May 2026

AI3011

DATA-CENTRIC PREPROCESSING FOR OBJECT DETECTION IN UNSTRUCTURED INDIAN ROAD SCENARIOS

MLPR End Semester Presentation

Daiwik Chilukuri, Farhan Hussain, Niksh Hiremath

PRESENTATION OUTLINE

01. Problem Statement

02. Literature Review

03. Dataset & EDA

04. ML Methodology & Challenges

05. Results & Performance Metrics

PROBLEM STATEMENT

How do data-centric preprocessing techniques such as augmentation and small object aware training affect object detection performance on unstructured Indian road scenes?

SIGNIFICANCE & APPLICATIONS

- **ADAS & Autonomous Vehicles** - Safer navigation in unstructured Indian traffic
- **Traffic Monitoring** - Real time detection of vulnerable road users & road anomalies
- **Cost-Effective AI Deployment** - Preprocessing based improvements without model redesign
- **Research Foundation** - Data centric baselines for emerging market scenarios
- **Global Relevance** - Applicable to similar unstructured environments worldwide

Credit @JonBbC_TechGeek

IDD: A Dataset for Exploring Problems of Autonomous Navigation in Unconstrained Environments (IEEE WACV 2019)[1]

Overview

- Introduces Indian Driving Dataset (IDD) for unstructured road scenes
- Focus on semantic segmentation in complex Indian traffic environments

Methodology / Findings

- Collected 10,000+ annotated images from 182 drives in Indian cities
- Evaluated segmentation models (DRN, ERFNet, DeepLab)
- Models show significant performance drop compared to Cityscapes

Limitations

- Focus only on segmentation, not object detection
- No analysis of training pipelines or preprocessing techniques

Source: Varma et al. (2019)

[1] Varma et al., WACV 2019 | IIIT-Hyderabad, Intel, UCSD

Evaluation of Detection and Segmentation Tasks on Driving Datasets (IAPR CVIP 2021)[2]

Overview

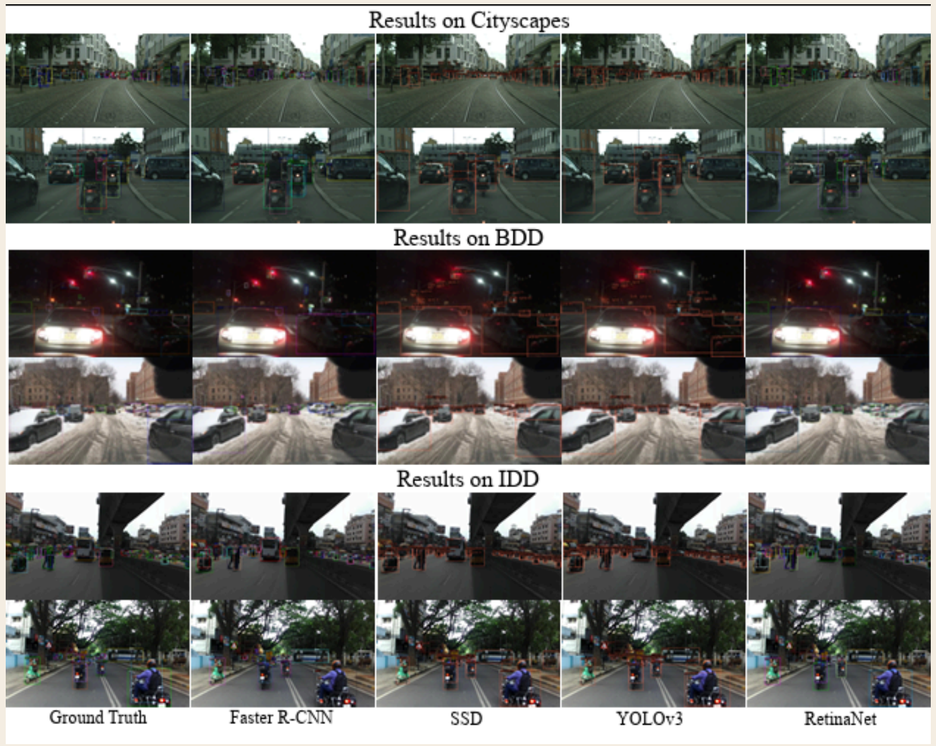
- Evaluates detection and segmentation models across Cityscapes, BDD100K, and IDD datasets
- Investigates cross dataset generalization

Methodology / Findings

- Trained detectors: Faster R-CNN, SSD, YOLOv3, RetinaNet
- Models trained separately on each dataset
- mAP significantly lower on IDD compared to Cityscapes
- Models were trained on one dataset and evaluated on different datasets to measure cross dataset generalization.

Limitations

- No COCO → IDD transfer learning experiments
- Fixed training pipeline, no augmentation or imbalance analysis



Source: Singh et al. (2021)

[2] Singh et al., CVIP 2021 | IIIT-Hyderabad, UC Berkeley, Intel

ORDER: Open-World Object Detection on Road Scenes (NeurIPS Workshop ML4AD 2021)[3]

Overview

- Addresses open world object detection in road scenes
- Experiments on BDD and IDD datasets

Methodology / Findings

- Built on Faster R-CNN + ResNet-50
- Introduced Feature-Mix, Focal Regression, Curriculum Training
- Improved detection of unknown objects and small objects

Limitations

- Focus on open world detection complexity
- Not suited for simple closed set detection analysis

Source: Singh et al. (2021)

[3] Singh et al., NeurIPS ML4AD 2021 | IIIT-Hyderabad, IIT-Hyderabad, IIT-Delhi

DriveIndia: An Object Detection Dataset for Diverse Indian Traffic Scenes (TiHAN-IIT Hyderabad, 2025)[4]

Overview

- Large scale Indian traffic detection dataset
- 66k images, 24 classes including road anomalies

Methodology / Findings

- Benchmarked detectors: YOLOv5, YOLOv8, RT-DETR, EfficientDet
- Best performance: YOLOv8 (~78% mAP50)
- Strong results for common vehicles, weaker for rare classes

Limitations

- Focus on dataset creation + benchmarking
- No study of training recipe or preprocessing effects

Source: Kumar et al.(2025)

[4] Kumar et al., arXiv 2025 | IIT Hyderabad

LITERATURE REVIEW SYNTHESIS

Paper	Dataset	Detector	Preprocessing Study	Gap We Fill
Varma et al.	IDD (seg)	None	None	No detection; no data-centric study
Singh et al. CVIP	IDD, BDD	FRCNN / SSD / YOLO	None	No augmentation
ORDER Singh et al.	IDD, BDD	FRCNN-based	Partial (loss side)	Model-side only; no data-centric baseline
DriveIndia Kumar et al.	DriveIndia	YOLO / RT-DETR	None	No training recipe study
Ours	**IDD 95K**	**YOLOv11s, YOLO26s, FRCNN**	**Full 3x3 ablation**	**First systematic data-centric study on IDD 95K**

IDD DETECTION 95K[5]

Real World -Complexity : Unlike controlled environments, the IDD-95K captures extreme Indian road conditions, featuring heavily mixed traffic with auto-rickshaws, stray animals, billboards, and frequent overlaps of pedestrians and vehicles.

Why Chosen: Established baseline; enables comparison with prior research

Data Collection: Front-facing stereo cameras, 182 sequences in Bangalore/Hyderabad

Ethical Concerns: Public roads, academic license, object class focus only

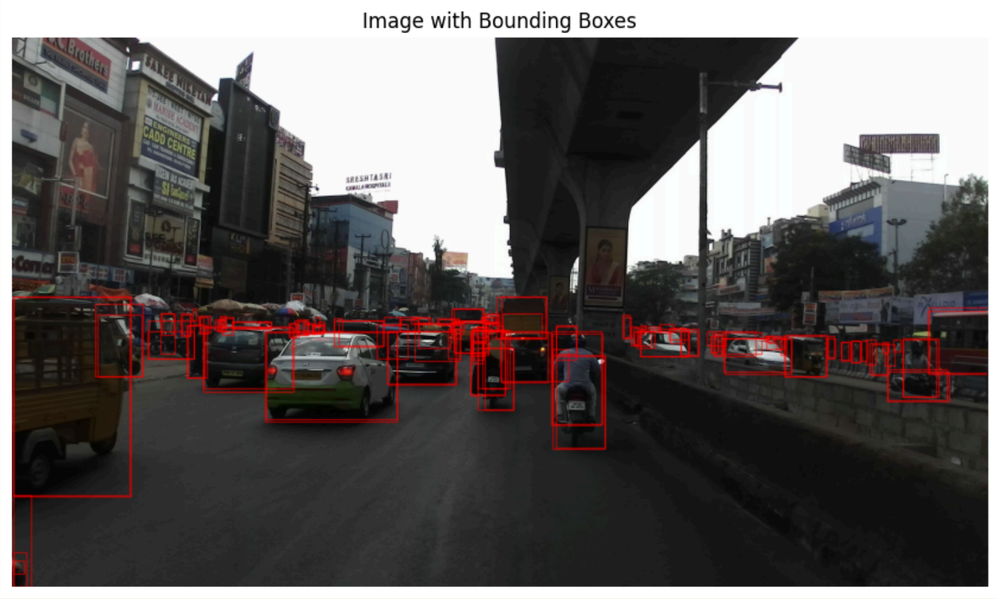
Features & Datapoints:

- 95,155 images (JSON Format)
- Train: ~65,328 images
- Validation: ~9,977 images
- Test: ~19,850 images
- 13 critical classes

Labels (13): car, bus, autorickshaw, truck, motorcycle, rider, person, bicycle, ego vehicle, traffic sign, traffic light, pole, vehicle fallback

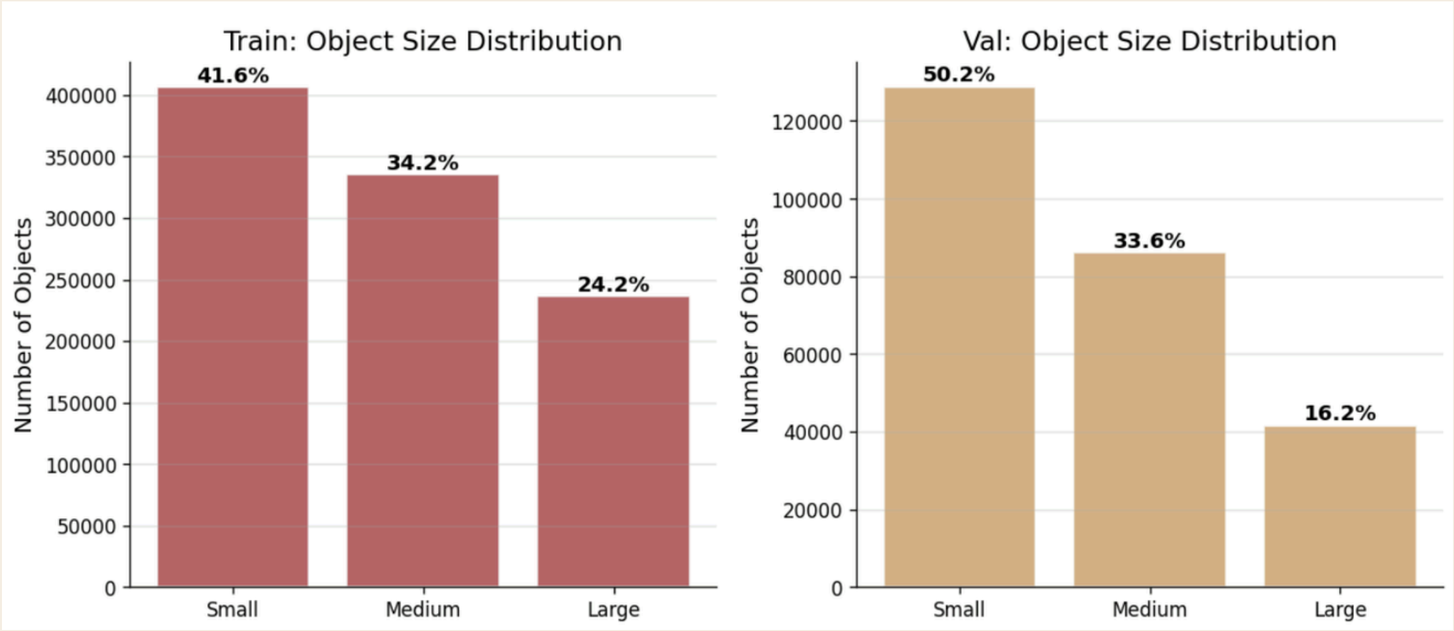
[5] INSAAN Dataset | IIIT Hyderabad

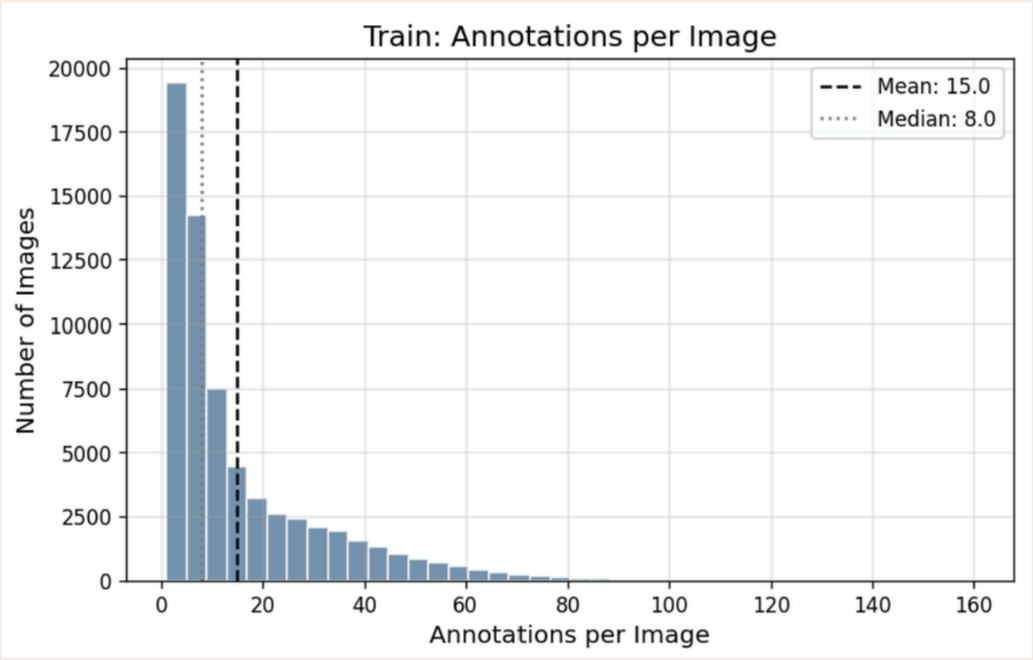
A GLIMPSE INTO IDD 95K

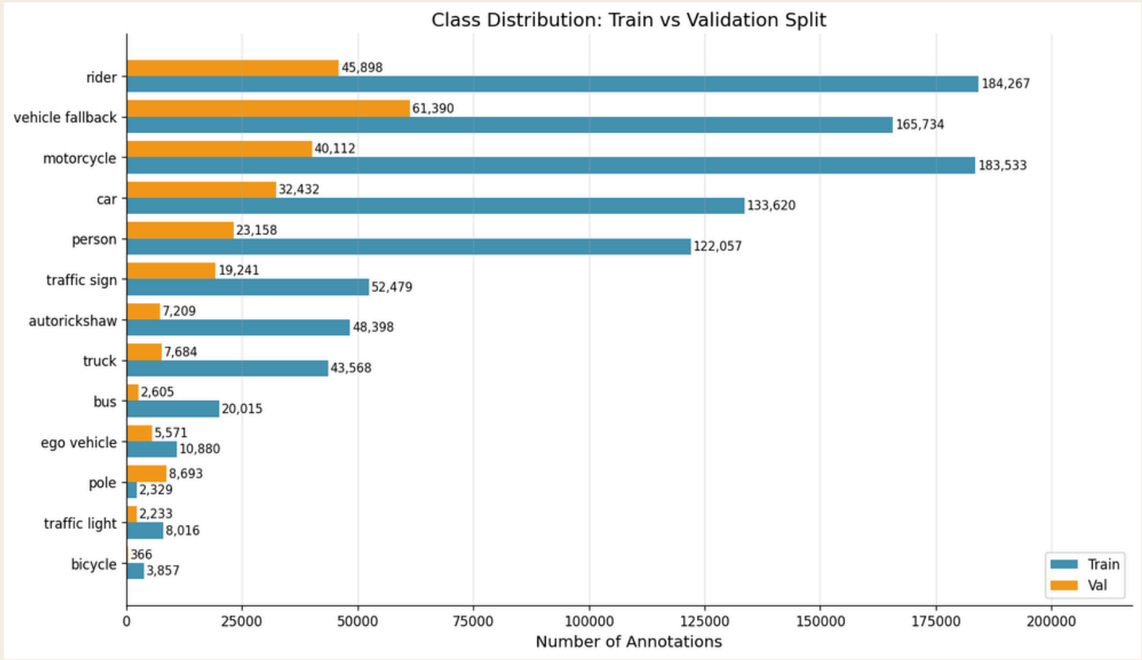


EDA PLOTS

PART 1

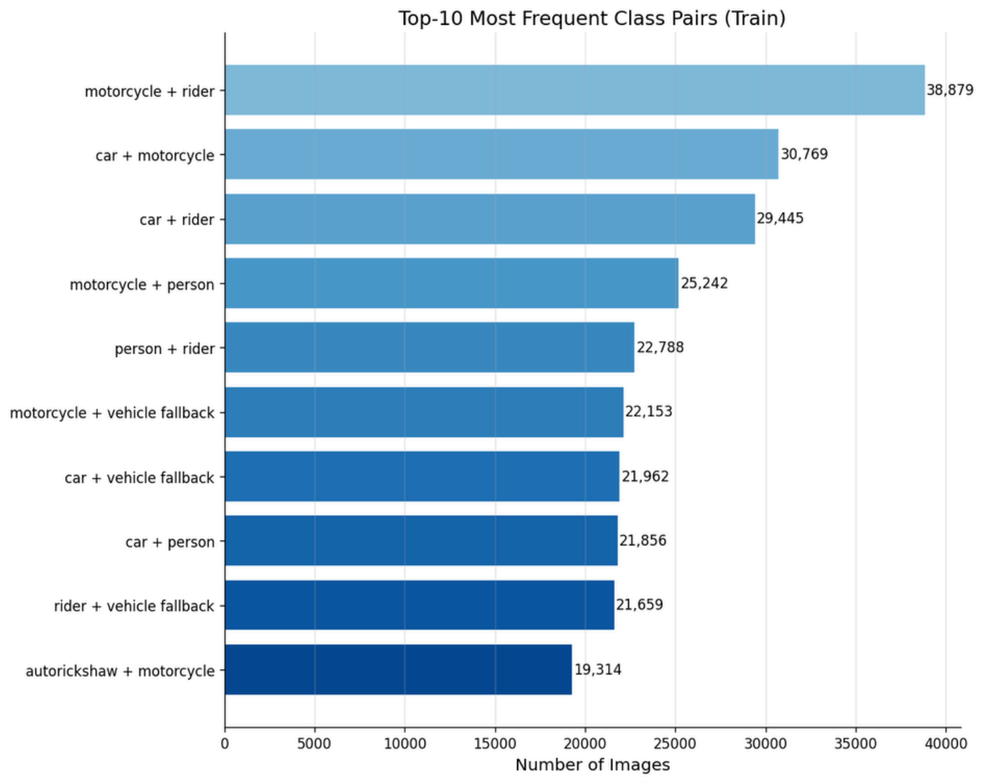


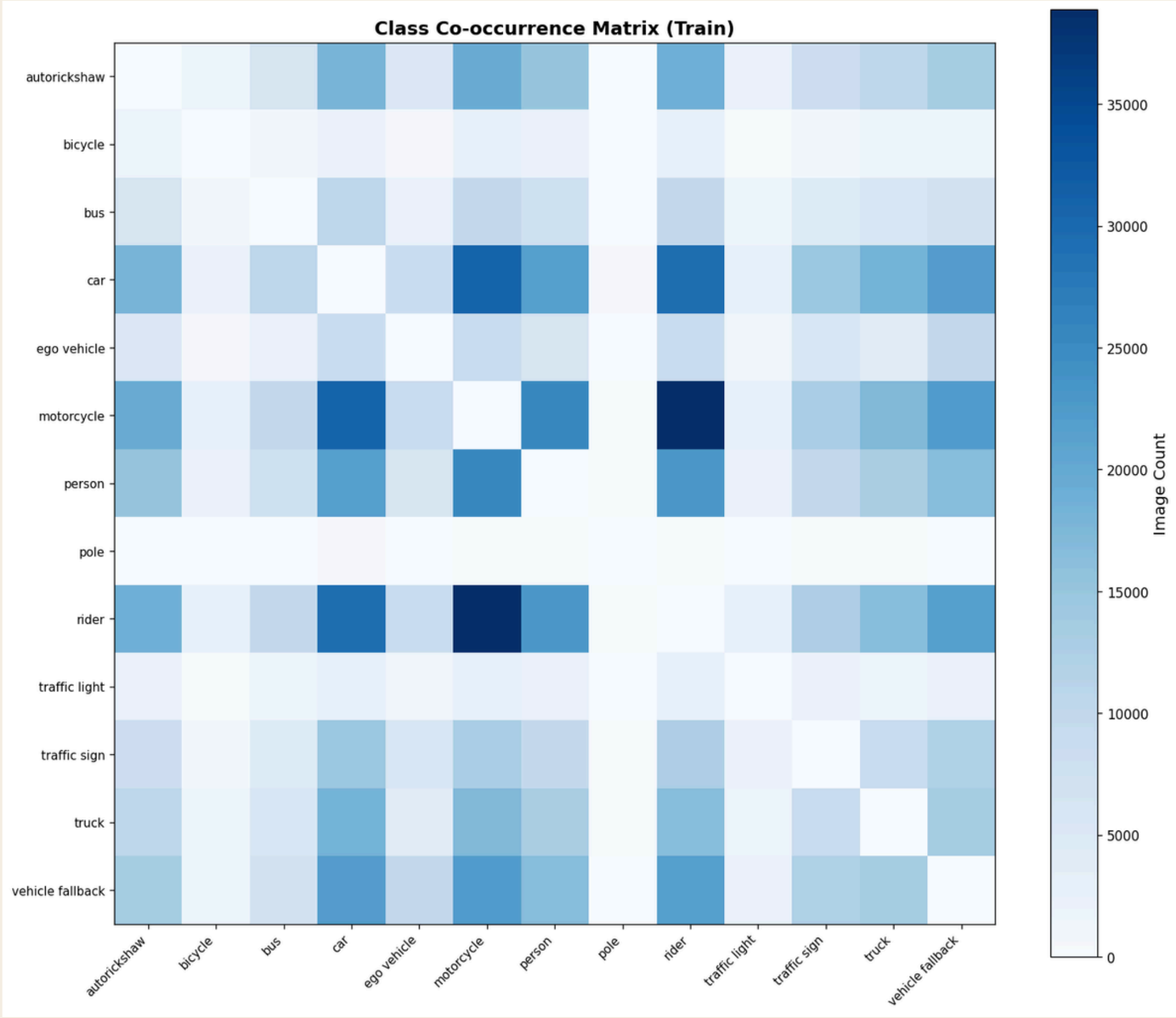




EDA PLOTS

PART 2





PRE-PROCESSING ON IDD95K DATASET

Annotation Parsing

- Original: JSON bounding box format
- Converted to: YOLO txt format (normalised cx, cy, w, h) for YOLO models; FRCNN JSON format for Faster R-CNN
- Label symlinks created alongside image files for Ultralytics compatibility

Train/Val/Test Split Design

- The dataset provided a pre-defined train/val split (65,328 train + 9,977 val images = 75,305 images)
- We combined both and performed our own stratified random split to create a dedicated held-out test set
- Final split: Train 54,220 (72%) / Val 6,024 (8%) / Test 15,061 (20%)
- Verified class balance across all three splits such that each class maintains ~72%/8%/20% ratio

Class Pruning

- Removed 'pole': annotation inconsistency across images; ambiguous geometry
- Removed 'vehicle fallback': catch-all category (227K annotations would dominate training but provide no learnable signal for specific vehicle types)
- Final: 11 clean, semantically distinct classes

No Missing Features: Pixel values are complete across all images

Features Dimensionality Reduction: Explicit dimensionality reduction not required as it is taken care in convolution operations of CNN

ML METHODOLOGY

Objective:

- Isolate the effect of preprocessing such that architecture and training setup kept constant across all models

Models Trained (3 models x 3 configs = 9 total):

- YOLOv11s - single stage, anchor free, fast inference
- YOLOv26s - newer single stage architecture (2026)
- YOLO "s" (small) variants were selected for real-time, deployment-feasible inference.
- Faster R-CNN ResNet50-FPN - two stage, anchor based detector
- All initialised from COCO pretrained weights

How The Models Work

- YOLOv11s - divides image into grid, predicts boxes and classes in a single forward pass
- YOLO26s - same single-stage approach, improved neck and head architecture over v11
- Faster R-CNN - first generates region proposals via RPN, then classifies each region separately; two-stage process gives higher accuracy at cost of speed
- All use FPN (Feature Pyramid Network) - detects objects at multiple scales simultaneously

