Vehicle Object Detection in the Context of Bangladesh

MD. ASIF HAIDER* and MASHIYAT MAHJABIN PRAPTY*, Bangladesh University of Engineering and Technology, Bangladesh

Fig. 1. Vehicle Detection instances

In this report, we delve into the exploration of cutting-edge methods for vehicle detection under the diverse and challenging road conditions prevalent in Bangladesh. We outline some efficient approaches of detecting vehicles. Specifically, we investigate the performance of different YOLO configurations, from YoloV6L6 to the advanced YoloV8, alongside novel transformer-based models like rtDETR and CoDETR.

Additional Key Words and Phrases: Vehicle Detection, Detection Transformer, YOLO

ACM Reference Format:

Md. Asif Haider and Mashiyat Mahjabin Prapty. 2018. Vehicle Object Detection in the Context of Bangladesh. In *Woodstock '18: ACM Symposium on Neural Gaze Detection, June 03–05, 2018, Woodstock, NY*. ACM, New York, NY, USA, 37 pages. https://doi.org/XXXXXXX.XXXXXXX

*Both authors contributed equally to this research.

Manuscript submitted to ACM

1 INTRODUCTION

Autonomous cars are like the future on wheels, aiming to drive themselves without a human. Despite the big leaps forward, there's still a lot to figure out, especially when it comes to teaching these cars to drive in all kinds of places and situations. One big issue is finding the right kind of data to teach these cars about the rare and tricky situations they might face on the road. Most of the time, the information we use to teach these cars doesn't come from every corner of the globe. This is a problem, especially in places like the Indian subcontinent, where the roads and traffic can be very different from what we might find in other parts of the world.

In this report, we focus on Dhaka, Bangladesh, to make these smart cars better at spotting and understanding vehicles on the road in real time. We outline the performance of modern, state-of-the-art deep learning algorithms and models on a carefully curated set of data called the BadODD Dataset, which is a fancy name for a huge collection of traffic objects from all over Bangladesh. The dataset is designed to help fill in the gaps that other datasets miss.

We focus on training the models using popular open-source machine learning tools and frameworks, along with exploring various types of architectures to draw a clear comparison across different approaches. We train, evaluate and finally test/infer the model on BadODD dataset. Finally, we also infer on another relevant, yet slightly different dataset titled DhakaAI dataset with our already finetuned model weights.

2 METHODOLOGY

2.1 BadODD Dataset

- Coverage of Bangladesh Districts
  The BadODD [1] dataset covers 9 districts in Bangladesh: Sylhet, Dhaka, Rajshahi, Mymensingh, Maowa, Chittagong, Sirajganj, Sherpur, and Khulna. It includes various road scenarios such as urban and rural areas, highways, and expressways, providing a diverse representation of Bangladesh's road infrastructure. Data collection was done using smartphone cameras to ensure authenticity and to capture real-world driving conditions.
- Image and Object Statistics
  The dataset comprises 9,825 images capturing different lighting conditions and road scenarios, including day and nighttime datasets. We used a total of 5896 images from the dataset, with 13 distinct classes representing various vehicles and objects commonly found on Bangladeshi roads. The images annotated with rectangular bounding boxes.The dataset contains a wide range of road types, including towns, expressways, highways, and village roads. This diversity in locations aims to challenge algorithms to perform well across various driving contexts commonly encountered on Bangladesh roads.
- Annotation Details
  The classification of vehicles was redefined based on their specific characteristics rather than relying solely on locally or globally recognized names. This novel approach aims to accommodate the diverse array of vehicle types encountered on the roads of Bangladesh, ensuring scalability. Analysis of the dataset's class distribution reveals a noticeable imbalance, with certain classes such as Person, Autorickshaw, and Three Wheeler being more prevalent compared to others like wheelchair, train, and construction vehicle. A comparative evaluation of three object detection models: YOLO series, rtDETR and CoDETR, was conducted to assess their performance in terms of mean Average Precision (mAP) scores on the dataset.

2.2 Experimental Setup

In our experimental setup, we utilized the Kaggle environment for training our models, leveraging the GPU P100. We also used T4 and V100 GPUs from Google Colaboratory.

2.3 Data Spliting

For model training, evaluation and testing, we employed a stratified split of the BadODD dataset. 70% of the data was used for training, 15% for validation and the rest 15% for testing. The class-wise distribution of the train set is shown in the figure below:

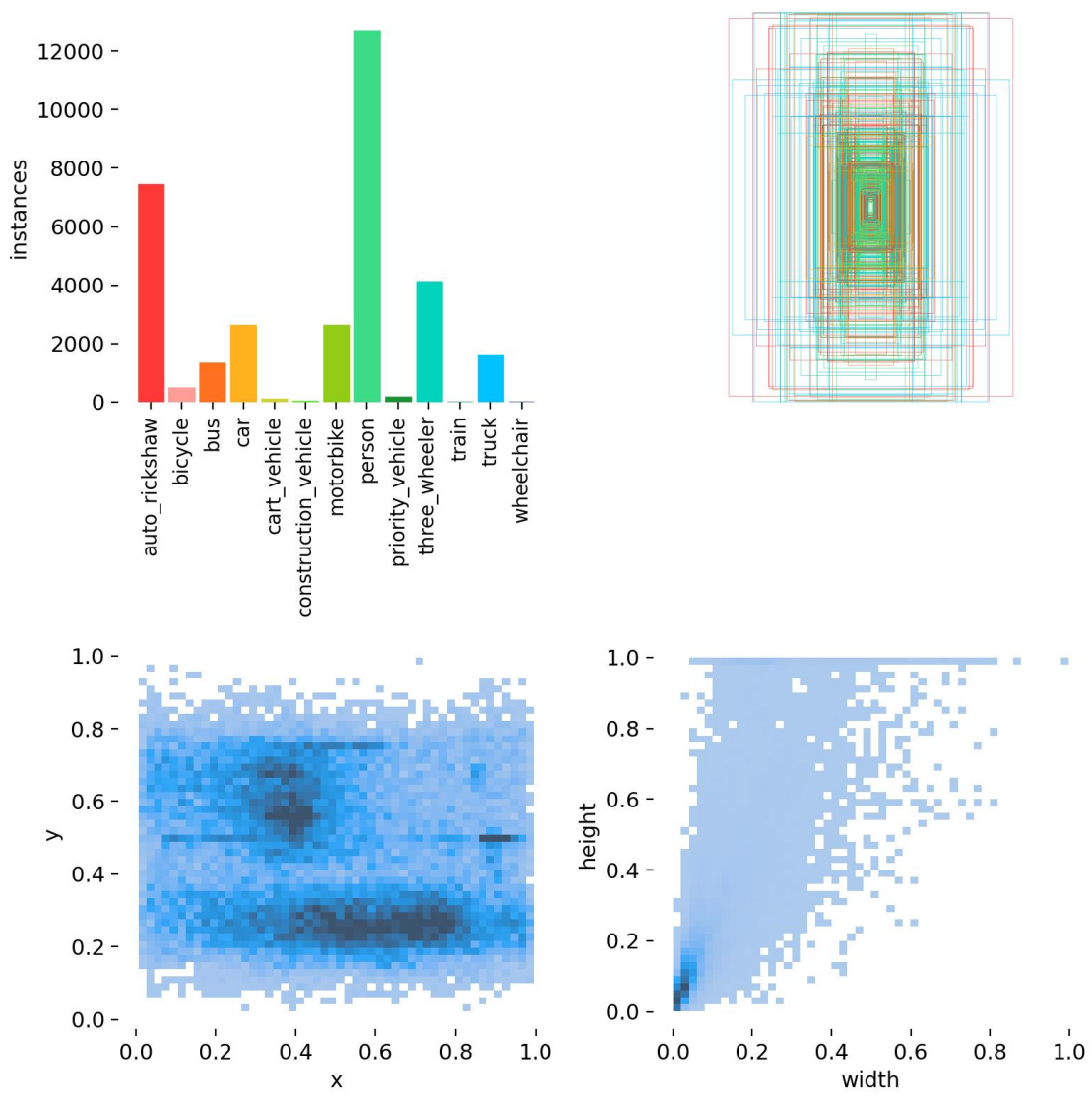

Fig. 2. Distribution of Classes in the Train set

2.4 Model Selection

We tried several variants of YOLO namely YOLOv8s, YOLOv8l, YOLOv8xl and YOLOv6l6. We also experimented with two variants of Transformer based architecture: rtDETR and CoDETR. Each of the models are discussed below.

2.4.1 YOLOv8. YoloV8 represents the latest advancement in the YOLO (You Only Look Once) series, designed for real-time object detection with enhanced accuracy and efficiency. It incorporates cutting-edge neural network architectures and optimization techniques to significantly improve detection speed and precision across a wide range of object classes.

YoloV8 further refines its predecessors' methodologies by optimizing model architecture for better generalization and lower computational overhead, making it highly suitable for deployment in resource-constrained environments.

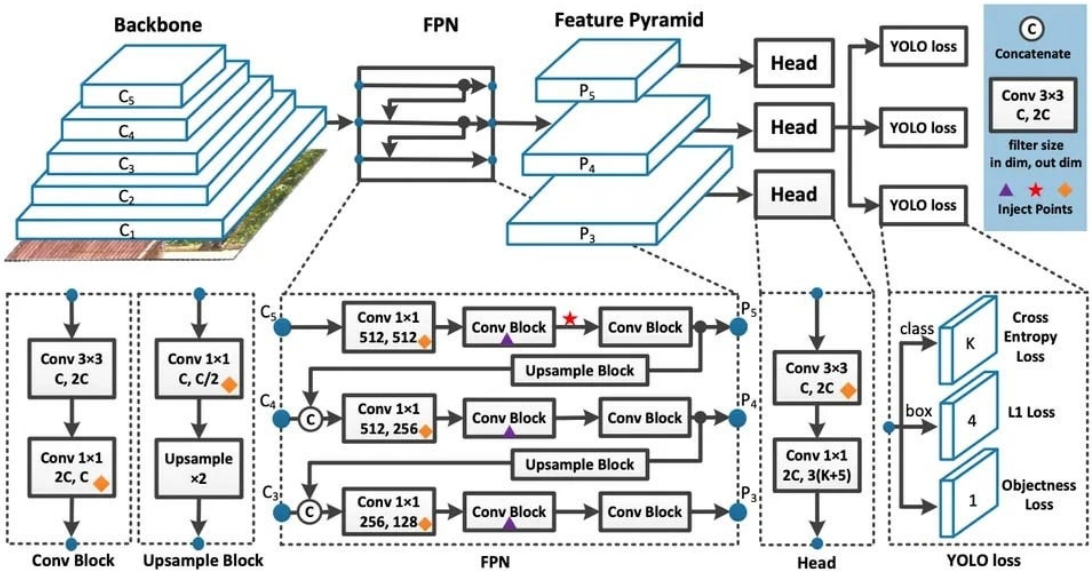

Fig. 3. YOLOv8 Architecture

YoloV8 introduces a neural network architecture that capitalizes on advancements in transformer models, convolutional neural network (CNN) optimizations, and sophisticated training techniques to achieve state-of-the-art object detection performance. At its core, YoloV8 leverages an evolved backbone that incorporates elements of Vision Transformers (ViTs) to enhance its feature extraction capabilities, allowing for more nuanced understanding of complex images. This is coupled with a redesigned detection head that employs depth-wise separable convolutions for efficiency, and attention mechanisms to focus on relevant features. The architecture also utilizes techniques such as automatic model pruning and knowledge distillation to streamline the model for faster inference speeds while maintaining high accuracy.

