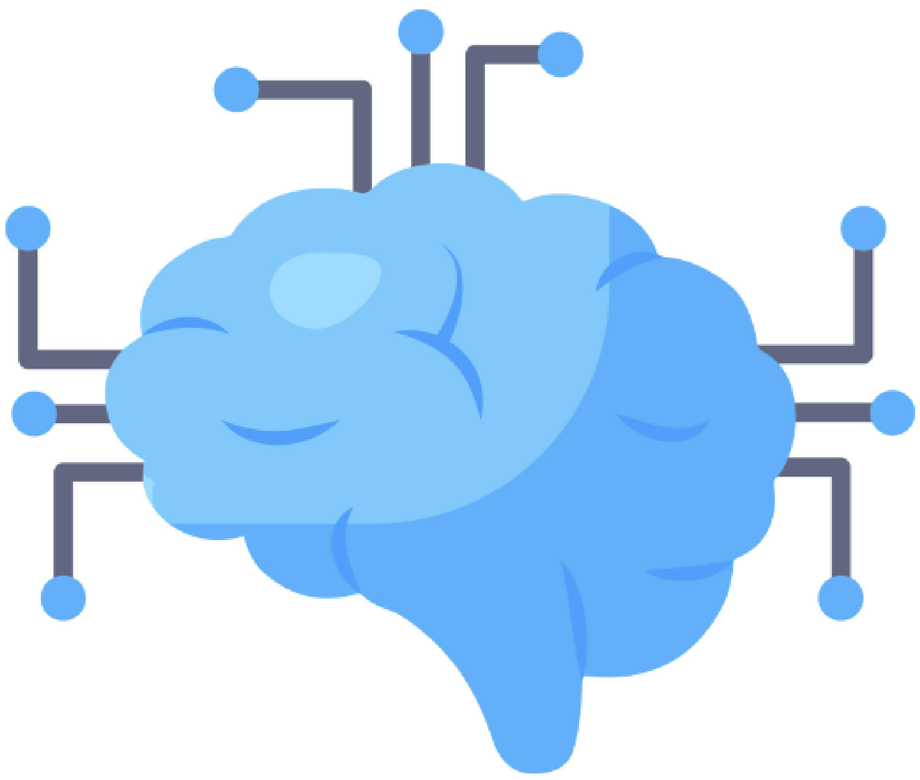


CAR DETECTION & CLASSIFICATION

Technical Test Submission – PNU ISLAB AI 2025

Stefanus Yudi Irwan – June 2025

Table of Content

1 Objective, Output & Constraint

Objective

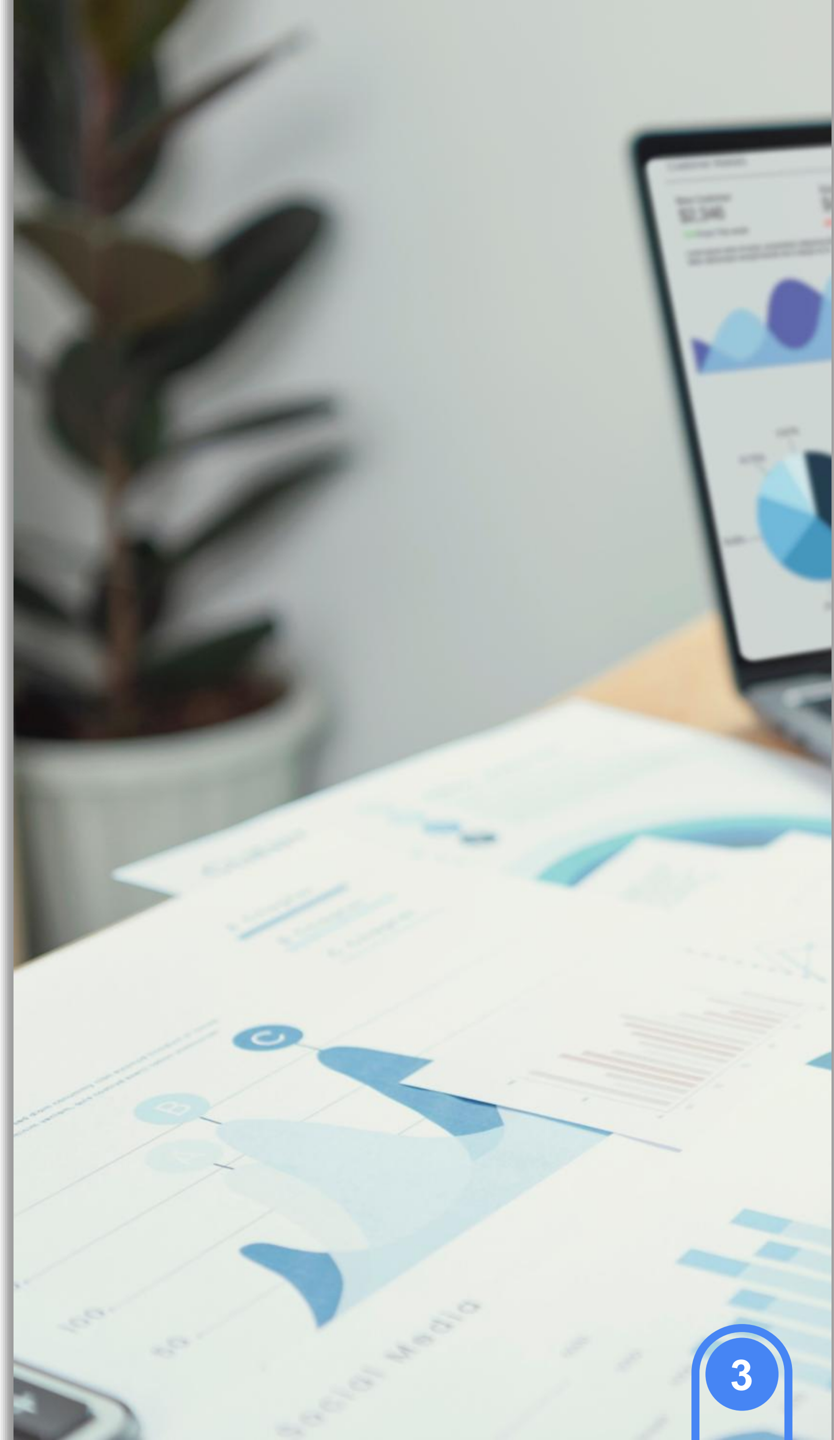
- The goal of this project is to develop a Car Retrieval System capable of detecting multiple cars in a scene and accurately classifying them into vehicle types with a specific focus on Indonesian vehicles.

Output

- The system produces two outputs: (1) bounding boxes from an object detection model that identify cars in an input image or video, and (2) a car classification output that labels each detected car based on its category.

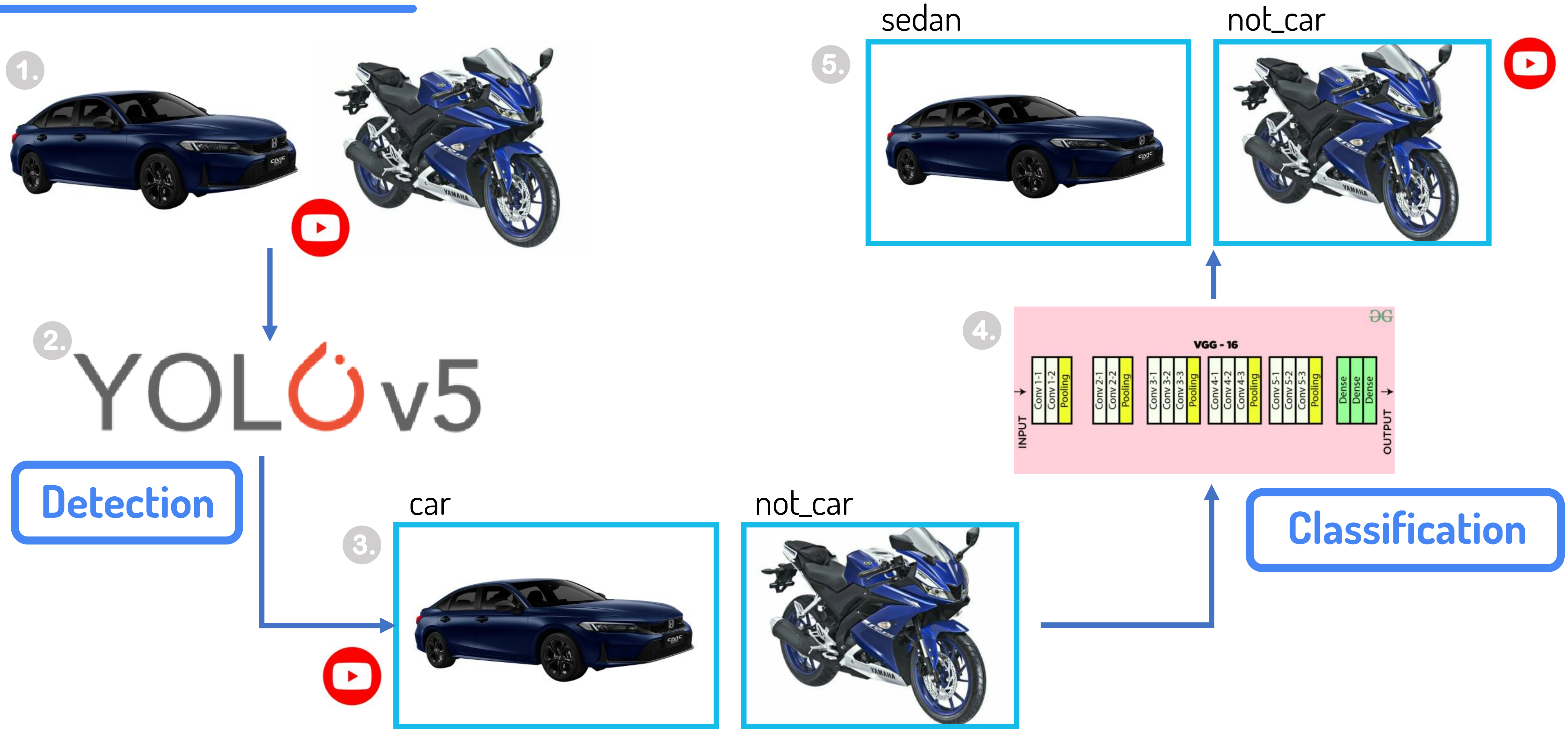
Constraint

- The detection and classification models must be developed separately, not merged into a single model. The classifier must be trained using TensorFlow or PyTorch and should support at least 8 different Indonesian car types.



② Solution Architecture

Solution Architecture



- This solution uses **YOLOv5** to detect objects in video frames, isolating potential cars from the scene. The cropped objects are then passed to a **VGG16** classifier, which distinguishes between cars (e.g., sedans) and non-car objects (e.g., motorcycles), enabling fine-grained classification after detection.

