A Comparative Study of DETR and YOLO on Traffic Object Detection

Huaze Liu
Harvey Mudd College
Claremont, California
hualiu@g.hmc.edu

Qingyang Zeng
Harvey Mudd College
Claremont, California
bozeng@g.hmc.edu

Abstract

For years, YOLO was demonstrated as the golden standard in object detection that features real-time speed and efficiency [7]. On the other hand, Object Detection models with an embedding of a Transformer architecture like DETR opened new paradigm insights into an area capable of global contextual understanding and capturing long-range dependencies within images [2]. In this work, we have compared YOLO and DETR's performances in carrying out traffic object detection tasks under the constraints of transfer learning and efficiency. Pre-trained and fine-tuned models are evaluated on the KITTI dataset [4], with Metric scores including mean Average Precision (mAP), class accuracy, and inference time. While YOLO demonstrates superior efficiency in most scenarios, DETR exhibits promise in transferability and robustness considering our small-size custom dataset with no more than 8k images. Our findings provide nuanced insights into the suitability of these models across various applications which offers a foundation for future advancements in object detection research. For the code reference, please check this GitHub repository. For the dataset reference, please check KITTI dataset on Roboflow. [8]

1. Introduction

Traffic object detection is critical for intelligent transportation systems. Efficient detection of objects such as pedestrians, vehicles, and road signs is essential for safety, traffic flow, and real-time decision making.

Deep Learning models have revolutionized object detection. YOLO ("You Only Look Once"), is a benchmark for real-time detection due to its speed and precision [7]. Meanwhile, Transformer based models, such as DEtection TRansformer (DETR), have introduced global contextual understanding, leveraging the Transformer architecture to capture long-range dependencies in images [2, 3]. In this project, we compared DETR's performance including transferability and efficiency to YOLO in traffic object detection with object detection subset of the KITTI dataset [4] which includes 7464 images. Our main questions are:

1. DETR and YOLO's performance under standalone and transfer learning settings.
2. Trade-offs between inference time, mean Average Precision (mAP), and class accuracy across object sizes.

Our findings highlight DETR's promising transferability and robustness considering a small dataset we used for the sake of computation power and training time. Meanwhile, YOLO demonstrates its efficiency and simplicity for object detection tasks. These insights contribute to advancements in traffic object detection, guiding model selection for diverse applications.

2. Related Work

2.1. YOLO: You Only Look Once

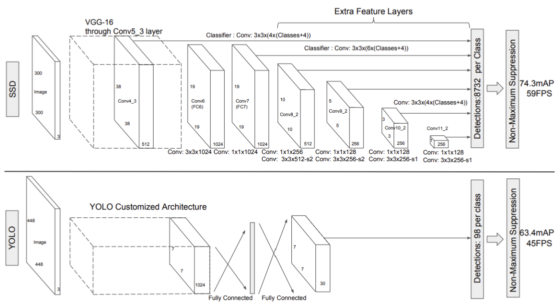

Figure 1. Framework of YOLO, illustrating its single-stage architecture for object detection. Adapted from the SSD paper [5].

"You Only Look Once" (YOLO) deep learning-based framework revolutionized object detection in 2015 by Joseph Redmon. This framework offers a unified and real-time solution. [7] Unlike traditional multi-stage detectors, YOLO processes the entire image in a single pass, treating object detection as a regression problem. To start with, the image is divided into a grid, with each cell predicting bounding boxes, confidence scores, and class probabilities. This approach enables YOLO to achieve real-time performance, processing up to 45 frames per second (FPS), as introduced in the paper cited. [5]. For this study, we selected YOLOv8, one of the iterations of the YOLO series published in 2023. This represents a relatively stable and advanced version of YOLO. The pretrained model provided by Ultralytics is widely recognized for its robustness and ease of use, making it ideal for this comparative study on traffic object detection [9].

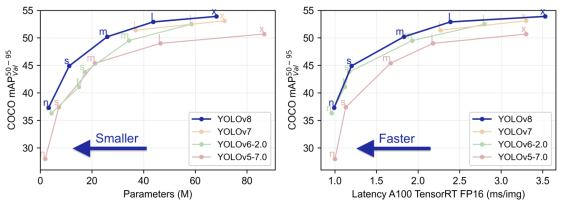

Figure 2. Comparison of YOLOv8, YOLOv7, YOLOv6, and YOLOv5 on COCO in terms of mAP and latency, adapted from Ultralytics documentation [9]. YOLOv8 demonstrates superior performance in both accuracy and speed.

