**Intercountnet - intersection-vehicle-counting-using-**

**computer-vision-networks AI Models**

Swathi Subramanyam Pabbathi

Shiley-Marcos School of Engineering, University of San Diego

## AAI-521-03: Applied Computer Vision for AI

Dr. Saeed Sardari and Dr. Roozbeh Sadeghian

December 03rd, 2025

**Contents**

[**Abstract** 3](#_heading=h.gjdgxs)

[**Introduction and Dataset Overview**](#_heading=h.30j0zll) 4

[**Data Cleaning and Preparation**](#_heading=h.1ksv4uv) 5

[**Exploratory Data Analysis (EDA)**](#_heading=h.44sinio) 6

[**Model Selection**](#_heading=h.2jxsxqh) 7

[**Model Training and Evaluation**](#_heading=h.z337ya) 8

[**Results**](#_heading=h.3j2qqm3) 12

[**Key Findings and Discussion** 1](#_heading=h.1y810tw)6

[**Build versus Buy Analysis** 13](#_heading=h.4i7ojhp)

[**Conclusion and Recommendations** 1](#_heading=h.2xcytpi)7

**References** 20

[**Appendix** 21](#_heading=h.3whwml4)

## 

## **Abstract**

The objective of this project is to develop an automated, computer-vision-based system for multi-class vehicle detection and counting at urban intersections. As cities grow denser, manual traffic studies and traditional sensor-based monitoring fail to deliver the accuracy, scalability, and real-time analytics required for effective transportation planning. This project leverages a dataset of over 17,000 annotated intersection traffic images to build InterCountNet an AI-powered vehicle counting pipeline. Three state-of-the-art deep learning models were trained and evaluated: **YOLOv8, Faster R-CNN, and DETR**, using transfer learning and data augmentations to ensure robustness under occlusion, varying lighting, and dense traffic conditions. Model performance was compared using metrics such as **mAP, precision, recall,** and inference speed, along with qualitative evaluation of detection consistency across challenging scenes. Among the tested models, **YOLOv8n** demonstrated the best balance of accuracy and real-time performance, making it the most suitable for deployment in city-scale traffic analytics systems. The resulting InterCountNet framework enables automatic extraction of per-class vehicle counts (car, bus, truck, motorcycle, bicycle) and produces structured insights that support traffic engineering, congestion mitigation, and smart city decision-making. This work highlights the effectiveness of modern deep learning models in transforming traditional traffic management into a data-driven, camera-only solution.

## **Introduction**

In this project, we aim to develop and implement an AI-driven computer vision system for automated vehicle detection, classification, and counting at urban intersections. As cities expand and traffic volumes increase, traditional monitoring techniques such as manual counting, induction loops, and radar sensors struggle to provide accurate, real-time, and scalable insights. InterCountNet leverages deep learning based object detection to overcome these limitations and deliver reliable traffic analytics that support data-driven transportation management.

InterCountNet fits within the smart mobility and smart city ecosystems. In the context of smart mobility, it enables optimization of signal timings, reduction of congestion, and enhanced road safety through consistent, high-resolution traffic insights. From a broader smart city perspective, the system supports infrastructure planning, resource allocation, and policy development by providing precise vehicle-class statistics across diverse traffic environments.

To achieve these objectives, we trained and evaluated three state-of-the-art deep learning models YOLOv8, Faster R-CNN, and DETR on a dataset of more than 17,000 annotated intersection images. This comparative analysis highlights the strengths and trade-offs of each architecture, addressing real-world considerations such as detection accuracy, inference speed, and robustness under occlusion and heavy traffic density. The evaluation demonstrates that YOLOv8n offers the best balance of precision and real-time performance, making it the most suitable candidate for deployment in operational traffic monitoring systems.

This project also explores build-versus-buy considerations that organizations must evaluate when choosing between traditional sensor-based systems and advanced deep learning based solutions. By implementing a scalable, camera-only approach, InterCountNet provides a cost-effective and high-performance alternative to expensive physical sensors, empowering city agencies with actionable insights for efficient and sustainable traffic management.

