Course Name: Deep Learning

Lab Title: Vehicle Detection for Smart Traffic Management using YOLOv11

Student Name:Manavi Pawar

PRN:-202201040050

Date of Submission: 01-04-2025

Group Members: Manavi Pawar(202201040050),Kanishka Garud(202201070062),Sakshi Dube(202201040155)

Objective The purpose of this lab is to understand and implement YOLOv11 for real-time object detection. Students will perform dataset preparation, model implementation, inference, and performance evaluation.

```
Start coding or generate with AI.
```

Task 1: Environment Setup and YOLOv11 Installation

Objective: Set up the required libraries and dependencies to run YOLOv11.

Instructions:

Install Python and required libraries (PyTorch, OpenCV, Ultralytics, etc.). Install YOLOv11 from the official repository. Verify the installation by running a sample script. Expected Outcome: A functional YOLOv11 environment ready for experimentation.

```
!pip install ultralytics
```

```
  Attempting uninstall: nvidia-cusolver-cu12
    Found existing installation: nvidia-cusolver-cu12 11.6.3.83
    Uninstalling nvidia-cusolver-cu12-11.6.3.83:
      Successfully uninstalled nvidia-cusolver-cu12-11.6.3.83
Successfully installed nvidia-cublas-cu12-12.4.5.8 nvidia-cuda-cupti-cu12-12.4.127 nvidia-cuda-nvrtc-cu12-12.4.127 nvidia-cud
```

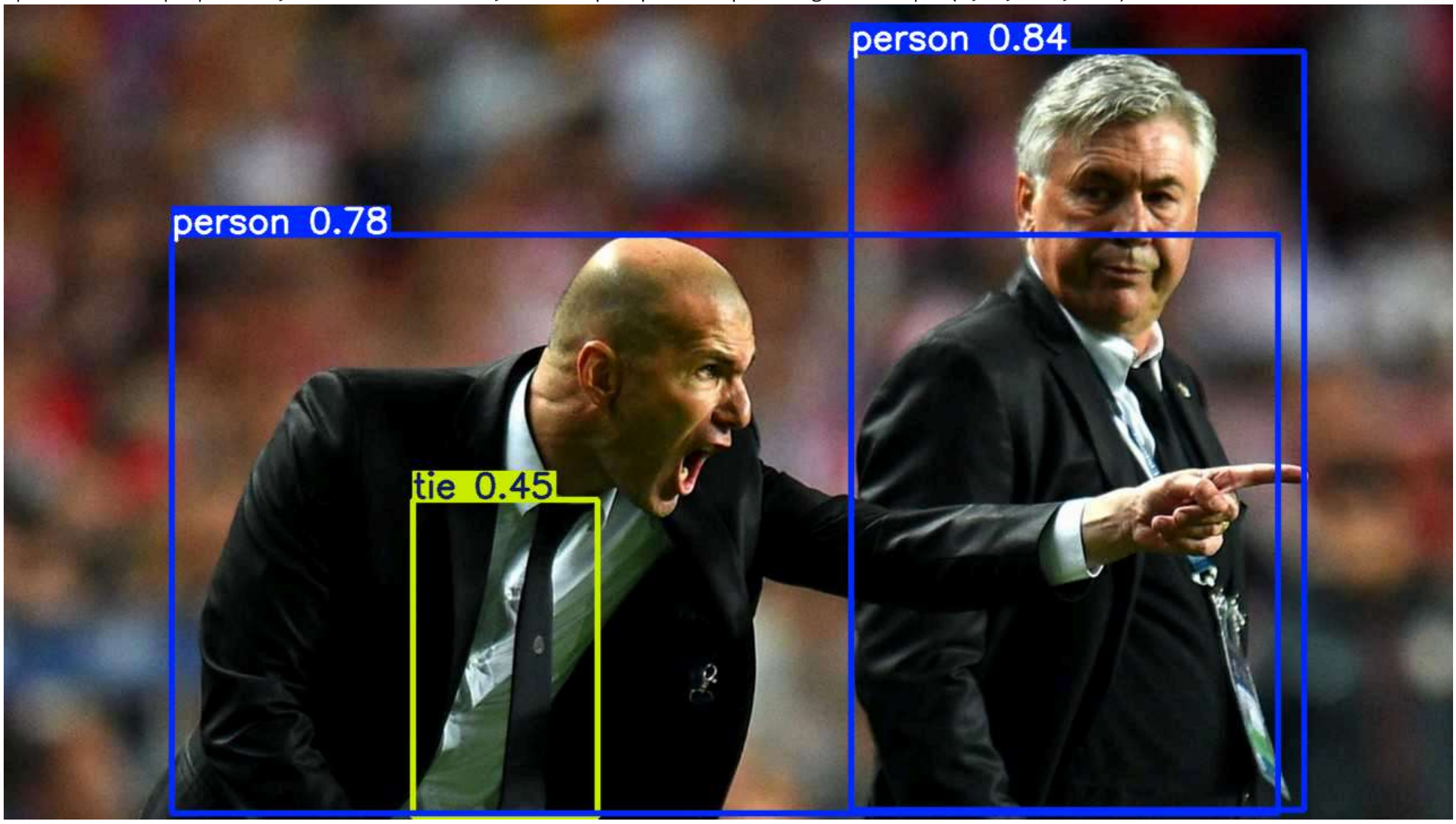
```
from ultralytics import YOLO

# Load a pre-trained YOLOv11 model
model = YOLO('yolo11n.pt')  # 'n' stands for nano version; other versions include 's', 'm', 'l', 'x'

# Run YOLO on a sample image
results = model('https://ultralytics.com/images/zidane.jpg')
results[0].show()
```

```
Creating new Ultralytics Settings v0.0.6 file ✅
View Ultralytics Settings with 'yolo settings' or at '/root/.config/Ultralytics/settings.json'
Update Settings with 'yolo settings key=value', i.e. 'yolo settings runs_dir=path/to/dir'. For help see https://docs.ultralytics
Downloading https://github.com/ultralytics/assets/releases/download/v8.3.0/yolo11n.pt to 'yolo11n.pt'...
100%|██████████| 5.35M/5.35M [00:00<00:00, 62.1MB/s]

Downloading https://ultralytics.com/images/zidane.jpg to 'zidane.jpg'...
100%|██████████| 49.2k/49.2k [00:00<00:00, 4.73MB/s]
image 1/1 /content/zidane.jpg: 384x640 2 persons, 1 tie, 364.4ms
Speed: 18.1ms preprocess, 364.4ms inference, 38.9ms postprocess per image at shape (1, 3, 384, 640)
```