2.2. Detection Transformer (DETR)

DETR is a paradigm shift to the current object detection approaches by embedding the Transformer architecture, designed for natural language processing tasks, into an object detection framework. The DETR was introduced by Carion et al. in 2020 [2]. Rather than using traditional hand-design components such as anchor boxes and non-maximum suppression, it formulates object detection as a set prediction problem, where a fixed number of object queries are matched to ground truth objects via a bipartite matching process.

As shown in Figure 3, DETR employs a traditional CNN backbone to acquire a 2D representation of an input image. The model compresses it and enhances it with a positional encoding prior to inputting it into a transformer encoder. A transformer decoder receives a limited set of learnt positional embeddings, referred to as object queries, and also attends to the encoder output. Each output embedding from the decoder is processed through a shared feedforward network (FFN) that predicts either a detection (class and bounding box) or a ”no object” classification. [2]. By simultaneously attending to all components of the image, DETR collects long-range dependencies and global contextual information, hence enabling effective object detection in intricate scenarios. Despite its efficacy, DETR's implementation of Transformers incurs significant computational overhead particularly during training. This fact makes it unsuitable for resource-limited applications. Conversely, the management of occlusion and congested environments renders it a highly attractive model for applications requiring substantial durability and adaptability.

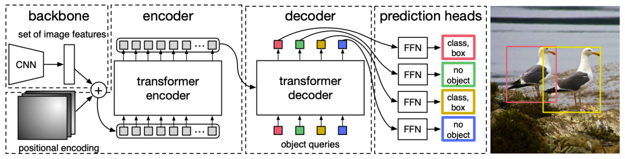

Figure 3. Model Architecture of DETR [2]

2.3. KITTI Dataset

The KITTI dataset is a widely used benchmark for autonomous driving research. It provides diverse real-world scenarios for evaluating detection models [4]. In this project, we utilized the Object Detection subset of the KITTI dataset, accessed through Roboflow, as prepared by Sebastian Krauss [8]. This version includes annotated data across eight object classes: Car, Cyclist, Misc, Pedestrian, Person_sitting, Tram, Truck, and Van. The Object Detection subset is divided into 5,223 images for training, 1,495 images for validation, and 746 images for testing.

3. Methodology

3.1. Data Preprocessing

Data Preparation for YOLOv8

To ensure compatibility with the YOLOv8 model's input requirements, the dataset was resized using a stretched resolution of 640x640 pixels [8]. No additional data augmentation was applied during preprocessing. This resizing ensures consistency with the pretrained YOLOv8 architecture while maintaining the integrity of the original annotations.

In our study, we used the training and validation sets to fine-tune and evaluate the performance of YOLOv8 and DETR models, taking advantage of the diverse scenarios and object classes in the KITTI dataset. Additionally, we used the training, validation, and testing sets to evaluate YOLOv8's baseline performance and YOLOv8 with transfer learning. This ensures a comprehensive analysis of detection accuracy and efficiency.

Data Preparation for DETR

While the YOLOv8 model relied on a uniform resized input with minimal preprocessing, our DETR-based transfer learning workflow introduced a more complex on-the-fly data augmentation pipeline. Instead of resizing all images offline, we dynamically applied each training iteration with geometric and color-based transformations. In this way, generalization is enhanced by providing more varied input patterns to DETR. Concretely, this includes random

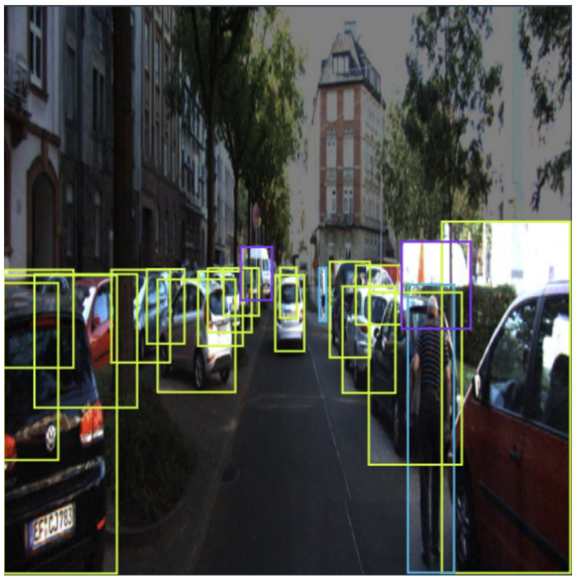

Figure 4. A sample image from the KITTI dataset resized to 640x640 pixels, showing annotations for multiple object classes.

perspective transforms, horizontal flips, brightness and contrast adjustments, and hue-saturation-value (HSV) shifts for data augmentations with Albumentations [1], which happens only during training. This is carefully wrapped up in a custom PyTorch dataset class. Unlike the YOLOv8 preprocessing, this allows the model to see slightly different versions of the same image across epochs and prevents overfitting by making it more robust.