## **Dataset Overview**

The InterCountNet project utilizes the **Intersection Traffic Dataset (Version 10)** sourced from Roboflow Universe. This dataset contains real-world traffic images captured at busy urban intersections from various camera angles, lighting conditions, and weather scenarios. It is specifically designed for multi-class vehicle detection tasks, making it ideal for training and benchmarking object detection models such as YOLOv8, Faster R-CNN, and DETR.

The dataset includes **17,000+ labeled images**, each annotated with bounding boxes for five major vehicle categories: Car, Bus, Bicycle, License Plate, Motorcycle

Annotations are provided in **YOLO format**, with each label file containing class ID, normalized bounding box center coordinates (x, y), and box dimensions (width, height). The dataset ensures diversity in traffic density, camera viewpoints, occlusions, and scale variations—factors critical for building robust vehicle detection models.

To prepare the data for training, the images are split into **train, validation, and test** subsets. The dataset supports common object-detection formats and integrates smoothly with PyTorch, TorchVision, and YOLO training pipelines.

This rich and varied dataset enables effective learning, model comparison, and reliable evaluation of detection performance across multiple deep learning architectures.

## **Link:** <https://universe.roboflow.com/machine-learning-class-eiri5/intersection-traffic-piimy/dataset/10>

## **Data Cleaning and Preparation**

The dataset underwent a rigorous cleaning and preprocessing workflow to ensure high-quality training data for the object detection models. Since object detection performance is highly sensitive to label accuracy, each image annotation pair was carefully validated using custom Python scripts and visualization tools.

### ● Handling Corrupted and Invalid Files

All images and label files were scanned to identify corrupted entries. Any files that failed to load, contained unreadable pixel data, or had mismatched label files were removed from the dataset. This step ensured that only valid image annotation pairs were used for training and evaluation.

### ● Annotation Validation and Label Correction

The YOLO-format labels were parsed and checked for structural correctness. Several issues—such as labels with missing values, out-of-range bounding box coordinates, or negative widths/heights were identified and corrected or removed. Bounding boxes were clamped to image boundaries to avoid training instabilities caused by invalid annotations.

### ● Removing Images Without Objects

### Some images in the dataset contained **no annotated vehicles** (empty .txt files). Since the goal of the project is multi-class vehicle detection and counting, such images were excluded from the training set. This prevented class imbalance issues and ensured that each training sample contributed useful learning signals.

### ● Ensuring Annotation-Image Consistency

Custom scripts checked that every image had a corresponding annotation file and vice versa. Orphaned images or labels were removed to maintain dataset integrity. Additionally, class IDs in annotations were cross-verified with the dataset’s defined class mapping (car, bus, truck, motorcycle, bicycle).

### ● Train-Validation-Test Split

The cleaned dataset was split into **training, validation, and test** subsets to ensure balanced class representation across all sets. This enabled reliable model evaluation and consistent comparison between YOLOv8, Faster R-CNN, and DETR.

## **Exploratory Data Analysis (EDA)**

Exploratory Data Analysis (EDA) was conducted to gain insights into the visual characteristics, annotation quality, and class distribution within the intersection traffic dataset. The analysis included visualizing random images, overlaying annotations, and examining class frequencies to assess dataset balance and reliability.

### Key Findings:

● **Images Without Annotations:** Several images contained visible vehicles but had *no annotations*, indicating missing or incomplete labels. These were flagged for removal during data cleaning.

● **Brightness & Lighting Variations:** The dataset includes images with significant differences in brightness bright daylight, shadows, dusk lighting which highlights the need for brightness/contrast augmentation.

● **Annotation Quality Issues:** Some bounding boxes were misaligned, too small, extended outside image boundaries, or entirely missing for distant vehicles. These inconsistencies required manual or scripted correction.

● **Class Imbalance:** A strong imbalance was observed **cars and motorcycles** dominate the dataset, while **buses and bicycles** appear in much smaller numbers.