2.4.2 YOLOv6. YoloV6 [2] is a another variant of the popular YOLO family, known for its balance between speed and accuracy in object detection tasks. This variant employs a deeper and wider network architecture, tailored to leverage large-scale datasets for improved detection performance in complex scenes.

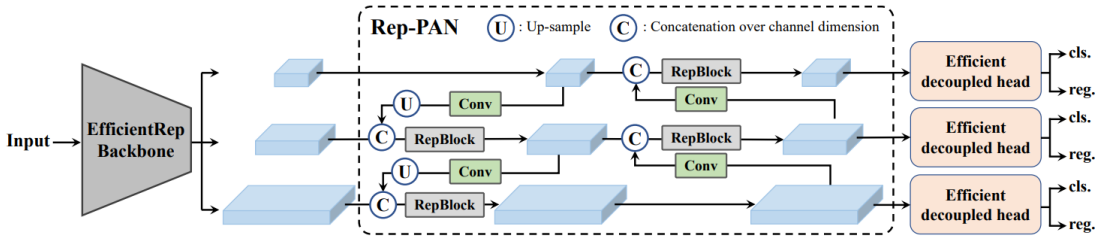

Fig. 4. YOLOv6 Architecture

YoloV6L6, as a specialized variant of the YoloV6 family, focuses on a hybrid architecture that combines the strengths of CNNs with attention-based mechanisms to optimize for both accuracy and inference speed. The "L6" configuration refers to a deeper and wider network structure, integrating six layers of convolutional blocks that are specifically designed to process high-resolution inputs more effectively. This model employs a sophisticated backbone, such as EfficientRep, augmented with additional convolutional layers for richer feature extraction at various scales. Spatial pyramid pooling and cross-stage partial connections (CSP) are key components, enhancing the network's ability to handle multi-scale objects and reduce computational costs. Attention modules, such as the Squeeze-and-Excitation (SE) blocks, are incorporated to refine feature channels, emphasizing informative features and suppressing less useful ones. This architectural refinement ensures that YoloV6L6 achieves a delicate balance between the depth and width of the network.

2.4.3 rtDETR. rtDETR [3] (real-time Detection Transformer) represents a significant leap in the application of transformer models to real-time object detection tasks. It ingeniously adapts the transformer architecture, known for its efficacy in natural language processing, to the visual domain, focusing on efficiency and speed to meet real-time requirements.

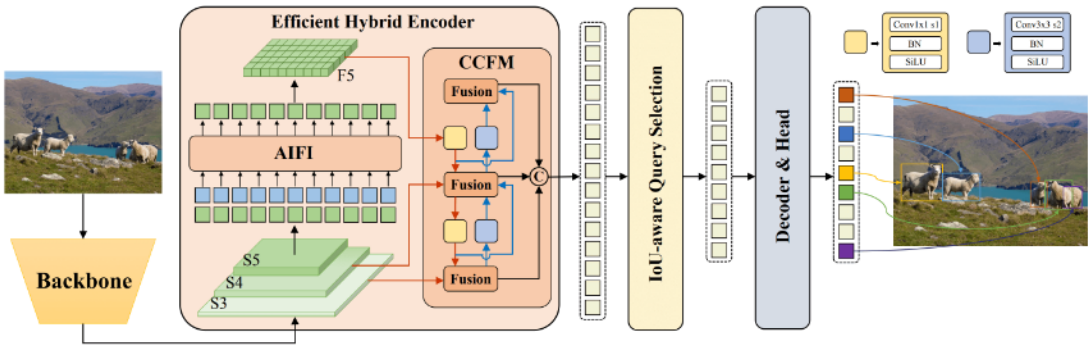

Fig. 5. rtDETR Architecture

rtDETR introduces a lightweight transformer encoder-decoder structure, optimized to reduce computational complexity while maintaining high detection performance. Key to its design is the use of a compact set of learnable object queries and a reduced-resolution feature map to minimize the amount of data processed by the transformer layers. This approach, combined with a carefully designed positional encoding scheme, allows rtDETR to achieve fast inference times. Additionally, rtDETR employs a novel training strategy that integrates knowledge distillation directly within the transformer architecture, leveraging pre-trained models to enhance its learning efficiency and accuracy without significantly increasing computational demands.

2.4.4 CoDETR. CoDETR [5], standing for DEtection TRansformer with Collaborative Hybrid Assignments, introduces a novel approach in the realm of object detection by integrating a collaborative hybrid assignments training scheme to enhance the efficiency and effectiveness of DETR (DEtection TRansformer)-based models.

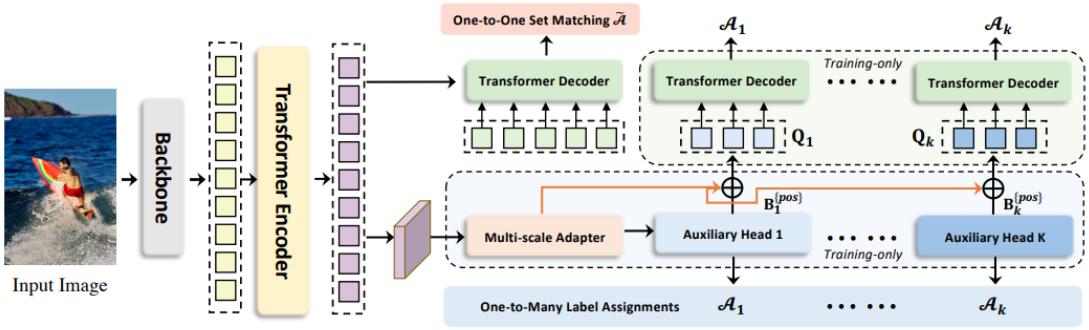

Fig. 6. CoDETR Architecture

CoDETR improves the encoder's learning capability within end-to-end detectors through the employment of multiple parallel auxiliary heads. These heads are supervised by one-to-many label assignments, diversifying the learning signals and enhancing feature extraction capabilities. The model enhances the decoder's attention learning by introducing extra customized positive queries. These queries are derived from the positive coordinates identified by the auxiliary heads, thereby refining the model's ability to focus on relevant features for accurate object detection.

2.5 Model Performance

Model	Epoch	Batch Size	mAP50 Score
YOLOv8l	100	16	0.602
YOLOv6l6	20	16	0.548
rtDETR	35	8	0.611
CoDETR with Swin-L	1	1	0.05