3.2. Experiment Workflow

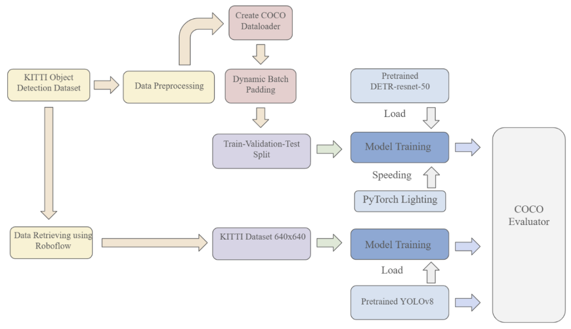

Figure 5. Overview of the experiment workflow, including dataset preparation, model implementation, evaluation, and analysis.

Our experiment workflow was designed to comprehensively evaluate the performance of YOLOv8 and DETR models for traffic object detection using the KITTI dataset. The workflow consists of the following key stages:

Model Implementation: Pretrained YOLOv8 and DETR models were used as the starting point. The models were fine-tuned using the training and validation sets, while their baseline performance was directly evaluated using the test set.

Evaluation Metrics: We focused on key metrics such as mean Average Precision (mAP), class-wise accuracy, and inference time to compare the models' efficiency and robustness across varying object sizes and scenarios.

Analysis: Results were analyzed to highlight the trade-offs between YOLOv8's real-time performance and DETR's robustness and adaptability, particularly in scenarios involving smaller datasets or more complex environments.

The workflow, as illustrated in Figure 5, provides a structured approach for evaluating object detection models. By incorporating cross-comparison metrics and curves, the analysis ensures consistency and reliability in comparing YOLOv8 and DETR under diverse conditions.

4. Results

The evaluation metrics depicted in Figure 6 highlight clear performance differences across various configurations of DETR and YOLOv8. Starting with the IoU threshold-based metrics (top plot), we observe that the fully trained YOLO v8 ($66.7\% - 84.0\%$) consistently outperforms trained DETR($41.5\% - 71.1\%$) and DETR trained with augmented training set ($38.0\% - 65.5\%$), which achieved notably higher mAP values at all IoU threshold. This suggests that YOLOv8, after fine-tuning, is better able to precisely localize and classify objects. Among the DETR models, the normally trained DETR and the trained DETR with augmentation (Trained DETR (Aug)) both show improvement over the pre-trained DETR baseline, particularly at the stricter IoU thresholds, indicating that our experiment made DETR effectively transfer-learning from KITTI dataset.

When examining performance across different object sizes (bottom plot), the trends remain consistent. YOLO v8 maintains its advantage across small, medium, and large objects detection compared to DETR and DETR with augmented training set on KITTI. Additionally, we can observe that both detection frameworks are good at detecting larger objects with a higher mAP.

Model	Mean Inference Time (s)	Std Dev (s)
YOLO v8	0.0087	0.0003
DETR	0.0163	0.0005
DETR (Aug)	0.0166	0.0043

Table 1. Mean inference time with standard deviation for YOLOv8, DETR, and DETR with data augmentation.

