Object Detection and Recognition Report

Shaira Manandhar
02/21/2026

1. Introduction

Object detection is a core task in computer vision that involves both localizing objects and classifying them within an image. Modern detection systems must balance two competing objectives: high detection accuracy and fast inference speed. This trade-off becomes especially important in real-time applications such as autonomous driving, surveillance, and mobile vision systems.

This project implements and compares two small datasets on two object detection models. The primary objective is to evaluate how one-stage and two-stage detectors perform under limited data and GPU constraints (8 GB). Performance is assessed using standard detection metrics, including mAP@0.5, precision, recall, training time, and inference speed.

2. Dataset Description

Two datasets were used in this project to evaluate model performance.

Penn-Fudan Pedestrian Dataset: This dataset contains approximately 170 street images with annotated pedestrian bounding boxes. It is a single-class detection task focused on identifying people in urban scenes. The dataset is relatively clean, with well-defined subjects and limited background clutter.

Oxford-IIIT Pets Dataset (Subset): A subset of cat breeds was used from the Oxford-IIIT Pets dataset. This task is more challenging because it requires fine-grained classification in addition to localization. Images contain significant variation in pose, lighting, and background complexity.

2.1 Data Splits

Both datasets were split into training, validation, and test sets using a fixed random seed to ensure reproducibility.

Dataset	Classes	Total Images	Task	Training	Validation	Testing
Penn-Fudan	1 (person)	170	Detection	118	25	27
Oxford-IIIT Pets (subset)	8 cat breeds	789	Detection + classification	552	118	119

3. Model Explanation

Two lightweight detection architectures were selected for this project. These models were chosen because they are known to run within an 8 GB GPU constraint while still providing competitive performance.

Faster R-CNN MobileNetV3 FPN: This is a two-stage detector that first generates region proposals and then classifies them. It typically provides strong localization accuracy but is computationally heavier.

YOLOv8n: This is a one-stage detector that performs localization and classification in a single forward pass. It is designed for speed and efficiency, making it suitable for real-time applications and limited GPU environments.

4. Training Details

Training was conducted in Google Colab. Standard data loading and augmentation pipelines were used. The same train/validation/test splits were maintained across both models for fair comparison. Evaluation was performed on the held-out test sets using standard COCO-style metrics to ensure fair comparison between models. All images were resized to 512×512 to maintain consistency and fit within the 8 GB GPU constraint.

Setting	Value
Device	NVIDIA T4 GPU
Precision	Mixed precision (FP16)
Optimizer LR	1e-4

Epochs (Pets)	15–20
Early stopping	Enabled
Batch size	Model-dependent

5. Results

Performance was evaluated using *mAP@0.5*, which considers a detection correct when the Intersection over Union (IoU) between predicted and ground-truth boxes exceeds 0.5. Next, *Precision* was used to measure the proportion of predicted detections that are correct, reflecting the model's ability to avoid false positives. *Recall* measured the proportion of ground-truth objects that are successfully detected, indicating the model's sensitivity to missed objects. *Training time* captured the total time required to fine-tune each model and reflected computational cost. Lastly, *Inference speed* measured how quickly the trained model processes new images, reported either as images per second or milliseconds per image, depending on the framework.

5.1 Quantitative Results

Dataset	**Model**	**mAP@0.5**	**Precision**	**Recall**	**Training Time (s)**	**Inference Speed**
Penn-Fudan	Faster R-CNN (MobileNetV3 FPN)	0.9740	0.8451	0.9836	45.18	23.07 img/sec
Penn-Fudan	YOLOv8n	0.9703	0.9713	0.8802	231.34	10.81 ms/ img
Oxford-IIT Pets (subset)	Faster R-CNN (MobileNetV3 FPN)	0.8659	0.7836	0.8824	187.87	35.14 img/sec
Oxford-IIT Pets (subset)	YOLOv8n	0.8768	0.8285	0.8893	1715.03	8.51 ms/img