3 RESULTS

3.1 rtDETR

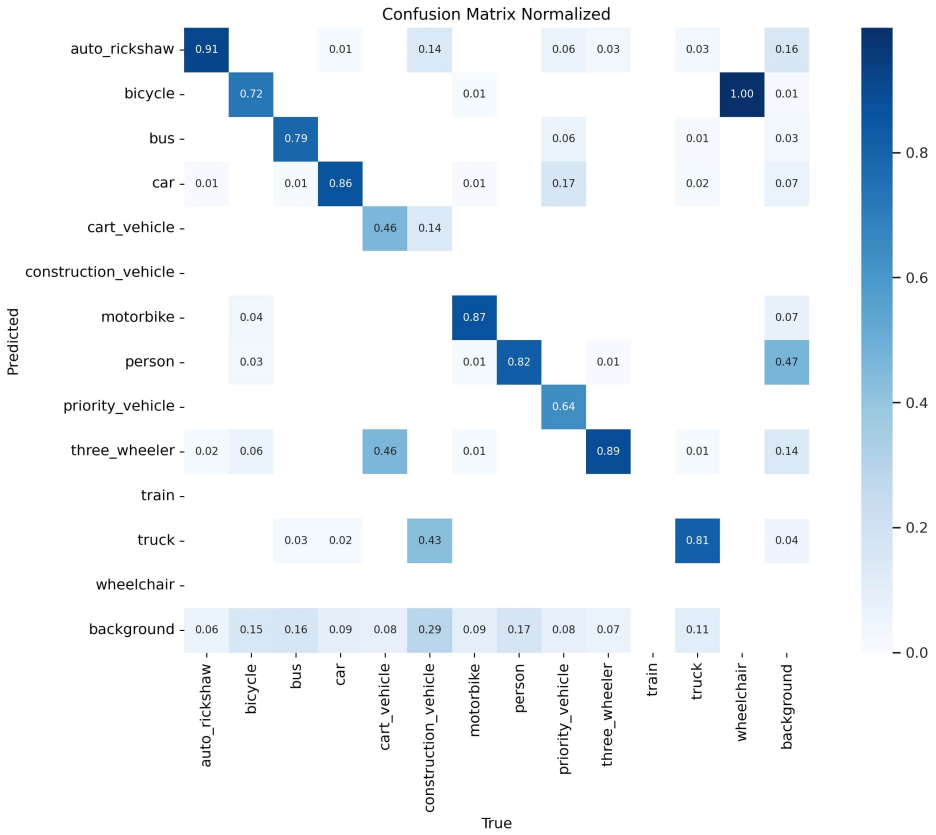

Fig. 7. Confusion Matrix for Validation Set

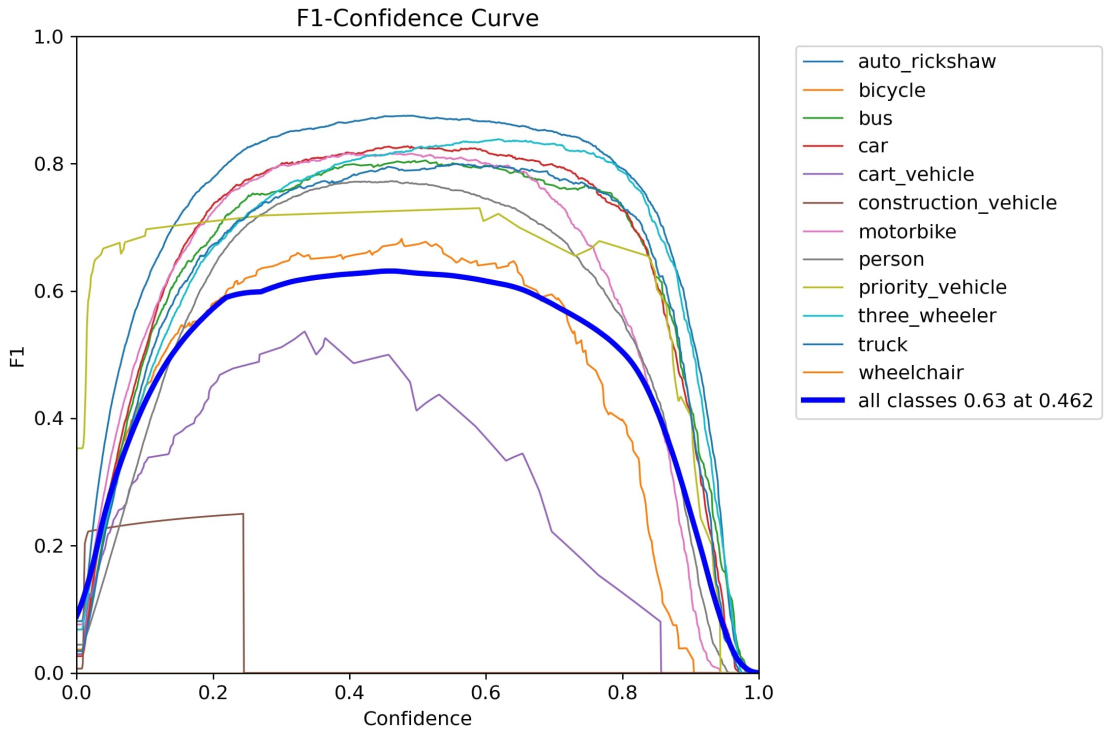

Fig. 8. F1 Curve for Validation Set

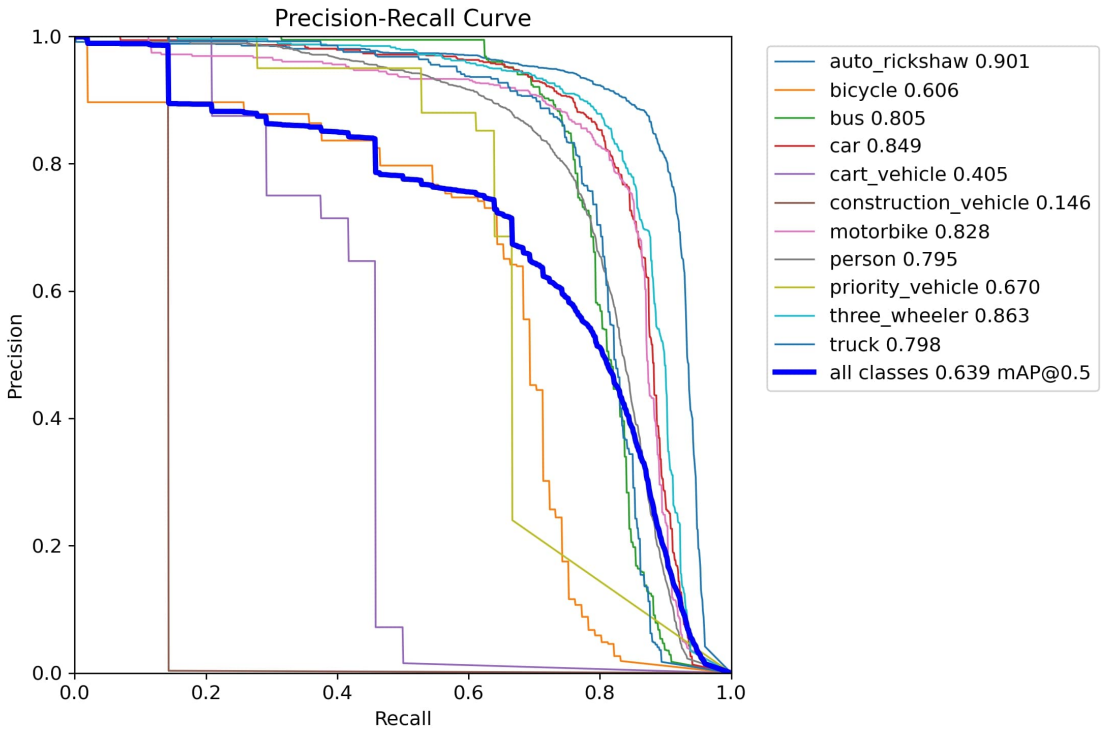

Fig. 9. PR Curve for Validation Set

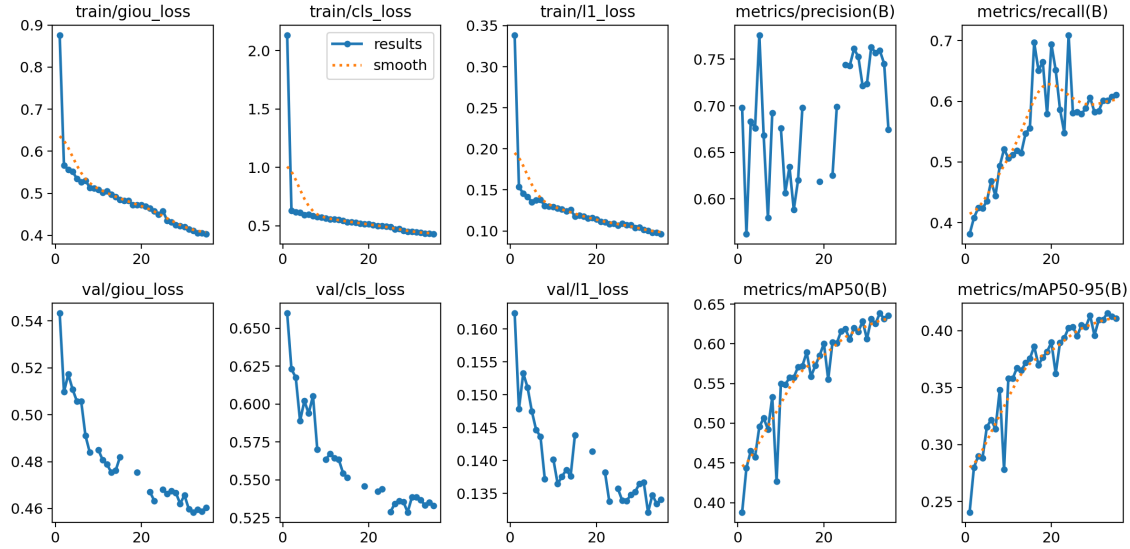

Fig. 10. Losses

3.1.1 *Validation.*

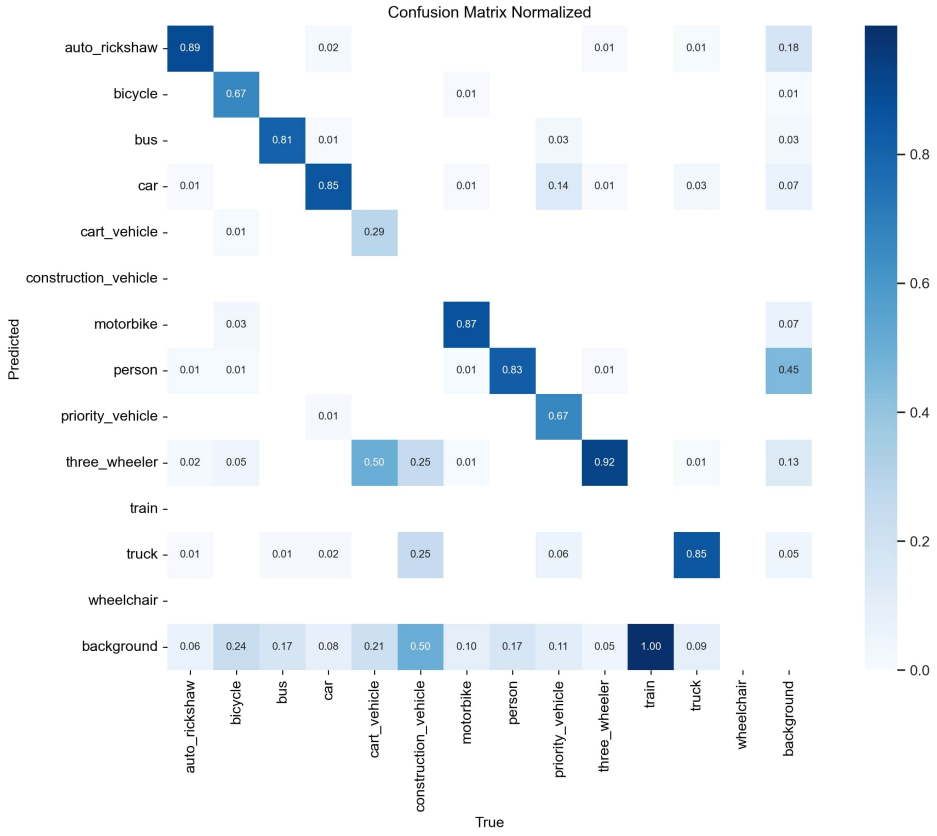

Fig. 11. Confusion Matrix for Test Set

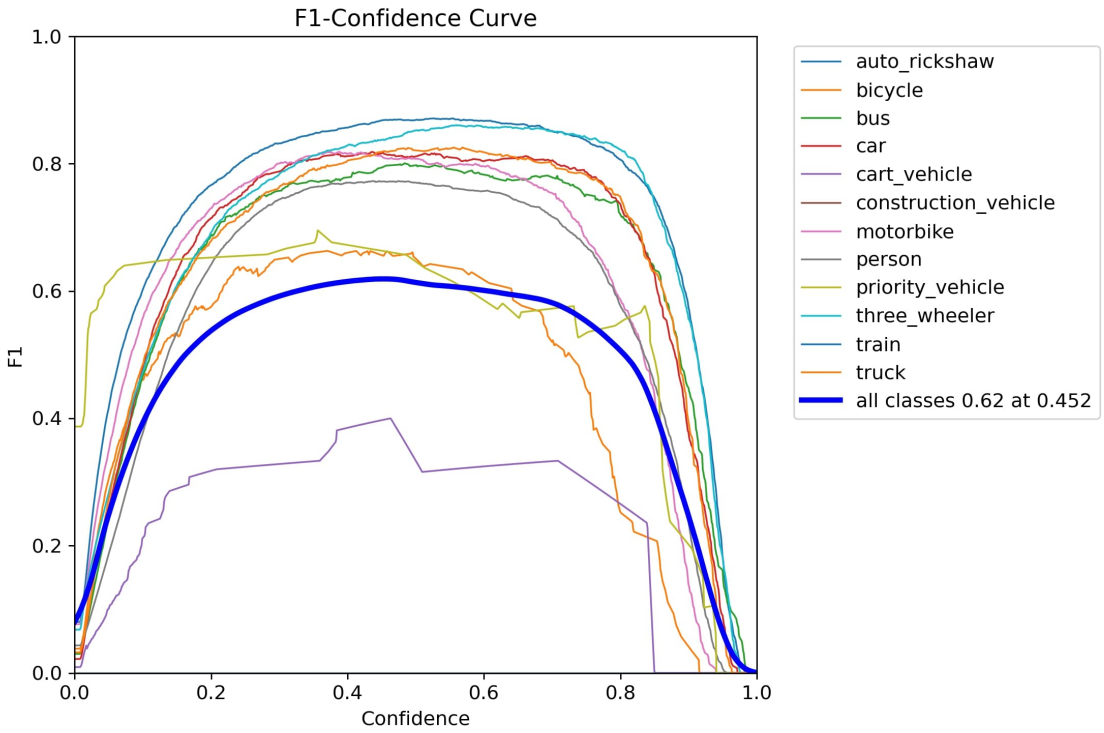