③ Data Collection & Preprocessing

Data Collection

Detection

roboflow

- For detection model dataset is collected and marked using Roboflow website, annotating between car and not car

Image Data

Label:
- Car
- Not Car

Composition:
- Train: 1822
- Valid: 604
- Test: 603

Classification

Bing

- For classification model, dataset is collected using Microsoft Bing Image Crawler library in Python to get sample data for 10 classification of car type

Image Data

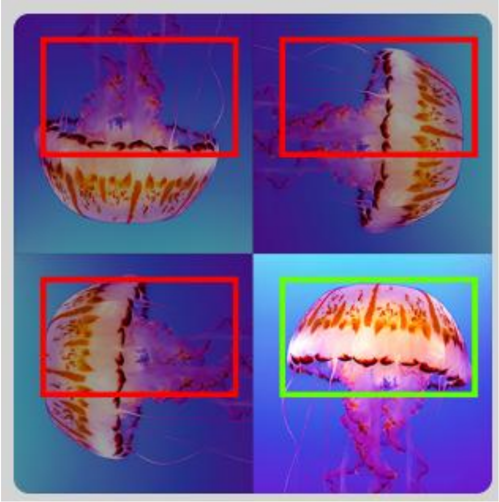
Label:
- Bajaj
- Double Cabin
- Jeep
- Bus
- Hatchback
- Minivan
- MPV
- Pickup
- Sedan
- SUV
- Truck

Composition:
- Train: 1177
- Valid: 154
- Test: 171

Data Preprocessing & Augmentation : Detection

preprocessing & augmentation features of Roboflow is used to enrich the data for detection model: as follows

1 Auto Orient



2 Flip (Horizontal)

3 Rotation (between -10 and 10 deg)

4 Resize

Data Preprocessing & Augmentation : Detection

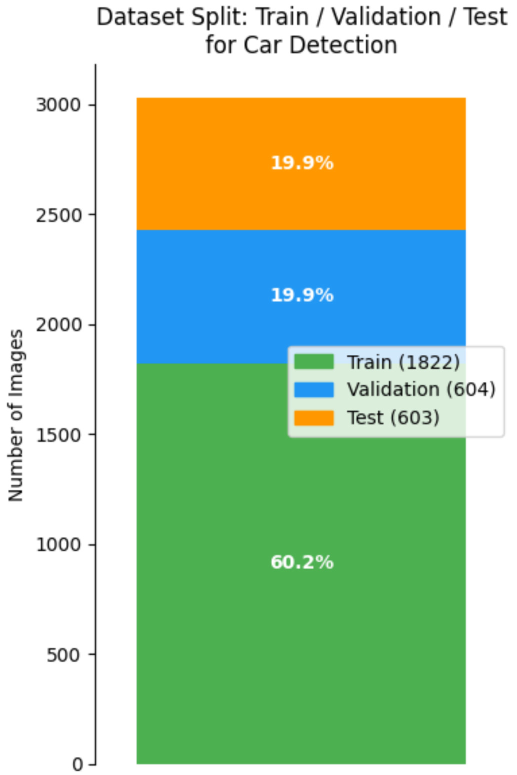
preprocessing & augmentation features of Roboflow is used to enrich the data for detection model:

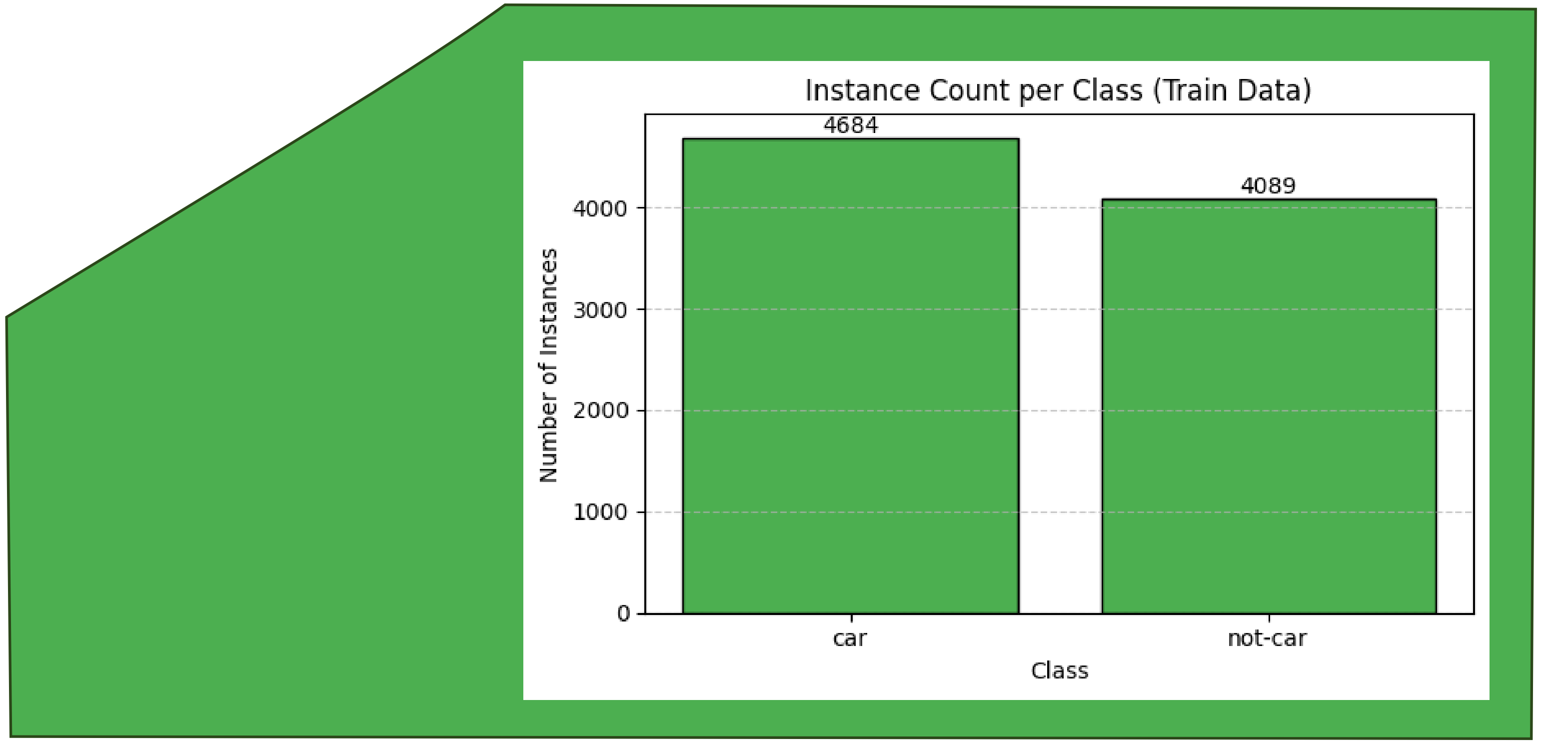
5 Noise (1.01% of pixels)

6 Brightness (between -15% and 15%)

7 Blur (up to 2.5 px)

EDA: Detection





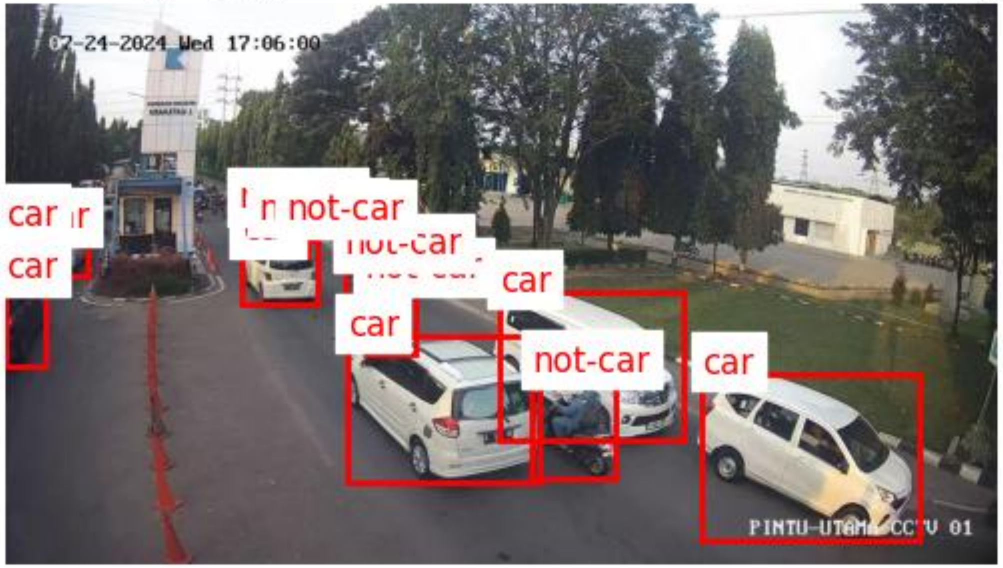
- For car detection model total of image used to train, validate and test data is around 3000 images, with composisition of training data around 60%, and 20% for respective validation and test data. Within the training data itself contains instance of 4684 car and 4089 not car.

EDA: Detection

This is example of training data which will be used to train car detector, showing box to identify car object and not-car object

File: 24052_truk_jpg.rf.b5626c018f28a4190db86ad6d2556a7e.jpg

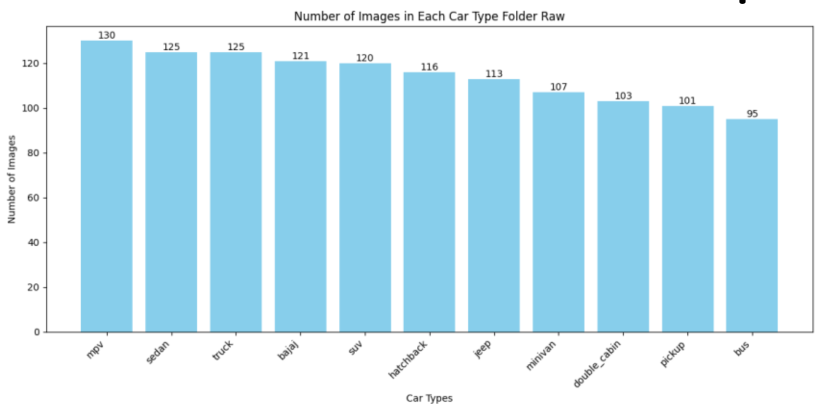
File: 15041_bis_jpg.rf.a8ceabaf9698f99373b2bdfa897e5540.jpg