● **Missing License Plate Class:** Although the dataset defines *license plate* as a class, EDA confirmed **zero annotations** for this category, making it unusable for model training.

● **Scene Diversity:** Images vary widely in perspective (side-view, aerial, overhead), traffic density, and scale, offering good variability for training robust detection models.

### Insights:

● **Vehicle Count Distribution:** The dataset is heavily skewed toward high-frequency classes (car and motorcycle). This imbalance influences model learning and motivated augmentation strategies to improve detection of minority classes.

● **Annotation Presence & Quality:** Visualization of bounding boxes revealed inconsistencies that affected downstream tasks. Cleaning invalid labels and removing empty-annotation images improved dataset reliability significantly.

● **Lighting & Environmental Variability:** The diverse exposure levels across images indicate that models must learn to detect vehicles under both bright and low-light conditions. Brightness augmentation and contrast normalization are therefore essential.

● **Dataset Suitability for Object Detection:** After removing corrupted and unlabeled images, the cleaned dataset provides strong class diversity (excluding license plates), good scene variability, and a sufficient number of bounding boxes for training YOLOv8, Faster R-CNN, and DETR.

## **Model Selection**

## AI Algorithms and Model Selection

I experimented with three state-of-the-art object detection architectures **YOLOv8**, **Faster R-CNN**, and **DETR** to identify the most effective model for multi-class vehicle detection and counting at urban intersections. Faster R-CNN served as a strong two-stage baseline, offering high accuracy through region proposal networks, but it showed slower inference speed and reduced real-time suitability. DETR, a transformer-based detection framework, demonstrated strong performance on complex scenes but required longer training, struggled with small objects, and was sensitive to annotation inconsistencies. In contrast, **YOLOv8**, a modern one-stage detector optimized for real-time applications, delivered superior performance by balancing accuracy, speed, and robustness. It handled varying lighting conditions, occlusions, and dense traffic scenarios more effectively than the other models. Empirical results showed YOLOv8 achieving higher mAP scores and significantly faster inference times, validating its suitability for a scalable, real-time intersection traffic counting system.

## Model Selection and Optimization Strategy

After comparing all three architectures, **YOLOv8n** was finalized as the optimal model for InterCountNet. While Faster R-CNN offers strong accuracy, its computational overhead makes it less practical for continuous traffic monitoring. DETR provides a transformer-based alternative capable of capturing global context but struggles with small objects and requires extensive training resources. YOLOv8n demonstrated the best trade-off between detection accuracy, model size, and realtime performance critical factors for deployment in smart city environments. To further enhance performance, we applied targeted data augmentations (brightness/contrast adjustments, blurring, affine transforms) to improve robustness across diverse lighting and camera perspectives. Hyperparameters such as confidence threshold, IoU threshold, and learning rate were tuned to stabilize training and reduce false detections. This iterative refinement strategy ensured that YOLOv8n delivered reliable per-class vehicle counts while maintaining deployment-grade inference speed, establishing it as the most practical and effective solution for automated intersection traffic analytics.

## **Model Training and Evaluation**

The model training and evaluation phase involved experimenting with three state-of-the-art deep learning architectures **YOLOv8l**, **Faster R-CNN**, and **DETR** to determine the most effective framework for multi-class vehicle detection and counting. Each model was trained using carefully tuned hyperparameters, evaluated on both training and validation datasets, and analyzed using loss curves, IoU curves, and mAP metrics.

## YOLOv8l Training and Performance

The YOLOv8l model was trained using a combination of **AdamW** optimizer and an extensive augmentation pipeline designed to improve robustness under varying lighting, occlusions, and small-object scenarios. Key hyperparameters included a learning rate of **0.001**, final learning rate of **0.01**, image size of **1024**, and mosaic, HSV, scaling, and translation augmentations.