Fig. 12. F1 Curve for Test Set

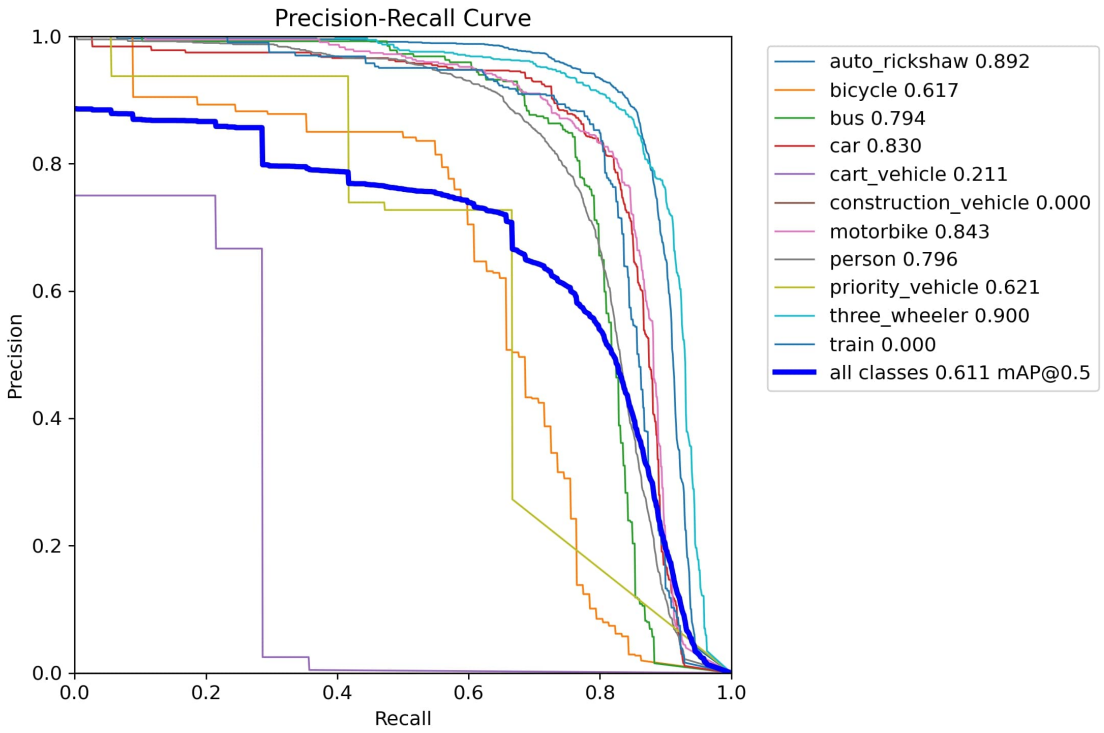

Fig. 13. PR Curve for Test Set

Fig. 14. Example Batch Prediction for Test Set

3.1.2 *Testing.*

3.2 YOLOv8l

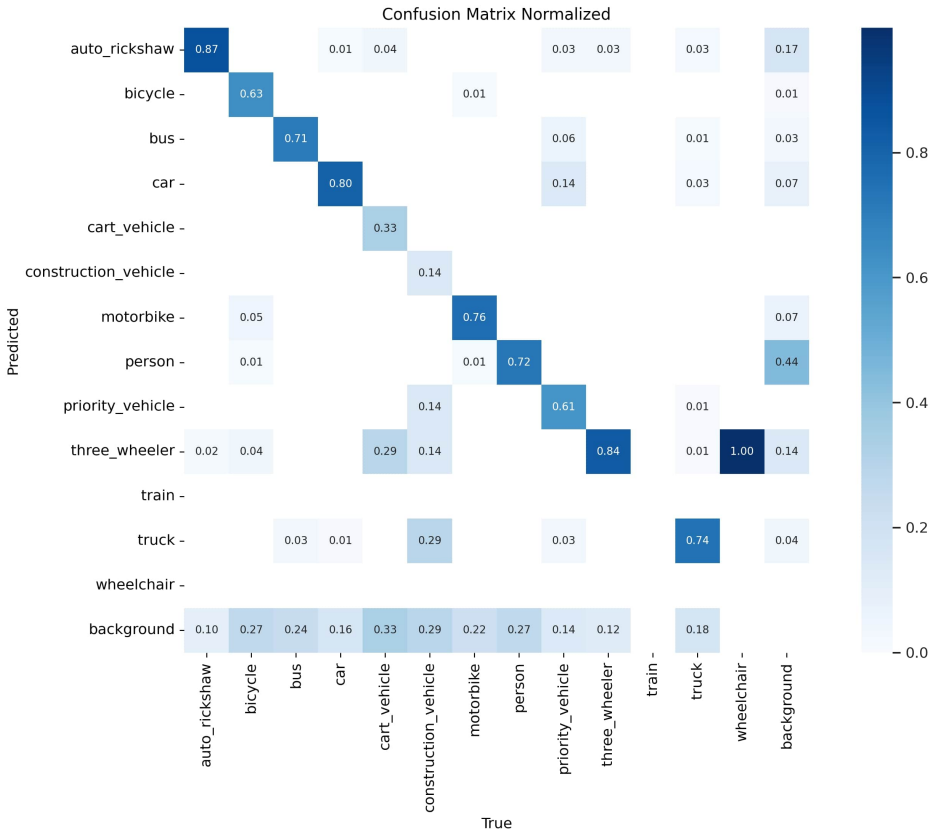

Fig. 15. Confusion Matrix for Validation Set

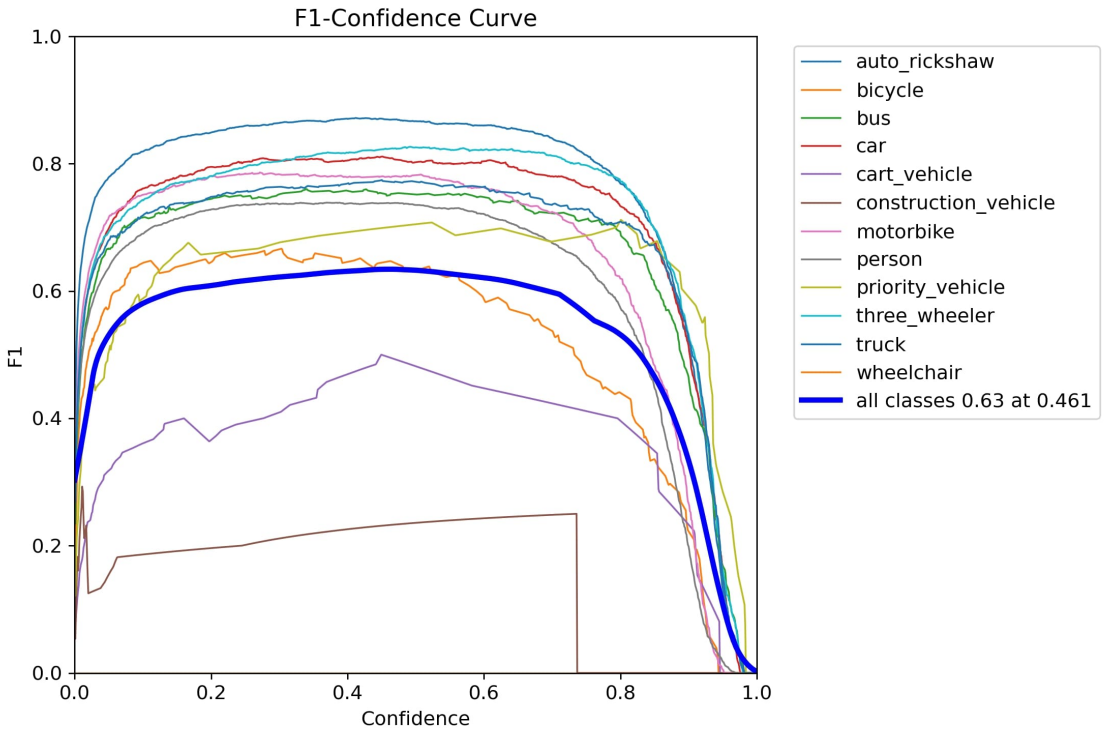

Fig. 16. F1 Curve for Validation Set

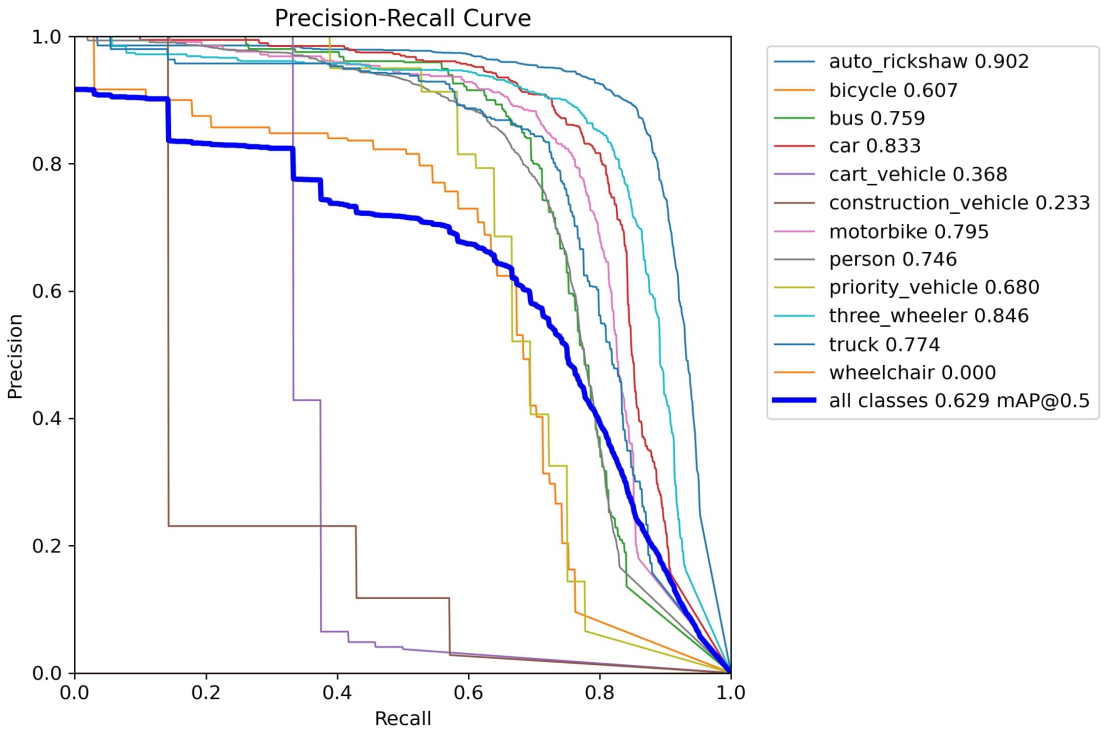

Fig. 17. PR Curve for Validation Set

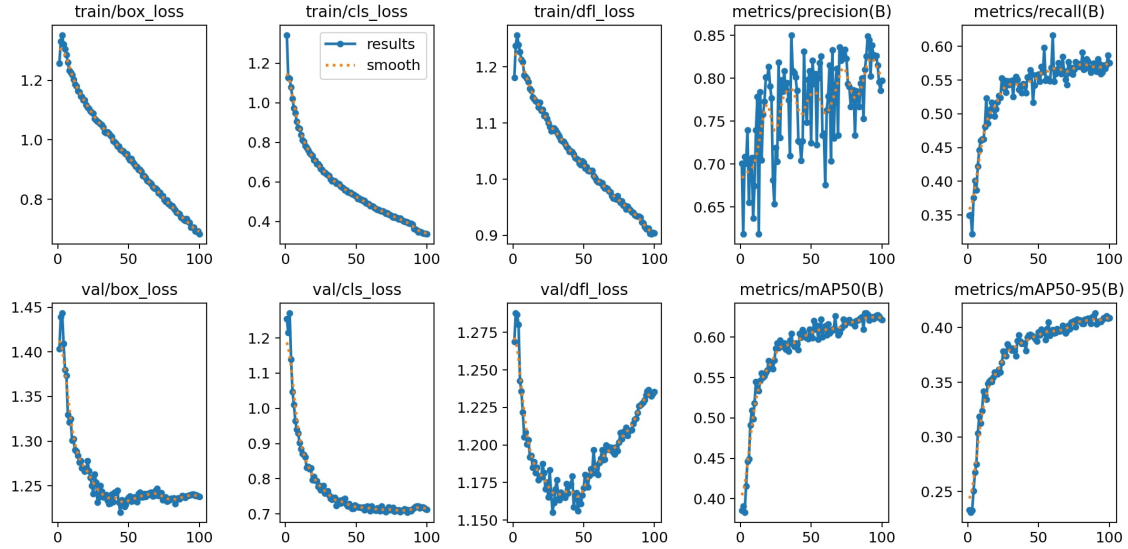

