Object Detection on BDD 100K dataset

Bosch Global Software Technologies

Project Report

Author:
Mritunjoy Halder

Summary

An end-to-end object detection framework was developed for the BDD100K dataset, integrating data analysis, model training, and evaluation within a reproducible Docker environment. An interactive `Streamlit` dashboard was implemented to analyze class distributions, dataset imbalances, and visual anomalies, providing key insights into data characteristics. The study explores two detection architectures, YOLO and Faster R-CNN. The YOLO model was trained and evaluated using both standard metrics and a CLIP-assisted scoring mechanism to enhance semantic understanding. The Faster R-CNN pipeline utilized a pretrained Hugging Face model and a custom lightweight MobileNet-based variant optimized for limited computational resources. Comprehensive evaluation was performed across both models to assess performance trade-offs between accuracy, efficiency, and interpretability. All workflows were containerized to ensure portability and reproducibility, with results summarized through quantitative metrics and qualitative visualizations presented in the accompanying report.

1 Exploratory Data Analysis

A detailed exploratory data analysis (EDA) was performed on the BDD100K object detection dataset to assess data quality, class imbalance, and structural characteristics before model development. All analyses were conducted through a Dockerized `Streamlit` dashboard for reproducibility and interactivity.

1.1 Annotation Parsing and Structure

Dataset annotations were parsed from JSON files to extract bounding boxes and class labels. The resulting mappings between images and annotated objects were converted into structured DataFrames for analysis. Figure 1 shows the same.

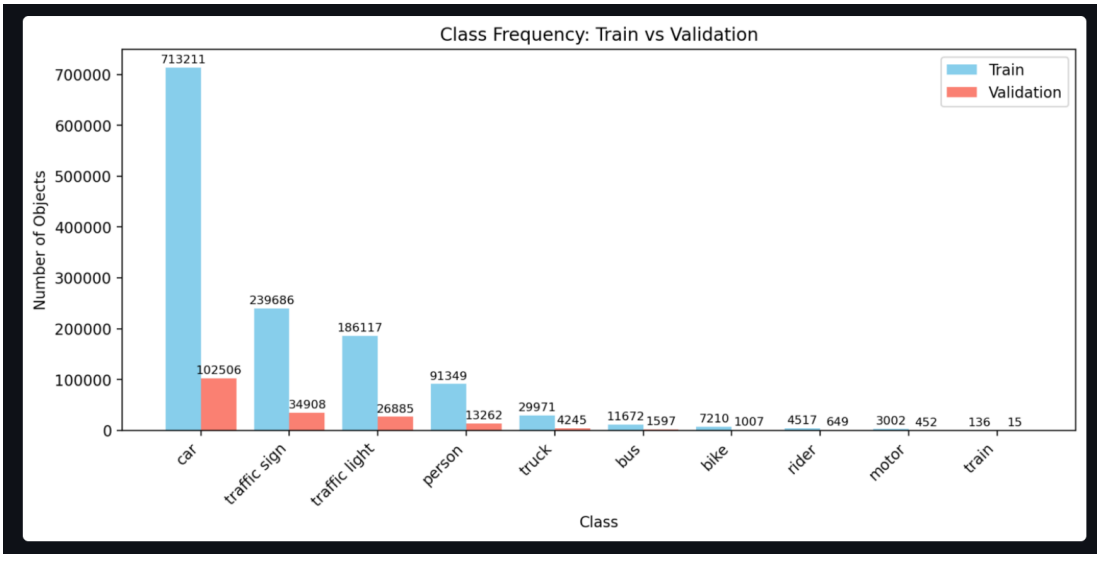


Figure 1: Annotation extraction workflow ensuring consistent and validated bounding box parsing across the dataset.

1.2 Class Distribution and Imbalance

Object frequency analysis (Figure 2) revealed severe class imbalance, with *car* and *person* dominating the dataset, while *rider* and *train* were underrepresented. The imbalance ratio

exceeded 10:1, indicating a strong bias toward vehicular classes. This motivated the use of weighted losses and targeted augmentations in later training phases.