Additionally, in terms of inference efficiency, as shown

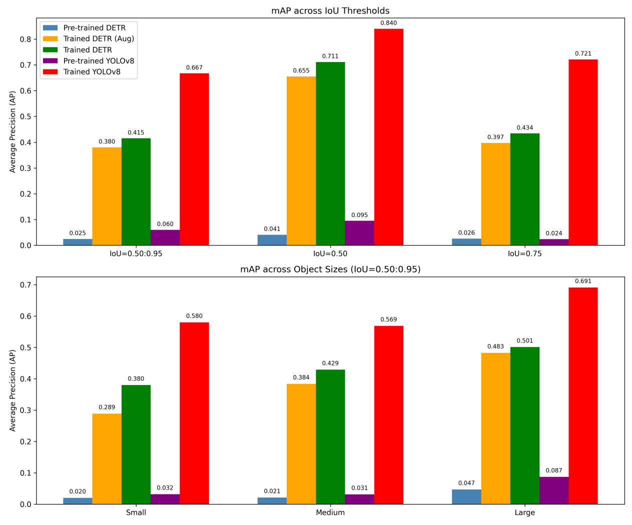

Figure 6. mean Average Precision comparison of two detection frameworks

in Table 1, YOLO v8 achieves the fastest mean inference time at 0.0087 seconds per image, nearly twice as fast as the DETR-based models. Both the baseline DETR and the augmented DETR variants require approximately 0.017 to 0.018 seconds per inference, indicating a slower processing pipeline. Notably, while data augmentation improved the detection performance for DETR, it also introduced a higher variance in inference times. Overall, these results suggest that YOLOv8 is more computationally efficient.

5. Analysis & Discussion

In terms of DETR's performance, the visualization of self-attention maps from various layers and heads of the DETR model reveals a pattern that helps explain the sub-optimal mAP observed in DETR's performance compared to YOLO v8. In the provided figures (Fig.7), the highlighted attention regions fail to strongly correlate with semantically meaningful areas of the scene, such as distinct objects or their boundaries with our transfer-learned DETR model. Instead, the attention weights appear dispersed and lack a strongly coherent focus on salient targets like vehicles or pedestrians.

This broad, mid-intensity distribution suggests that the model currently has difficulty in concentrating its representational capacity on the most critical parts of the image. When self-attention does not mostly prioritize meaningful object features, the downstream object queries cannot fully reliably infer high-quality bounding boxes and class labels, which ultimately leads to lower detection accuracy and reduced mAP. Although DETR's transformer architecture is designed to learn object-centric representations through attention, the current results indicate that the training or dataset conditions might not have been sufficient to guide the model towards sharper, more discriminative attention patterns. Considering the size of the used dataset (with no more than 7k images for training and validation), further optimization and additional data augmentation techniques are necessary to help the model better differentiate objects from the background and enhance the resulting mAP.

Observing YOLO's performance on KITTI dataset (more details shown in Appendix A), we can conclude that we find that YOLO not only achieves higher mAP scores but also maintains a more stable attention distribution over object regions. Its one-stage detection pipeline, combined with extensive pre-training on large-scale data, enables faster convergence on smaller datasets and more robust localization of vehicles, pedestrians, and other traffic participants. Consequently, YOLO's efficiency and its stronger correlation between predicted bounding boxes and actual object boundaries highlight why it remains a preferred choice in settings where both accuracy and inference speed are critical.

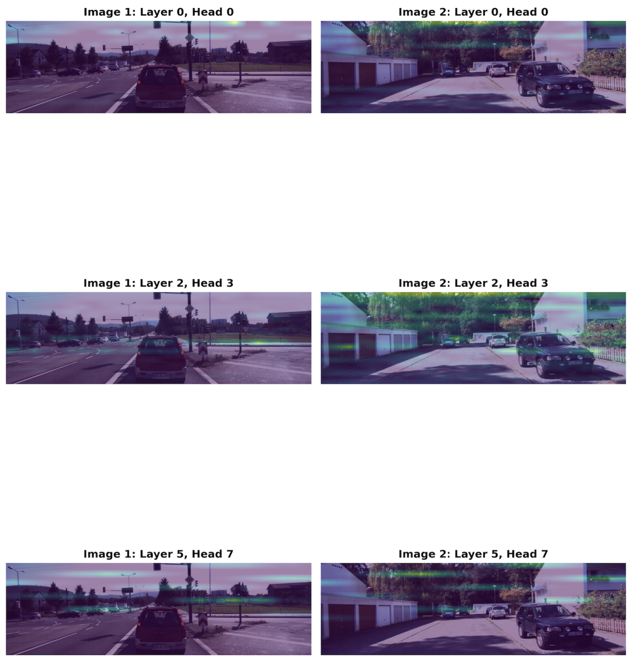

Figure 7. Attention Overlays for Test Images at Different Layers & Heads

6. Conclusion

In this study, we conduct a comprehensive comparison between DETR and YOLO v8 for traffic object detection using the tiny KITTI dataset, comprising approximately 8,000 photos. From our experiment results, YOLO v8 surpassed DETR in several aspects including mAP and inference time. We did expect a lower inference efficiency for DETR due to its intricate model structure and higher operating complexity, but we anticipated a higher detection accuracy for DETR than YOLO v8 before the experiment, particularly considering its ability to use global contextual cues effectively even with significantly constrained data. The features of DETR make it particularly advantageous for real-time, latency-sensitive applications such as autonomous driving. The result show that both models significantly profited from transfer learning, attaining elevated mAP values and enhanced robustness to difficult object categories and fluctuating environmental variables.