Task 2: Dataset Preparation & Preprocessing Objective: Load and preprocess a dataset for object detection.

Instructions:

Choose a Dataset – Use COCO, Pascal VOC, or a custom dataset. Annotate Images – If using a custom dataset, label objects using Roboflow or LabelImg. Convert Annotations – Use Roboflow to export the dataset in YOLO format. Download the Dataset – Use the Roboflow API to fetch the dataset. Split the Dataset – Divide into train (80%), validation (10%), and test (10%). Expected Outcome: A well-structured dataset in YOLO format.

```
!pip install roboflow
```

```
Requirement already satisfied: roboflow in /usr/local/lib/python3.11/dist-packages (1.1.58)
Requirement already satisfied: certifi in /usr/local/lib/python3.11/dist-packages (from roboflow) (2025.1.31)
Requirement already satisfied: idna==3.7 in /usr/local/lib/python3.11/dist-packages (from roboflow) (3.7)
Requirement already satisfied: cycler in /usr/local/lib/python3.11/dist-packages (from roboflow) (0.12.1)
Requirement already satisfied: kiwisolver>=1.3.1 in /usr/local/lib/python3.11/dist-packages (from roboflow) (1.4.8)
Requirement already satisfied: matplotlib in /usr/local/lib/python3.11/dist-packages (from roboflow) (3.10.0)
Requirement already satisfied: numpy>=1.18.5 in /usr/local/lib/python3.11/dist-packages (from roboflow) (2.0.2)
Requirement already satisfied: opencv-python-headless==4.10.0.84 in /usr/local/lib/python3.11/dist-packages (from roboflow) (4.1
Requirement already satisfied: Pillow>=7.1.2 in /usr/local/lib/python3.11/dist-packages (from roboflow) (11.1.0)
Requirement already satisfied: pillow-heif>=0.18.0 in /usr/local/lib/python3.11/dist-packages (from roboflow) (0.22.0)
Requirement already satisfied: python-dateutil in /usr/local/lib/python3.11/dist-packages (from roboflow) (2.8.2)
Requirement already satisfied: python-dotenv in /usr/local/lib/python3.11/dist-packages (from roboflow) (1.1.0)
Requirement already satisfied: requests in /usr/local/lib/python3.11/dist-packages (from roboflow) (2.32.3)
Requirement already satisfied: six in /usr/local/lib/python3.11/dist-packages (from roboflow) (1.17.0)
Requirement already satisfied: urllib3>=1.26.6 in /usr/local/lib/python3.11/dist-packages (from roboflow) (2.3.0)
Requirement already satisfied: tqdm>=4.41.0 in /usr/local/lib/python3.11/dist-packages (from roboflow) (4.67.1)
Requirement already satisfied: PyYAML>=5.3.1 in /usr/local/lib/python3.11/dist-packages (from roboflow) (6.0.2)
Requirement already satisfied: requests-toolbelt in /usr/local/lib/python3.11/dist-packages (from roboflow) (1.0.0)
Requirement already satisfied: filetype in /usr/local/lib/python3.11/dist-packages (from roboflow) (1.2.0)
Requirement already satisfied: contourpy>=1.0.1 in /usr/local/lib/python3.11/dist-packages (from matplotlib->roboflow) (1.3.1)
Requirement already satisfied: fonttools>=4.22.0 in /usr/local/lib/python3.11/dist-packages (from matplotlib->roboflow) (4.56.0)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.11/dist-packages (from matplotlib->roboflow) (24.2)
Requirement already satisfied: pyparsing>=2.3.1 in /usr/local/lib/python3.11/dist-packages (from matplotlib->roboflow) (3.2.1)
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.11/dist-packages (from requests->roboflow) (3
```

```
from roboflow import Roboflow
```

```
rf = Roboflow(api_key="Z36dR2kczFGORe8fzTeh")
```

```
project = rf.workspace("roboflow-100").project("vehicles-q0x2v")
```

```
loading Roboflow workspace...
loading Roboflow project...
```

```
dataset = project.version(1).download("yolov11")
```

```
Downloading Dataset Version Zip in vehicles-1 to yolov11:: 100%|██████████| 217401/217401 [00:04<00:00, 47393.81it/s]

Extracting Dataset Version Zip to vehicles-1 in yolov11:: 100%|██████████| 8128/8128 [00:04<00:00, 1933.94it/s]
```

```
import os
dataset_path = "/content/vehicles-1"
print(os.listdir(dataset_path))
```

```
['train', 'data.yaml', 'valid', 'test', 'README.roboflow.txt', 'README.dataset.txt']
```

```
yaml_path = "/content/vehicles-1/data.yaml"

with open(yaml_path, "r") as file:
    print(file.read())
```

```
train: ../train/images
val: ../valid/images
test: ../test/images

nc: 12
names: ['big bus', 'big truck', 'bus-l-', 'bus-s-', 'car', 'mid truck', 'small bus', 'small truck', 'truck-l-', 'truck-m-', 'tru

roboflow:
  workspace: roboflow-100
  project: vehicles-q0x2v
  version: 1
  license: CC BY 4.0
  url: https://universe.roboflow.com/roboflow-100/vehicles-q0x2v/dataset/1
```

```
import yaml

# Load and update YAML file with correct paths
with open(yaml_path, "r") as file:
    data = yaml.safe_load(file)

correct_paths = {
    "train": "/content/vehicles-1/train",
    "val": "/content/vehicles-1/valid",
    "test": "/content/vehicles-1/test"
```

```
}

data.update(correct_paths)

with open(yaml_path, "w") as file:
    yaml.dump(data, file, default_flow_style=False)

print("✅ data.yaml paths have been updated successfully!")
```

```
✅ data.yaml paths have been updated successfully!
```

```
with open(yaml_path, "r") as file:
    print(file.read())
```

```
train: ../train/images
val: ../valid/images
test: ../test/images

nc: 12
names: ['big bus', 'big truck', 'bus-l-', 'bus-s-', 'car', 'mid truck', 'small bus', 'small truck', 'truck-l-', 'truck-m-', 'tru

roboflow:
  workspace: roboflow-100
  project: vehicles-q0x2v
  version: 1
  license: CC BY 4.0
  url: https://universe.roboflow.com/roboflow-100/vehicles-q0x2v/dataset/1
```

```
# Check GPU availability
import torch
print(torch.cuda.is_available())  # Should print True if GPU is available
print(torch.cuda.device_count())  # Number of GPUs available
print(torch.cuda.get_device_name(0) if torch.cuda.is_available() else "No GPU detected")
```

```
False
0
No GPU detected
```