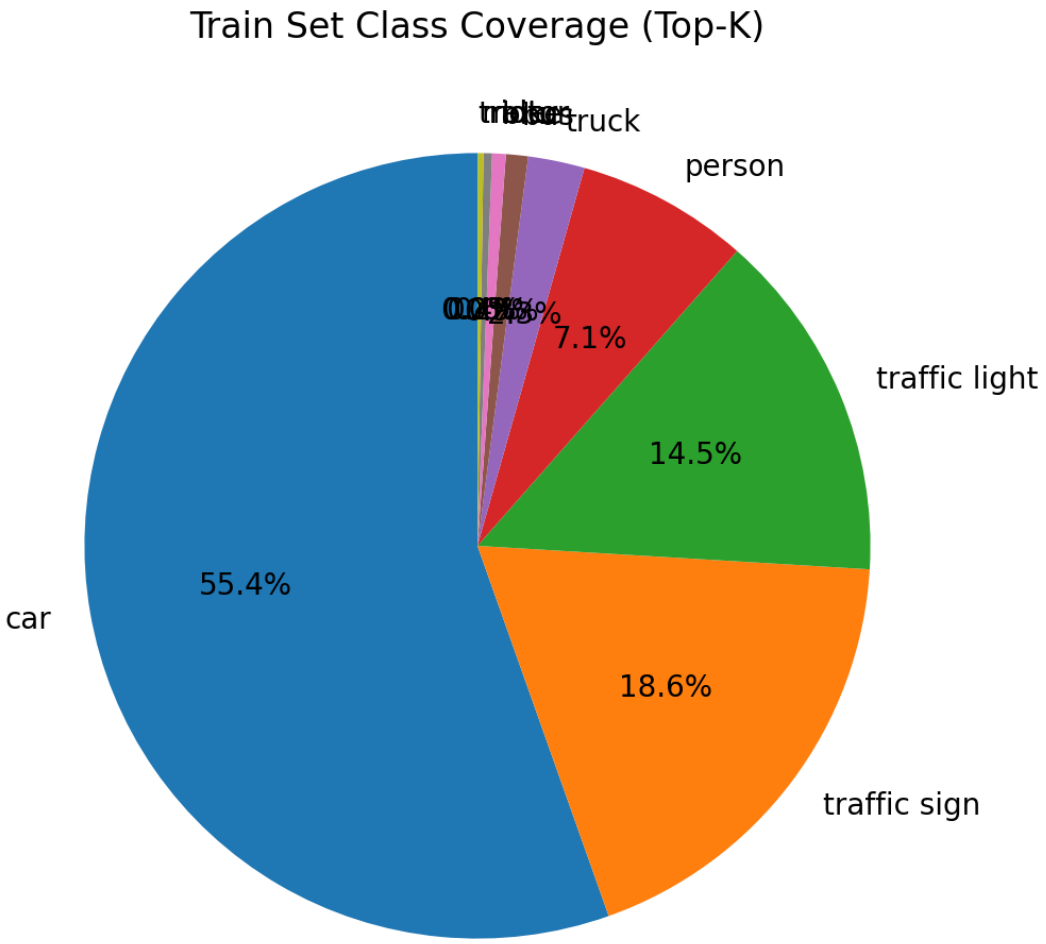


Figure 2: Train validation class frequency comparison showing dominant vehicular classes and sparsity in minor ones.

1.3 Visibility and Scale Variation

Average object visibility, measured as the ratio of bounding box area to image area, is shown in Figure 4. Smaller objects such as *traffic sign* and *rider* exhibited significantly lower visibility, implying higher detection difficulty. A weak negative correlation between object density and visibility suggests that crowded scenes tend to contain smaller, less discernible instances. In Figute 3 we rank categorize interms of detection difficulty.

4

1.4 Aspect Ratio and Shape Insights

Bounding box aspect ratios were analyzed to guide anchor generation. As shown in Figure 5, most objects exhibit near-square shapes, while elongated ones such as *bus* and *truck* deviate significantly.