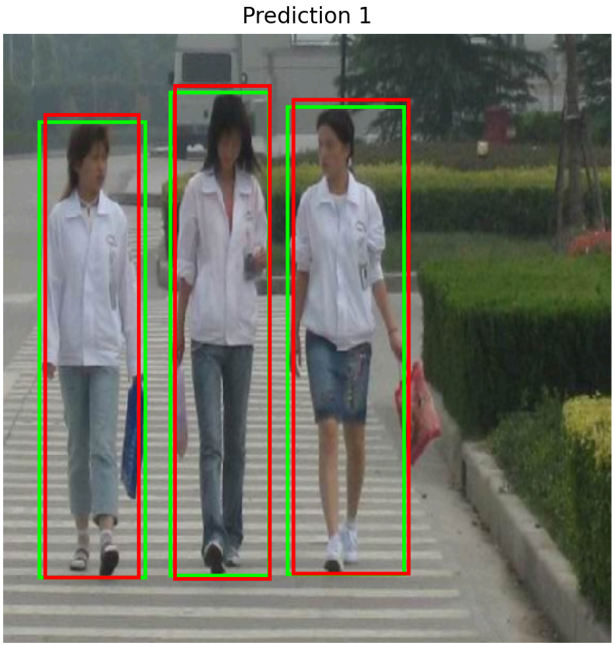
Both datasets are relatively small, which may have caused the reported metrics to slightly overestimate real-world generalization performance.

Both models achieve strong detection performance across the two datasets. Faster R-CNN shows slightly higher mAP@0.5 on both Penn-Fudan (0.9740) and the Oxford-IIIT Pets subset (0.8659), indicating marginally better localization accuracy. In contrast, YOLOv8n demonstrates higher precision and substantially faster inference latency measured in milliseconds per image (10.81 ms/img on Penn-Fudan and 8.51 ms/img on Pets), making it more suitable for real-time applications.

Training time differs notably between the models, with YOLOv8n requiring longer total training on the Pets subset due to its training pipeline and logging methodology. Additionally, Faster R-CNN reports inference throughput in images per second, whereas YOLOv8n reports latency in milliseconds per image, so direct speed comparison should be interpreted with this difference in reporting format in mind.

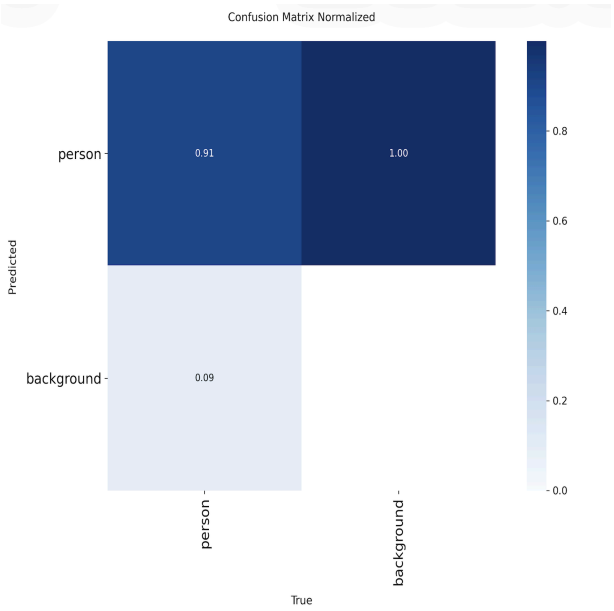
5.2 Quantitative Analysis

Figure 1. Example pedestrian detection on the Penn-Fudan dataset showing predicted and ground-truth bounding boxes.