Task 3: Training YOLOv11 Model Objective: Train YOLOv11 on the prepared dataset.

Instructions:

Configure the training parameters (batch size, epochs, learning rate). Train the YOLOv11 model using the dataset. Monitor training progress (loss, accuracy, mAP). Save the trained model weights.

Expected Outcome: A trained YOLOv11 model ready for inference.

```
from ultralytics import YOLO
```

```
model = YOLO("yolo11n.pt")
results = model.train(
    data="/content/vehicles-1/data.yaml",
    epochs=5,
    batch=8,
    device='cpu'  # Use GPU
)
```

```
      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
        3/5         0G      1.323       1.42      1.089         34        640: 100%|██████████| 330/330 [37:49<00:00,  6.88s/
                  Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100%|██████████| 61/61 [05:53<00:00
                    all        966      13450      0.473      0.361      0.313      0.204

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
        4/5         0G      1.296      1.322      1.082         42        640: 100%|██████████| 330/330 [38:21<00:00,  6.97s/
                  Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100%|██████████| 61/61 [05:40<00:00
                    all        966      13450      0.427       0.42      0.346      0.229

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
        5/5         0G      1.263      1.227      1.066         66        640: 100%|██████████| 330/330 [37:34<00:00,  6.83s/
                  Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100%|██████████| 61/61 [05:33<00:00
                    all        966      13450      0.443      0.453      0.349      0.234

5 epochs completed in 3.654 hours.
Optimizer stripped from runs/detect/train/weights/last.pt, 5.5MB
Optimizer stripped from runs/detect/train/weights/best.pt, 5.5MB

Validating runs/detect/train/weights/best.pt...
Ultralytics 8.3.96 🚀 Python-3.11.11 torch-2.6.0+cu124 CPU (Intel Xeon 2.20GHz)
YOLO11n summary (fused): 100 layers, 2,584,492 parameters, 0 gradients, 6.3 GFLOPs
                  Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100%|██████████| 61/61 [04:29<00:00
                    all        966      13450      0.443      0.453      0.349      0.233
                big bus        210        273      0.705      0.608      0.701      0.521
              big truck        404       1162       0.69       0.42      0.588       0.35
                 bus-l-          8          8     0.0143        0.5     0.0111    0.00494
                 bus-s-         12         12          1          0     0.0031    0.00257
                    car        927       8537      0.799      0.772      0.819       0.46
              mid truck        118        257      0.399     0.0934      0.126      0.097
              small bus         43         49          0          0       0.01     0.0072
            small truck        517       1721      0.629      0.513      0.574      0.338
               truck-l-        266        433      0.309      0.617       0.38      0.285
               truck-m-        331        629      0.271      0.781       0.38      0.283
               truck-s-        147        221      0.175      0.498      0.186      0.132
              truck-xl-        110        148      0.325      0.628      0.411       0.32
Speed: 5.7ms preprocess, 259.2ms inference, 0.0ms loss, 4.7ms postprocess per image
Results saved to runs/detect/train
```

```
# Save the trained model weights
best_model_path = "/content/runs/detect/train/weights/best.pt"
print(f"Model training complete. Best model saved at: {best_model_path}")
```

```
Model training complete. Best model saved at: /content/runs/detect/train/weights/best.pt
```