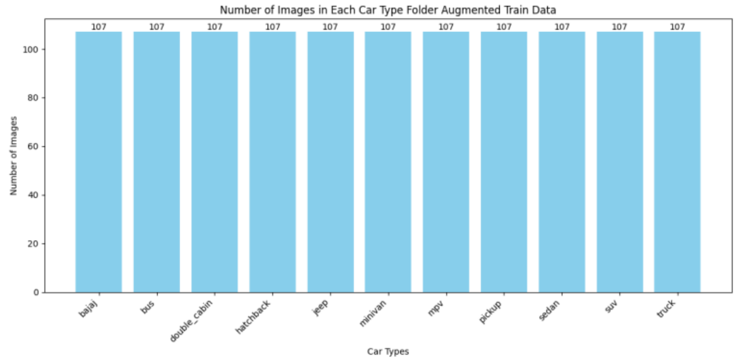


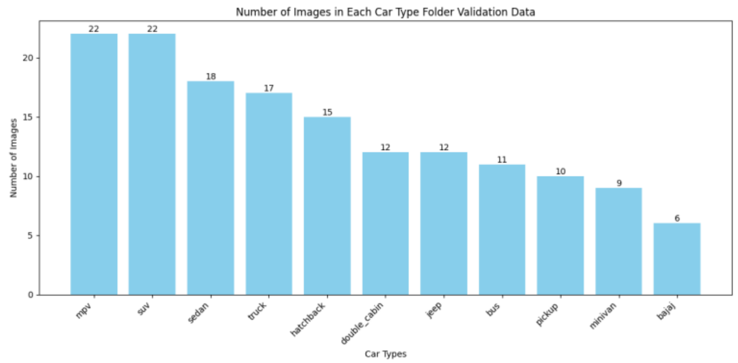
Data Preprocessing : Classification

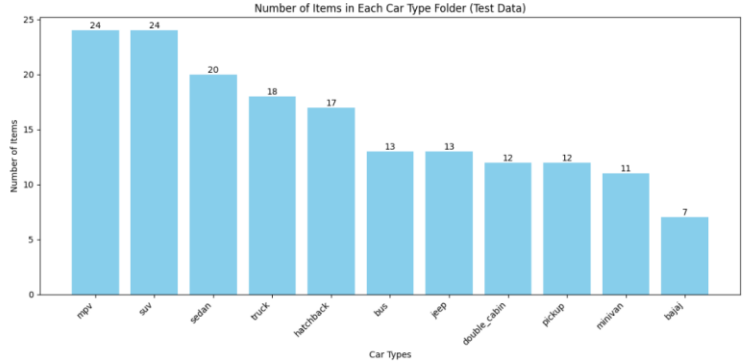
Augmentation

Crawl → Crop → Manual Inspection → Split





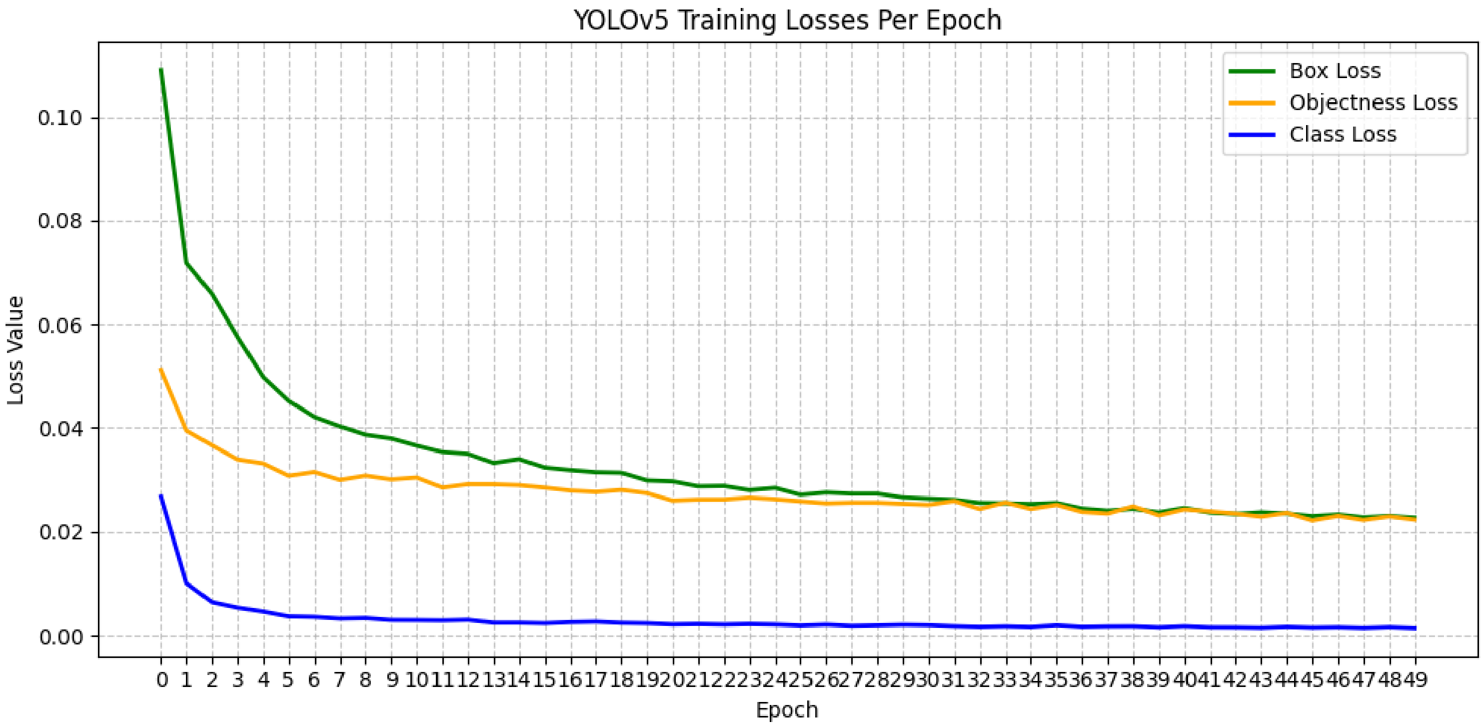




- The raw data obtained from web crawling were cropped, manually inspected, and split into training, validation, and test sets.
- To ensure balanced label distribution in the training set, image augmentation techniques were applied.

④ Model Training and Evaluation

Detection Model Analysis : Training Result



Insight.

Box Loss

- Meaning: Measures error in predicting bounding box coordinates.
- Trend: Sharp decline in early epochs, followed by gradual stabilization.
- Interpretation: The model quickly learned to localize objects within the first 10–15 epochs.

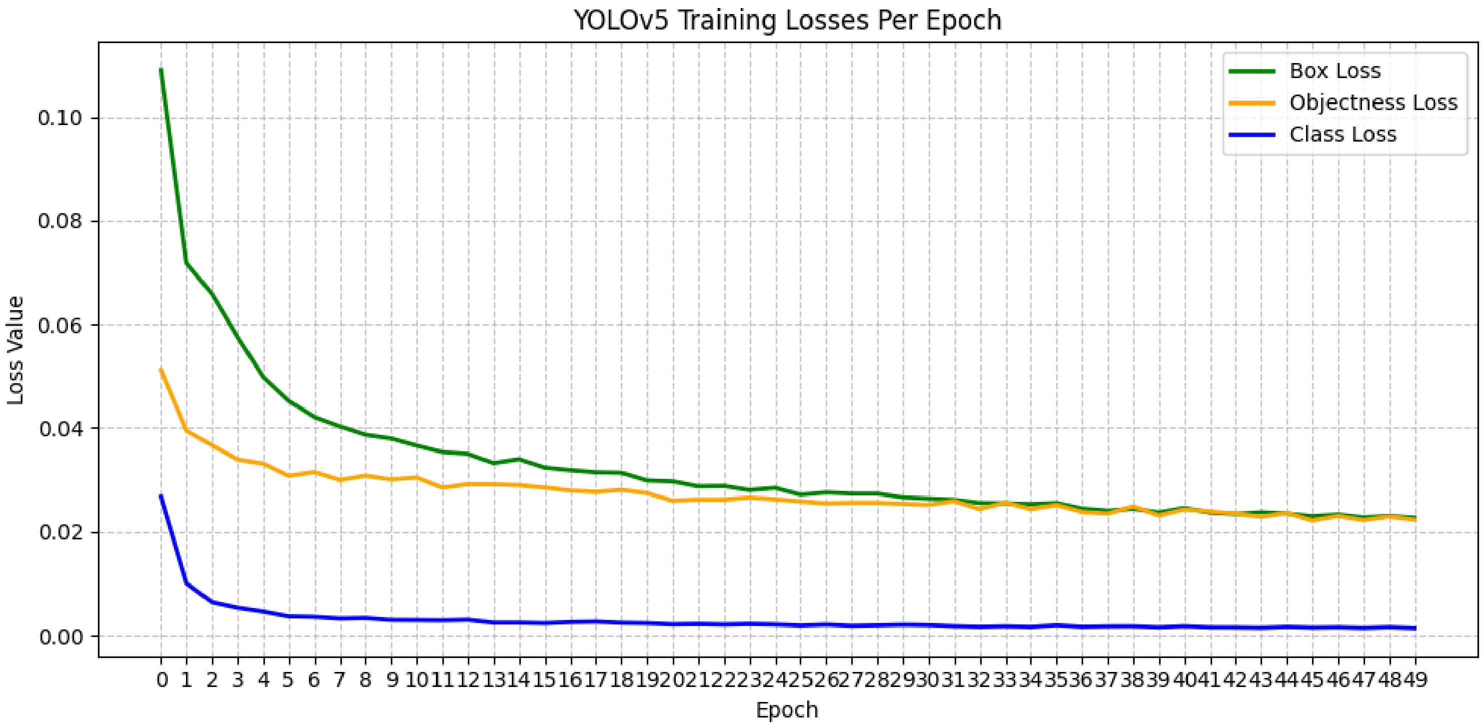
Objectness Loss

- Meaning: Reflects model confidence in object presence (vs. background)
- Trend: Steady decline, stabilizing around epoch 30.
- Interpretation: The model progressively improved at recognizing when an object is present.

Class Loss

- Meaning: Measures classification errors on detected objects
- Trend: Starts low and approaches zero quickly.
- Interpretation: The model learned the object classes effectively from early epochs.

Detection Model Analysis : Training Result



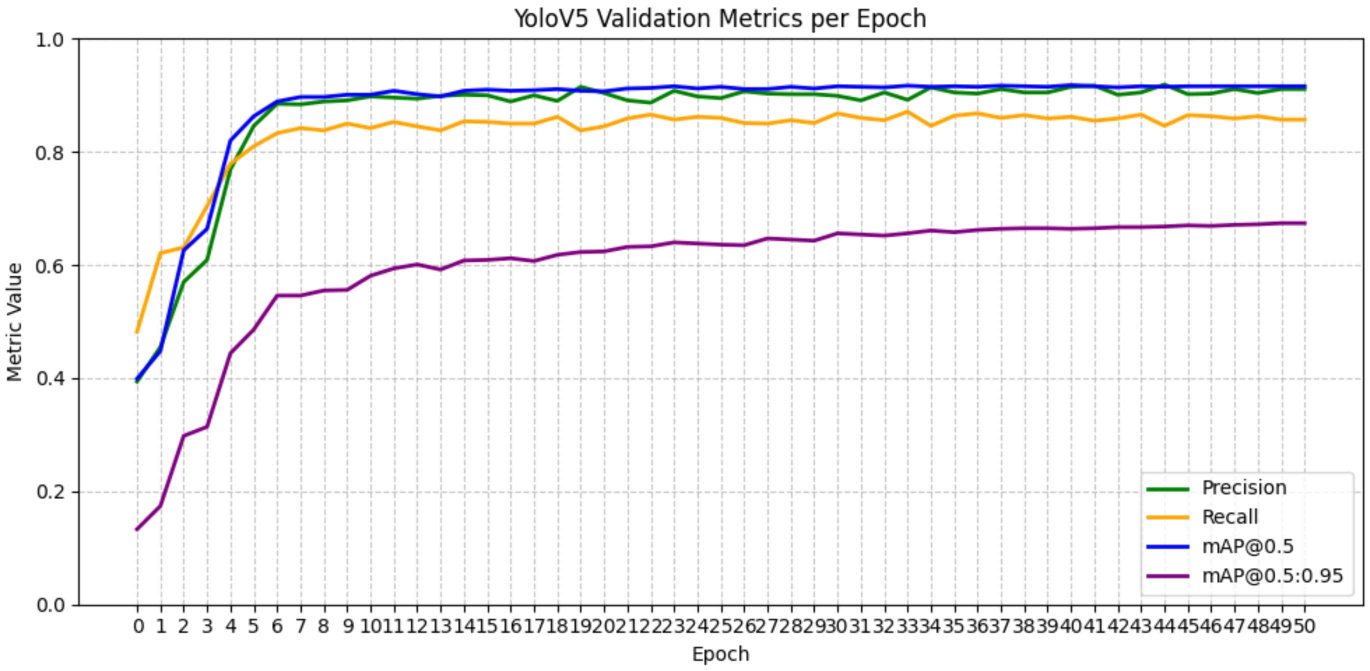
Insight.

- **Conclusion**

  - The model converged successfully, with all loss components reducing over time.

  - No clear signs of overfitting are observed from the training loss alone.

Detection Model Analysis : Validation Result



Insight.

Precision

- Definition: The proportion of predicted positives that are true positives.
- Insight: Precision rose rapidly within the first 5–7 epochs and stabilized above 0.88, indicating the model reliably avoids false positives.

