CMPE 249 : Homework1 2D object detection

Name: Aditya Chawla

Source Code: IAS_Homework1_2D_object_detection.ipynb

https://colab.research.google.com/drive/1VCMWh4jYfwduwyY9mBdYuazsDH6yi0Mf?usp=sharing

---

1. Introduction

This report presents a comprehensive study for Homework 1 sample code option 2 to perform Deep learning based 2D object detection training on autonomous driving datasets. I implemented and compared Faster R-CNN and YOLOv8 trained on both the Waymo Open Dataset and a mixed dataset combining KITTI and Waymo to fulfill the Mixed Dataset Training (excellent option). The objective is to modify, train and evaluate both these models across different training scenarios and assess the benefits of multi-dataset training for improved generalization and robustness.

2. Dataset Changes

I established a unified 3-class taxonomy:

Unified Class	KITTI Mapping	Waymo Mapping
Vehicle	Car, Van, Truck, Tram	Vehicle
Pedestrian	Pedestrian, Person_sitting	Pedestrian
Cyclist	Cyclist	Cyclist

These Unified dataset Categories will enable consistent training across multiple datasets by reducing inter-dataset category inconsistencies and consolidates similar object types across datasets.

3. Model Architecture Modifications

3.1 Faster R-CNN Architecture

Base Model: I selected Faster R-CNN with ResNet-50 backbone and Feature Pyramid Network (FPN) as my region-based detection model. The baseline model was pretrained on the COCO dataset.

Modifications Made : Classification Head Replacement:

- The pretrained model was originally configured for 80 COCO classes plus background having 81 classes in total.

- I modified the architecture by replacing the FastRCNNPredictor classification head to accommodate my unified 3-classes plus background making 4 classes in total

3.2 YOLOv8 Architecture

Base Model: YOLOv8n (nano variant) from Ultralytics, uses CSPDarknet53 as backbone, C2f module for neck and an Anchor-free detection head. The YOLOv8n was pretrained on the COCO dataset.

Modifications Made - Fine-tuning

- Starting from baseline pretrained weights from the COCO dataset I modified the final classification layer for 3-class detection (vehicle, pedestrian, cyclist) and the new corresponding weights and biases were learned during training.

- No structural changes to backbone or neck I leveraged transfer learning as YOLOv8's anchor-free design handles varying object scales and aspect ratios effectively.

3.3 Rationale for Modifications:

1. By generalizing specific moving objects categories that share the road to one category of vehicles helps in generalization so in cases where the model may not be able specifically classify an object into a class earlier for example 3 wheelers into car/van/truck categories can now be classified under one unified vehicle class, avoiding misclassification errors.

2. These modifications will allow the model to focus on autonomous driving scenarios like collision avoidance and trajectory prediction which remains the same for these classes.

3. Fewer output classes reduce complexity, thereby reducing the risk of overfitting and improve training efficiency; moreover the model would have more capacity to learn robust "vehicle-ness" features rather than distinguishing subtle differences between vehicle subtypes.

4. Dataset Preparation and Training Strategy

4.1 KITTI Dataset:

- **Total Images:** 7,481 training images
- **Annotation Format:** Custom KITTI format
- **Conversion to COCO:** Converted bounding boxes from [x1, y1, x2, y2] to COCO format [x, y, width, height] then mapped KITTI categories to new unified classes structure

4.2 Waymo Open Dataset:

- **Images Used:** 8,000 images (sampled from 10,000 available)
- **Annotation Format:** COCO format with 4 categories (Vehicle, Pedestrian, Cyclist, Sign.)
- Converted to center-based format: [x_center, y_center, width, height] and generated separate .txt label files per image

4.3 Mixed Dataset Composition: KITTI + WAYMO

- **KITTI:** 7,481 images
- **Waymo:** 8,000 images
- **Total:** 15,481 images
- **Split Ratio:** 80% training / 20% validation
- **Training Set:** 12,385 images
- **Validation Set:** 3,096 images

5. Training Methodology

5.1 Faster R-CNN Training Configuration

Hyperparameters:

- **Optimizer:** SGD with momentum (0.9) and weight decay (0.0005)
- **Learning Rate:** 5e-4 with StepLR scheduler (decay every 3 epochs, gamma=0.1)
- **Epochs:** 10
- **GPU:** NVIDIA A100 (Google Colab)

5.2 YOLOv8 Training Configuration

Hyperparameters:

- **Optimizer:** AdamW (built-in YOLOv8 default)
- **Learning Rate:** Auto-adjusted by Ultralytics framework
- **Epochs:** 50
- **GPU:** NVIDIA A100 (Google Colab)