### Training Insights

* **Loss Curves:** The box loss, classification loss, and distribution focal loss showed a consistent downward trend across 20 epochs, indicating stable optimization.
* **Precision & Recall:** Precision fluctuated initially but stabilized around ~0.78, while recall increased steadily to ~0.84 by epoch 20.
* **mAP Performance:**
  + **mAP50** improved continuously and reached high performance (>0.80) by epoch 20.
  + **mAP50–95**, a stricter metric, also increased consistently, ending above **0.65**.

These results demonstrate that YOLOv8l learned effectively despite class imbalance and annotation inconsistencies. Its fast convergence, strong recall, and high mAP underscore its suitability for real-time intersection vehicle counting.

## Faster R-CNN Training and Performance

Faster R-CNN was trained using the **SGD optimizer** with a conservative learning rate (**0.0001**) and momentum (**0.9**). This configuration prioritizes stability over speed, which is essential for two-stage detectors.

Training Insights:

* **Loss Curve:** The model exhibited a steady decline in training loss across 45 epochs, while the validation loss stabilized midway, indicating controlled learning with minimal overfitting.
* **IoU Curve:** IoU increased rapidly in early epochs and plateaued around **0.75 (val)** and **0.80 (train)**.
* **mAP Curve:** Validation mAP50–95 improved consistently but plateaued below YOLOv8, indicating slower convergence and limitations in detecting small or distant vehicles.

Faster R-CNN demonstrated strong foundational accuracy but fell short in inference speed and flexibility, making it less suitable for real-time deployment.

## DETR Training and Performance

The DETR transformer model was trained using **AdamW** with an extremely low learning rate (**2e-5**), consistent with transformer stability requirements.

### Training Insights

* **Loss Behavior:** DETR showed a gradual decrease in both training and validation loss, characteristic of transformer-based models that require long training periods.
* **Performance Trend:** Despite stable optimization, DETR underperformed on small and distant vehicles (common in intersection scenes) and required significantly longer training to reach competitive accuracy levels.

While DETR captured global context well, its slow convergence and sensitivity to small-object annotations made it less effective for this dataset compared to YOLOv8 and Faster R-CNN.

Comparision Evaluation

|  |  |  |  |
| --- | --- | --- | --- |
| **Model** | **mAP50–95** | **Strengths** | **Limitations** |
| YOLOv8l | Highest | Fast, accurate, robust to lighting, strong on small objects | Initial precision fluctuations |
| Faster R-CNN | Moderate | Strong accuracy, stable training | Slower inference, less adaptive to real-time needs |
| DETR | Lowest | Excellent global context via transformer architecture | Slow training, struggles with small/distant objects |

|  |
| --- |
| **Train and Validation Loss Curves for YoLo8l** |
| A group of graphs showing loss  AI-generated content may be incorrect. |

Fig1: Train and Validation Loss Curves for YOLO8l

|  |  |
| --- | --- |
| **Training and Validation Loss curves for Faster R-CNN** | **Training and Validation Loss curves for DETR** |
| A graph of loss curve  AI-generated content may be incorrect. | A graph of a line graph  AI-generated content may be incorrect. |

Fig2: train and validation loss curves for Faster RCNN and DETR

## **Results**

The evaluation of the three object detection models YOLOv8l, Faster R-CNN, and DETR showed clear differences in accuracy, robustness, and real-time suitability. YOLOv8l delivered the strongest overall performance, consistently achieving the highest mAP50–95 and recall scores across the validation set. The model demonstrated the ability to accurately detect vehicles of varying sizes, lighting conditions, and occlusions, with smooth and steadily decreasing loss curves indicating strong generalization. Faster R-CNN performed moderately well, capturing large and medium-sized vehicles reliably but exhibiting slower inference times and lower mAP scores compared to YOLOv8. DETR, while effective at capturing global scene context, struggled with small and distant objects and required significantly longer training to converge. Visual inspection of detection outputs confirmed that YOLOv8 produced the most precise bounding boxes and the most accurate per-class vehicle counts, with minimal missed detections. However, occasional precision fluctuations were observed in early epochs, suggesting that further tuning could improve stability. Overall, the results strongly validate YOLOv8l as the most accurate and deployment-ready model for automated intersection vehicle counting in real-world smart city contexts.