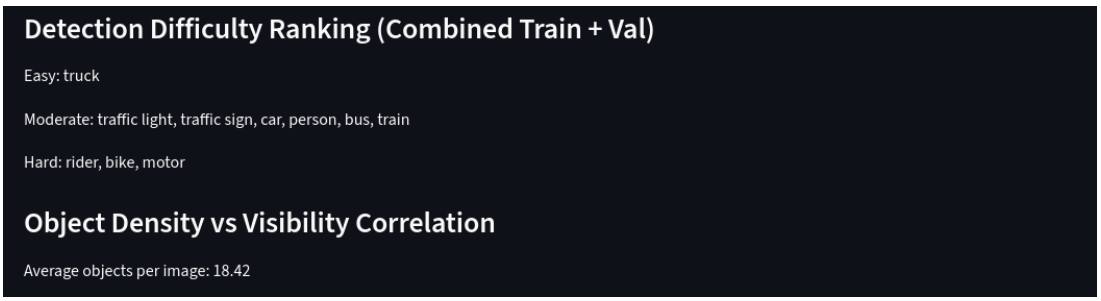
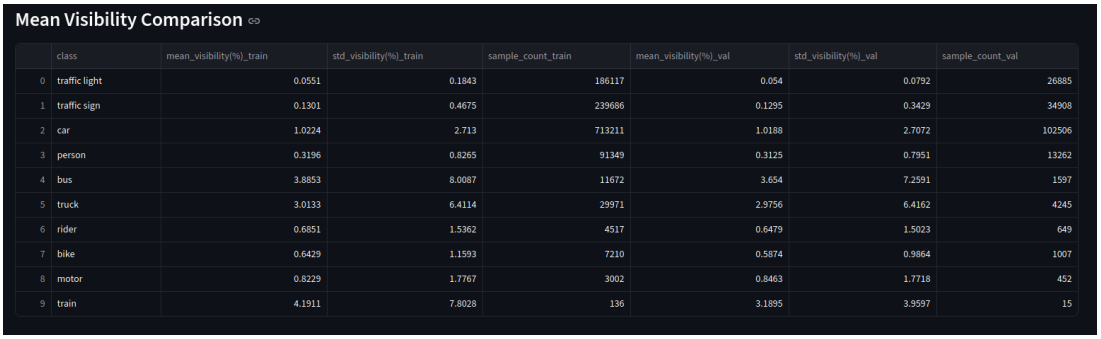


Figure 3: Ranking object detection diffculty depending on scale and visibility.



	class	mean_visibility(%)_train	std_visibility(%)_train	sample_count_train	mean_visibility(%)_val	std_visibility(%)_val	sample_count_val
0	traffic light	0.0551	0.1843	186117	0.054	0.0792	26885
1	traffic sign	0.1301	0.4675	239686	0.1295	0.3429	34908
2	car	1.0224	2.713	713211	1.0188	2.7072	102506
3	person	0.3196	0.8265	91349	0.3125	0.7951	13262
4	bus	3.8853	8.0087	11672	3.654	7.2591	1597
5	truck	3.0133	6.4114	29971	2.9756	6.4162	4245
6	rider	0.6851	1.5362	4517	0.6479	1.5023	649
7	bike	0.6429	1.1593	7210	0.5874	0.9864	1007
8	motor	0.8229	1.7767	3002	0.8463	1.7718	452
9	train	4.1911	7.8028	136	3.1895	3.9597	15

Figure 4: Mean visibility versus object density. Classes with lower visibility are prone to detection errors in dense scenes.