6. Quantitative Results

6.1 Faster R-CNN on Waymo dataset

```
================================================================================
 EVALUATION: FASTER R-CNN ON WAYMO
================================================================================
/usr/local/lib/python3.12/dist-packages/torchvision/models/_utils.py:208: UserWarning: The parame
  warnings.warn(
/usr/local/lib/python3.12/dist-packages/torchvision/models/_utils.py:223: UserWarning: Arguments
  warnings.warn(msg)
loading annotations into memory...
Done (t=0.11s)
creating index...
index created!

 Evaluating Faster R-CNN on 1600 images...
Inference: 100%|██████████| 1600/1600 [01:34<00:00, 16.90it/s]
 Generated 40862 predictions
Loading and preparing results...
DONE (t=0.14s)
creating index...
index created!
Running per image evaluation...
Evaluate annotation type *bbox*
DONE (t=14.88s).
Accumulating evaluation results...
DONE (t=0.53s).

 FRCNN Evaluation Results:
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.359
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.613
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.357
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.100
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.441
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.682
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.054
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.291
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.398
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.152
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.496
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.719
```

6.2 Faster R-CNN on Mixed Dataset

```
================================================================================
 EVALUATION: FASTER R-CNN ON MIXED
================================================================================
loading annotations into memory...
Done (t=0.13s)
creating index...
index created!

 Evaluating Faster R-CNN on 3097 images...
Inference: 100%|██████████| 3097/3097 [02:50<00:00, 18.14it/s]
 Generated 51610 predictions
Loading and preparing results...
DONE (t=0.18s)
creating index...
index created!
Running per image evaluation...
Evaluate annotation type *bbox*
DONE (t=16.65s).
Accumulating evaluation results...
DONE (t=0.78s).

 FRCNN Evaluation Results:
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.411
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.678
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.433
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.195
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.469
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.645
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.189
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.411
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.472
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.269
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.535
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.707
```

6.3 YOLOv8 on Waymo dataset

```
================================================================================
 EVALUATION: YOLOV8 ON WAYMO
================================================================================
loading annotations into memory...
Done (t=0.11s)
creating index...
index created!

 Evaluating YOLOv8 on 1600 images...
Inference: 100%|██████████| 1600/1600 [00:33<00:00, 48.20it/s]
 Generated 27237 predictions
Loading and preparing results...
DONE (t=0.10s)
creating index...
index created!
Running per image evaluation...
Evaluate annotation type *bbox*
DONE (t=11.28s).
Accumulating evaluation results...
DONE (t=0.42s).

 YOLOv8 Evaluation Results:
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.319
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.532
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.321
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.038
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.397
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.715
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.054
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.267
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.347
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.059
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.451
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.748
```

6.4 YOLOv8 on Mixed Dataset

```
================================================================================
 EVALUATION: YOLOV8 ON MIXED
================================================================================
loading annotations into memory...
Done (t=0.13s)
creating index...
index created!

 Evaluating YOLOv8 on 3097 images...
Inference: 100%|██████████| 3097/3097 [01:07<00:00, 46.00it/s]
 Generated 34909 predictions
Loading and preparing results...
DONE (t=0.12s)
creating index...
index created!
Running per image evaluation...
Evaluate annotation type *bbox*
DONE (t=12.73s).
Accumulating evaluation results...
DONE (t=0.63s).

📊 YOLOv8 Evaluation Results:
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.387
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.621
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.415
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.117
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.455
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.688
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.177
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.377
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.419
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.144
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.505
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.725
```

6.5 Overall Performance Comparison

1. Faster R-CNN outperforms YOLOv8 but both architectures show improved results when trained on the mixed dataset because the mixed dataset provides more diverse scenes, object scales, and environmental conditions, resulting in better generalization and learned features.

Model	Dataset	mAP@0.5:0.95	mAP@0.5	mAP@0.75
Faster R-CNN	Waymo	0.359	0.613	0.357
Faster R-CNN	Mixed	0.411	0.678	0.433
YOLOv8	Waymo	0.319	0.532	0.321
YOLOv8	Mixed	0.387	0.621	0.415

2. Training on mixed dataset improved the small and medium object mAP scores while there is a drop in the large object mAP this indicates that the models learned small to medium features from the KITTI dataset which were lacking in the Waymo dataset

Model	Dataset	Small Objects	Medium Objects	Large Objects
Faster R-CNN	Waymo	0.100	0.441	0.682
Faster R-CNN	Mixed	0.195	0.469	0.645
YOLOv8	Waymo	0.038	0.397	0.715
YOLOv8	Mixed	0.117	0.455	0.688