|  |  |
| --- | --- |
| MAP metric for YOLO8l model | Confusion matrix in evaluations for YOLO8l Model |
| A comparison of a graph  AI-generated content may be incorrect. |  |
| Precision-Recall Curve | F1-Confidence Curve |
|  |  |

Fig3: Evaluation results

compare the results actual values and predicted values from the above plots in fig: **Evaluation results**

**Observations**

## Detection Performance Overview

● The YOLOv8 model demonstrated strong multi-class detection capability, with consistently high precision and recall for the major classes—car, bus, and motorcycle.

● The model accurately detected small and dense objects in highly congested scenes, as shown in the qualitative prediction outputs.

● Minor misclassifications occurred primarily between visually similar categories (e.g., small cars vs. motorcycles) in heavy traffic.

## Confusion Matrix Insights

● Cars and motorcycles exhibited the highest correct predictions, with **11,347** correct car detections and **35,640** correct motorcycle detections.

● Buses were also detected well, with strong separation from other classes and minimal cross-class confusion.

● Bicycles showed moderate performance, with some confusion against background due to their smaller size and lower representation in the dataset.

● **License-plate detection was weak** (very few correct predictions), which aligns with earlier EDA findings that license plates were rarely annotated in the dataset.

## Precision–Recall Curve Observations

● Cars, buses, and motorcycles achieved excellent PR curves, with AUC values around **0.94–0.98**, demonstrating strong discrimination ability.

● Bicycles reached a precision of **0.846**, reflecting acceptable but lower performance due to small object size and class imbalance.

● License plates displayed very poor PR curve behavior (0.375 AP), confirming insufficient training signals.

● The overall model PR curve achieved **mAP@0.5 = 0.824**, indicating high detection reliability across all object classes.

## F1–Confidence Curve Insights

● Optimal global F1 score of **0.79** was achieved at a confidence threshold of **0.357**, balancing precision and recall.

● Cars, buses, and motorcycles maintained high F1 scores (>0.90) across a wide confidence range, showing strong robustness.

● Bicycles showed moderate F1 behavior, reflecting sensitivity to lighting, occlusion, and distance.

● License plate curves again showed instability and low F1 values due to missing annotations.

## Class-wise Performance Metrics

● **Car:** Highest performance overall, with **P = 0.937**, **R = 0.948**, **mAP50 = 0.978**, **mAP50–95 = 0.851**.

● **Motorcycle:** Also very strong, with **mAP50 = 0.973** and **mAP50–95 = 0.757**, aided by high instance count (37,919).

● **Bus:** Performed exceptionally well with **mAP50 = 0.938** and **mAP50–95 = 0.842**, despite fewer training examples.

● **Bicycle:** Moderate detection capability, with **mAP50–95 = 0.703**, impacted by class imbalance and small-object difficulty.

● **License-plate:** Weakest performance (mAP50–95 = 0.242), consistent with limited ground-truth labels.

## Model Strengths

● Excellent generalization across major traffic classes, especially in dense intersection scenes.

● High recall (**0.852 overall**) indicates the model rarely misses major objects of interest.

● Strong performance on small and partially occluded vehicles due to 1024×1024 training resolution and augmentations.

● Fast inference (~5.3 ms per image) supports real-time deployment on video feeds.

## Model Limitations

● Underperformance on license-plate class caused by insufficient labeled samples.

● Moderate confusion for bicycles, particularly in night scenes or low-contrast environments.

● Slight early-epoch precision fluctuations, though they stabilized with training.

## Qualitative Evaluation

● Visual inspection of detection outputs shows highly accurate bounding boxes for cars and motorcycles, even in cluttered nighttime environments.

● The model captures object boundaries well, with minimal false positives in background regions.