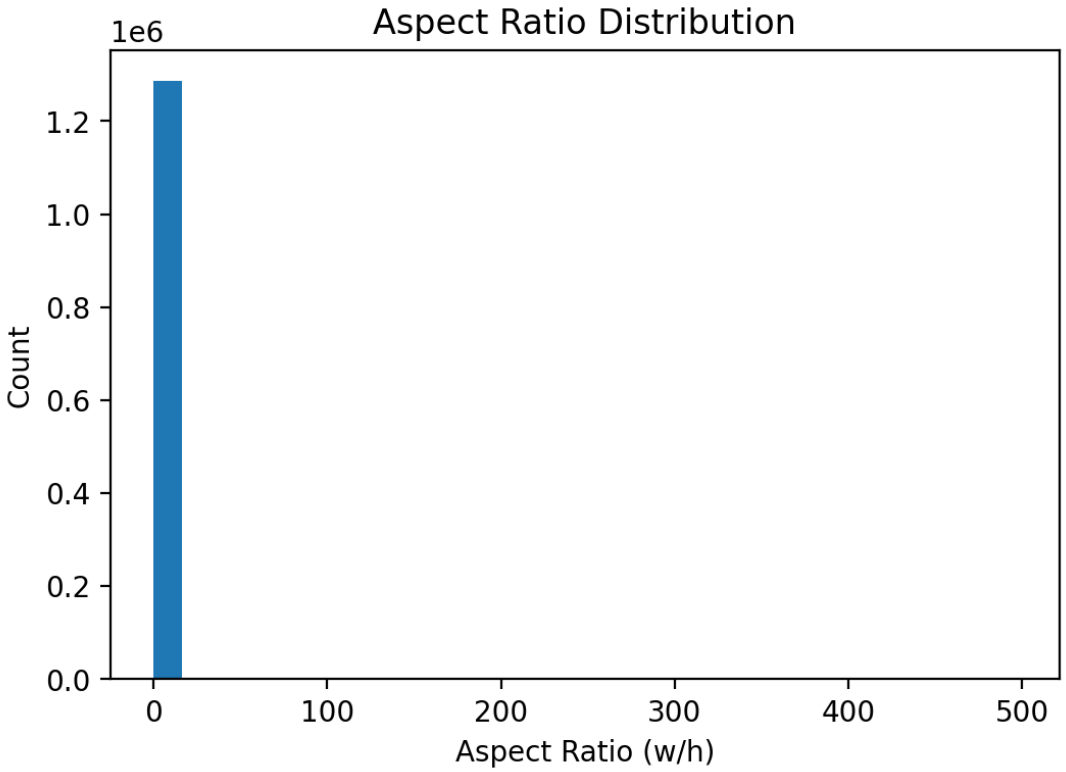


Figure 5: Aspect ratio distribution showing shape variability across classes informative for anchor box tuning.

1.5 Inter-Class Co-Occurrence

The co-occurrence matrix (Figure 6) revealed frequent contextual pairings such as *car traffic light* and *car person*. These spatial dependencies highlight potential confusion zones that could

influence false-positive rates during evaluation.

Figure 6: Inter-class co-occurrence matrix highlighting contextually linked object pairs likely to cause detection ambiguity.

1.6 Remarks

The EDA established that the dataset suffers from class imbalance and visibility-based difficulty, with distinct structural diversity across object categories. These findings informed later design choices such as using CLIP-assisted semantic scoring and lightweight backbones to improve detection robustness under varying scale and context.

2 Model Development and Selection

Following the exploratory data analysis, two object detection models were selected for experimentation: **YOLOv11** and **Faster R-CNN**. The choice of these models was guided by their complementary strengths in detection accuracy, computational efficiency, and adaptability to diverse hardware settings.

2.1 YOLOv11

The `YOLOv11` model was chosen as the primary detection framework due to its superior balance between inference speed and mean average precision (mAP). Compared to transformer-based models such as DETR and RefineDet, which achieve competitive accuracy at significantly higher computational costs, YOLOv11 provides a faster, single-stage detection pipeline that remains robust under real-time conditions. Its performance efficiency and stable convergence make it particularly suitable for large-scale datasets such as BDD100K, where rapid iteration and evaluation are essential.

2.2 Faster R-CNN

The `Faster R-CNN` model was used to demonstrate flexible training capability and serve as a baseline for region-based detection. It was configured with a lightweight MobileNet backbone to accommodate limited GPU resources but the accuracy was low. It is only choosen to show custom training pipeline.

In summary, YOLOv11 was selected as the preferred model owing to its superior speed accuracy trade-off, while Faster R-CNN served as a lightweight, interpretable alternative suited for controlled training and evaluation.

3 Model Evaluation

The trained models were evaluated to quantify detection performance and to analyze the effect of semantic re-scoring through the proposed CLIP-assisted framework. Two evaluation settings were implemented for the YOLO model, followed by validation of the Faster R-CNN baseline.