Significantly, DETR's encouraging outcomes in data-scarce environments provide optimism for situations where abundant, fully annotated datasets are not readily available. The Transformer-based design inherently supports intricate item distributions and contextual interactions, enabling the model to perform effectively with a constrained training dataset. DETR exhibits greater computational requirements, yet fails to achieve YOLOv8's superiority in real-time responsiveness; nonetheless, this compromise may be justified if global reasoning and scalability are essential.

In conclusion, our findings indicate that practitioners must meticulously evaluate their priorities: YOLOv8 may be preferred for its speed and efficiency, whereas DETR may be favored for its promising superior contextual comprehension, particularly with smaller or more specialized datasets. Our work facilitates further study of the testing of other DETR variants, including real-time DETR and deformable DETR [6, 10]. Additional studies with more diverse datasets and advanced data augmentation techniques are also expected to further elucidate the strengths, and to a lesser extent the flaws, of each methodology. This work establishes a foundation for enhancing model selection in intelligent transportation systems, facilitating future advancements in traffic object recognition and related areas.

7. Acknowledgements

We would like to express our sincere gratitude to Professor Calden Wloka and all the class of CSCI153 for providing invaluable insights and constructive feedback throughout the course of this research. Particularly, Prof. Wloka's expertise and thoughtful guidance in project ideation greatly contributed to the development and refinement of this work.

References

[1] Alexander Buslaev, Alexander Parinov, Evgenii Khvedchenya, Vladimir I. Iglovikov, and Alexey A. Kalinin. Albumentations: Fast and flexible image augmentations. *Information*, 11(2), 2020. 3

[2] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *European Conference on Computer Vision (ECCV)*, pages 213–229, 2020. 1, 2

[3] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations (ICLR)*, 2021. 1

[4] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3354–3361, 2012. 1, 2

[5] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C. Berg. Ssd: Single shot multibox detector. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 21–37. Springer, 2016. 1, 2

[6] Tong Lv, Yiqi Wang, Shaoyu Zhao, Wenqiang Zhang, Zhengkai Wang, Zeming Liu, Xiangyu Zhang, and Jian Sun. Rt-detr: Real-time object detection with transformer. *arXiv preprint arXiv:2303.11840*, 2023. 5

[7] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 779–788, 2016. 1

[8] Roboflow and Sebastian Krauss. Kitti dataset for object detection. https://universe.roboflow.com/, 2023. Accessed: 2023-12-08. 1, 2

[9] Ultralytics. Ultralytics yolov8 documentation, 2023. 2

[10] Xizhou Zhu, Weijian Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable detr: Deformable transformers for end-to-end object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 834–844, 2021. 5

APPENDIX A: More Performance Analysis for YOLO v8 on KITTI Dataset

In this section, we present the evaluation results for the pretrained YOLOv8 model and the YOLOv8 model fine-tuned with transfer learning on the KITTI dataset. The evaluation includes key performance metrics such as precision (P), recall (R), mean Average Precision at IoU=0.5 (mAP@50), and mean Average Precision at IoU=0.5:0.95 (mAP@50-95).

Table 2 summarizes the precision and recall metrics for all classes, while Table 3 highlights the detection accuracy through mAP metrics.

Class	Model	P	R
All	Pretrained	0.165	0.0275
	Transfer Learning	0.873	0.768
Car	Pretrained	0.323	0.00342
	Transfer Learning	0.941	0.924
Cyclist	Pretrained	0.167	0.00709
	Transfer Learning	0.897	0.745
Misc	Pretrained	0.00129	0.0108
	Transfer Learning	0.950	0.817
Pedestrian	Pretrained	0.000	0.000
	Transfer Learning	0.935	0.574
Tram	Pretrained	0.308	0.150
	Transfer Learning	0.892	0.868
Truck	Pretrained	0.519	0.0491
	Transfer Learning	0.980	0.935
Van	Pretrained	0.000	0.000
	Transfer Learning	0.922	0.916

Table 2. Precision (P) and Recall (R) metrics for YOLOv8 models on the KITTI test dataset. Metrics are reported for pretrained and fine-tuned models. The fine-tuned YOLOv8 model demonstrates significantly improved precision and recall across all classes.