● Detected vehicle counts are consistent with human annotations, supporting reliable downstream analytics like traffic volume assessment

**Key Findings and Discussion**

* **YOLOv8 delivered the best overall performance**, achieving the highest mAP and recall across major classes and proving most suitable for real-time vehicle detection.
* **Class imbalance strongly influenced results**, with high accuracy for cars and motorcycles but weaker performance for bicycles and license plates due to limited annotations.
* **The model demonstrated strong robustness**, accurately detecting vehicles in dense, occluded, and low-light intersection scenes with minimal false positives.
* **Error analysis revealed improvement areas**, including better handling of small objects and addressing missing or inconsistent annotations for underrepresented classes.

**Build vs. Buy Analysis**

1. **Building a Custom Detection Model Enables Tailored Performance**

Developing custom models such as YOLOv8, Faster R-CNN, and DETR allows full control over training data, class definitions, and performance optimization. YOLOv8, in particular, can be fine-tuned to intersection-specific traffic patterns, camera angles, and lighting conditions—capabilities that generic, off-the-shelf solutions cannot guarantee.

1. **Off-the-Shelf Solutions Lack Flexibility and Granular Control**

Commercial traffic analytics platforms often provide fixed detection classes, limited customization, and proprietary black-box models. They typically cannot adapt to unique requirements like detecting motorcycles in dense urban traffic, adjusting for imbalanced datasets, or improving performance through annotation corrections areas where YOLOv8 clearly excelled during experimentation.

1. **Cost Efficiency Favors a Build Approach When Dataset & Expertise Exist**

Once the dataset is prepared and domain expertise is available, training YOLOv8 or Faster R-CNN incurs minimal operational cost compared to subscription-based commercial systems. The inference speed achieved (5–6 ms per frame) enables real-time deployment on edge devices without recurring licensing fees. This makes a build approach more cost-effective for large-scale or long-term implementation.

1. **Buying May Still Be Beneficial for Rapid Deployment or Maintenance Support**

Purchased solutions reduce engineering complexity and offer vendor-managed maintenance, model updates, and integration support. However, the evaluation of DETR and Faster R-CNN highlights that more complex models require substantial compute resources and expertise factors that commercial platforms abstract away. Buying may be preferred when organizations lack ML infrastructure or need fast deployment without model training cycles.

## **Conclusion and Recommendations**

# **Conclusions**

1. **YOLOv8 Outperforms Other Models for Real-Time Detection**

The comparative analysis demonstrated that YOLOv8 consistently achieved the highest mAP50–95, precision, and recall across major vehicle classes. Its ability to detect small, occluded, and high-speed vehicles makes it the most suitable deep learning model for real-time multi-class traffic analytics at intersections.

1. **Model Architecture Matters for Practical Deployment**

While Faster R-CNN delivered strong accuracy in controlled conditions, and DETR offered rich global context through transformer attention, both were significantly slower and computationally heavier than YOLOv8. This reinforces that one-stage detectors remain the best choice for real-time operational environments.

1. **Dataset Quality and Annotation Consistency Are Critical**

Performance variation across classes especially the weak results for license plates and moderate results for bicycles highlighted bounding boxes, and complete annotations. High-quality datasets are essential for reliable model generalization in computer vision applications.

1. **Deep Learning Alone Has Limitations for Tiny and Distant Objects**

The project revealed that tiny objects (like distant bicycles or license plates) remain challenging due to scale variation and annotation scarcity. In real-world commercial systems, companies often rely on **RADAR and LiDAR sensors** to capture 3D spatial information, which is then converted into 2D representations or fused with camera data for training. This hybrid approach significantly improves the detection of distant or very small objects compared to camera-only solutions.

1. **Scalability and Real-Time Processing Are Feasible with YOLOv8**

YOLOv8’s lightweight architecture and fast inference speed (≈5–6 ms per frame) make it suitable for deployment on edge devices or real-time traffic control centers. Its performance validates that deep learning-based vision systems can effectively support urban traffic monitoring and smart city infrastructure when optimized properly.the importance of class balance, accurate

# **Recommendations**

1. **Adopt YOLOv8 for Real-Time Traffic Analytics**

Given its superior detection accuracy, speed, and scalability, YOLOv8 should be the primary model for intersection-level vehicle counting and monitoring in production environments.