3.1 YOLO Evaluation

The baseline evaluation employed the `YOLOv11` model trained on the BDD100K dataset. Standard evaluation metrics such as mean Average Precision (mAP) at thresholds of 0.5 and 0.5:0.95, precision, and recall were computed using the built-in `ultralytics` validation API. The model achieved strong detection performance across major classes, with stable generalization on unseen validation data.

3.2 CLIP-Assisted Re-Scoring (Proposed Improvement)

To enhance semantic alignment between visual predictions and class labels, a CLIP-based post-evaluation refinement was introduced. In this approach, each predicted region from YOLO was re-scored using a similarity measure between the region's CLIP-encoded image features and corresponding class text embeddings. The final confidence score was computed as a weighted combination of the YOLO confidence and CLIP similarity, effectively re-ranking predictions based on visual semantic consistency.

This modification improved classification confidence for ambiguous instances, particularly in cases of inter-class visual overlap (*e.g.*, rider vs. person, car vs. truck). The proposed CLIP-assisted evaluation achieved more semantically coherent detections and reduced false positives in contextually complex scenes.

3.3 Faster R-CNN Evaluation

The Faster R-CNN model was evaluated under two configurations: one using a pretrained model fine-tuned on the BDD dataset, and another trained with a lightweight MobileNet backbone. Evaluation was performed using the COCO-style mAP metric. Although the Faster R-CNN models demonstrated acceptable precision, they lagged behind YOLOv11 in both inference speed and overall mAP, validating YOLO's suitability for real-time applications under computational constraints.

3.4 Summary

The evaluation demonstrated that the CLIP-assisted YOLO approach provides a meaningful improvement in semantic consistency without retraining. While the baseline YOLOv11 achieved strong quantitative results, the integration of CLIP post-processing enhanced qualitative detection reliability, establishing it as a promising direction for lightweight semantic refinement in object detection pipelines.

3.5 Evaluation Metrics

The evaluation follows the COCO detection standard, using Average Precision (AP) and Recall (AR) at multiple Intersection-over-Union (IoU) thresholds to ensure a comprehensive performance assessment. The primary metric, **AP@[.5:.95]**, represents the mean of AP values computed over IoU thresholds ranging from 0.5 to 0.95 in increments of 0.05, providing a balanced measure of both localization and classification accuracy.

AP@0.5 focuses on detection precision at a single IoU threshold of 0.5, capturing coarse localization accuracy, while **AP@0.75** reflects stricter localization performance under higher overlap criteria. These precision metrics quantify how confidently the model identifies objects at varying levels of positional tolerance.

These metrics were selected to provide a holistic evaluation of the models: AP captures precision under varying confidence thresholds, while AR quantifies the completeness of detections. Together, they offer a robust comparison of both standard YOLO and the proposed CLIP-assisted approach in terms of accuracy, recall coverage, and semantic reliability across diverse object categories.

3.6 Results

The YOLO model with ultralytics package is trained on RTX 3050 GPU (4GB VRAM) for 50 epoch on BDD dataset. Table 1 shows the mAP Precision and Recall of each classes and overall dataset.

Due to time constraints and GPU constraints the CLIP evaluation was shown on a small sample of 20 images with YOLO. A full evaluaiton is still needed. One drawback with CLIP was it was taking time reducing the overall speed and increasing latency which is not acceptable for Real time scenarios. The comparison is shown in Table 2. But we can see there is a slight increase in mAP@100 which is crucial for object detection in wild although it performance is inferior in mAP<100

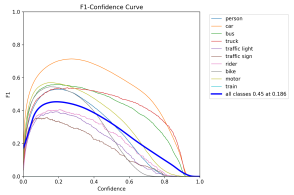
Table 1: YOLOv11 baseline evaluation on the BDD100K validation set showing per-class performance metrics. The model achieves an overall mAP@0.5 of 0.529 and mAP@0.5:0.95 of 0.337.

A more detailed study on non dominant classes may prove fruitfull.