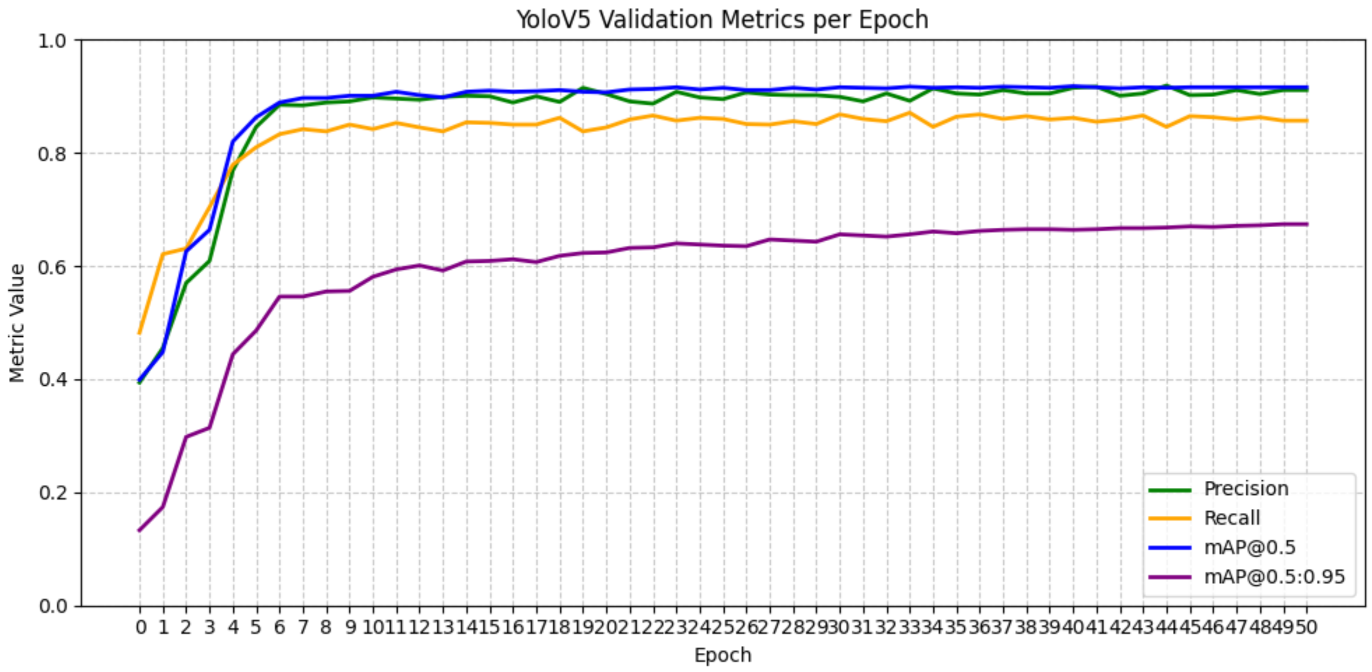
Recall

- Definition: The proportion of true positives that were correctly predicted.
- Insight: Followed a similar trend to precision, stabilizing around 0.85–0.88. This shows the model is successfully detecting most relevant objects.

mAP@0.5

- Definition: Mean Average Precision using an IoU threshold of 0.5.
- Insight: Peaked quickly (~0.91) and remained stable, a strong indicator of high-quality detection with loose localization tolerance.

Detection Model Analysis :Validation Result



Insight.

- **mAP@0.5:0.95**

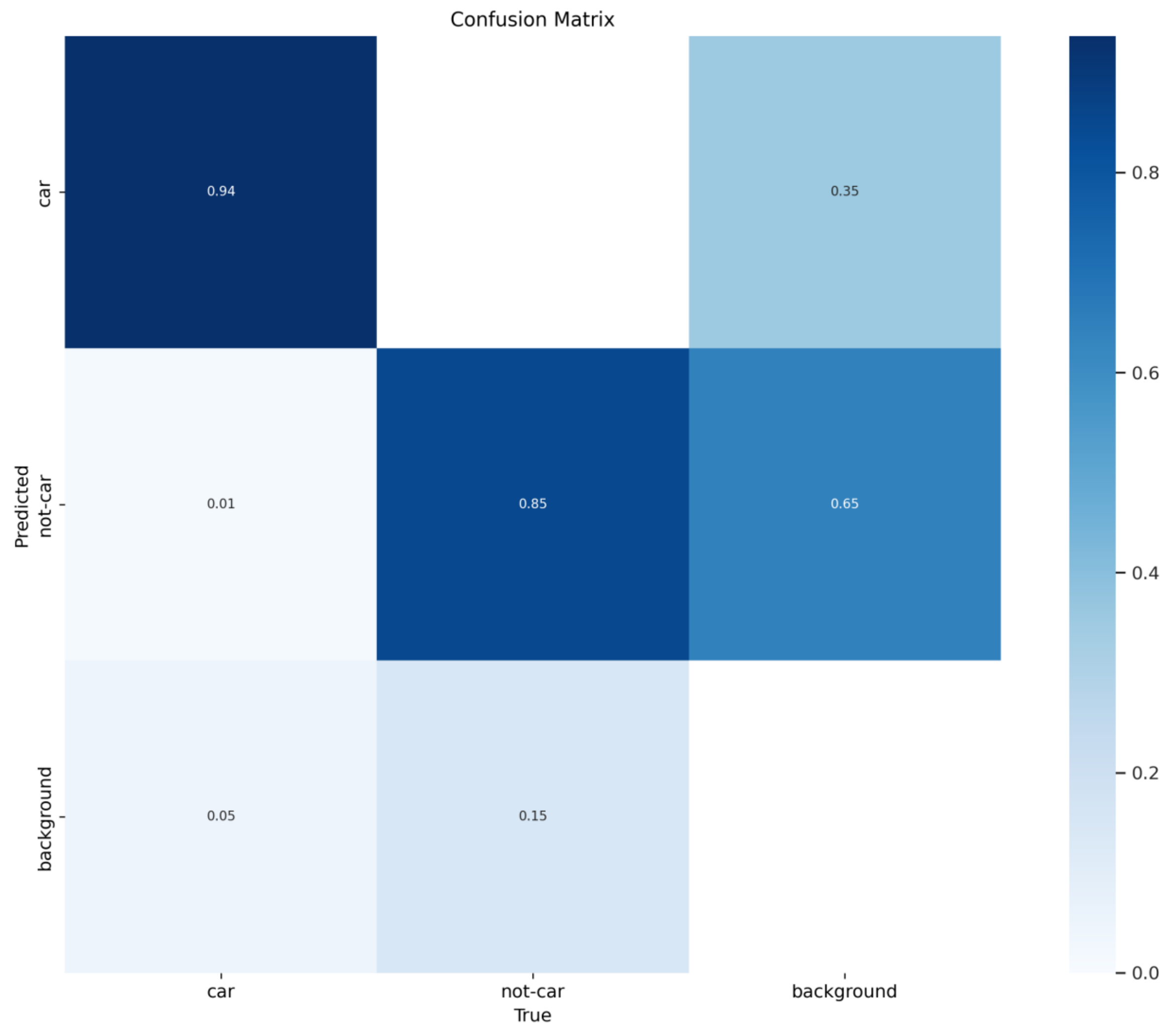
  - Definition: A stricter average precision measure across IoU thresholds [0.5:0.95].
  - Insight: Grew steadily, plateauing near 0.67 — a healthy indicator of robustness under tighter localization demands.

- **Summary**

  - All validation metrics show rapid improvement early in training and converge smoothly.
  - The high and stable precision, recall, and mAP values confirm that:
    - The model is accurate, robust, and generalizes well to unseen data.
    - There are no signs of overfitting, as validation performance continues to improve or remains stable throughout.

Detection Model Analysis: Confusion Matrix

This confusion matrix visualizes the classification performance of the car detection model across three categories: car, not-car, and background (default from YoloV5). It helps assess how accurately the model distinguishes between these classes during testing.



Insight.

True Label: car

- 94% of car objects were correctly predicted as car.
- 1% were misclassified as not-car.
- 5% were incorrectly labeled as background.

Interpretation: The model performs very well at detecting cars.

True Label: not-car

- 85% of not-car objects were correctly predicted.
- 15% were misclassified as background.

Interpretation: Good performance, but with some tendency to miss objects and consider them background.

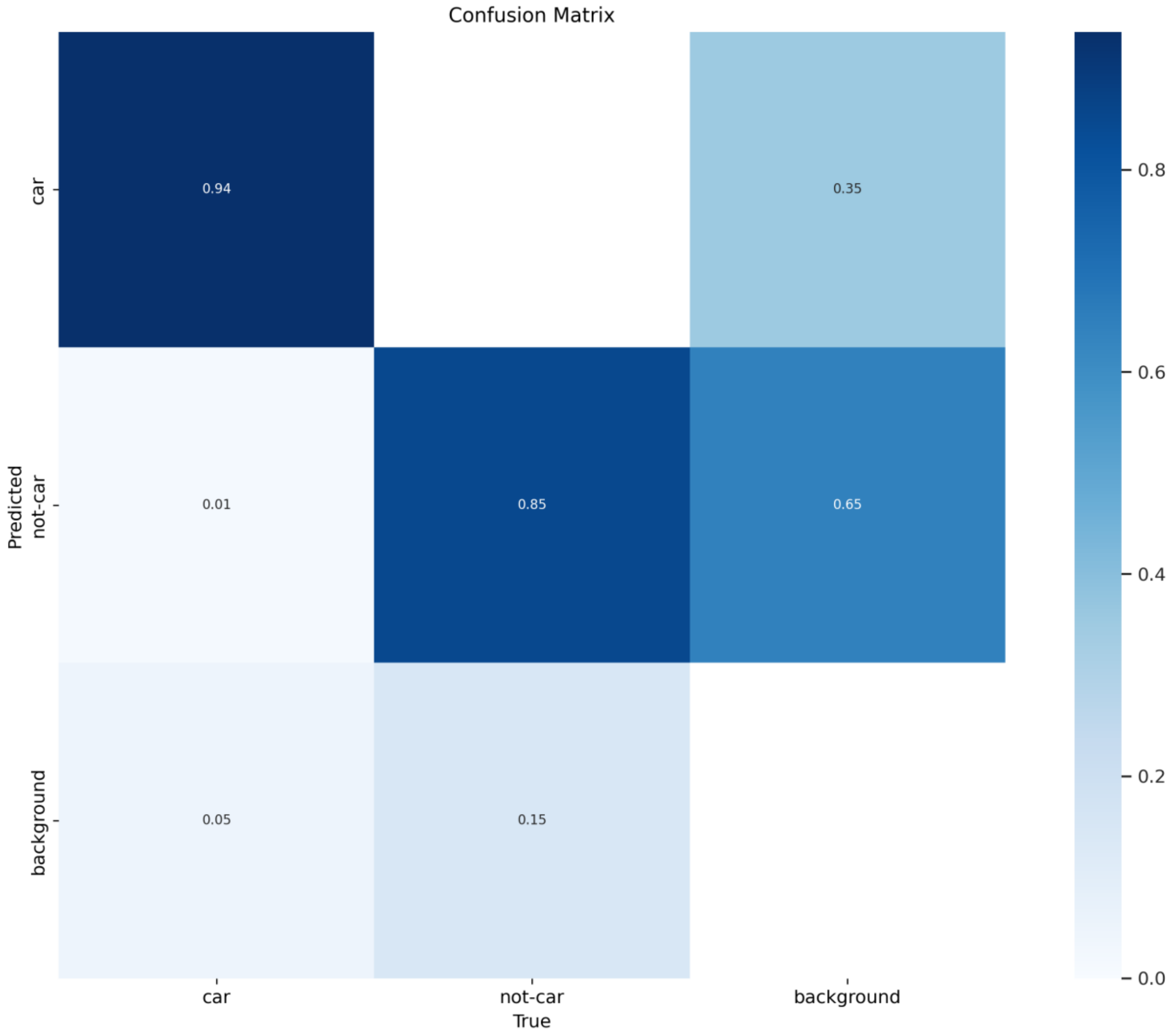
True Label: background

- 65% of the background regions were incorrectly predicted as not-car.
- Only 35% of the background was correctly ignored.

Interpretation: The model is overpredicting not-car on the background, leading to false positives.

Detection Model Analysis: Confusion Matrix

This confusion matrix visualizes the classification performance of the car detection model across three categories: car, not-car, and background (default from YoloV5). It helps assess how accurately the model distinguishes between these classes during testing.



Insight.