ML METHODOLOGY

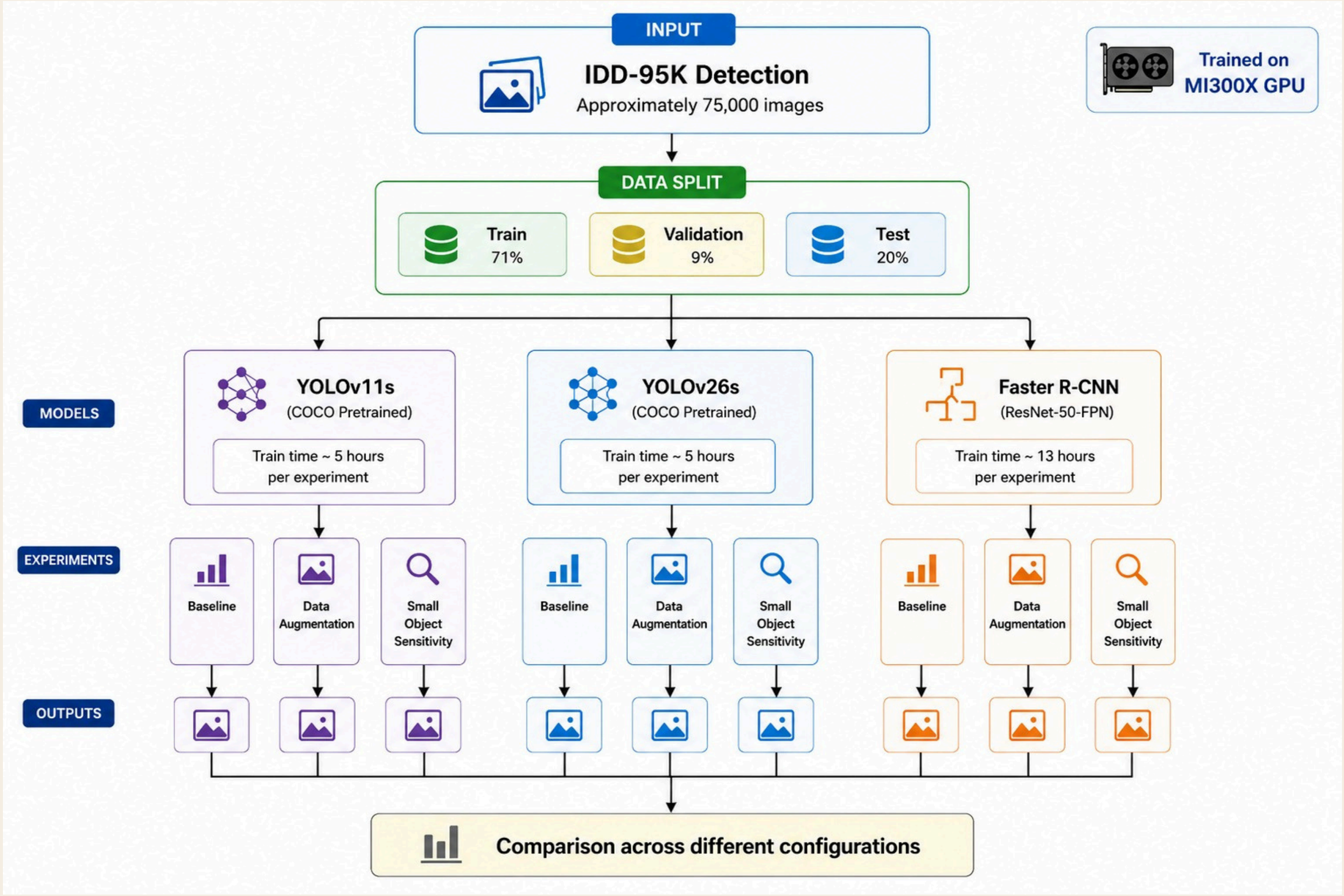
3 Preprocessing Conditions:

- Baseline - zero augmentation, pure fine-tune
- Data Augmentation techniques[6] - hsv_h = 0.015 - hue shift ±1.5%
  hsv_s = 0.7 - saturation shift ±70%
  hsv_v = 0.4 - brightness shift ±40%
  scale = 0.5 - random scale ±50%
  translate = 0.1 - random translation ±10%
  erasing = 0.4 - random erasing probability 40%
- Small Object Sensitivity - input resolution changed from 640 → 1280px, no augmentation

First Benchmark on IDD 95K Detection : Prior work (Varma et al., Singh et al.) was conducted on IDD 10K (2019). IDD 95K Detection is a significantly larger and more recent release with no published detection benchmark. Our 3×3 ablation is the first systematic evaluation on this dataset ,establishing baseline numbers for the community.

[6] ULTRALYTICS GITHUB REPOSITORY

ML METHODOLOGY



CHALLENGES FACED

Compute access :

- Free platforms (Colab) → 24+ hrs/model with disconnections making 9 models training infeasible
- Raised $900, trained on AMD MI300X GPUs; still took 90+ total GPU hours

Dataset failure - DriveIndia discarded mid-project

- Image count received via email didn't match the paper's claims → credibility concerns
- Fully discarded mid-project; pivoted to IDD 95K, forcing experimental redesign

Ego Vehicle Overfitting (YOLO Aug Models)

- AP spiked to 0.921 in YOLO26s data augmentation model - learned a positional shortcut, not a real visual feature
- Ego vehicle appears in every image, same position, same size → model memorised location, not appearance
- Realised this late in the project; no corrective action (e.g. class-specific augmentation suppression or reweighting) was applied thus acknowledged as a limitation

UNDERSTANDING THE METRICS

- **$mAP_{0.5}$** : A detection is counted correct only if the predicted box overlaps the ground truth by ≥50%. Averaged across all 11 classes.

- **$mAP_{0.5:0.95}$** : Stricter version averaged across 10 IoU thresholds (0.50 to 0.95). Rewards precise box localisation, not just rough overlap.

- **Precision** : Of all boxes the model predicted, what fraction were actually correct.

- **Recall** : Of all real objects in the scene, what fraction did the model successfully detect.

- **AP_S / AP_M / AP_L** : Detection accuracy computed separately by object size. Small: area < 1024 px², Medium: 1024–9216 px², Large: > 9216 px².

- **Per-Class $AP_{0.5}$** : mAP@50 broken down individually for each of the 11 classes. Reveals which object types the model handles well and which it consistently misses

RESULTS & PERFORMANCE METRICS

All 9 models evaluated on a held-out test set of 15,061 images across 11 classes.

Model	Experiment	mAP@0.5	mAP@0.5:0.95	Precision	Recall	AP_S	AP_M	AP_L
YOLOv11s	baseline	0.541	0.334	0.718	0.483	0.073	0.255	0.695
YOLOv11s	data_aug	0.623	0.408	0.795	0.564	0.081	0.271	0.729
YOLOv11s	small_obj	0.623	0.402	0.786	0.540	0.115	0.301	0.647
YOLOv26s	baseline	0.575	0.356	0.744	0.526	0.086	0.271	0.672
YOLOv26s	data_aug	0.633	0.417	0.812	0.558	0.089	0.279	0.683
YOLOv26s	small_obj	0.666	0.428	0.757	0.583	0.138	0.303	0.644
FRCNN	baseline	0.661	0.409	—	—	0.110	0.271	0.563
FRCNN	data_aug	0.646	0.247	—	—	0.105	0.255	0.581
FRCNN	small_obj	0.666	0.394	—	—	0.127	0.268	0.473

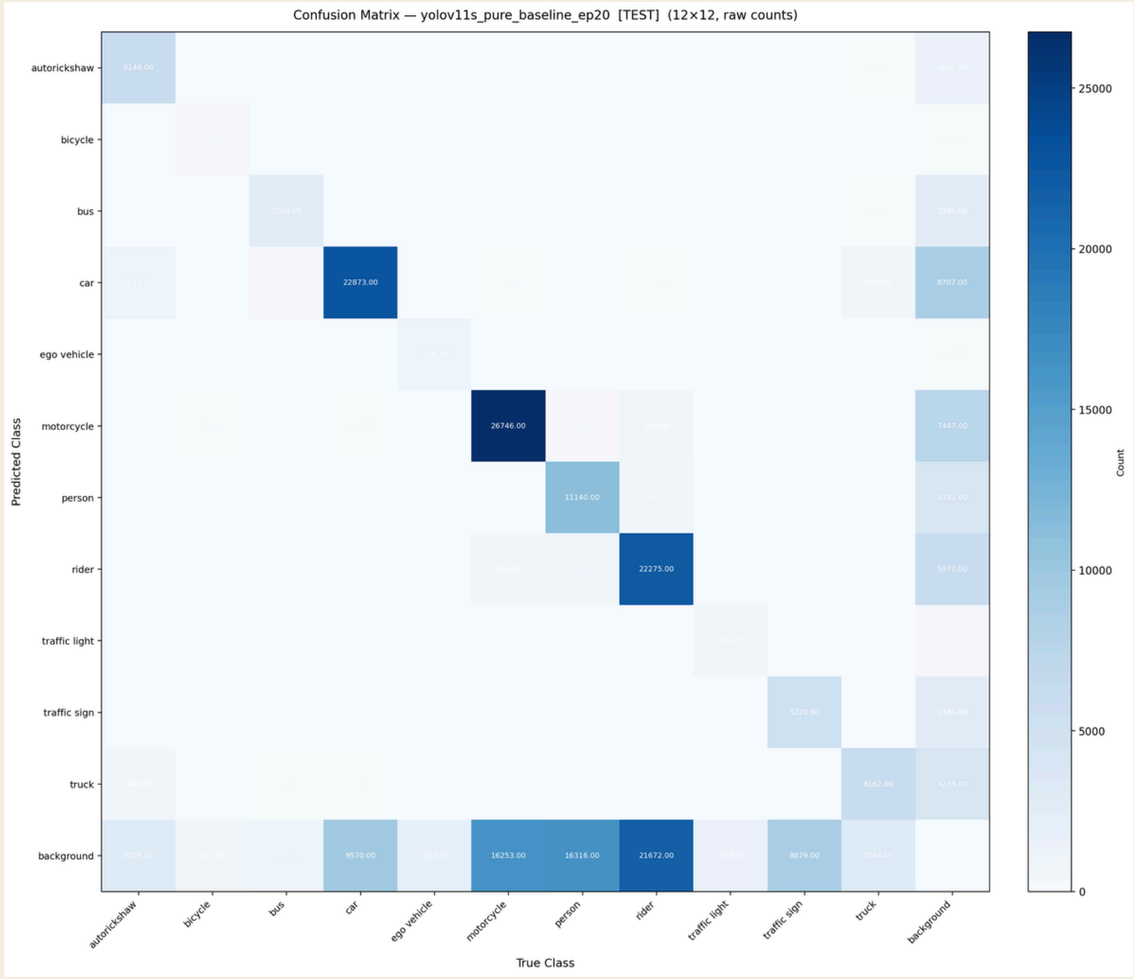
AP_S/AP_M/AP_L use COCO area thresholds: Small < 1024 px², Medium 1024–9216 px², Large ≥ 9216 px².