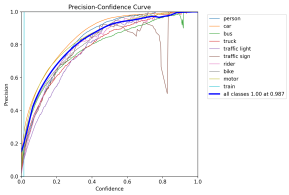
Class	Images	Instances	Precision (P)	Recall (R)	mAP@0.5	mAP@0.5:0.95
All	10000	185526	0.678	0.351	0.529	0.337
Person	3220	13262	0.796	0.399	0.611	0.342
Car	9879	102506	0.832	0.647	0.773	0.540
Bus	2689	4245	0.704	0.460	0.604	0.491
Truck	1242	1597	0.681	0.448	0.603	0.510
Traffic Light	578	1007	0.729	0.254	0.499	0.280
Traffic Sign	334	452	0.746	0.201	0.478	0.275
Rider	515	649	0.759	0.267	0.522	0.303
Bike	5653	26885	0.746	0.408	0.581	0.253
Motor	8221	34908	0.785	0.429	0.619	0.377
Train	14	15	0.000	0.000	0.000	0.000

Table 2: Comparison of YOLOv11 baseline and CLIP-assisted evaluation on the BDD100K dataset. The CLIP-based semantic re-scoring introduces a minor but consistent improvement in precision-oriented metrics without retraining the detector.

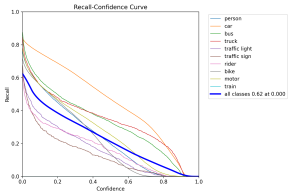
Metric	YOLOv11 (Baseline)	YOLOv11 + CLIP (Proposed)
AP@[.5:.95] (mAP)	0.2534	0.2514
AP@0.5	0.4075	0.4051
AP@0.75	0.2339	0.2326
AR@1	0.1372	0.1357
AR@10	0.2821	**0.2827**
AR@100	0.3137	**0.3139**



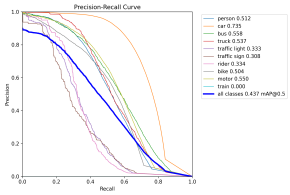
(a) F1-score vs confidence threshold, showing optimal trade-off between precision and recall.



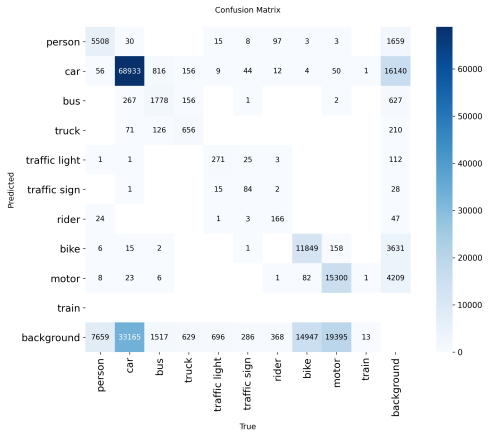
(b) Precision curve illustrating confidence thresholds maximizing detection accuracy.



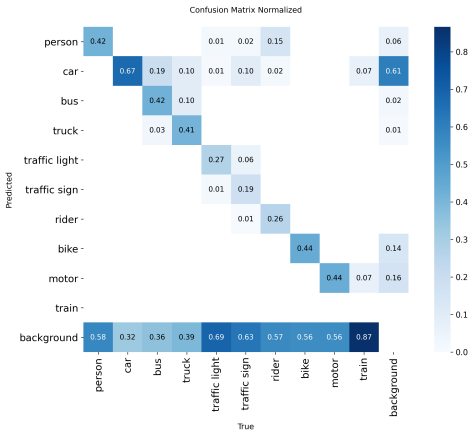
(c) Recall curve showing model sensitivity across varying confidence thresholds.



(d) Precision Recall curve summarizing detection trade-offs between true and false positives.

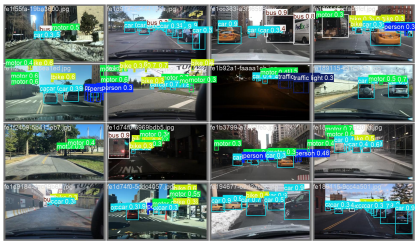


(e) Confusion matrix showing raw per-class prediction frequencies and misclassifications.