Fig. 18. Losses

3.2.1 *Validation.*

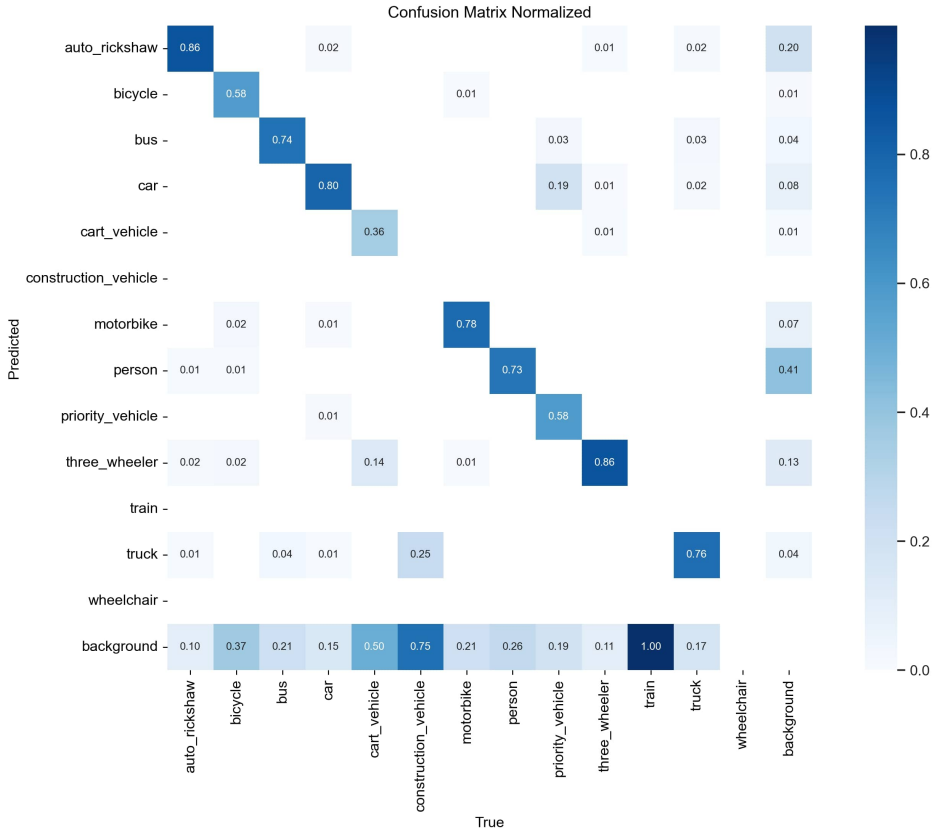

Fig. 19. Confusion Matrix for Test Set

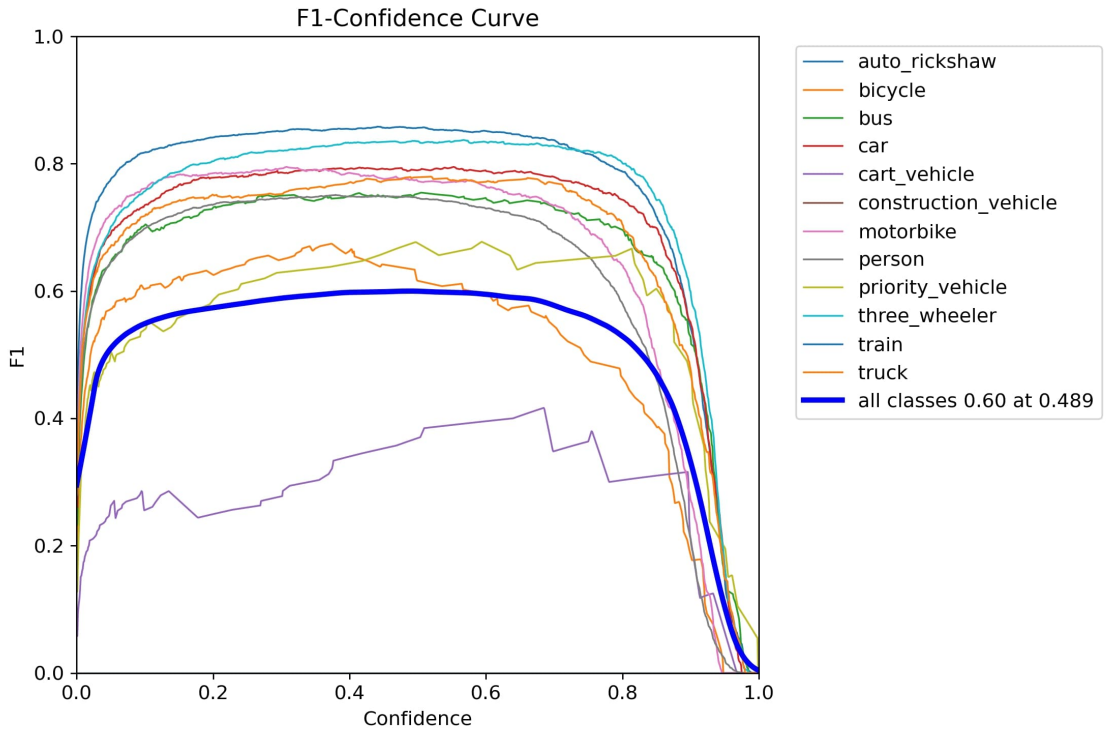

Fig. 20. F1 Curve for Test Set

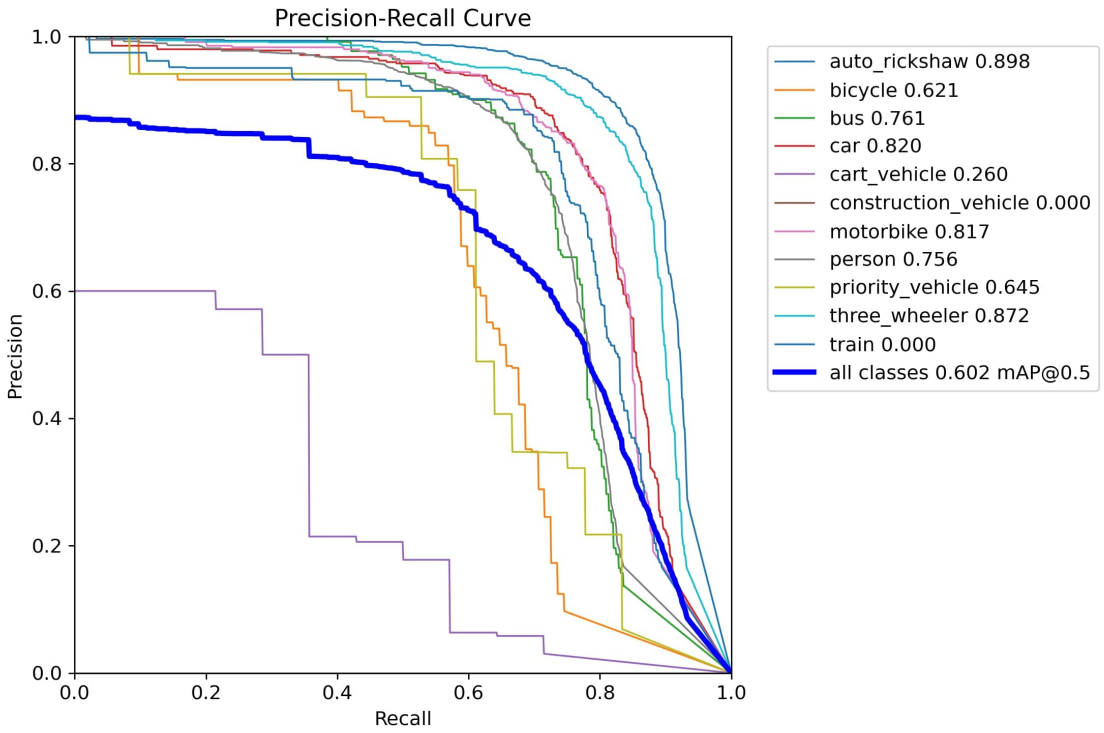

Fig. 21. PR Curve for Test Set

Fig. 22. Example Batch Prediction for Test Set

3.2.2 Testing.

3.3 YOLOv6l6

3.3.1 Training. The training process was completed in 0.823 hours with a mAP@0.5 score of 0.567 and mAP@0.5:0.95 score of 0.378.