RESULTS & PERFORMANCE METRICS

(i) YOLOv11s Baseline

Per class AP:

- autorickshaw = 0.6691
- bicycle = 0.3284
- bus = 0.6529
- car = 0.6888
- ego vehicle = 0.6314
- motorcycle = 0.6447
- person = 0.4284
- rider = 0.5374
- traffic light = 0.3843
- traffic sign = 0.3815
- truck = 0.604



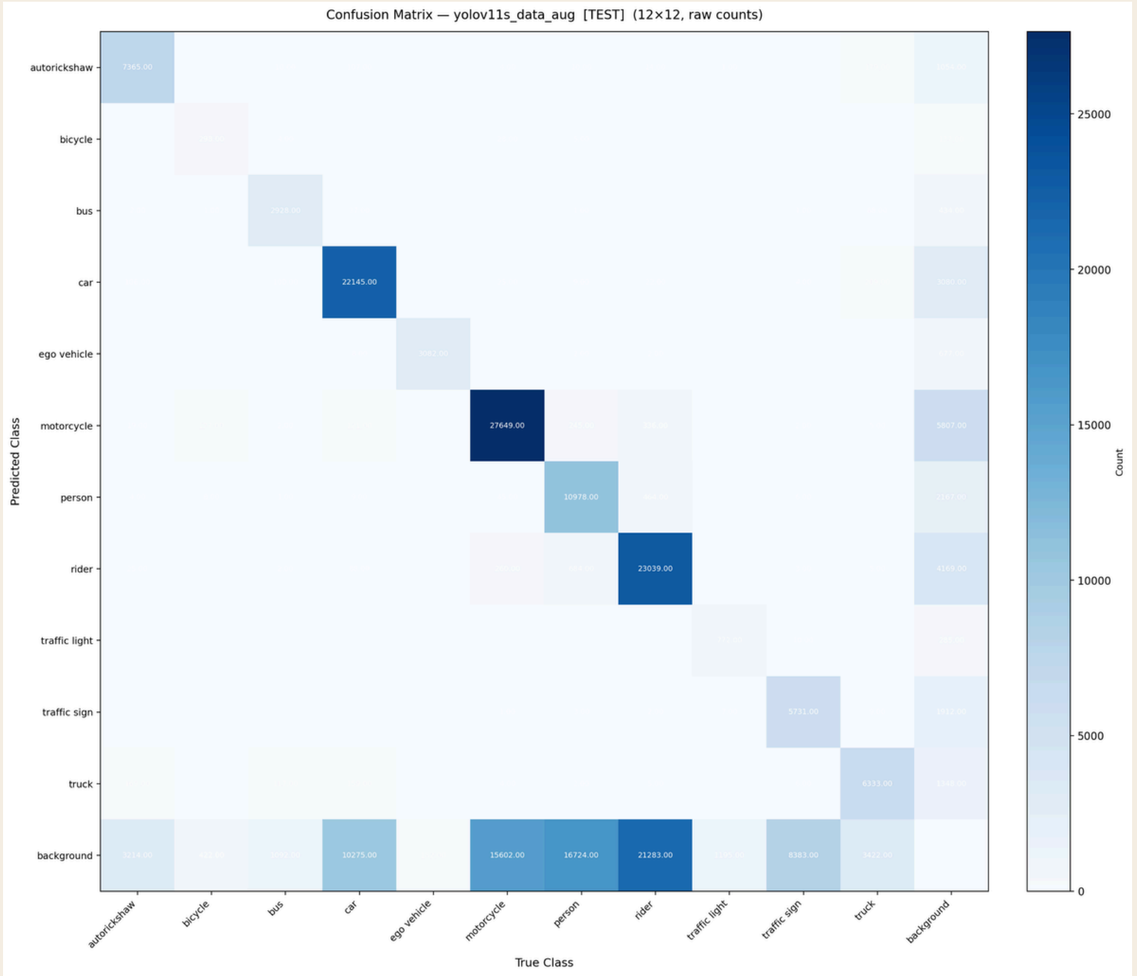
mAP@0.5	mAP@0.5:0.95	Precision	Recall	AP_S	AP_M	AP_L	Epochs	Train Time
0.541	0.3342	0.7183	0.4826	0.0729	0.2548	0.6949	20	~3.5h

RESULTS & PERFORMANCE METRICS

(ii) YOLOv11s Data Augmentation

Per class AP:

- autorickshaw = 0.7523
- bicycle = 0.3837
- bus = 0.7716
- car = 0.7403
- ego vehicle = 0.9212
- motorcycle = 0.6826
- person = 0.47
- rider = 0.5749
- traffic light = 0.426
- traffic sign = 0.4377
- truck = 0.6902



mAP@0.5	mAP@0.5:0.95	Precision	Recall	AP_S	AP_M	AP_L	Epochs	Train Time
0.6228	0.4084	0.7951	0.5644	0.0808	0.2711	0.7285	50*	~5h

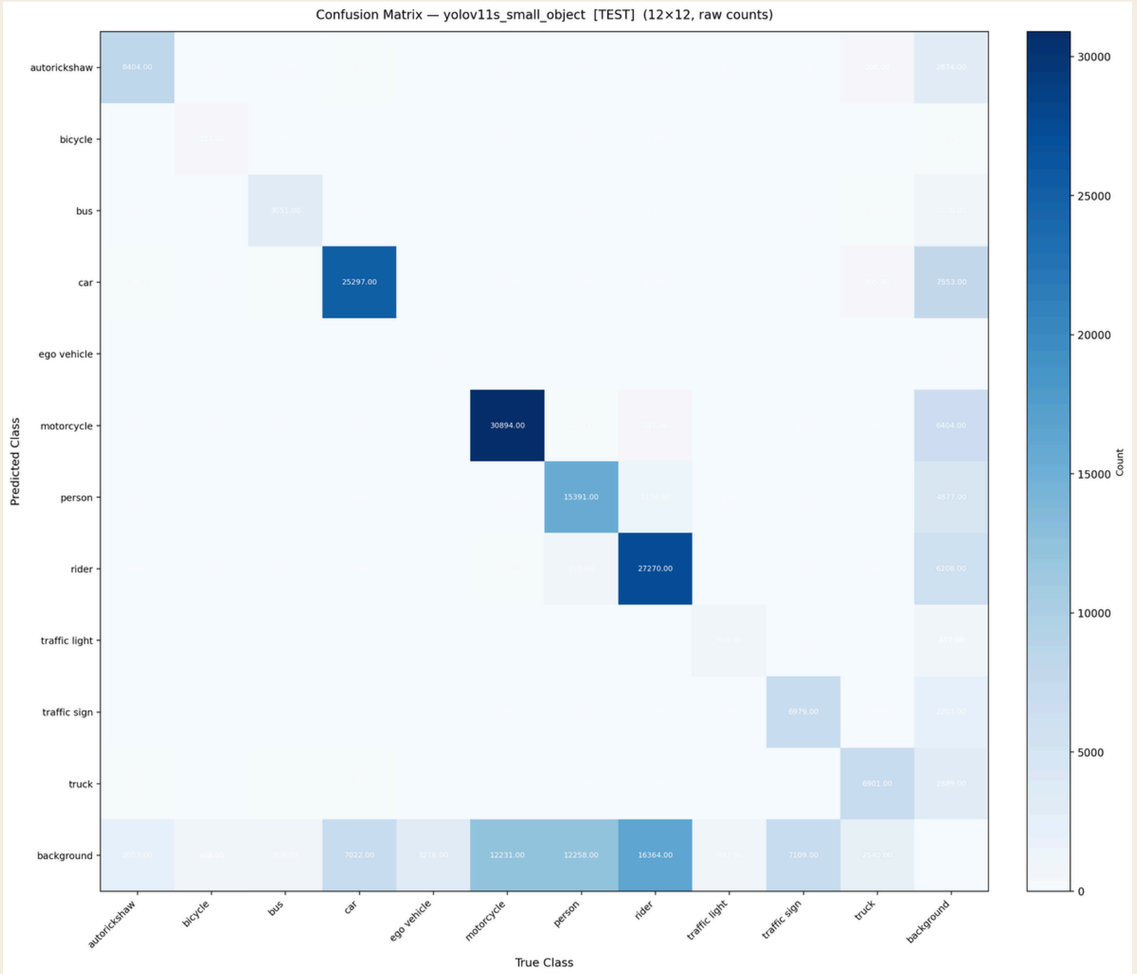
* Early stopping enabled

RESULTS & PERFORMANCE METRICS

(iii) YOLOv11s Small Object Handling

Per class AP:

- autorickshaw = 0.7939
- bicycle = 0.406
- bus = 0.7728
- car = 0.7935
- ego vehicle = 0.2918
- motorcycle = 0.7635
- person = 0.5902
- rider = 0.6688
- traffic light = 0.5034
- traffic sign = 0.5481
- truck = 0.719



mAP@0.5	mAP@0.5:0.95	Precision	Recall	AP_S	AP_M	AP_L	Epochs	Train Time
0.6228	0.4024	0.7863	0.5403	0.1153	0.3007	0.6465	50*	~5h

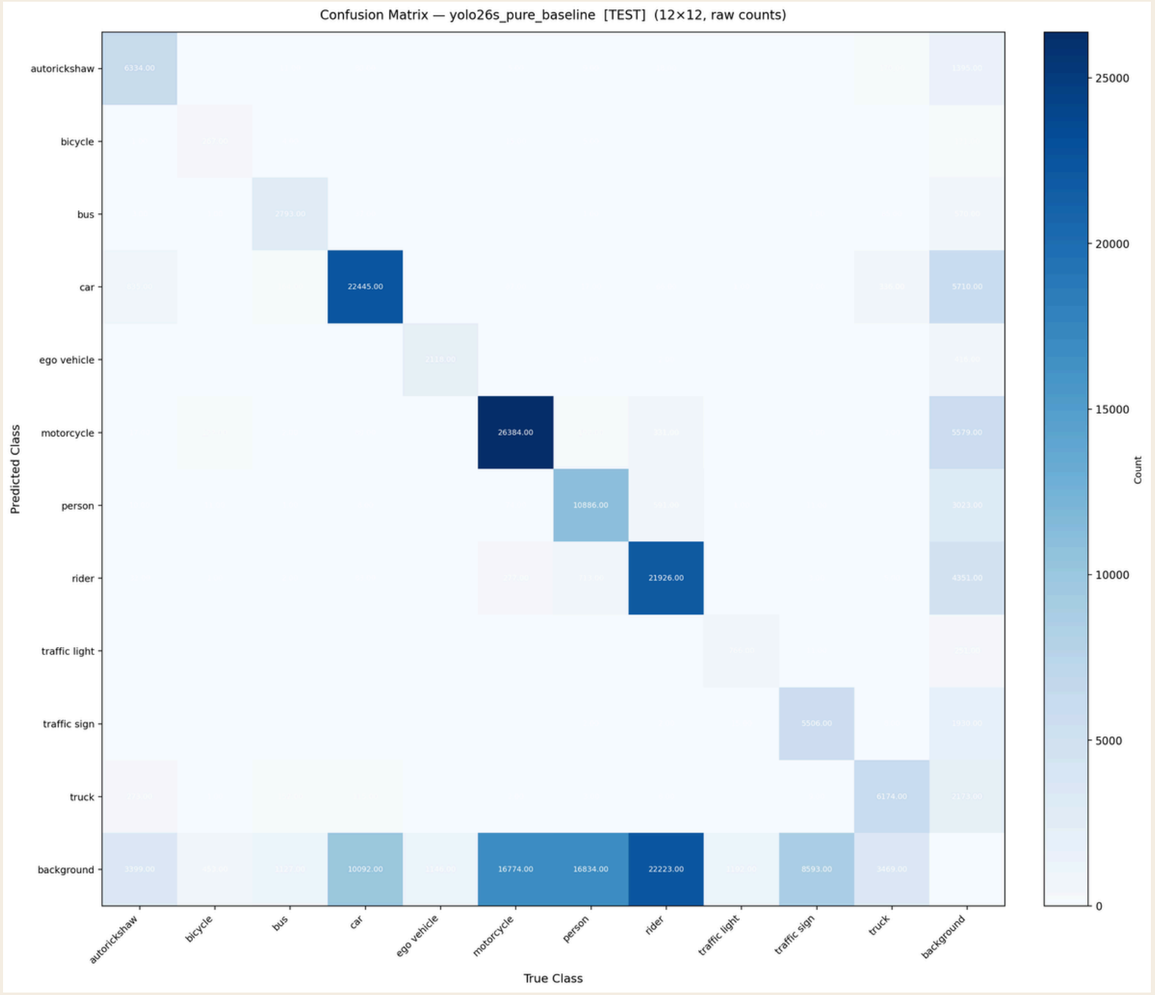
* Early stopping enabled

RESULTS & PERFORMANCE METRICS

(iv) YOLOv26s Baseline

Per class AP:

- autorickshaw = 0.6852
- bicycle = 0.3329
- bus = 0.7178
- car = 0.71
- ego vehicle = 0.7017
- motorcycle = 0.6716
- person = 0.447
- rider = 0.5632
- traffic light = 0.4321
- traffic sign = 0.4238
- truck = 0.6409



mAP@0.5	mAP@0.5:0.95	Precision	Recall	AP_S	AP_M	AP_L	Epochs	Train Time
0.5751	0.3564	0.7437	0.5256	0.0855	0.2712	0.6721	50*	~11h

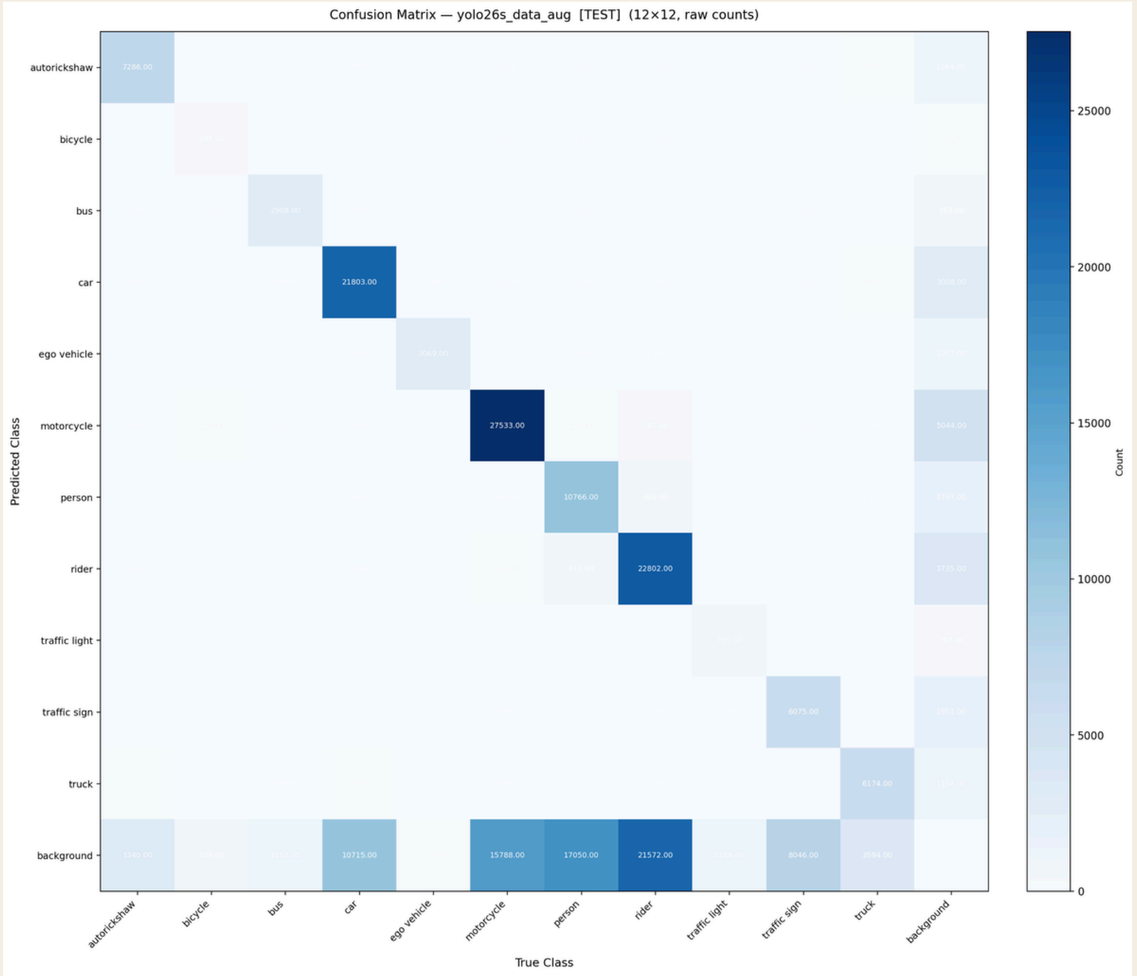
* Early stopping enabled

RESULTS & PERFORMANCE METRICS

(v) YOLOv26s Data Augmentation

Per class AP:

- autorickshaw = 0.75
- bicycle = 0.3974
- bus = 0.7704
- car = 0.7435
- ego vehicle = 0.8947
- motorcycle = 0.7047
- person = 0.4824
- rider = 0.5983
- traffic light = 0.4544
- traffic sign = 0.4764
- truck = 0.6942



mAP@0.5	mAP@0.5:0.95	Precision	Recall	AP_S	AP_M	AP_L	Epochs	Train Time
0.6333	0.4171	0.8118	0.5579	0.0894	0.2795	0.6831	50*	~5h