Precision and recall provide a measure of the model's ability to correctly detect objects (precision) and its effectiveness at identifying all objects in the dataset (recall). The fine-tuned YOLOv8 model achieves substantially higher precision and recall compared to the pretrained model, especially for classes such as *Car* and *Truck*.

The mean Average Precision (mAP) provides a comprehensive evaluation of detection accuracy across all object classes. The fine-tuned YOLOv8 model significantly outperforms the pretrained model in both mAP@50 and mAP@50-95, demonstrating its effectiveness in adapting to the KITTI dataset.

Class	Model	mAP@50	mAP@50-95
All	Pretrained	0.095	0.0578
	Transfer Learning	0.840	0.667
Car	Pretrained	0.163	0.119
	Transfer Learning	0.955	0.832
Cyclist	Pretrained	0.0881	0.00881
	Transfer Learning	0.848	0.589
Misc	Pretrained	0.000665	6.65e-05
	Transfer Learning	0.896	0.675
Pedestrian	Pretrained	0.000	0.000
	Transfer Learning	0.768	0.501
Tram	Pretrained	0.187	0.129
	Transfer Learning	0.920	0.732
Truck	Pretrained	0.273	0.206
	Transfer Learning	0.966	0.837
Van	Pretrained	0.000	0.000
	Transfer Learning	0.947	0.808

Table 3. mAP@50 and mAP@50-95 metrics for YOLOv8 models on the KITTI test dataset. The fine-tuned YOLOv8 model exhibits significant improvement in detection accuracy compared to the pretrained model.

Comparison of Model Behavior: Pretrained vs. Transfer Learning

This subsection compares the performance of the pretrained YOLOv8 model and the YOLOv8 model fine-tuned with transfer learning using key evaluation curves—F1-Confidence, Precision-Confidence, Precision-Recall, and Recall-Confidence.

The F1-Confidence curves (Figure 8) reveal improved balance between precision and recall after transfer learning, with the fine-tuned model peaking at 0.81, compared to the pretrained model's low F1 score.

The Precision-Confidence curves (Figure 9) show substantial precision gains, with the fine-tuned model achieving near-perfect precision at higher confidence thresholds, unlike the pretrained model's unstable trends.

The Precision-Recall curves (Figure 10) highlight improved recall for classes like *Trucks* (0.966) and *Vans* (0.947) in the fine-tuned model, while the pretrained model exhibits minimal recall.

The Recall-Confidence curves (Figure 11) emphasize enhanced recall, with the fine-tuned model exceeding 0.8 across multiple thresholds, compared to the pretrained model's recall below 0.2.

These results demonstrate that transfer learning significantly improves YOLOv8's detection performance on the KITTI dataset by enhancing the balance between precision and recall.

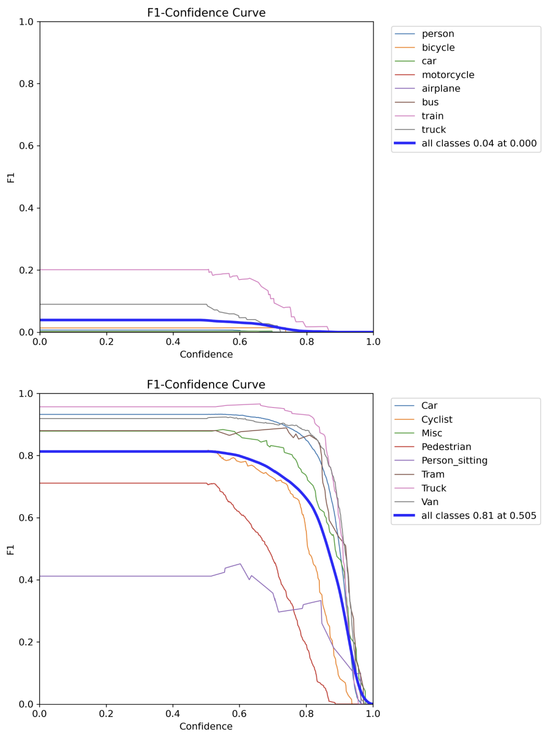

Figure 8. F1-Confidence Curves: Pretrained YOLOv8 (top) vs. Transfer Learning (bottom).