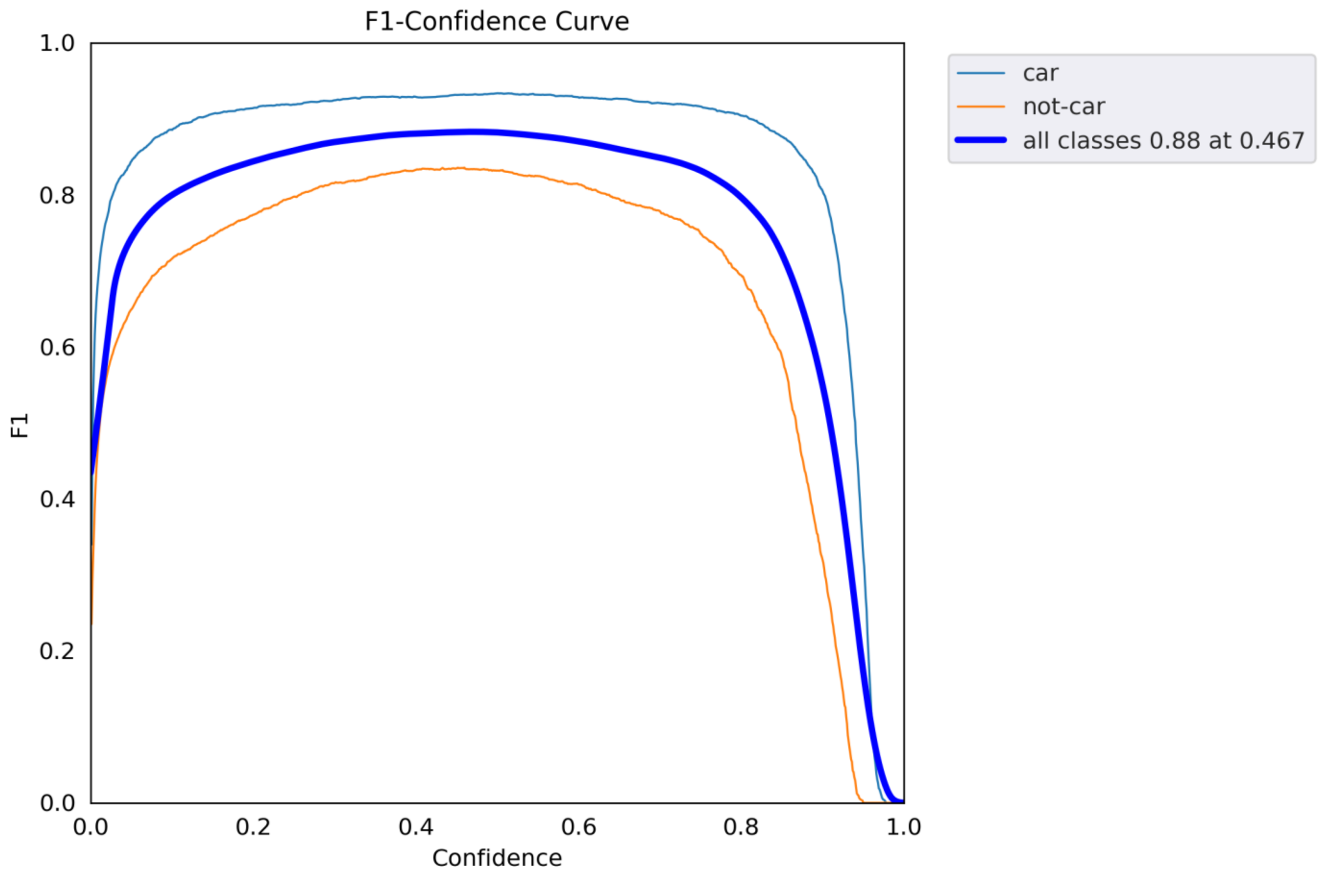
Conclusion

- The model performs best at identifying car objects with high confidence.
- There's room for improvement in distinguishing not-car vs background.

Potential action: Introduce more diverse background samples during training to reduce false positives.

Detection Model Analysis: F1-Confidence Curve

This curve illustrates the relationship between prediction confidence and the F1-score for each class (car, not-car) during evaluation. It helps determine the optimal confidence threshold to maximize model performance.



Insight.

All Classes

- Peak F1-score: 0.88 achieved at a confidence threshold of 0.467.

Interpretation: This is the recommended threshold for balancing precision and recall across all classes.

Car Class

- Consistently high F1-score across all thresholds, peaking close to 0.92.
- Interpretation: The model is very confident and accurate in predicting car objects.

Not-Car Class

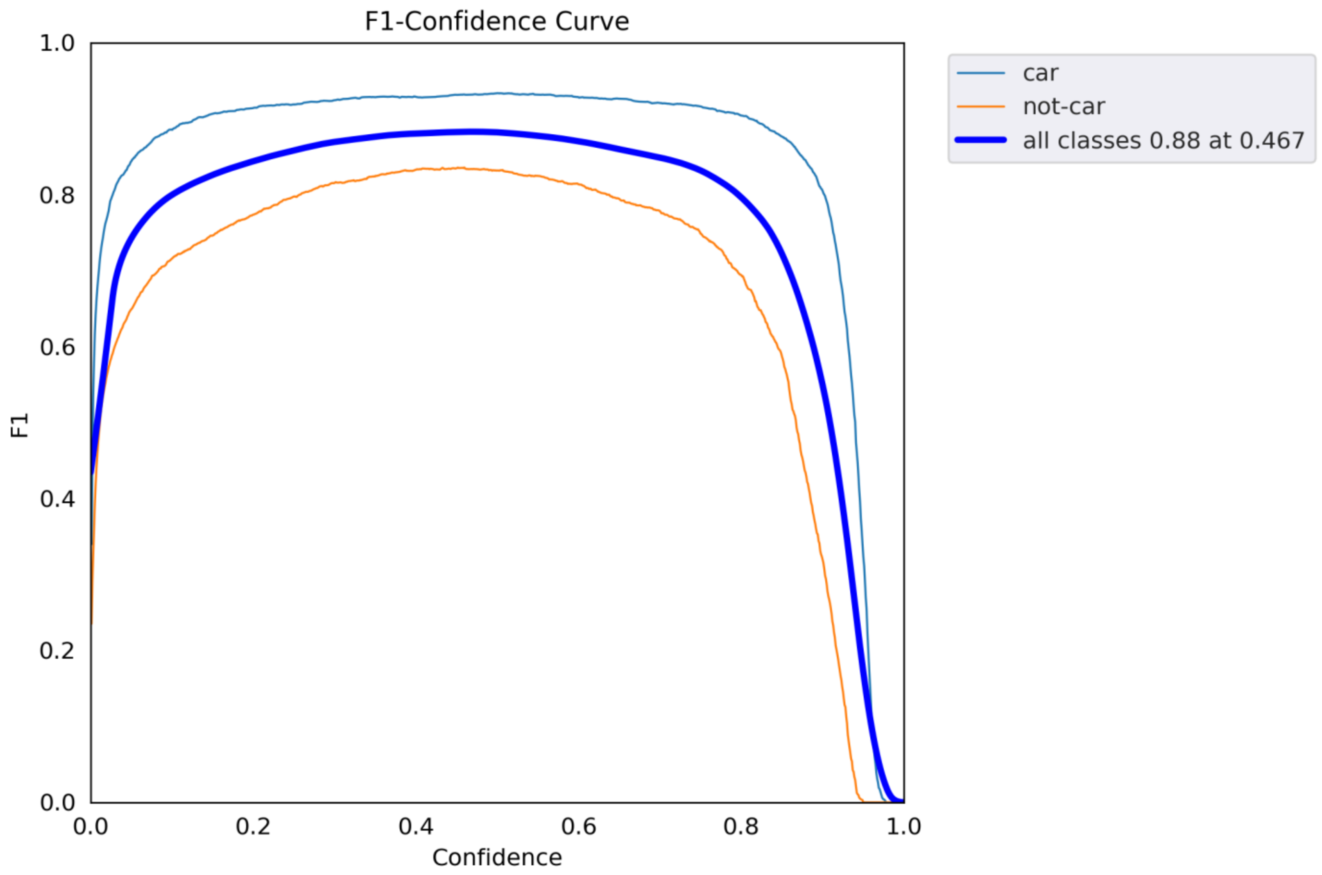
- Lower overall F1-score, peaking below 0.85.

Interpretation: Model is less confident and more prone to errors for not-car objects, possibly due to:

- Class imbalance
- Higher visual variation
- Ambiguous labeling

Detection Model Analysis: F1-Confidence Curve

This curve illustrates the relationship between prediction confidence and the F1-score for each class (car, not-car) during evaluation. It helps determine the optimal confidence threshold to maximize model performance.



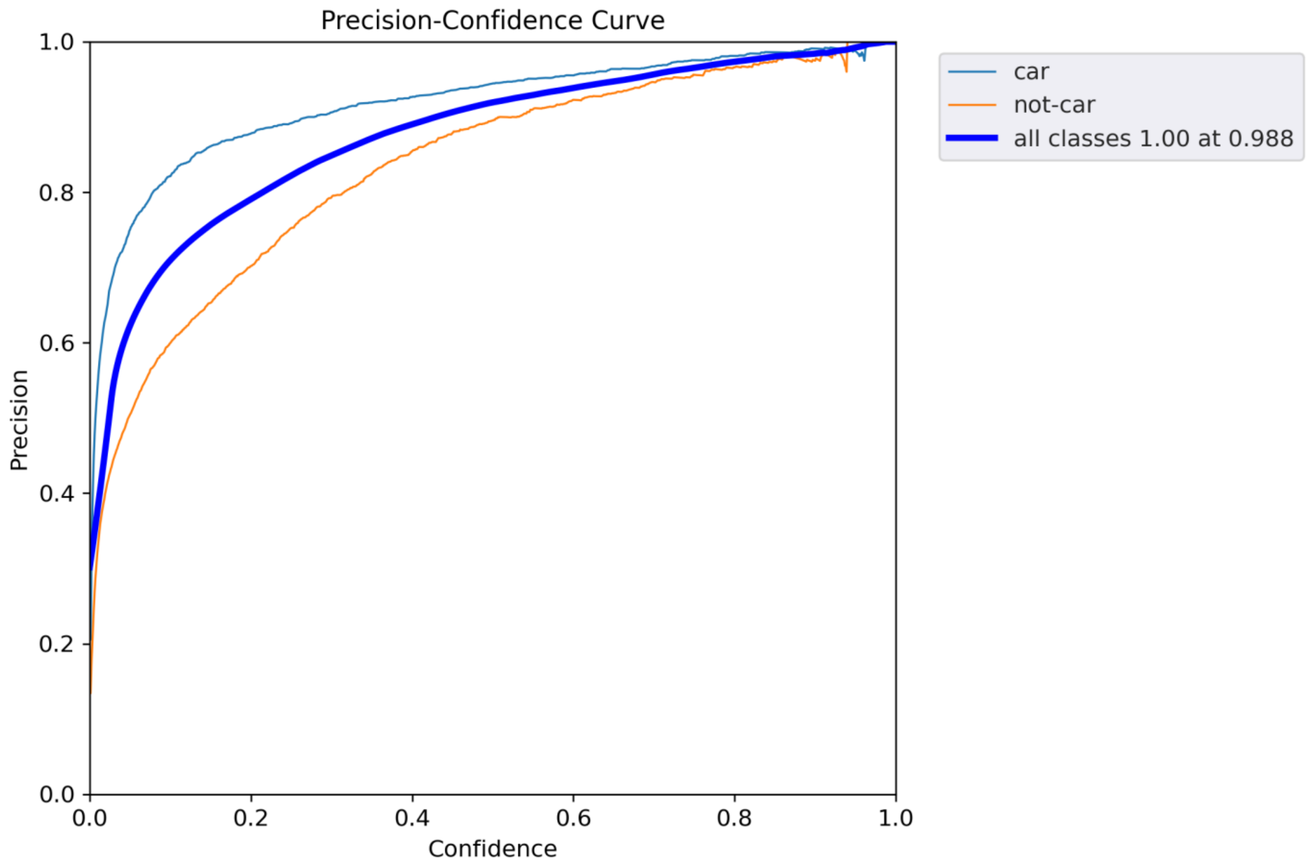
Insight.

Conclusion

- The optimal threshold of 0.467 can be used to fine-tune the model for deployment.
- The model is more reliable for detecting cars than other vehicles or objects (not-car).
- Consider rebalancing the dataset or adding hard examples to improve not-car classification.

Detection Model Analysis: P-Confidence Curve

This graph shows how precision changes with respect to confidence threshold for each class (car, not-car). It helps determine the threshold at which the model achieves maximum precision — i.e., the fewest false positives.



Insight.

All Classes

- Peak precision: 1.00 (100%) at a confidence threshold of 0.988.
Interpretation: The model makes perfectly precise predictions when it's very confident (≥ 0.988), though likely at the cost of lower recall.