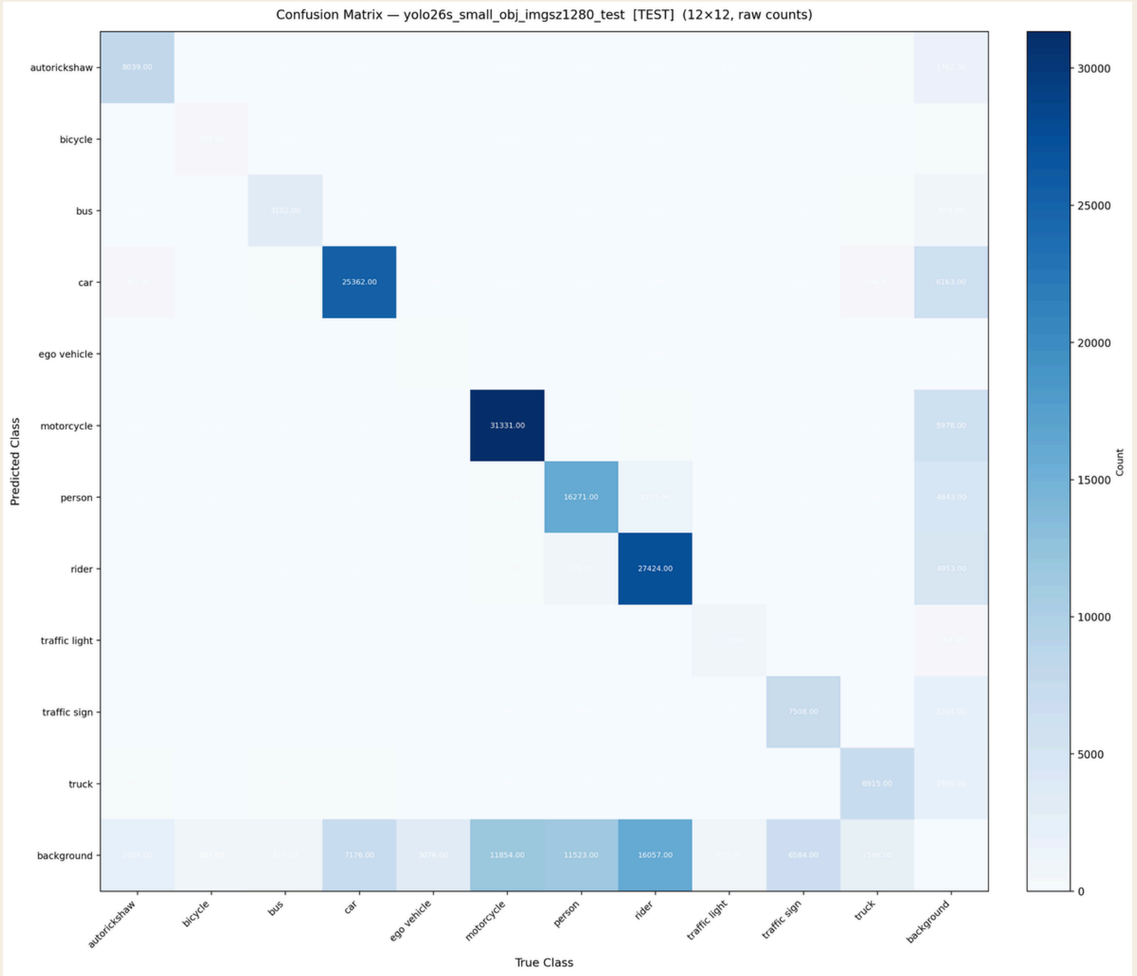
* Early stopping enabled

RESULTS & PERFORMANCE METRICS

(vi) YOLOv26s Small Object Handling

Per class AP:

- autorickshaw = 0.8006
- bicycle = 0.4535
- bus = 0.7891
- car = 0.8063
- ego vehicle = 0.4995
- motorcycle = 0.7819
- person = 0.6168
- rider = 0.6948
- traffic light = 0.559
- traffic sign = 0.5904
- truck = 0.7359



mAP@0.5	mAP@0.5:0.95	Precision	Recall	AP_S	AP_M	AP_L	Epochs	Train Time
0.6662	0.4283	0.7572	0.5828	0.138	0.3025	0.6439	50*	~12h

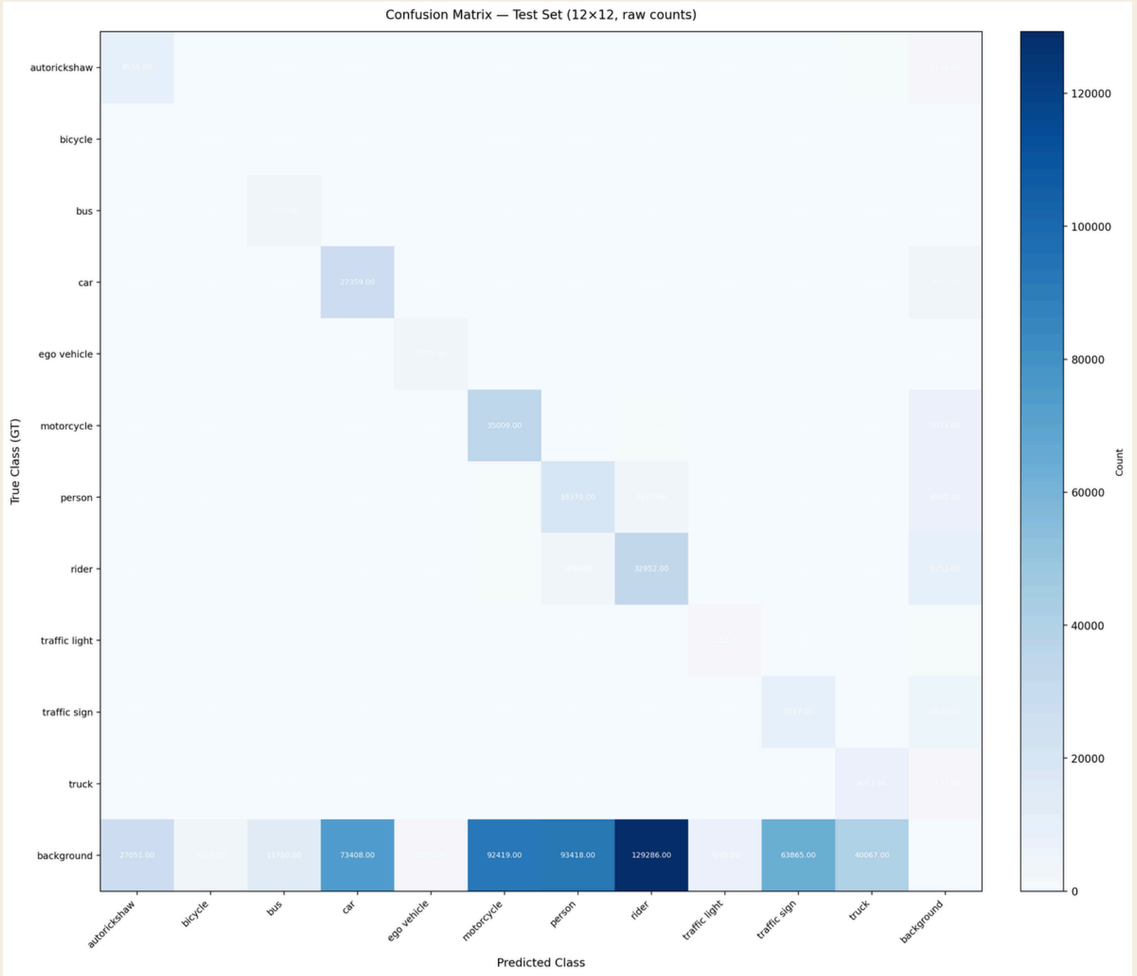
* Early stopping enabled

RESULTS & PERFORMANCE METRICS

(vii) FRCNN Baseline

Per class AP:

- autorickshaw = 0.7591
- bicycle = 0.4312
- bus = 0.7642
- car = 0.7841
- ego vehicle = 0.9134
- motorcycle = 0.7431
- person = 0.5560
- rider = 0.6591
- traffic light = 0.4538
- traffic sign = 0.5063
- truck = 0.6970