(f) Normalized confusion matrix highlighting relative per-class precision and recall.

(g) Ground truth annotations for validation batch 0.



(h) YOLOv11 predictions for validation batch 0 showing accurate detection alignment.

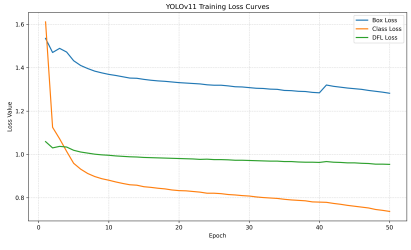
(i) Ground truth annotations for validation batch 1.

(j) YOLOv11 predictions for validation batch 1, showing consistent object localization.

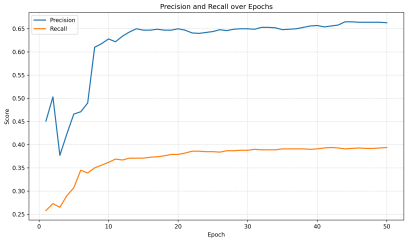
(k) Ground truth annotations for validation batch 2.

(l) YOLOv11 predictions for validation batch 2 illustrating minor detection misses.

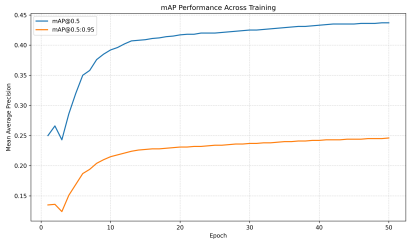
Figure 7: YOLOv11 validation visualizations showing quantitative performance curves (top), confusion matrices (middle), and qualitative comparisons between ground truth and predicted detections (bottom). The model exhibits strong precision recall consistency and reliable localization across varied scenes.



(a) Training loss curves for YOLOv11 showing the convergence of Box, Classification, and DFL losses across 50 epochs. The consistent decline indicates stable learning and effective optimization.



(b) Evolution of Precision and Recall over epochs. The steady improvement and stabilization after epoch 30 demonstrate strong generalization capability.



(c) mAP@0.5 and mAP@0.5:0.95 trends showing improved detection accuracy across multiple IoU thresholds. The plateau after epoch 40 indicates convergence.

Figure 8: Performance metrics visualization for YOLOv11 during training. Each subfigure collectively demonstrates progressive improvement in localization accuracy, classification precision, and overall detection quality.

3.7 Faster RCNN

3.8 Faster R-CNN Training and Evaluation

Faster R-CNN was employed to demonstrate the training and validation with dataloader code (as asked in assignment) of the proposed dataset and pipeline for region-based object detectors.

A lightweight `MobileNet` backbone was integrated to reduce computational overhead while retaining spatial representation power, making it suitable for constrained GPU environments. The model was trained for 20 epochs using the same BDD configuration and augmentation strategy as the YOLO experiments, leveraging the `YoloDataModule` for seamless data handling and transformation.

To assess the model's capability, evaluations were conducted using both the custom-trained weights and a pretrained Faster R-CNN model fine-tuned on the BDD dataset (HugoHE/faster-rcnn-bdd-finetune). Mean Average Precision (mAP) and Average Recall (AR) metrics were used for comparison, following the COCO-style evaluation protocol.

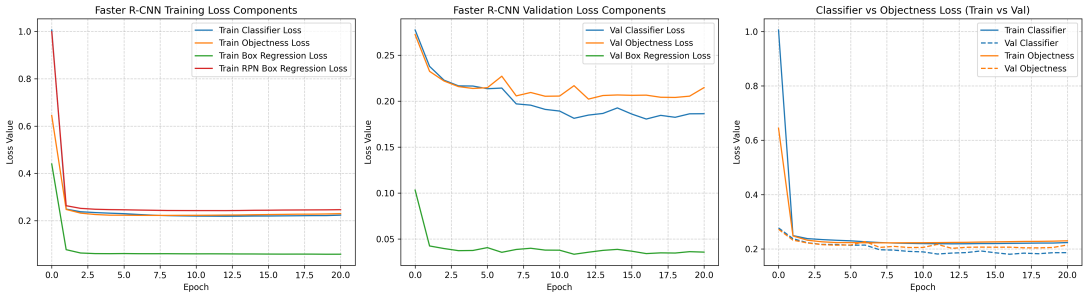


Figure 9: Inter-class co-occurrence matrix highlighting contextually linked object pairs likely to cause detection ambiguity.