Car Class (Light Blue Line)

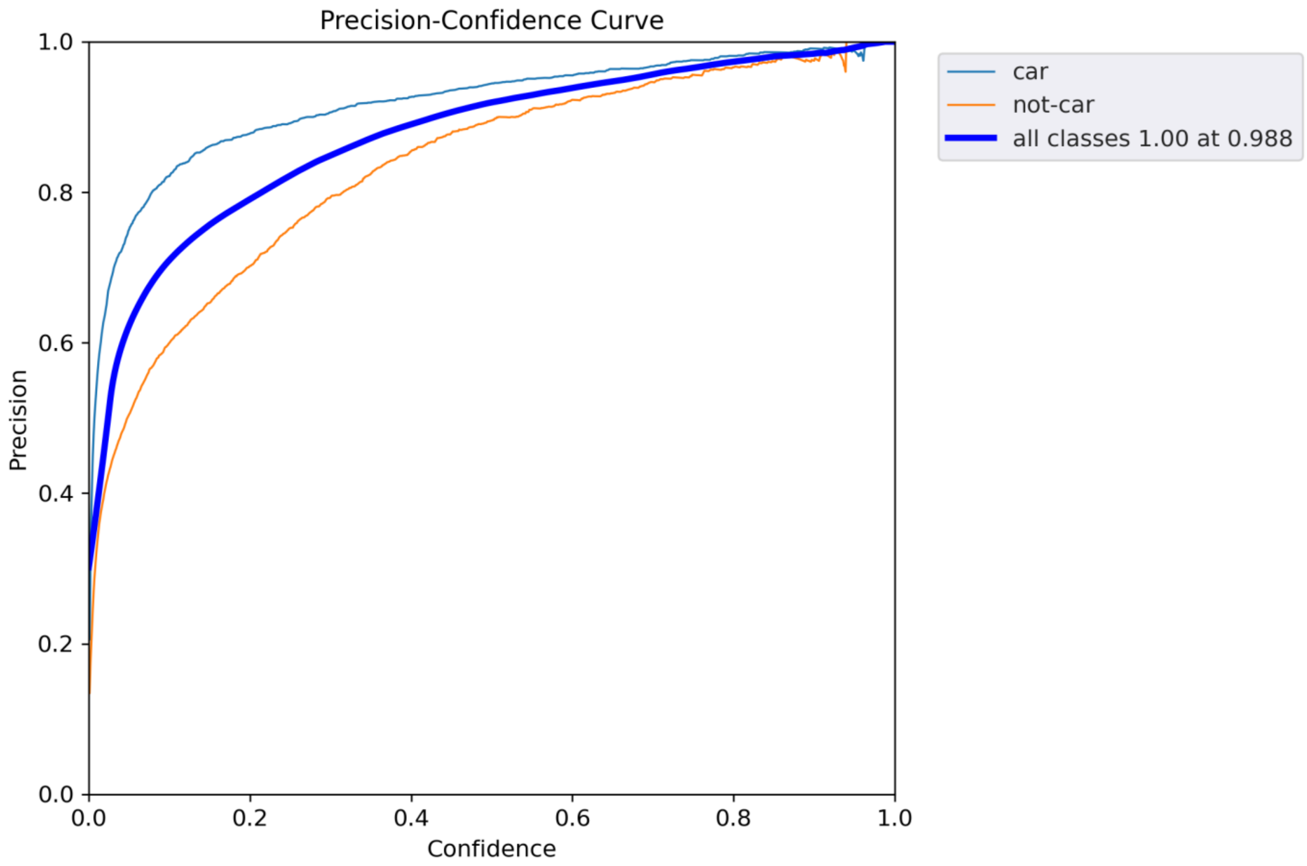
- Precision starts high and stays consistently strong across all thresholds.
Interpretation: The model is highly accurate at identifying cars, even at lower confidence levels.

Not-Car Class (Orange Line)

- Precision improves as the confidence threshold increases but remains below car class throughout.
Interpretation: There are more false positives in the not-car category, indicating the model finds it harder to distinguish these reliably.

Detection Model Analysis: P-Confidence Curve

This graph shows how precision changes with respect to confidence threshold for each class (car, not-car). It helps determine the threshold at which the model achieves maximum precision — i.e., the fewest false positives.



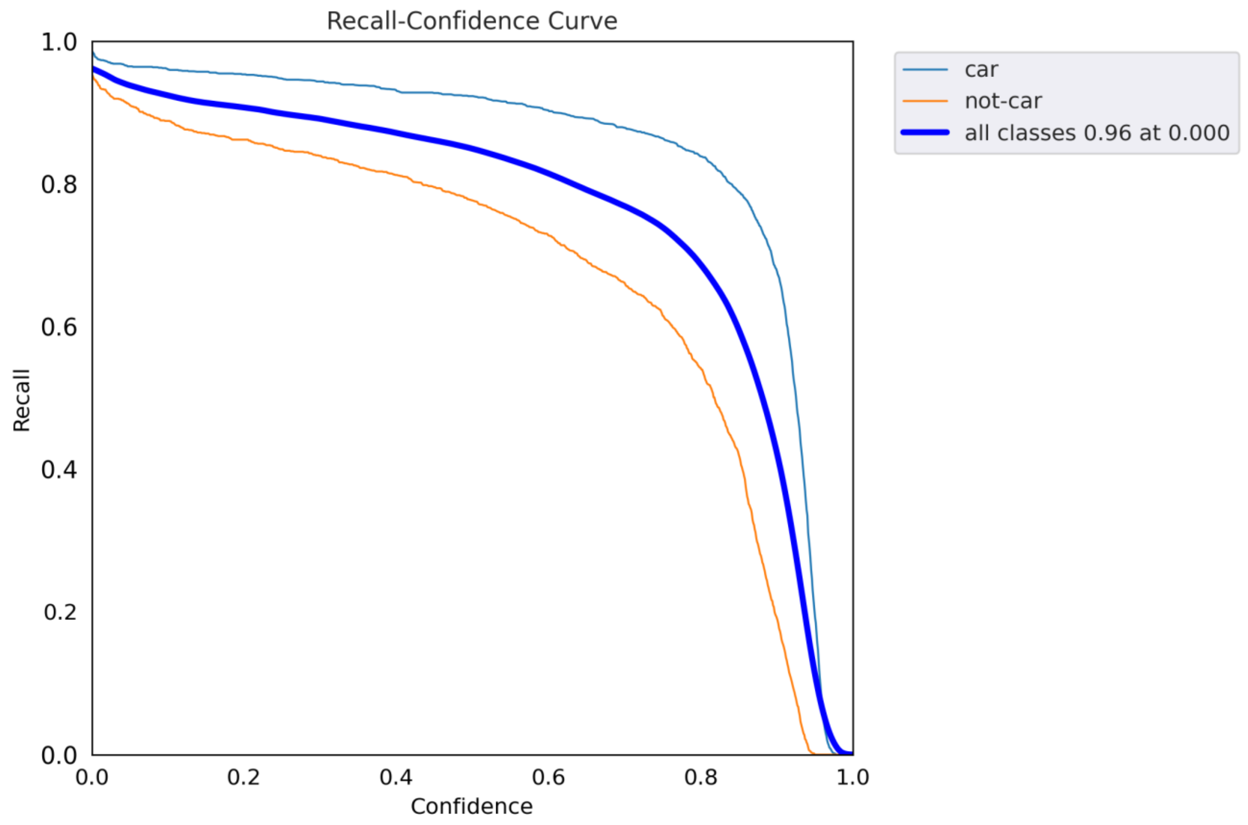
Insight.

Conclusion

- To maximize model reliability and minimize false detections, a high confidence threshold (e.g., 0.988) should be used if precision is prioritized.
- For a balanced trade-off between precision and recall, combine this with the F1-Confidence Curve to select a more suitable threshold (e.g., ~0.467 from the F1 graph).
- Suggest adding more diverse not-car examples to improve class separation and reduce misclassification.

Detection Model Analysis: R-Confidence Curve

This graph illustrates how recall varies with the confidence threshold for each class (car, not-car). It shows how well the model detects all true positives, especially at different certainty levels.



Insight.

All Classes

- Peak recall: 0.96 (96%) at confidence threshold = 0.0.
Interpretation: At very low confidence, the model detects nearly all true positives (high recall) but likely includes more false positives — hence lower precision.

Car Class

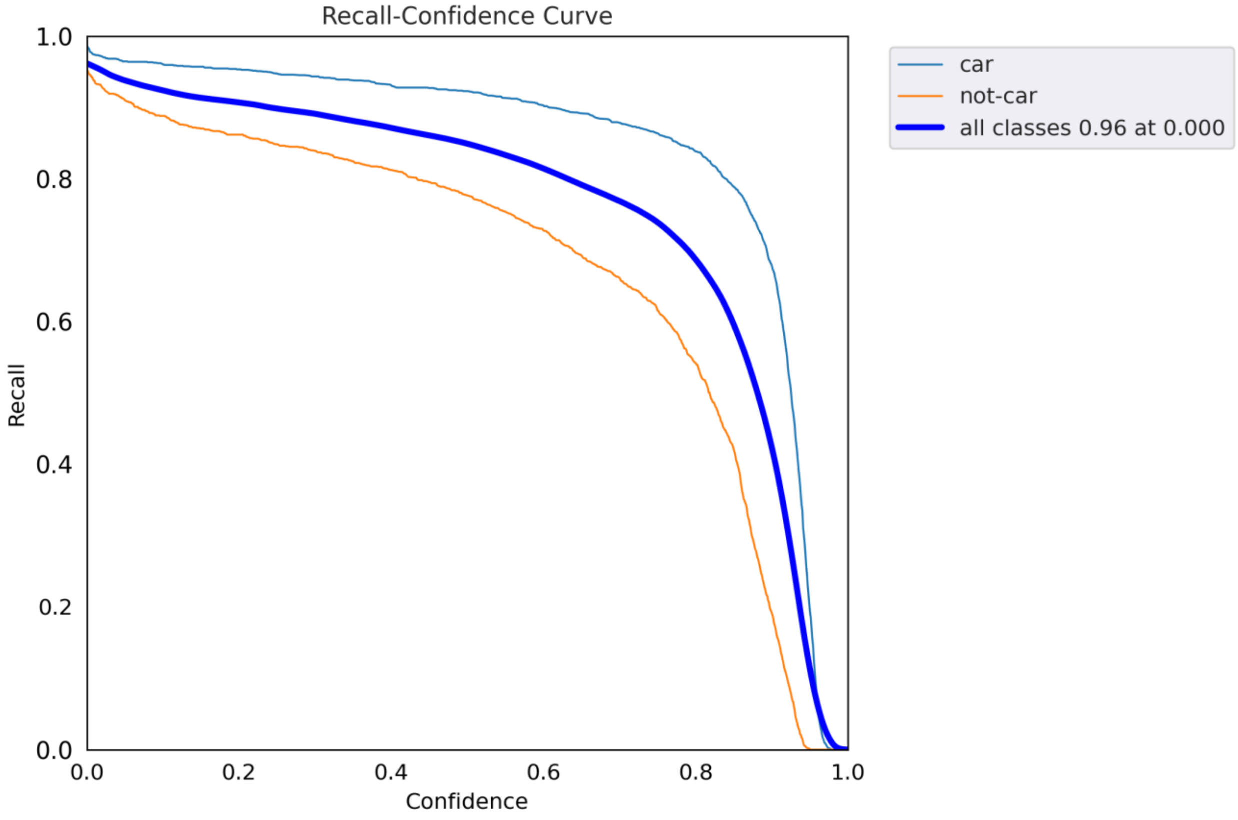
- Recall remains consistently high and only declines sharply near confidence = 1.0.
Interpretation: The model is very sensitive to the presence of cars and tends to detect them successfully, even at higher thresholds.

Not-Car Class

- Recall is lower and more sensitive to confidence changes compared to car.
Interpretation: The model misses more not-car objects as the threshold increases, indicating it struggles more with this class.

Detection Model Analysis: R-Confidence Curve

This graph illustrates how recall varies with the confidence threshold for each class (car, not-car). It shows how well the model detects all true positives, especially at different certainty levels.