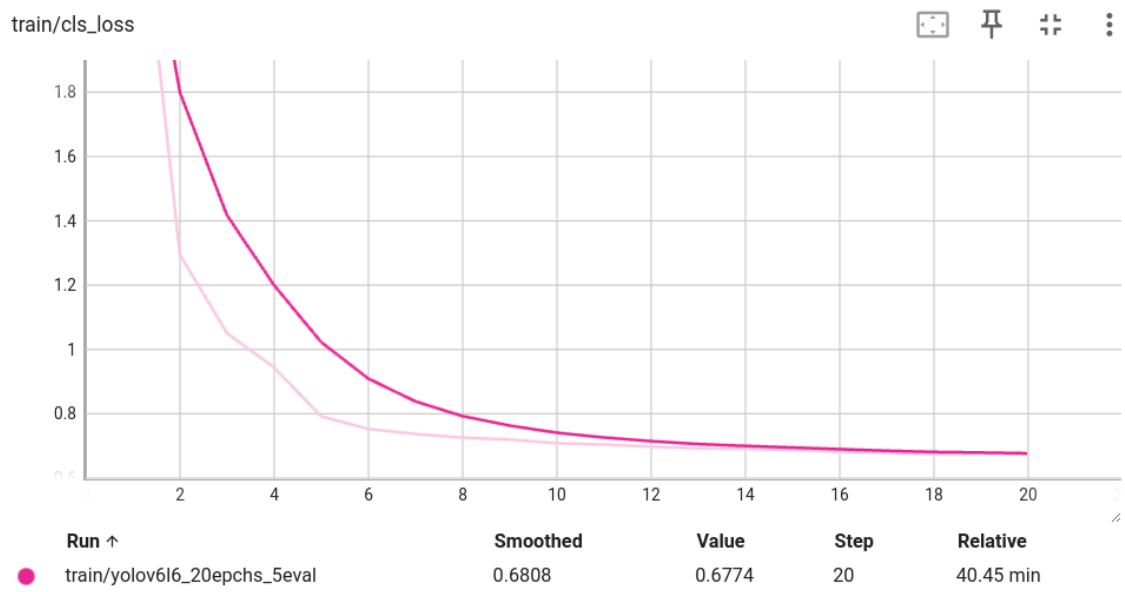

Fig. 23. Training class loss for 20 epochs

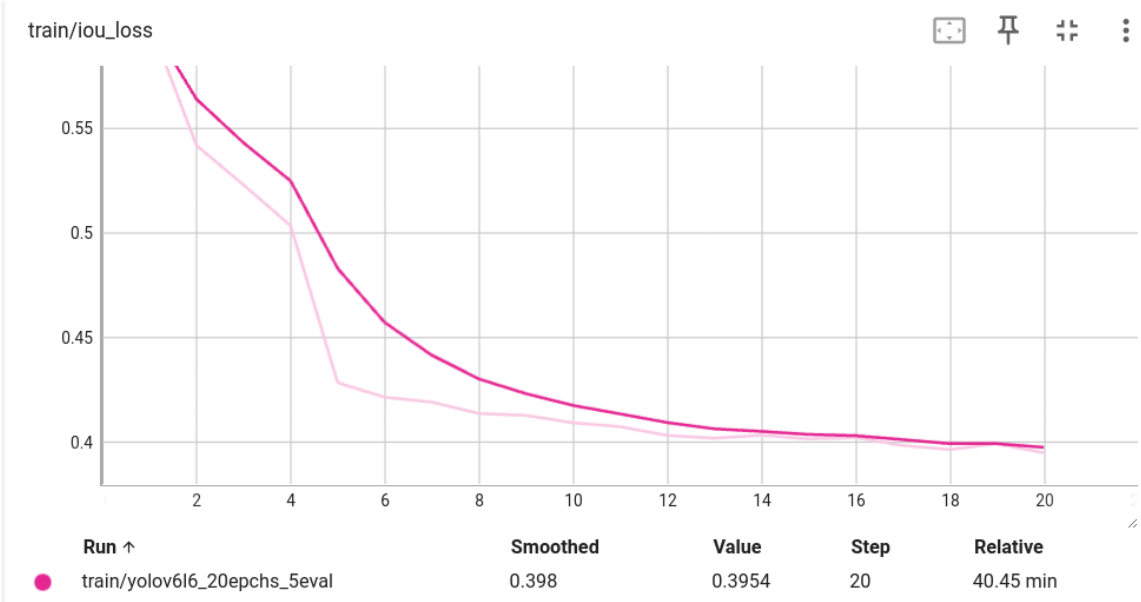

Fig. 24. Training IoU loss for 20 epochs

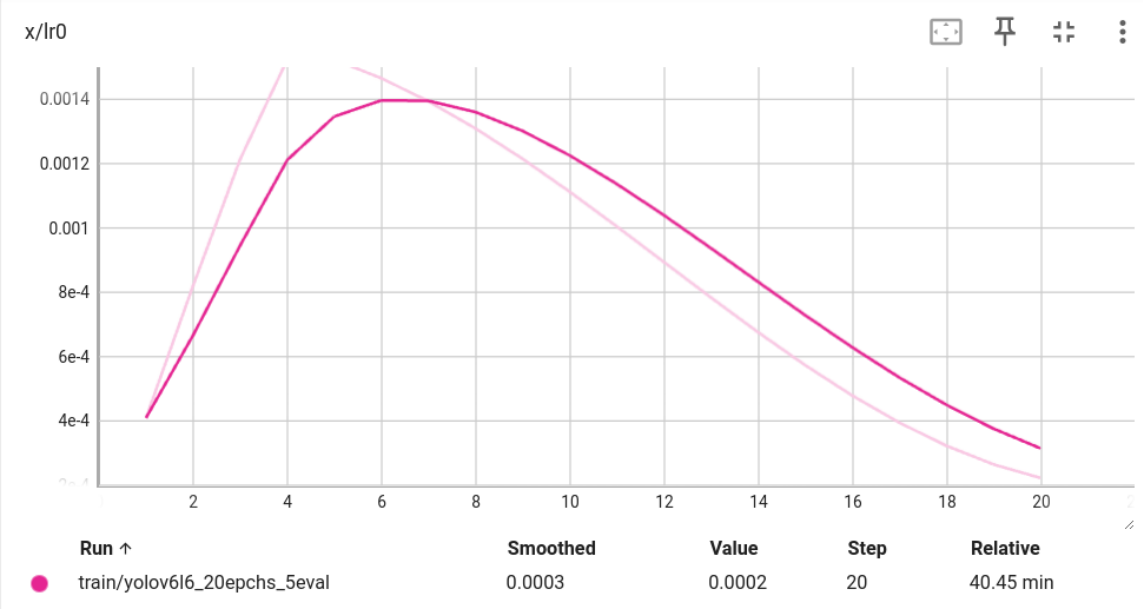

Fig. 25. Learning rate

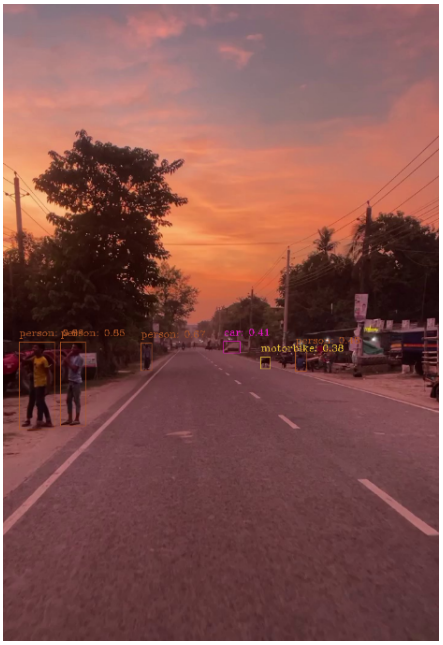

Fig. 26. Example inference

Fig. 27. Example inference

Table 1. Validation Performance

Class	Labeled_images	Labels	P@.5iou	R@.5iou	F1@.5iou	mAP@.5	mAP@.5:.95
all	**884**	**7136**	**0.619**	**0.63**	**0.624**	**0.568**	**0.378**
auto_rickshaw	633	1592	0.877	0.8	0.837	0.88	0.643
bicycle	96	101	0.758	0.49	0.595	0.514	0.272
bus	188	295	0.837	0.71	0.768	0.778	0.547
car	371	635	0.832	0.75	0.789	0.825	0.581
cart_vehicle	20	24	0.25	0.29	0.269	0.221	0.127
construction_vehicle	7	7	0.1	0.14	0.117	0.0181	0.0113
motorbike	372	577	0.814	0.69	0.747	0.727	0.392
person	716	2711	0.848	0.65	0.736	0.728	0.426
priority_vehicle	31	36	0.523	0.63	0.571	0.533	0.427
three_wheeler	396	810	0.864	0.76	0.809	0.836	0.583
train	0	0	-1	0.99	198	nan	nan
truck	251	347	0.876	0.63	0.733	0.755	0.524
wheelchair	1	1	0	0	0	0	0

3.3.2 Validation. The validation score turns out to be 0.568 for mAP@0.5 and 0.378 for mAP@0.5:0.95. The **nan** value in the **train** class indicates that there was no instance of train found in the validation set.

Table 2. Test Performance

Class	Labeled_images	Labels	P@.5iou	R@.5iou	F1@.5iou	mAP@.5	mAP@.5:.95
all	**885**	**6860**	**0.577**	**0.63**	**0.602**	**0.548**	**0.365**
auto_rickshaw	626	1595	0.869	0.81	0.838	0.878	0.644
bicycle	89	102	0.761	0.5	0.604	0.533	0.294
bus	179	273	0.854	0.64	0.732	0.753	0.554
car	330	533	0.857	0.74	0.794	0.825	0.587
cart_vehicle	13	14	0.333	0.35	0.341	0.149	0.0903
construction_vehicle	4	4	0	0	0	0	0
motorbike	375	558	0.899	0.65	0.754	0.743	0.412
person	708	2604	0.811	0.68	0.74	0.737	0.424
priority_vehicle	29	36	0.476	0.55	0.51	0.342	0.235
three_wheeler	400	792	0.861	0.83	0.845	0.871	0.629
train	1	1	0	0	0	0	0
truck	258	348	0.795	0.68	0.733	0.744	0.511
wheelchair	0	0	-1	0.99	198	nan	nan

3.3.3 Testing. In the unseen test set, the model achieves 0.548 in mAP@0.5 and 0.365 in mAP@0.5:0.95. The **nan** value in the **wheelchair** class indicates that there was no instance of wheelchair found in the test set.

3.4 CoDETR

We explored several data augmentation strategy in the training pipeline of the CoDETR model. The performed augmentation are as follow:

- Random flip (with 0.5 probability)
- Random resize
- Random crop (of 384 x 600 size)

After 1 epoch of full training, the validation and test set achieves a mAP@0.5 of 0.054. Increasing epochs is likely to imporve the model's generalized performance over the dataset.

4 INFERENCE ON DHAKAAI DATASET

4.1 DhakaAI Dataset

Dhaka-AI [4]: Dhaka Traffic Detection Challenge Dataset is a dataset for traffic object detection tasks emphasizing the distinctive traffic conditions in Dhaka, Bangladesh. The primary objective of this challenge was to evaluate the effectiveness of advanced techniques in automated traffic detection using AI and ICT solutions, especially in the context of a diverse urban environment like Dhaka.

The dataset consists of 3953 images with 24318 labeled objects belonging to 21 different classes including car, bus, motorbike, and other: three-wheelers (CNG), rickshaw, truck, pickup, minivan, suv, van, bicycle, auto-rickshaw, human hauler, wheelbarrow, ambulance, minibus, taxi, army vehicle, scooter, policecar, and garbagevan. We choose to proceed with only those images for inference where overlapping classes of objects are found compared to the BadODD dataset. Hence, the total number of final labeled images to infere on was 2804.