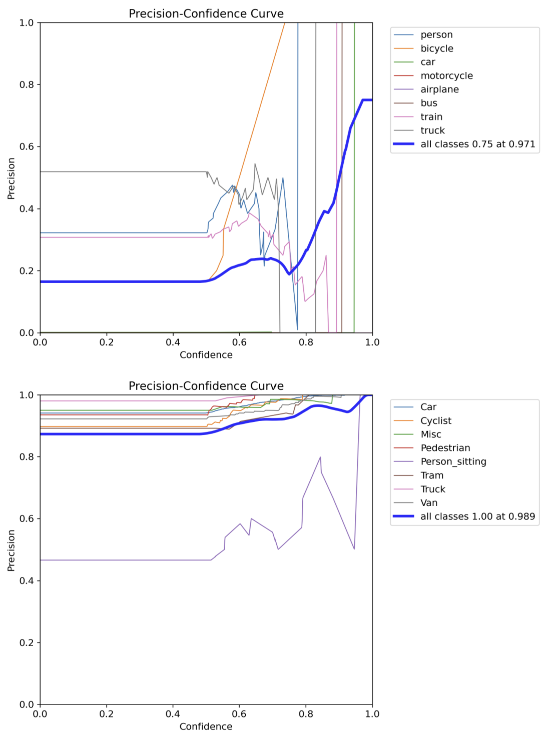

Figure 9. Precision-Confidence Curves: Pretrained YOLOv8 (top) vs. Transfer Learning (bottom).

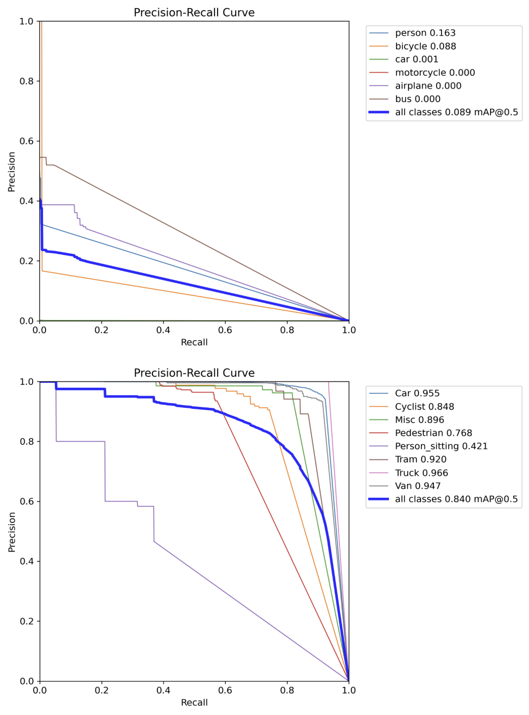

Figure 10. Precision-Recall Curves: Pretrained YOLOv8 (top) vs. Transfer Learning (bottom).

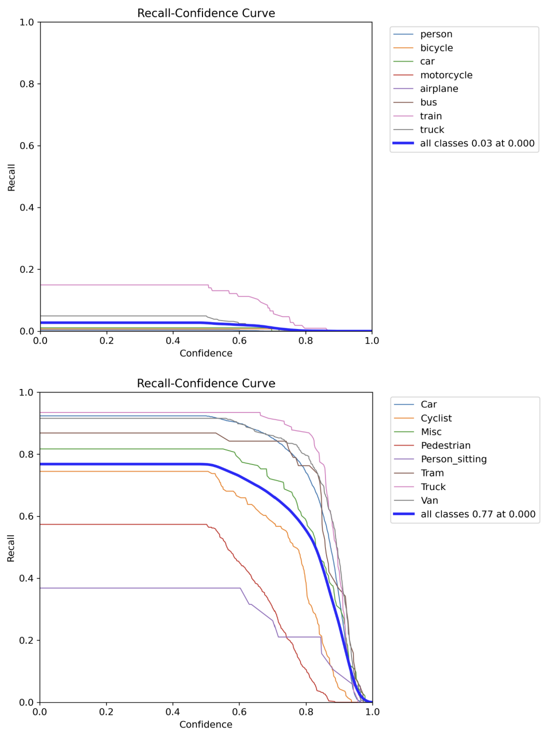

Figure 11. Recall-Confidence Curves: Pretrained YOLOv8 (top) vs. Transfer Learning (bottom).