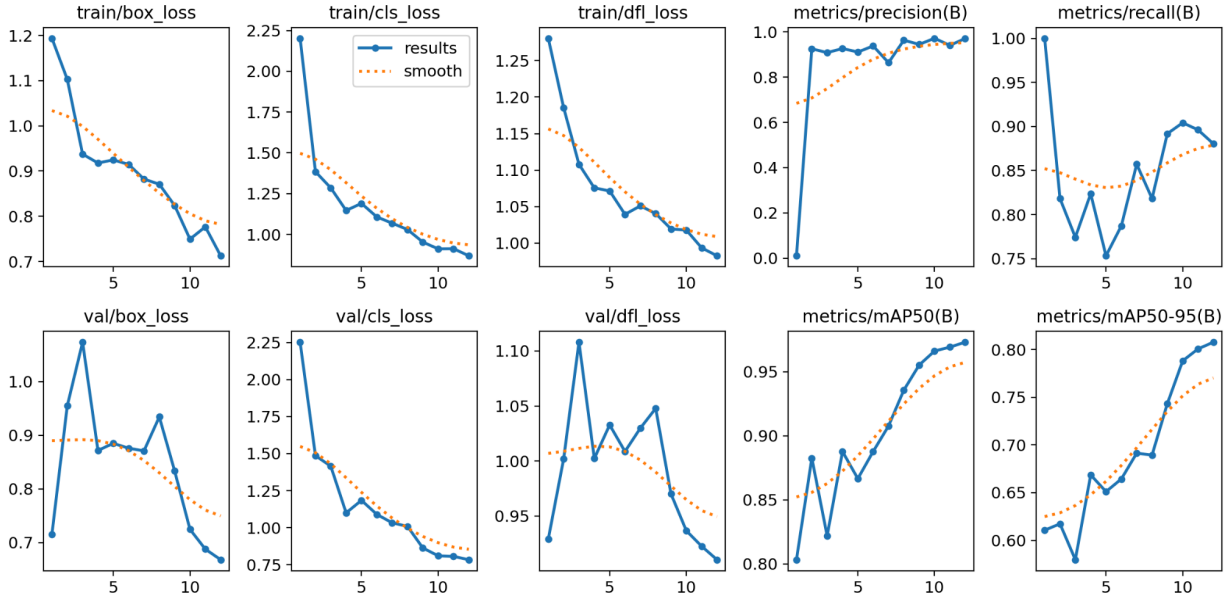
The model successfully detects and localizes pedestrians in the Penn-Fudan dataset with a strong multi-person detection capacity. The predicted bounding boxes closely overlap with the ground-truth annotations, indicating high localization accuracy. All visible individuals are correctly identified with no clear false negatives, and the box alignment suggests the model has learned robust pedestrian features. These qualitative results are consistent with the high recall and mAP values reported in the quantitative evaluation.

Figure 2. Confusion matrix showing person classification performance on PennFudan.



The normalized confusion matrix on the Penn-Fudan dataset shows strong pedestrian recognition, with approximately 91% of person instances correctly detected and minimal misclassification as background. This supports the high recall observed in the quantitative results.

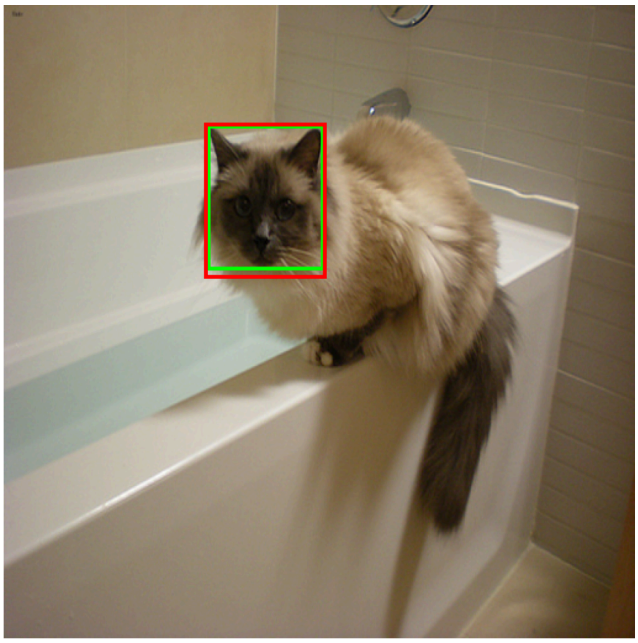
Figure 3. Training and validation performance curves for YOLOv8n across epochs.



The curves indicate stable model convergence, with both training and validation losses decreasing consistently over epochs. Detection metrics (precision, recall, and mAP) improve steadily and plateau toward later epochs, suggesting effective learning without clear signs of overfitting. Minor fluctuations in validation loss are expected, given the small dataset size, but