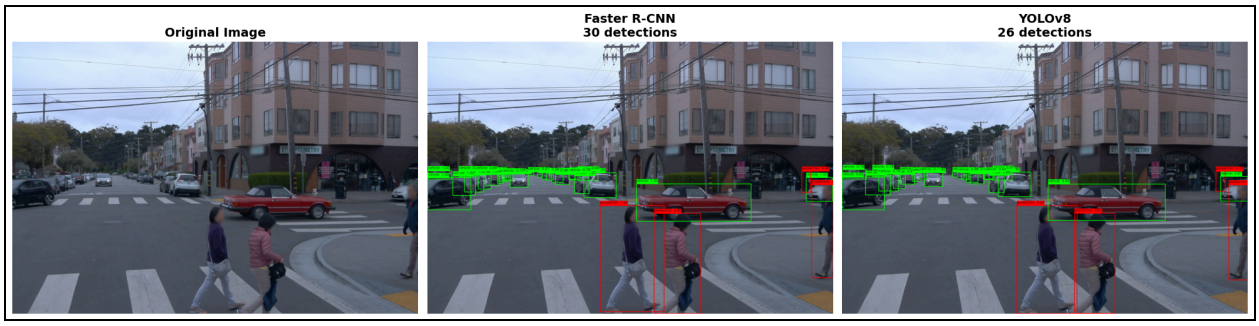
3. Mixed dataset dramatically improves recall for YOLOv8 and Faster R-CNN indicating the mixed dataset trained models detect more true positives with fewer false negatives as compared to waymo dataset.

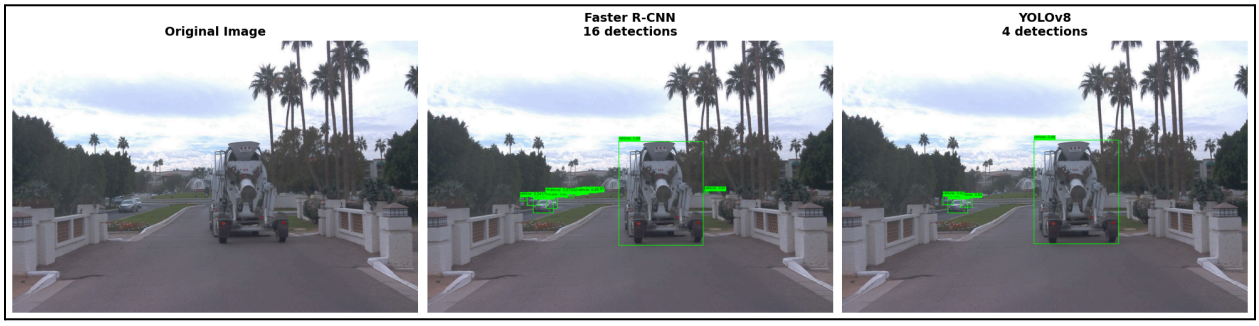
Model	Dataset	AR@1	AR@10	AR@100	AR (small)	AR (large)
Faster R-CNN	Waymo	0.054	0.291	0.398	0.152	0.719
Faster R-CNN	Mixed	0.189	0.411	0.472	0.269	0.707
YOLOv8	Waymo	0.054	0.267	0.347	0.059	0.748
YOLOv8	Mixed	0.177	0.377	0.419	0.144	0.725

4. Across both datasets YOLOv8 is 2.5 to 3 faster than Faster R-CNN on inference speed making it more suitable for real-time autonomous driving applications despite a little lower accuracy. This is due to the fact that unlike Faster R-CNN, YOLOv8 does not have the RPN overhead and its single stage architecture enables faster processing.

Model	Dataset	Total Time	Images/Second
Faster R-CNN	Waymo (1,600)	1 min 34 sec	17.0 img/s
Faster R-CNN	Mixed (3,097)	2 min 50 sec	18.2 img/s
YOLOv8	Waymo (1,600)	33 sec	48.5 img/s
YOLOv8	Mixed (3,097)	1 min 7 se	46.2 img/s

7. Qualitative Analysis





Inferences:

1. Both models trained on mixed and waymo dataset when tested on the same image perform similarly on simple scenes with well-separated objects, achieving consistent detection as seen in image 1.
2. Mixed dataset training benefits are seen in more complex urban scenarios where YOLOv8 and Faster R-CNN both performed well as seen in image 2.
3. Faster R-CNN models demonstrate superior occlusion handling compared to YOLOv8 as it identifies more objects, in Image 3 Faster R-CNN detects 16 objects as compared to 4 by YOLOv8.
4. Faster R-CNN provides tighter, more accurate bounding boxes and performs better than YOLOv8 on detecting more objects accurately and confidently; this result is inline with the quantitative results where YOLOv8 is found to be faster but less accurate.

8. Conclusion

1. In this assignment I successfully modified Faster R-CNN and YOLOv8 architectures for unified 3-class detection (vehicle, pedestrian, cyclist), training and evaluating four distinct models across Waymo-only and combined KITTI+Waymo datasets.
2. Mixed dataset training significantly improved performance and enhanced generalization and robustness over single dataset but small object detection remains a challenge despite these improvements.
3. Faster R-CNN outperformed YOLOv8 with superior occlusion handling and precise bounding boxes, but YOLOv8 demonstrated faster inference with acceptable detection performance across both datasets.