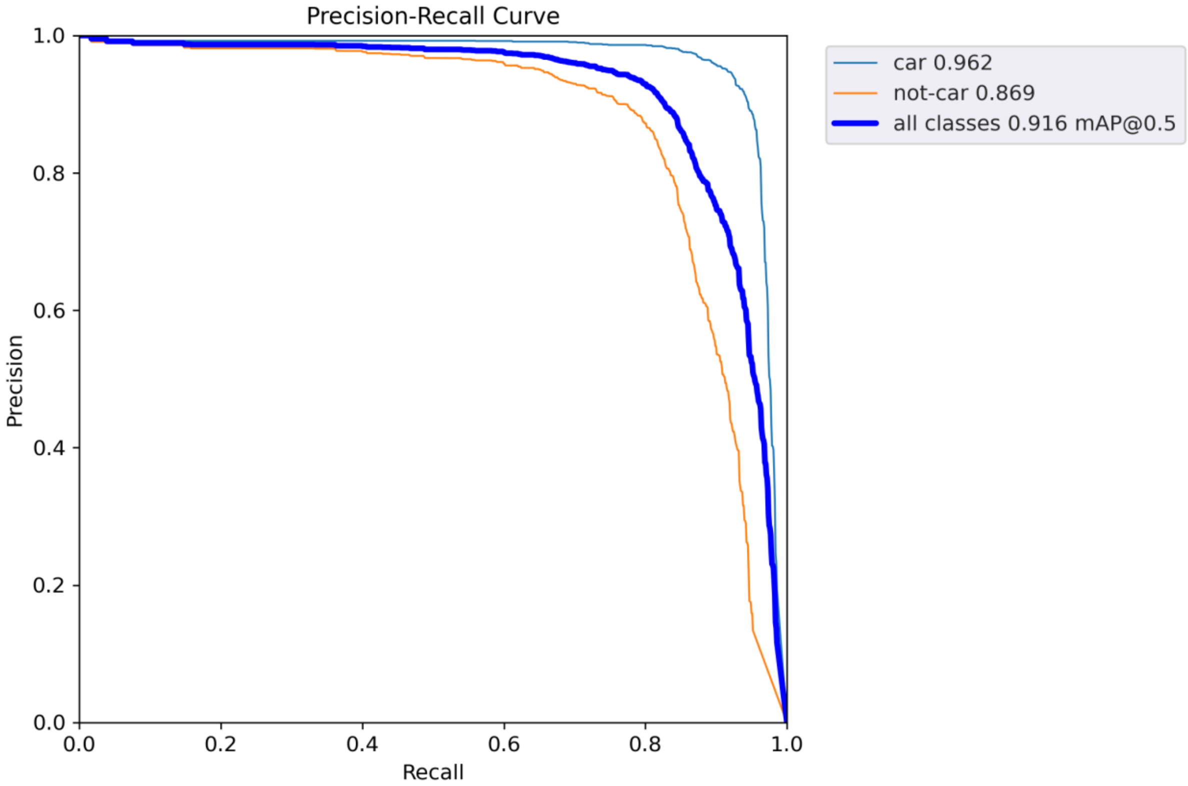
Insight.

Conclusion

- To maximize recall, a lower threshold (e.g., ~0.0 to 0.2) is ideal, especially for detecting all possible true objects.
- However, this should be balanced with precision (from the previous curve), as high recall at low thresholds often comes with more false positives.
- For a real-world deployment, select a threshold based on F1-score to balance both.

Detection Model Analysis: PR Curve

The PR curve shows the trade-off between precision and recall across thresholds. It's especially useful for imbalanced datasets like yours (e.g., car vs not-car).



Insight.

- **All Classes**

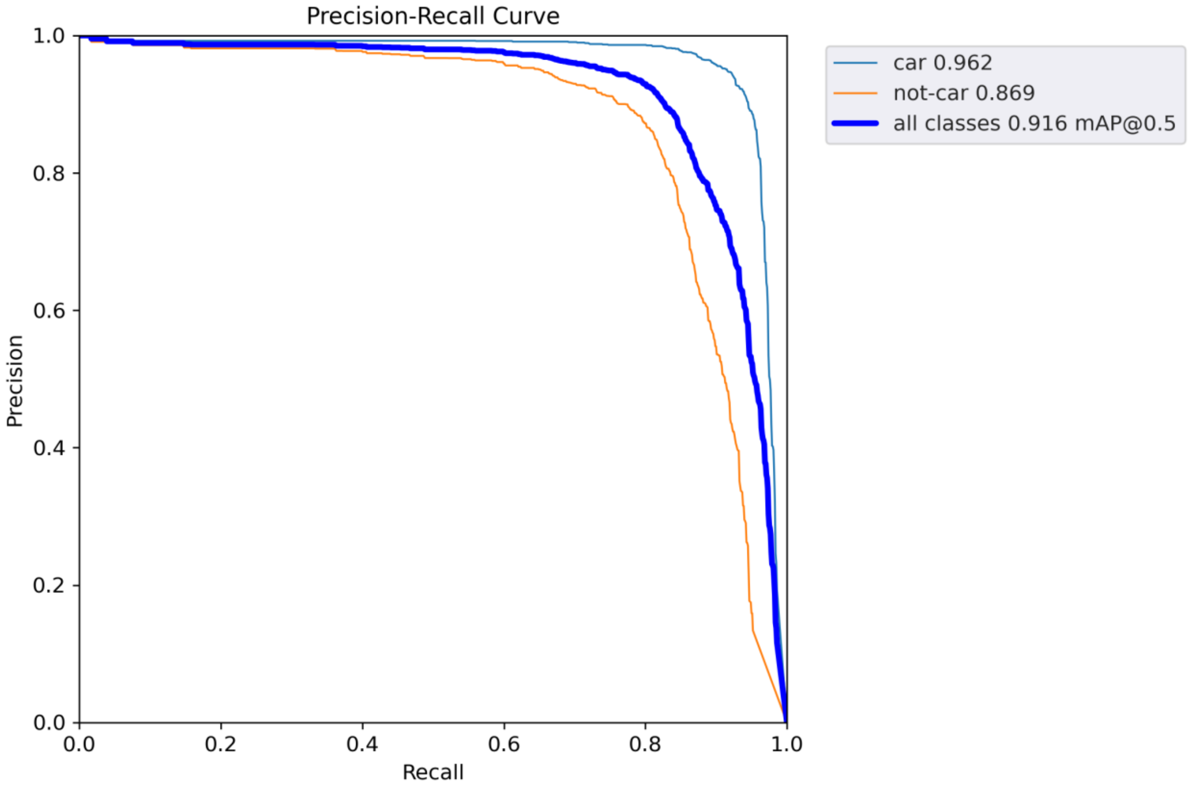
  - Very high performance overall.
  - Precision stays near 1.0 for most of the recall range.
  - Slight drop at high recall levels, as expected.
  Interpretation: The model balances precision and recall well, indicating reliable detections without many false positives or false negatives.

- **Car Class**

  - Highest performance of both classes (AP = 0.962).
  - Precision remains extremely high across nearly all recall values.
  - Sharp decline only near perfect recall.
  Interpretation: The model almost always detects cars accurately, making it highly reliable for car classification.

Detection Model Analysis: PR Curve

The PR curve shows the trade-off between precision and recall across thresholds. It's especially useful for imbalanced datasets like yours (e.g., car vs not-car).



Insight.

Not-Car Class

- Lower but still strong performance (AP = 0.869).
- Precision drops more quickly as recall increases, indicating more false positives at higher recall.

Interpretation: The model finds it harder to distinguish not-car objects without sacrificing precision.

Conclusion

- With a mean AP of 0.916, the model is highly performant overall.
- It's better at detecting car than not-car, which is typical when class imbalance or subtle features affect classification.
- Thresholds can be fine-tuned to prioritize either precision (for fewer false alarms) or recall (for fewer missed detections).

Detection Model Analysis: Test Result

```
📦 all
Precision        : 0.894
Recall           : 0.862
mAP@0.5          : 0.912
mAP@0.5:0.95     : 0.670

📦 car
Precision        : 0.934
Recall           : 0.931
mAP@0.5          : 0.967
mAP@0.5:0.95     : 0.777

📦 not-car
Precision        : 0.853
Recall           : 0.793
mAP@0.5          : 0.857
mAP@0.5:0.95     : 0.562
```

💡Insight.

- **all**
  - Precision: The model correctly predicted 89.4% of detected objects.
  - Recall: It captured 86.2% of all ground-truth objects.
  - mAP@50: Strong localization performance at IoU 0.5.
  - mAP@0.5:0.95: Indicates good performance across stricter IoU thresholds.

  Interpretation: The model performs well in both detection and localization across all classes.

- **car**
  - Precision: Very few false positives.
  - Recall: Nearly all true car instances were detected.
  - mAP@0.5: Excellent bounding box alignment.
  - mAP@0.5:0.95: High robustness under varying IoU thresholds.

  Interpretation: car class detection is highly reliable and consistent, with strong precision and recall balance.

- **not-car**
  - Precision: Fair accuracy, though more false positives than for car.
  - Recall: Some missed detections for not-car instances.
  - mAP@0.5: Good bounding box accuracy.
  - mAP@0.5:0.95: Weaker performance under strict IoU conditions.

  Interpretation: not-car class performance is lower, suggesting potential data imbalance or difficulty in class definition.

Detection Model Analysis: Test Result

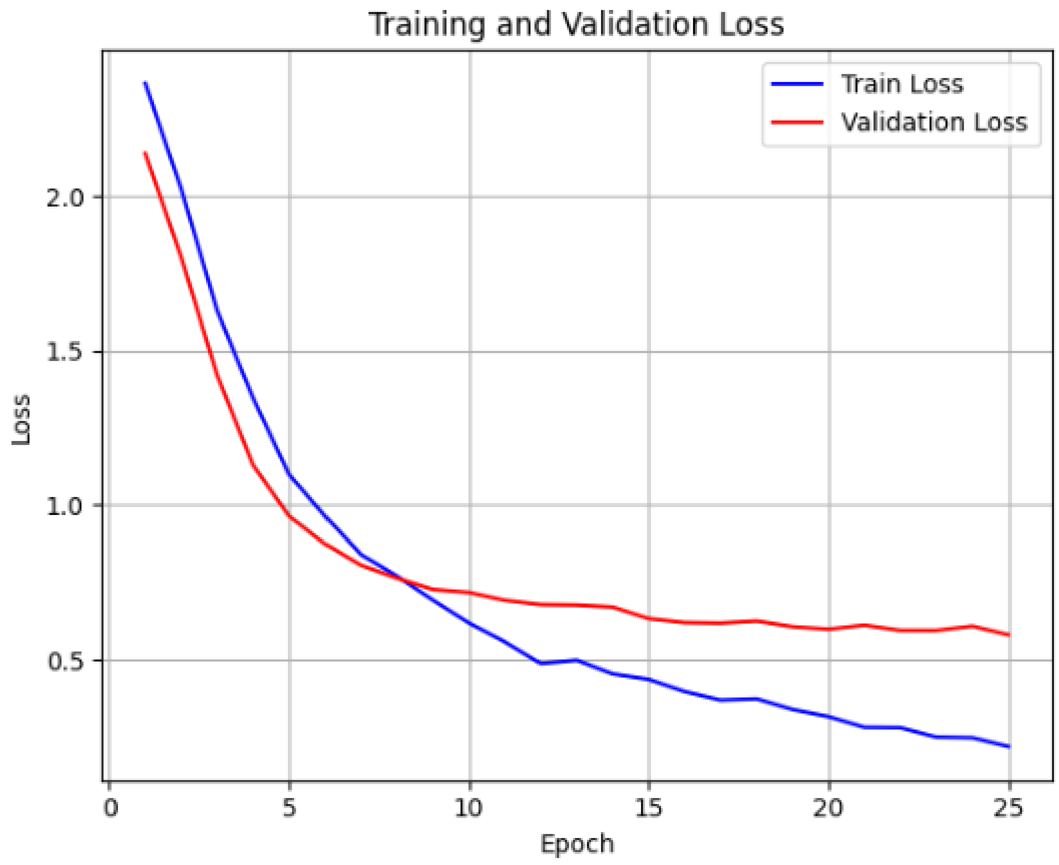
```
📦 all
Precision        : 0.894
Recall           : 0.862
mAP@0.5          : 0.912
mAP@0.5:0.95     : 0.670

📦 car
Precision        : 0.934
Recall           : 0.931
mAP@0.5          : 0.967
mAP@0.5:0.95     : 0.777

📦 not-car
Precision        : 0.853
Recall           : 0.793
mAP@0.5          : 0.857
mAP@0.5:0.95     : 0.562
```

💡Insight.