3.9 Faster R-CNN Evaluation and Comparative Analysis

Two variants of Faster R-CNN were evaluated to assess their detection capability: a custom-trained lightweight model (20 epochs, MobileNet backbone) and a pretrained model fine-tuned on the BDD dataset. Both were evaluated using COCO-style metrics on the validation split.

Custom-trained Model. The custom-trained model, optimized for GPU constraints, showed extremely low overall performance ($mAP@0.5 = 0.00037$), with per-class AP values near zero except for minor detections in the `car` category. The results suggest severe underfitting due to limited epochs and lightweight backbone capacity. However, the model maintained structural consistency across epochs as indicated by stable loss curves.

Pretrained Model. The pretrained Faster R-CNN, in contrast, achieved reasonable detection on the `person` class ($AP@0.5 = 46.07\%$), but negligible performance across other categories ($AP@0.5 < 1\%$ for all others). This behavior indicates class-specific overfitting, where the model's strong prior on human-centric datasets biases it toward the `person` category, reducing its generalization to diverse traffic objects.

Comparative Discussion. When compared to YOLOv11, the performance disparity becomes evident. YOLOv11 achieved a significantly higher $mAP@0.5$ of 0.529 and maintained balanced detection across multiple categories such as `car`, `bus`, and `bike`. Moreover, YOLOv11's one-stage architecture ensured faster inference and better utilization of contextual cues, whereas Faster R-CNN struggled to generalize with limited data and computational constraints.

Table 3: Comparison between custom-trained and pretrained Faster R-CNN models baseline. The pretrained model exhibits overfitting on the `person` class

Model	**Backbone**	**Epochs**	**mAP@0.5**	**Dominant Class (AP@0.5)**
Faster R-CNN (Custom)	MobileNet-V3 (light)	20	0.00037	Car (0.057)
Faster R-CNN (Pretrained)	ResNet-50 (BDD)	-	0.4607	Person (46.07)

References

[1] Yu, Fisher, Wenqi Xian, Yingying Chen, Fangchen Liu, Mike Liao, Vashisht Madhavan, and Trevor Darrell. *BDD100K: A Diverse Driving Dataset for Heterogeneous Multitask Learning.* Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2020.

[2] Jocher, Glenn, Ayush Chaurasia, and Jing Qiu. *YOLOv11: Real-Time Object Detection and Segmentation.* Ultralytics Technical Report, 2024.

[3] Ren, Shaoqing, Kaiming He, Ross Girshick, and Jian Sun. *Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks.* Advances in Neural Information Processing Systems (NeurIPS), 2015.

[4] Radford, Alec, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, et al. *Learning Transferable Visual Models from Natural Language Supervision.* Proceedings of the International Conference on Machine Learning (ICML), 2021.

[5] Howard, Andrew, Mark Sandler, Grace Chu, Liang-Chieh Chen, Bo Chen, Mingxing Tan, Weijun Wang, Yukun Zhu, Ruoming Pang, Vijay Vasudevan, and Quoc V. Le. *Searching for MobileNetV3.* Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), 2019.

[6] Lin, Tsung-Yi, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr DollÃ¡r, and C. Lawrence Zitnick. *Microsoft COCO: Common Objects in Context.* European Conference on Computer Vision (ECCV), 2014.

[7] Paszke, Adam, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. *PyTorch: An Imperative Style, High-Performance Deep Learning Library.* Advances in Neural Information Processing Systems (NeurIPS), 2019.

[8] Treuille, Adrien, and Thiago Teixeira. *Streamlit: An Open-Source Framework for Interactive Machine Learning and Data Science Applications.* Streamlit Inc., Technical Documentation, 2020.

[9] Merkel, Dirk. *Docker: Lightweight Linux Containers for Consistent Development and Deployment.* Linux Journal, 2014(239):2.