4.2 rtDETR

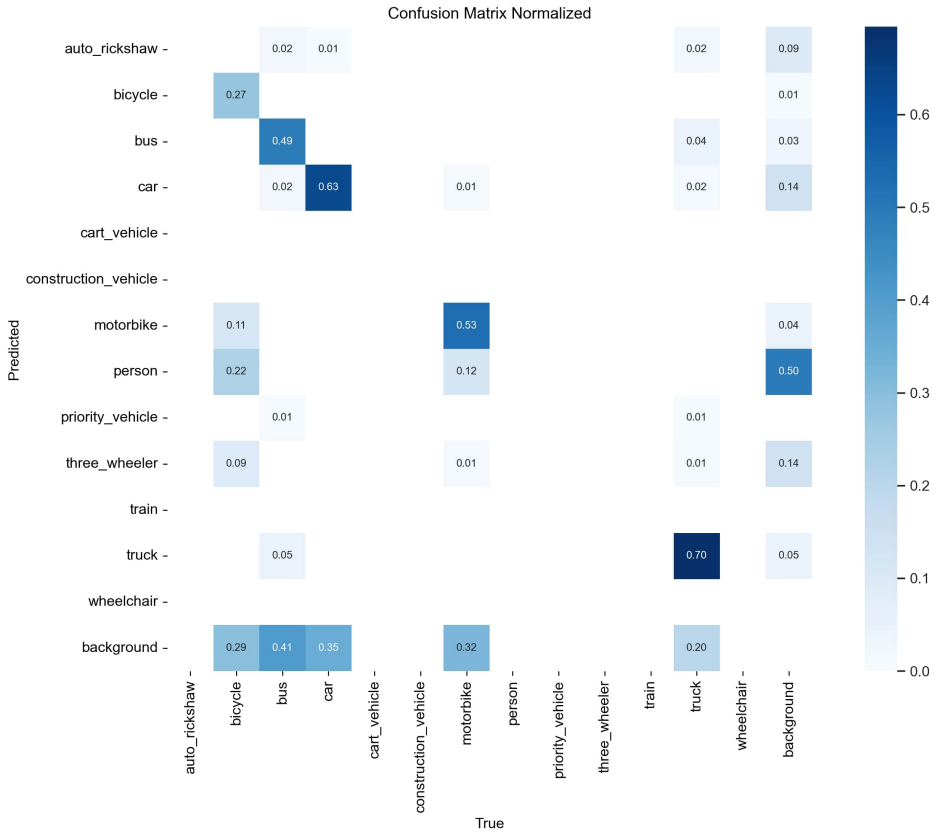

Fig. 28. Confusion Matrix

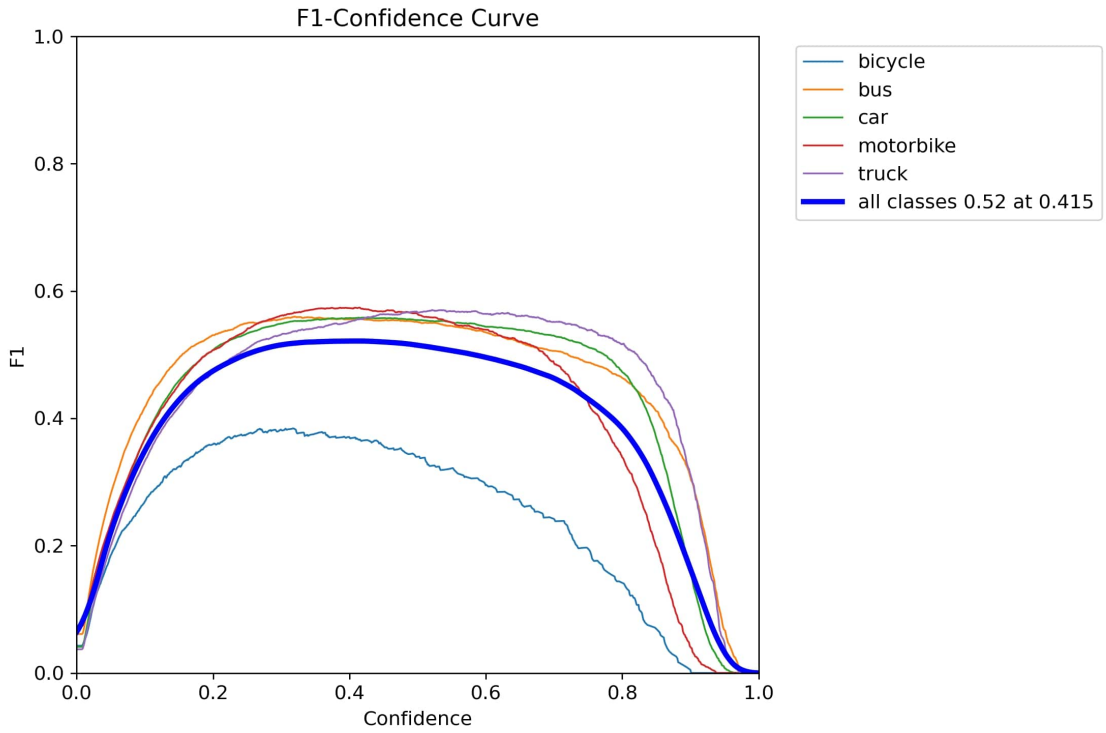

Fig. 29. F1 Curve

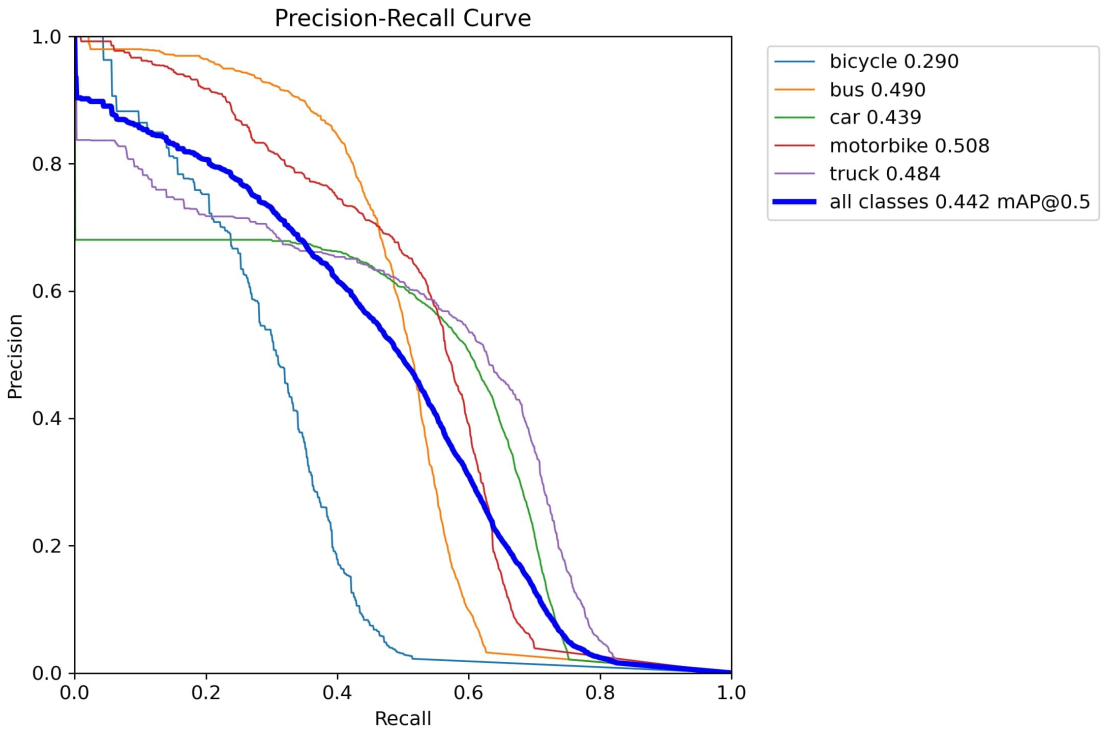

Fig. 30. PR Curve for Test Set

Fig. 31. Example Prediction

4.3 YOLOv8

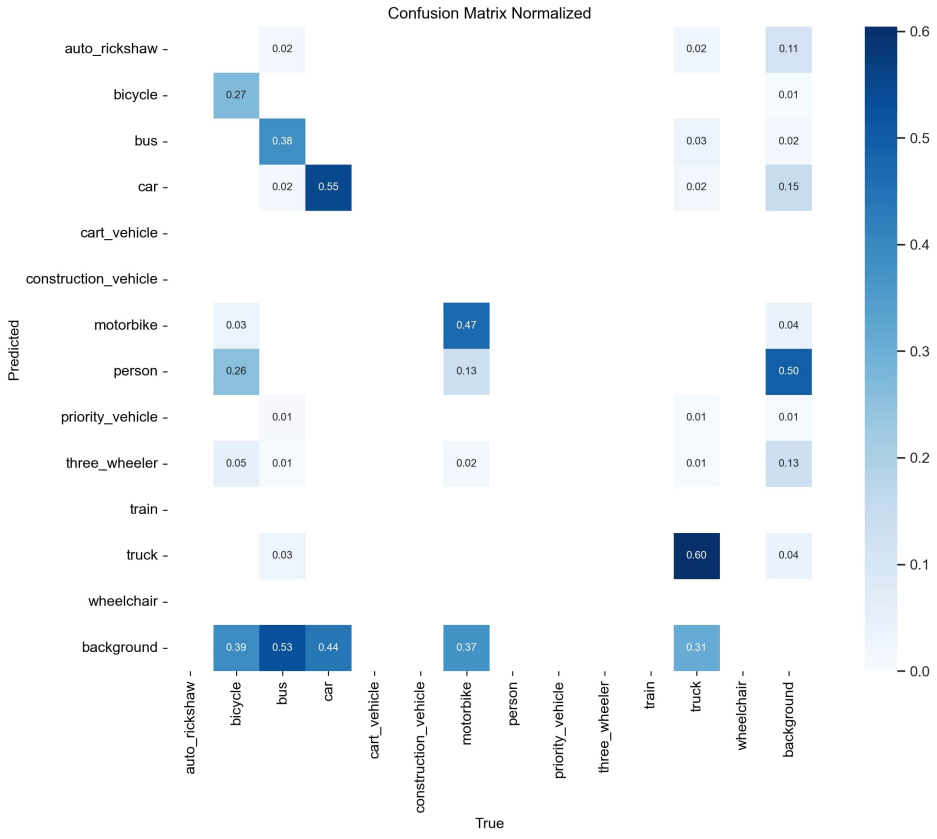

Fig. 32. Confusion Matrix

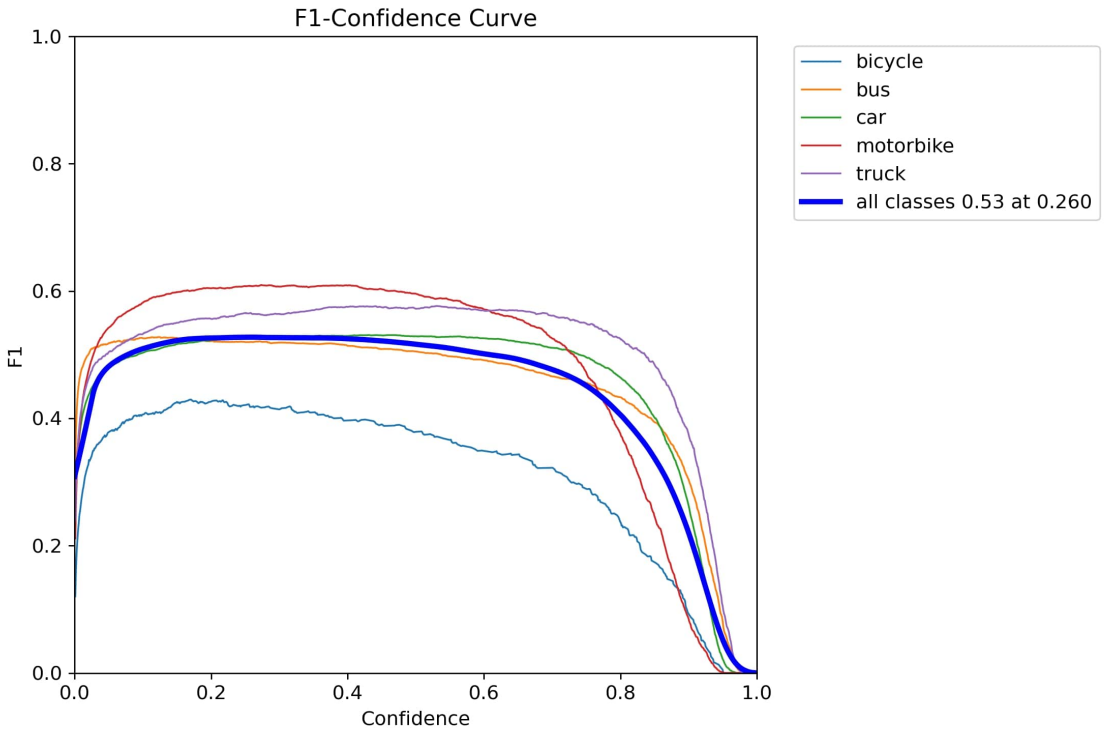

Fig. 33. F1 Curve

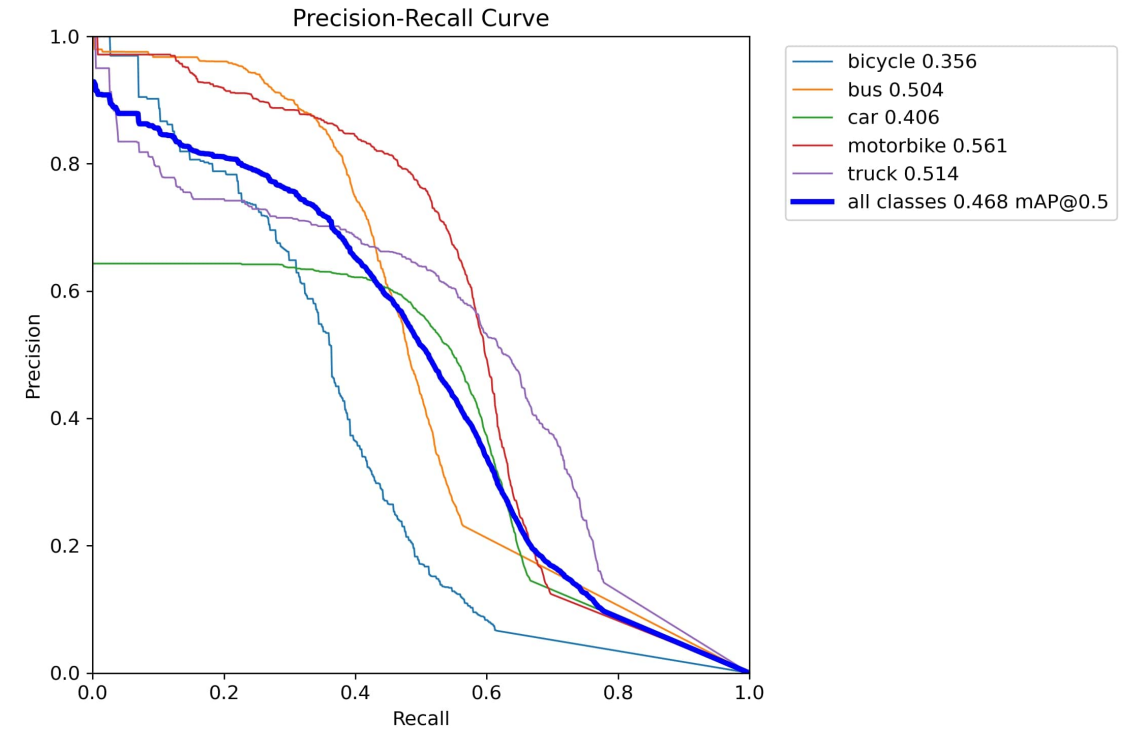

Fig. 34. PR Curve

Fig. 35. Example Prediction

4.4 YOLOv6l6

Here the model achieves 0.558 in mAP@0.5 and 0.323 in mAP@0.5:0.95. We ignored the rest of the classes discussed above because those classes were not present in the DhakaAI dataset.

Table 3. Inference on DhakaAI Dataset

Class	Labeled_images	Labels	P@.5iou	R@.5iou	F1@.5iou	mAP@.5	mAP@.5:.95
all	**2804**	**13044**	**0.595**	**0.56**	**0.577**	**0.558**	**0.323**
bicycle	352	459	0.609	0.37	0.46	0.409	0.151
bus	1556	3333	0.799	0.56	0.659	0.663	0.466
car	1621	5476	0.578	0.64	0.608	0.54	0.367
motorbike	1186	2284	0.728	0.57	0.639	0.607	0.238
truck	842	1492	0.591	0.62	0.605	0.57	0.395

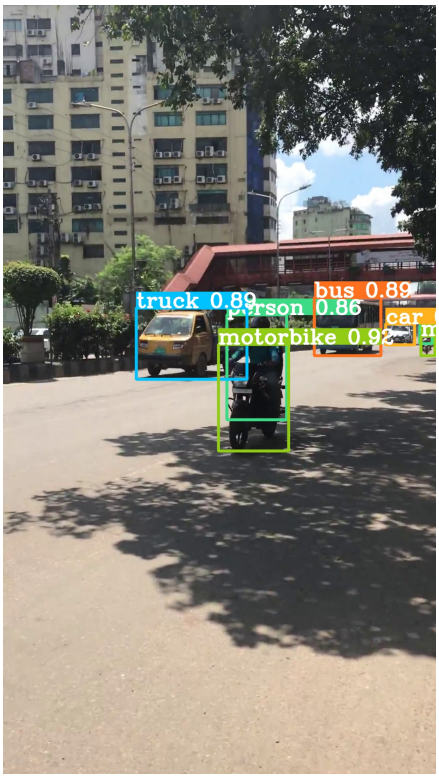

Fig. 36. Example inference

Fig. 37. Example inference

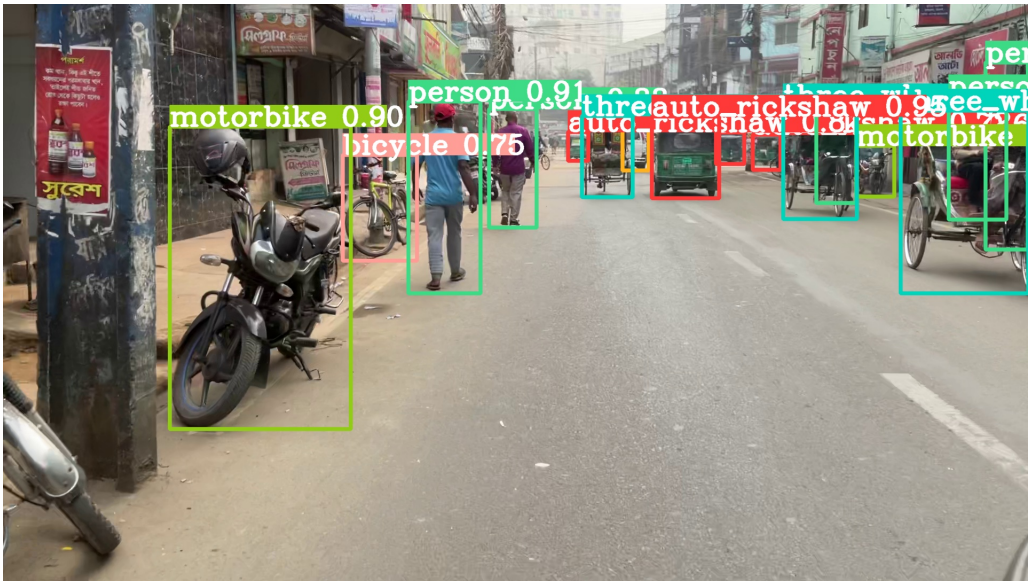

Fig. 38. Example inference

4.5 CoDETR

We followed the same data augmentation strategies and the same hyperparameter settings as trained already. The inference on this new dataset with reduced number of eligible classes achieves a similar mAP@0.5 score of 0.05.

5 CONCLUSION

