# 8. A PYTORCH IMPLEMENTATION OF OBJECT DETECTION WITH SINGLE SHOT DETECTOR

|  |  |
| --- | --- |
| **EX.N0 : 8** | **A PYTORCH IMPLEMENTATION OF OBJECT DETECTION WITH SINGLE SHOT DETECTOR** |
| **DATE : 18/03/2025** |

**AIM:**

To implement object detection using the Single Shot Detector (SSD) model in PyTorch.

# ALGORITHM:

Step 1: Import necessary libraries including PyTorch and TorchVision.

Step 2: Load a pre-trained SSD model (ssd300\_vgg16) from torchvision.models. Step 3: Set the model to evaluation mode and move it to GPU (if available).

Step 4: Read an input image and apply transformations.

Step 5: Pass the image through the model to get predictions.

Step 6: Post-process and display detected objects using bounding boxes and labels.

# PROGRAM:

import torch import torchvision

from torchvision import transforms import cv2

import numpy as np

model = torchvision.models.detection.ssdlite320\_mobilenet\_v3\_large(pretrained=True) model.eval()

COCO\_LABELS = [

' background ', 'person', 'bicycle', 'car', 'motorcycle', 'airplane', 'bus', 'train', 'truck', 'boat', 'traffic light', 'fire hydrant', 'N/A', 'stop sign',

'parking meter', 'bench', 'bird', 'cat', 'dog', 'horse', 'sheep', 'cow',

'elephant', 'bear', 'zebra', 'giraffe', 'N/A', 'backpack', 'umbrella', 'N/A',

'N/A', 'handbag', 'tie', 'suitcase', 'frisbee', 'skis', 'snowboard', 'sports ball', 'kite', 'baseball bat', 'baseball glove', 'skateboard', 'surfboard', 'tennis racket', 'bottle', 'N/A', 'wine glass', 'cup', 'fork', 'knife', 'spoon', 'bowl',

'banana', 'apple', 'sandwich', 'orange', 'broccoli', 'carrot', 'hot dog', 'pizza',

'donut', 'cake', 'chair', 'couch', 'potted plant', 'bed', 'N/A', 'dining table',

'N/A', 'N/A', 'toilet', 'N/A', 'tv', 'laptop', 'mouse', 'remote', 'keyboard',

'cell phone', 'microwave', 'oven', 'toaster', 'sink', 'refrigerator', 'N/A', 'book', 'clock', 'vase', 'scissors', 'teddy bear', 'hair drier', 'toothbrush'

]

transform = transforms.Compose([ transforms.ToTensor()

])

cap = cv2.VideoCapture('/content/big-buck-bunny-1080p-60fps-30sec.mp4') width = int(cap.get(cv2.CAP\_PROP\_FRAME\_WIDTH))

height = int(cap.get(cv2.CAP\_PROP\_FRAME\_HEIGHT)) fps = cap.get(cv2.CAP\_PROP\_FPS)

fourcc = cv2.VideoWriter\_fourcc(\*'mp4v')

out = cv2.VideoWriter('ssd\_output.mp4', fourcc, fps, (width, height)) while cap.isOpened():

ret, frame = cap.read() if not ret:

break

input\_tensor = transform(frame).unsqueeze(0) with torch.no\_grad():

detections = model(input\_tensor)[0] for i in range(len(detections['boxes'])):

score = detections['scores'][i].item() if score > 0.5:

box = detections['boxes'][i].numpy().astype(int) label = COCO\_LABELS[detections['labels'][i]]

cv2.rectangle(frame, (box[0], box[1]), (box[2], box[3]), (0, 255, 0), 2)

cv2.putText(frame, f'{label}: {score:.2f}', (box[0], box[1]-10), cv2.FONT\_HERSHEY\_SIMPLEX, 0.5, (0, 255, 0), 2)

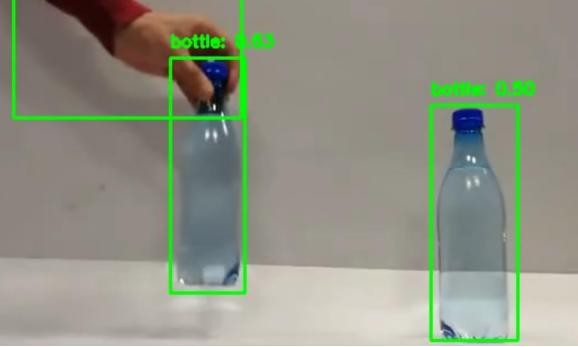
out.write(frame)

if cv2.waitKey(1) & 0xFF == ord('q'): break

cap.release() out.release()

cv2.destroyAllWindows()

# OUTPUT:

****

**RESULT:**

Thus the Program has been executed successfully and verified.