- **conclusion**

  - The model shows excellent performance for the car class, while not-car detection needs improvement.
  - Overall, the high mAP@0.5 values indicate solid localization ability.
  - The gap between mAP@0.5 and mAP@0.5:0.95 suggests room for improvement in tighter box predictions, especially for not-car.

Classification Mode Analysis: Train Result



Insight.

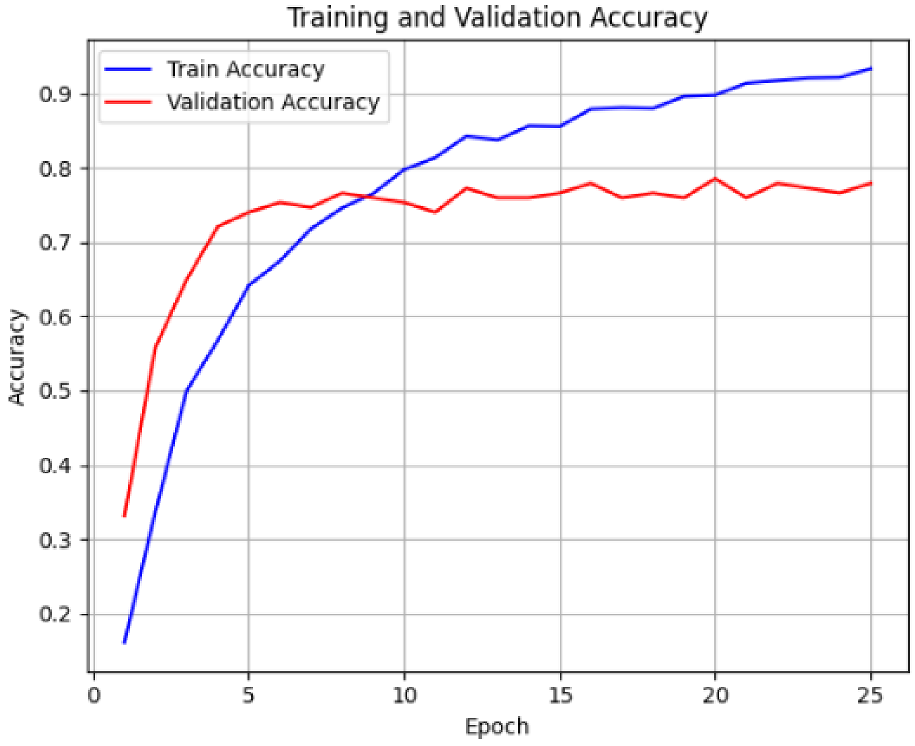
Loss Metrics Analysis

Training Loss shows a steady decrease from ~2.36 to ~0.22, indicating:
- Effective learning by the model
- Proper gradient descent optimization
- Appropriate learning rate selection

Validation Loss follows a similar decreasing trend but with:
- Slightly higher values than training loss (expected gap)
- Minor fluctuations indicating small overfitting after epoch 15

Key Observations:
- The curves converge nicely, suggesting good generalization
- No signs of severe overfitting (validation loss doesn't diverge)
- Best validation loss achieved around epoch 20

Classification Model Analysis: Train Result



Insight.

Accuracy Metrics Analysis

Training Accuracy grows consistently from 16% to 93%:
- Steep initial improvement (epochs 1-10)
- Gradual refinement in later epochs

Validation Accuracy shows:
- Similar growth pattern but with expected lower values
- Reaches peak performance (~78.6%) at epoch 20
- Small fluctuations in later epochs suggest model stability

Key Observations:
- 16% → 78.6% validation accuracy shows significant learning
- Training/validation gap remains reasonable (~15%)
- Early stopping triggered appropriately at epoch 25

Classification Model Analysis: Test Result

```
[2025-06-21 01:51:59] 🧪 Test Accuracy: 0.7251
[2025-06-21 01:51:59] 📉 Average Test Loss: 0.6859
[2025-06-21 01:51:59]
📊 Classification Report:
              precision    recall  f1-score   support

       bajaj       1.00      1.00      1.00         7
         bus       0.87      1.00      0.93        13
 double_cabin       0.90      0.75      0.82        12
   hatchback       0.55      0.65      0.59        17
        jeep       0.93      1.00      0.96        13
     minivan       0.60      0.55      0.57        11
         mpv       0.55      0.75      0.63        24
      pickup       1.00      0.83      0.91        12
       sedan       0.50      0.45      0.47        20
         suv       0.76      0.54      0.63        24
       truck       0.88      0.83      0.86        18

    accuracy                           0.73       171
   macro avg       0.78      0.76      0.76       171
weighted avg       0.74      0.73      0.73       171

[2025-06-21 01:51:59]
🧩 Confusion Matrix:
[[ 7,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0],
 [ 0, 13,  0,  0,  0,  0,  0,  0,  0,  0,  0],
 [ 0,  0,  9,  0,  1,  1,  0,  0,  0,  1,  0],
 [ 0,  0,  0, 11,  0,  0,  1,  0,  5,  0,  0],
 [ 0,  0,  0,  0, 13,  0,  0,  0,  0,  0,  0],
 [ 0,  0,  0,  0,  0,  6,  5,  0,  0,  0,  0],
 [ 0,  0,  0,  3,  0,  0, 18,  0,  0,  2,  1],
 [ 0,  0,  0,  0,  0,  1,  0, 10,  0,  0,  1],
 [ 0,  0,  0,  5,  0,  1,  4,  0,  9,  1,  0],
 [ 0,  0,  1,  1,  0,  0,  5,  0,  4, 13,  0],
 [ 0,  2,  0,  0,  0,  1,  0,  0,  0,  0, 15]]
```

💡 Insight.

Overall Performance

- Test Accuracy: 72.51% (slightly lower than validation accuracy of 78.57%)
- Average Test Loss: 0.6859 (consistent with validation loss trend)
- Performance Gap: ~6% drop from validation to test accuracy suggests mild overfitting

Class-Wise Performance Breakdown

Top Performers (F1 > 0.9)

✅ bajaj: Perfect 1.00 across all metrics (7 samples)
✅ bus: 0.87 precision, 1.00 recall (13 samples)
✅ jeep: 0.93 precision, perfect recall (13 samples)
✅ pickup: Perfect precision, 0.83 recall (12 samples)

Moderate Performers (0.7 < F1 ≤ 0.9)

🔸 double_cabin: 0.90 precision but 0.75 recall
🔸 truck: 0.88 precision, 0.83 recall

Problem Classes (F1 ≤ 0.7)

⚠️ hatchback: Low 0.55 precision, misclassified as sedan/MPV
⚠️ minivan: Struggles with recall (55%), confused with MPV
⚠️ MPV: 75% recall but low 55% precision
⚠️ sedan: Worst performer (0.47 F1), major confusion with hatchback
⚠️ SUV: 54% recall affected by MPV/sedan confusion

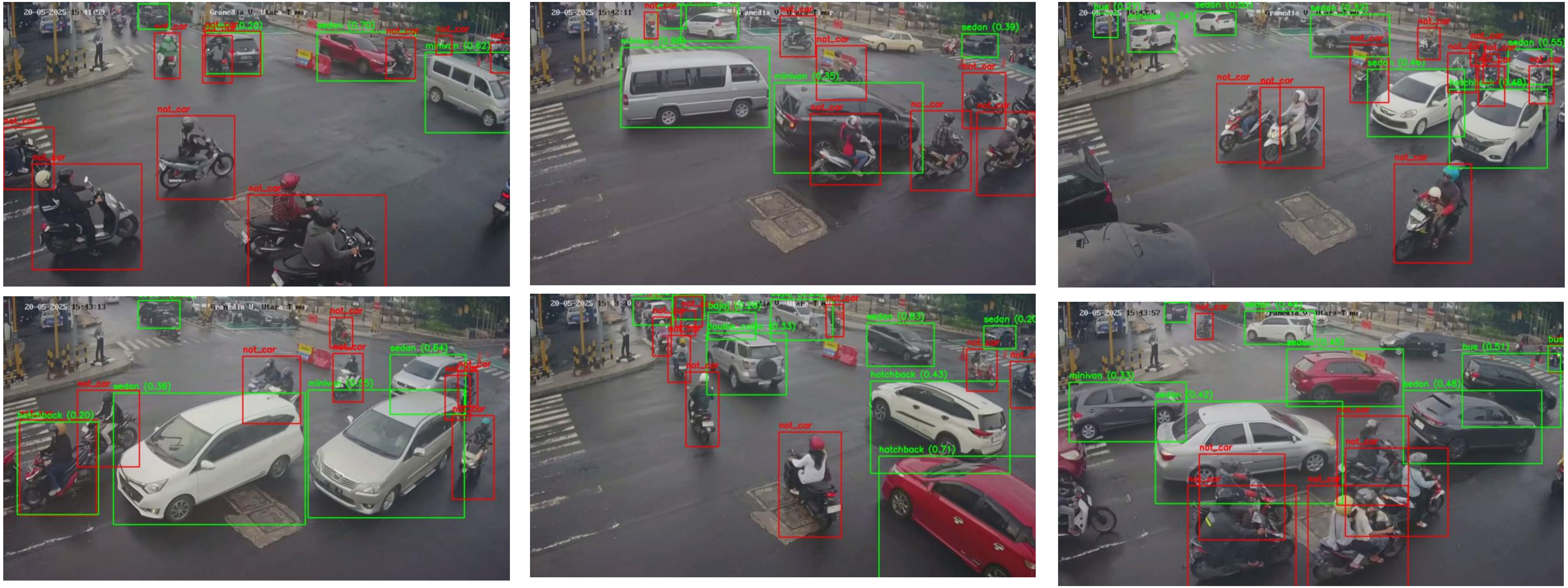
⑤ Demo

Car Detection Demo

Insight.

- The model demonstrates robust and practical object detection capabilities in real urban traffic environments.
- Especially effective for car identification, while not-car detection can still benefit from further tuning or data enrichment.

Car Detection and Classification Demo



Insight.

- While the object detection component demonstrates robust performance in identifying vehicles, the classification accuracy requires further refinement, particularly for similar-looking vehicle classes. Targeted improvements in feature extraction and model training will help bridge this performance gap for production-ready deployment.

5 Future Improvements

Future Improvements

🔧 1. Detection Model Improvements (YOLOv5)

✅ Current

- YOLOv5 trained on Roboflow-labeled dataset

🚀 Future Improvements

- Upgrade to **YOLOv8 or YOLO-NAS** for better accuracy and speed
- Add **object tracking** (e.g., DeepSORT, ByteTrack) for video analytics
- Enhance **data augmentation**:
  - MixUp, CutMix, Mosaic
  - Weather effects (rain, fog, glare)
- Apply **multi-scale training** for robust detection at different resolutions

🧠 2. Classification Model Enhancements

✅ Current

- VGG16 classifier trained on cropped detected cars

🚀 Future Improvements

- Use **more efficient models** like:
  - ResNet50, EfficientNet, ConvNeXt, MobileNetV3
- Add **attention mechanisms** (CBAM, SE-Blocks)
- Improve **class balance and data diversity**
- Add **background context** (cropped area + surrounding pixels)

🧪 3. Dataset and Labeling Quality

✅ Current

- Used Roboflow annotations and export to YOLO format

🚀 Future Improvements

- Add images from **varied lighting, locations, weather**
- Generate **synthetic data** (GANs, Unity simulations)
- Use **active learning** to retrain with uncertain predictions
- Label **additional attributes** (color, orientation, brand)

📈 4. Evaluation & Optimization

✅ Current

- Evaluated with PR, F1, P, and R curves

🚀 Future Improvements

- Include **confusion matrix** with labels
- Evaluate on **real-world CCTV or dashcam videos**
- Apply **model pruning, quantization** (TensorRT, ONNX)

🌐 5. System Pipeline & Deployment

✅ Current

- Manual pipeline with detection + cropped classification

🚀 Future Improvements

- Build **end-to-end video inference pipeline**
- Deploy using **Docker** on Jetson Nano / Raspberry Pi
- Create **web/mobile UI** for real-time image input
- Use **OpenCV** for real-time webcam integration

🗣️ 6. Explainability & Trust

🚀 Future Improvements

- Add **Grad-CAM or LIME** for classification explainability
- Show **prediction confidence scores**
- Allow **user feedback** for correcting misclassifications

Thank You