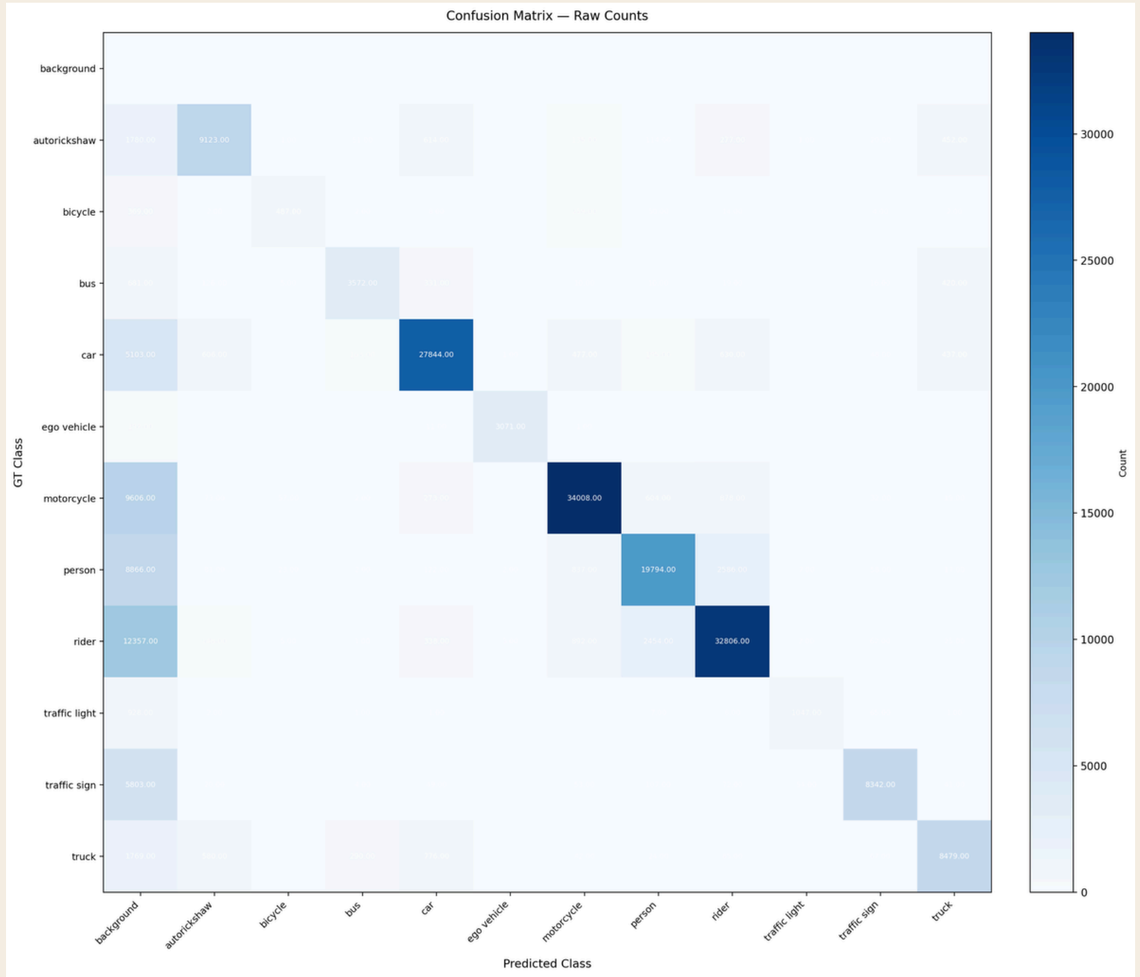
mAP@0.5	mAP@0.5:0.95	AP_S	AP_M	AP_L	Epochs	Train Time
0.6607	0.4091	0.1104	0.2711	0.5627	20	12h 15m

RESULTS & PERFORMANCE METRICS

(viii) FRCNN Data Augmentation

Per class AP:

- autorickshaw = 0.7445
- bicycle = 0.4383
- bus = 0.7600
- car = 0.7729
- ego vehicle = 0.9281
- motorcycle = 0.7214
- person = 0.5415
- rider = 0.6373
- traffic light = 0.3839
- traffic sign = 0.4632
- truck = 0.6797



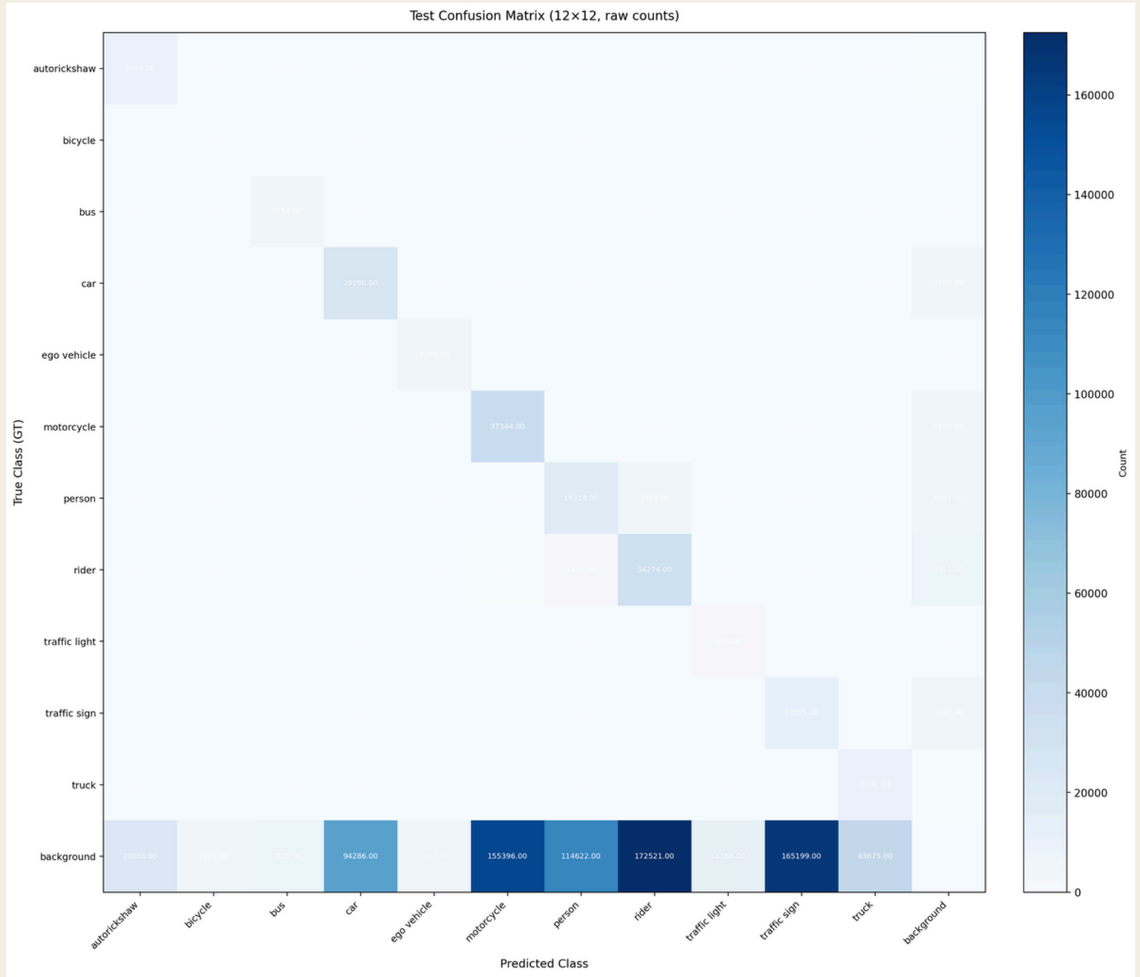
mAP@0.5	mAP@0.5:0.95	AP_S	AP_M	AP_L	Epochs	Train Time
0.6428	0.2487	0.103	0.2546	0.5953	40	21h 41m

RESULTS & PERFORMANCE METRICS

(ix) FRCNN Small Object Handling

Per class AP:

- autorickshaw = 0.7700
- bicycle = 0.4257
- bus = 0.7552
- car = 0.7906
- ego vehicle = 0.9118
- motorcycle = 0.7640
- person = 0.5589
- rider = 0.6612
- traffic light = 0.4589
- traffic sign = 0.5354
- truck = 0.6915



mAP@0.5	mAP@0.5:0.95	AP_S	AP_M	AP_L	Epochs	Train Time
0.6657	0.394	0.1272	0.2675	0.4728	20	14h 38m

DO THE RESULTS HOLD UP?

- **Augmentation consistently improves detection -** $mAP_{0.5}$ rose from 0.541→0.623 (YOLOv11s) and 0.575→0.633 (YOLO26s) over their baselines, proving that thoughtful data preparation alone drives meaningful accuracy gains
- **Higher resolution unlocks small object detection -** AP_s improved by 57–62% across both YOLO models simply by increasing input resolution from 640→1280px, with no changes to the model architecture whatsoever
- **Our AP_S is competitive with COCO standards -** The benchmark AP_s range on structured Western datasets is 10–15%; our best score of 0.138 matches this despite IDD being a far more cluttered and challenging environment
- **FRCNN augmentation hurt because of a scale conflict -** Faster R-CNN internally resizes images through its FPN; applying an additional RandomScale beforehand caused images to be resized twice, misaligning the feature pyramid with actual object sizes and degrading detection
- **YOLO26s with small object training is our strongest model -** Achieving 0.666 $mAP_{0.5}$, 0.428 $mAP_{0.5:0.95}$, and the best AP_s and AP_M across all 9 models, showing that a lightweight single-stage detector with the right preprocessing can match a heavier two-stage architecture

REFERENCES

[1] IDD: A Dataset for Exploring Problems of Autonomous Navigation in Unconstrained Environments

[2] Evaluation of Detection and Segmentation Tasks on Driving Datasets

[3] ORDER: Open-World Object Detection on Road Scenes

[4] DriveIndia: An Object Detection Dataset for Diverse Indian Traffic Scenes

[5] IDD Detection 40k

[6] ULTRALYTICS GITHUB REPOSITORY

May 2026

AI3011

THANK YOU

MLPR End Semester Presentation