Based on our analysis, we advocate for the adoption of rtDETR in object detection for Bangladesh's roads. Leveraging the BadODD dataset, our study demonstrates rtDETR's superiority over traditional methods like YOLOv8l. With a peak public score and private score, rtDETR outperforms other YOLO based models, showcasing its efficacy in real-world scenarios though currently models based on Co-DETR tend to perform better than rtDETR, as found in literature. So, there is scope for future improvement. Also, better result can be obtained exploring more robust data augmentation and image preprocessing strategies.

ACKNOWLEDGMENTS

To Hakim sir, who has inspired us to take on difficult challenges.

REFERENCES

[1] Mirza Nihal Baig, Rony Hajong, Mahdi Murshed Patwary, Mohammad Shahidur Rahman, and Husne Ara Chowdhury. 2024. BadODD: Bangladeshi Autonomous Driving Object Detection Dataset. arXiv:2401.10659 [cs.CV]

[2] Chuyi Li, Lulu Li, Hongliang Jiang, Kaiheng Weng, Yifei Geng, Liang Li, Zaidan Ke, Qingyuan Li, Meng Cheng, Weiqiang Nie, Yiduo Li, Bo Zhang, Yufei Liang, Linyuan Zhou, Xiaoming Xu, Xiangxiang Chu, Xiaoming Wei, and Xiaolin Wei. 2022. YOLOv6: A Single-Stage Object Detection Framework for Industrial Applications. arXiv:2209.02976 [cs.CV]

[3] Wenyu Lv, Yian Zhao, Shangliang Xu, Jinman Wei, Guanzhong Wang, Cheng Cui, Yuning Du, Qingqing Dang, and Yi Liu. 2023. DETRs Beat YOLOs on Real-time Object Detection. arXiv:2304.08069 [cs.CV]

[4] ASM Shihavuddin and Mohammad Rifat Ahmmad Rashid. 2020. *Dhaka Traffic Detection Challenge Dataset.* https://www.kaggle.com/datasets/rifat963/dhakaai-dhaka-based-traffic-detection-dataset

[5] Zhuofan Zong, Guanglu Song, and Yu Liu. 2022. DETRs with Collaborative Hybrid Assignments Training. arXiv:2211.12860 [cs.CV]