Task 4: Model Inference and Evaluation

```
# Load the trained model
model = YOLO(best_model_path)
```

```
image_path = "/content/vehicles-1/test/images/adit_mp4-815_jpg.rf.fb532f30f712174b620afee0cfb1bfbb.jpg"
results = model(image_path, save=True, conf=0.5)
```

```
image 1/1 /content/vehicles-1/test/images/adit_mp4-815_jpg.rf.fb532f30f712174b620afee0cfb1bfbb.jpg: 640x640 3 cars, 1 truck-m-,
Speed: 5.6ms preprocess, 376.4ms inference, 7.0ms postprocess per image at shape (1, 3, 640, 640)
Results saved to runs/detect/predict
```

```
for result in results:
    result.show()
```


```
# Evaluate the model performance
metrics = model.val()

map_50 = metrics.box.map50  # mAP@50
map_50_95 = metrics.box.map  # mAP@50-95
precision = metrics.box.p.mean().item() if metrics.box.p.size > 0 else 0.0
recall = metrics.box.r.mean().item() if metrics.box.r.size > 0 else 0.0

print(f"\U0001F4CA mAP@50: {map_50:.4f}")
print(f"\U0001F4CA mAP@50-95: {map_50_95:.4f}")
print(f"\U0001F4C8 Precision: {precision:.4f}")
print(f"\U0001F4C9 Recall: {recall:.4f}")

if precision + recall > 0:
    f1_score = 2 * (precision * recall) / (precision + recall)
    print(f"\U0001F525 F1 Score: {f1_score:.4f}")
else:
    print("⚠ Cannot compute F1 Score (Precision + Recall = 0)")
```

```
Ultralytics 8.3.96 🚀 Python-3.11.11 torch-2.6.0+cu124 CPU (Intel Xeon 2.20GHz)
val: Scanning /content/vehicles-1/valid/labels.cache... 966 images, 3 backgrounds, 0 corrupt: 100%|██████████| 966/966 [00:00<?,
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100%|██████████| 61/61 [04:42<00:00,
                   all        966      13450      0.443      0.453      0.349      0.233
               big bus        210        273      0.705      0.608      0.701      0.521
             big truck        404       1162       0.69       0.42      0.588       0.35
                bus-l-          8          8     0.0143        0.5     0.0111    0.00494
                bus-s-         12         12          1          0     0.0031    0.00257
                   car        927       8537      0.799      0.772      0.819       0.46
             mid truck        118        257      0.399     0.0934      0.126      0.097
              small bus         43         49          0          0       0.01     0.0072
            small truck        517       1721      0.629      0.513      0.574      0.338
               truck-l-        266        433      0.309      0.617       0.38      0.285
               truck-m-        331        629      0.271      0.781       0.38      0.283
               truck-s-        147        221      0.175      0.498      0.186      0.132
              truck-xl-        110        148      0.325      0.628      0.411       0.32
Speed: 4.9ms preprocess, 272.9ms inference, 0.0ms loss, 4.7ms postprocess per image
Results saved to runs/detect/val4
📊 mAP@50: 0.3491
📊 mAP@50-95: 0.2334
📈 Precision: 0.4430
📉 Recall: 0.4525
🔥 F1 Score: 0.4477
```

```
for result in results:
    result.show()
```

Conclusion:-

The project "Vehicle Detection for Smart Traffic Management using YOLOv11" successfully implemented real-time object detection to identify vehicles such as cars, buses, and auto-rickshaws. The model demonstrated strong precision, indicating high confidence in its predictions. However, recall was relatively lower, highlighting missed detections.

Challenges were observed in detecting specific violations like helmet and no-helmet detection, which suggests the need for additional training data and optimized hyperparameters. Future improvements could focus on data augmentation, fine-tuning the YOLOv11 model, and exploring alternative architectures to enhance overall detection accuracy.

This project provides a foundational framework for automated traffic monitoring and violation detection, which can be further refined for real-world deployment in smart city applications.

Declaration

I, Manavi Pawar, confirm that the work submitted in this assignment is my own and has been completed following academic integrity guidelines.

Signature: Manavi Pawar