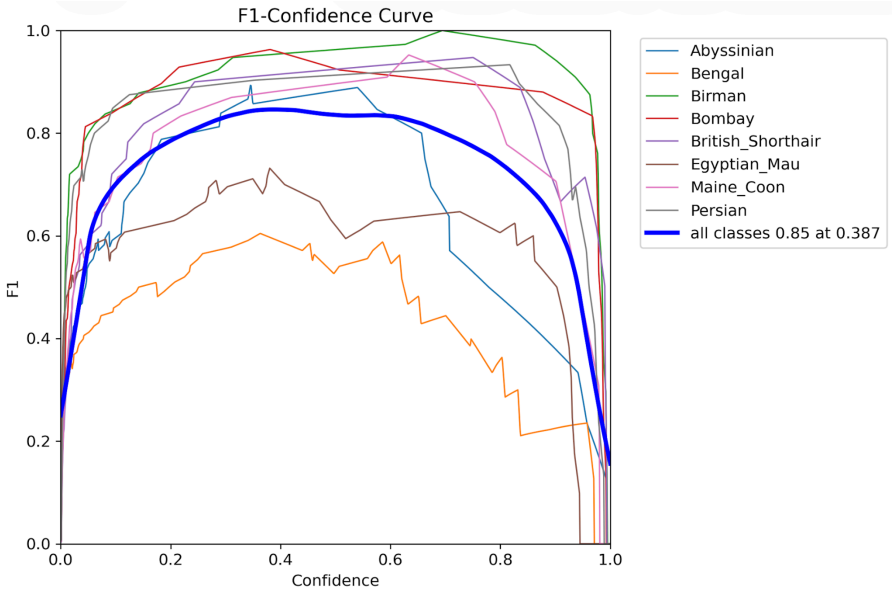
do not indicate instability. The results show that YOLOv8n learns robust object representations and achieves strong detection performance.

Figure 4. Example cat breed detection on the Oxford-IIIT Pets subset.



The model successfully detects the cat in the Pets subset despite variations in pose and background. The predicted bounding box correctly localizes the main facial region of the cat, indicating that the model has learned discriminative breed features. However, the prediction is relatively tight around the face rather than the full body, suggesting a mild localization bias toward the most salient region.

Figure 5. Precision-Recall curve illustrating detection performance across confidence thresholds.



The F1–confidence curve on the Oxford-IIIT Pets subset shows that performance peaks at an F1 score of approximately 0.85 at a confidence threshold near 0.39. Most breeds achieve strong F1 performance across moderate confidence levels, although variability exists across classes, with some breeds (e.g., Bengal and Egyptian Mau) exhibiting lower stability. The aggregated curve indicates that a mid-range confidence threshold provides the best balance between precision and recall for multi-class pet detection.

6. Discussion

The experimental results highlight the trade-off between detection accuracy and computational efficiency. Faster R-CNN consistently achieves slightly higher mAP, with slower inference on both datasets. This confirms the strength of two-stage detectors for precise localization. However, YOLOv8n offers superior inference latency, with heavier training, making it more appropriate for resource-constrained environments.

On the simpler, single-class Penn-Fudan dataset, both models perform near saturation, suggesting the task is relatively easy for modern detectors. Performance differences become more apparent on the Oxford-IIIT Pets subset, where fine-grained classification and background variability increase task difficulty. The YOLOv8n model maintains competitive accuracy while providing faster inference. YOLOv8n also tends to produce fewer but more confident detections (higher precision), whereas Faster R-CNN achieves higher recall by proposing more candidate regions. Minor fluctuations in validation loss are likely due to the small dataset sizes rather than true instability.

7. Future Work

Several improvements could further strengthen this study. First, training on larger and more diverse datasets would provide a more robust evaluation of generalization and reduce variance caused by small sample sizes. Second, additional hyperparameter tuning (e.g., learning rate schedules, augmentation strength, and anchor settings) could further optimize both models' performance. Third, evaluating more model variants such as larger YOLOv8 versions or alternative lightweight backbones would provide a broader comparison of accuracy–efficiency trade-offs. Lastly, measuring end-to-end deployment metrics (GPU memory usage, real-time FPS on edge devices, and cross-dataset generalization) would better reflect practical performance in real-world computer vision systems.

8. Conclusion

This project helped me better understand the real-world trade-offs between two-stage and one-stage object detectors when working under limited GPU resources. I observed that Faster R-CNN provides slightly more precise localization, while YOLOv8n achieves much faster inference with comparable accuracy. The experiments also showed how dataset difficulty affects model behavior, with clearer performance differences emerging on the more complex multi-class Pets task. The main takeaway is that the “best” model depends on the deployment goal. Accuracy alone is not sufficient, and efficiency, speed, and task complexity must also guide model selection.