyTorch, Torchvision models and transforms, OpenCV, PIL, NumPy, and matplotlib.**
3. **Load Pretrained SSD Model:**
   - **Load `ssd300_vgg16` model with pretrained weights (`torchvision.models.detection.ssd.SSD300_VGG16_Weights.DEFAULT`).**
   - **Set the model to evaluation mode to prevent training-specific behaviors (like dropout).**
4. **Prepare Image Transformation:**
   - **Use the transform associated with the loaded model weights to properly preprocess the image (resizing, normalization).**

5. Upload Image:
    - Upload an image manually using Google Colab’s file upload interface.
6. Preprocess Image:
    - Open the uploaded image using PIL and convert it to RGB.
    - Apply the transformation and add a batch dimension.
7. Run Prediction:
    - Feed the preprocessed image into the model.
    - Get the output predictions including bounding boxes, labels, and confidence scores.
8. Visualize Results:
    - Draw bounding boxes and labels on the image using OpenCV based on a confidence threshold (default 0.5).
    - Display the annotated image within Google Colab.

CODE;

```
# Step 1: Install necessary packages
!pip install -q torch torchvision matplotlib opencv-python


# Step 2: Import libraries
import torch
import torchvision
import torchvision.transforms as T
from PIL import Image
import matplotlib.pyplot as plt
import cv2
import numpy as np
from google.colab.patches import cv2_imshow

# Step 3: Load pretrained SSD model
weights = torchvision.models.detection.ssd.SSD300_VGG16_Weights.DEFAULT
model = torchvision.models.detection.ssd300_vgg16(weights=weights)
model.eval()

# Step 4: Define the transformation
transform = weights.transforms()

# Step 5: Upload image
from google.colab import files
uploaded = files.upload()

image_path = list(uploaded.keys())[0]
image = Image.open(image_path).convert("RGB")

# Step 6: Preprocess the image
img_tensor = transform(image).unsqueeze(0)

# Step 7: Predict
with torch.no_grad():
    preds = model(img_tensor)[0]

# Step 8: Visualize detections
def draw_boxes(image_pil, predictions, score_threshold=0.5):
    image = np.array(image_pil)
    boxes = predictions['boxes']
    labels = predictions['labels']
    scores = predictions['scores']
    categories = weights.meta["categories"]

    for box, label, score in zip(boxes, labels, scores):
        if score >= score_threshold:
            x1, y1, x2, y2 = box.int().tolist()
            cv2.rectangle(image, (x1, y1), (x2, y2), color=(0,255,0), thickness=2)
            text = f"{categories[label]}: {score:.2f}"
            cv2.putText(image, text, (x1, y1 - 10), cv2.FONT_HERSHEY_SIMPLEX,
                        0.5, (0, 255, 0), 2)

    cv2_imshow(image)

# Step 9: Draw boxes and show predictions
draw_boxes(image, preds)
```

Output:



Result:

Successfully performed object detection on the uploaded image using the pretrained SSD300 model. Detected objects are accurately localized and labeled with their corresponding class names and confidence levels.