1. **Incorporate Sensor Fusion for Advanced Detection Needs**

For scenarios requiring accurate detection of **tiny, distant, or partially occluded objects**, organizations should integrate **LiDAR, RADAR, or depth sensors**. These systems allow the generation of 3D point clouds that can be converted into high-quality 2D training samples, significantly enhancing model reliability.

1. **Improve Dataset Quality Through Re-Annotation and Balancing**

To further enhance performance, particularly for minority classes like bicycles and license plates, it is recommended to perform additional dataset balancing, re-annotation, and targeted small-object augmentation.

1. **Implement Continuous Training and Model Monitoring**

Deploying an automated retraining pipeline ensures the model adapts to new camera angles, seasonal changes, traffic patterns, and environmental variations. Continuous evaluation will help maintain accuracy as real-world conditions evolve.

1. **Evaluate Integration with Smart City Infrastructure**

The system should be integrated with traffic signal control platforms, transportation dashboards, and city analytics pipelines. Real-time vehicle counts and class-level statistics can support optimized traffic flow, congestion reduction, and long-term infrastructure planning.

These conclusions and recommendations highlight the importance of advanced object detection models in modern traffic analytics, enabling precise vehicle counting and supporting data-driven decision-making for safer, more efficient smart city infrastructure

**References**

Bochkovskiy, A., Wang, C.-Y., & Liao, H.-Y. M. (2020). *YOLOv4: Optimal speed and accuracy of object detection.* arXiv. <https://arxiv.org/abs/2004.10934>

Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., & Zagoruyko, S. (2020). *End-to-end object detection with transformers.* In *European Conference on Computer Vision (ECCV)*. <https://arxiv.org/abs/2005.12872>

Lin, T.-Y., Goyal, P., Girshick, R., He, K., & Dollár, P. (2017). *Focal loss for dense object detection.* In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. <https://arxiv.org/abs/1708.02002>

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., … & Chintala, S. (2019). *PyTorch: An imperative style, high-performance deep learning library.* In *Advances in Neural Information Processing Systems (NeurIPS)*. <https://arxiv.org/abs/1912.01703>

Redmon, J., & Farhadi, A. (2018). *YOLOv3: An incremental improvement.* arXiv. <https://arxiv.org/abs/1804.02767>

Ren, S., He, K., Girshick, R., & Sun, J. (2015). *Faster R-CNN: Towards real-time object detection with region proposal networks.* In *Advances in Neural Information Processing Systems (NIPS)*. <https://arxiv.org/abs/1506.01497>

Ultralytics. (2023). *Ultralytics YOLOv8 documentation.* Ultralytics. <https://docs.ultralytics.com/>

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). *Attention is all you need.* In *Advances in Neural Information Processing Systems (NeurIPS)*. [https://arxiv.org/abs/1706.03762](https://docs.ultralytics.com/)

Wang, C.-Y., Bochkovskiy, A., & Liao, H.-Y. M. (2022). *YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors.* arXiv. <https://arxiv.org/abs/2207.02696>

Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., … & Zheng, X. (2016). *TensorFlow: A system for large-scale machine learning.* In *12th USENIX Symposium on Operating Systems Design and Implementation (OSDI)*. <https://www.tensorflow.org/>

TensorFlow. (2023). *TensorFlow API documentation.* TensorFlow. <https://www.tensorflow.org/api_docs>

PyTorch. (2023). *PyTorch documentation.* PyTorch. <https://pytorch.org/docs/>

## **Appendix**

**Team Contributors:** I contributed to data cleaning, model training and evaluation,deployment and project reporting and presentations:

• Swathi Subramanyam Pabbathi

**GitHub Repository:** The complete code for this project, including cleaned datasets and model scripts is available at: GitHub Repository

**Content:**

* **Code repository:** <https://github.com/PSswathi/intercountnet-intersection-vehicle-counting-using-computer-vision>