

Phạm Duy Long

# **nhận dạng phương tiện bằng YOLO và RCNN**

# Mục tiêu Dự án

- Mục tiêu chính: Xây dựng và đánh giá hiệu suất của các mô hình phát hiện đối tượng (Object Detection) cho bài toán nhận dạng các phương tiện và đối tượng giao thông trên đường.
- Các mô hình được sử dụng: YOLOv8 (You Only Look Once) và Faster R-CNN.
- Ứng dụng tiềm năng: Hệ thống giao thông thông minh, xe tự lái, giám sát đô thị.

# Tổng quan về Object Detection

- Object Detection là gì?
  - Xác định vị trí (Bounding Box) và phân loại (Class Label) của các đối tượng trong một hình ảnh hoặc video.
  - Hơn cả phân loại hình ảnh: trả lời "đối tượng này ở đâu?" và "đó là đối tượng gì?".
- Tầm quan trọng: Nền tảng cho nhiều Ứng dụng AI thực tế.
- Các phương pháp phổ biến:
  - One-stage detectors (YOLO): Phát hiện nhanh, hiệu quả cho các Ứng dụng thời gian thực.
  - Two-stage detectors (Faster R-CNN): Thường chính xác hơn nhưng chậm hơn.



# Vấn đề với ID lớp và Giải pháp (Tiền xử lý nhãn)

Thứ hai, 10/01/2023

Chia sẻ bởi: Nguyễn Văn A

**Vấn đề:** Khi thực hiện việc gán nhãn một cách thủ công hoặc sử dụng các công cụ/dataset khác nhau, ID của các lớp có thể không đồng nhất. Ví dụ, có thể xuất hiện lớp không mong muốn hoặc ID không bắt đầu từ 0 liên tục.

Khi trả về giá trị Ground Truth, mô hình sẽ cung cấp ID lớp theo cách mà nó đã được đào tạo.

# Tại sao lại là vấn đề?

---

Mô hình cần các ID lớp liên tục trùng với dataset đã train và bắt đầu từ 0 để hoạt động chính xác.

Việc tồn tại các lớp không mong muốn (ví dụ: 'auto' từ một số dataset) có thể gây nhiễu loạn trong quá trình huấn luyện và đánh giá.

# Giải pháp

# Các bước thực hiện:

- Định nghĩa Ánh xạ ID: Tạo một từ điển để chuyển đổi từ ID cũ sang ID mới (loại bỏ lớp 'auto' và đánh số lại từ 0 đến 7 cho các lớp mong muốn).
- Duyệt và Chỉnh sửa File .txt
- Đọc từng dòng trong file nhãn YOLO gốc.
- Nếu class\_id là 0 (tương ứng với 'auto'), hãy bỏ qua dòng đó.
- Ánh xạ class\_id cũ (1-8) sang class\_id mới (0-7) theo bảng định nghĩa đã có.
- Ghi các dòng đã cập nhật vào thư mục nhãn mới (labels\_processed).
- Cập nhật File classes.txt: Tạo một file classes\_final.txt chỉ chứa 8 tên lớp mong muốn theo đúng thứ tự ID mới (0 đến 7)

```
with open(input_file_path, 'r', encoding='utf-8') as f: # Đảm bảo encoding
    for line in f:
        parts = line.strip().split()
        if parts:
            try:
                old_class_id = int(parts[0])

                if old_class_id == 0:
                    continue

                # Ánh xạ ID cũ sang ID mới
                if old_class_id in id_mapping_old_to_new:
                    parts[0] = str(id_mapping_old_to_new[old_class_id])
                    updated_lines.append(' '.join(parts))
                else:
                    print(f"Cảnh báo: ID lớp {old_class_id} trong file '{filename}' không có trong bảng ánh xạ")
            except ValueError:
                print(f"Cảnh báo: Không thể chuyển đổi class ID thành số trong file '{filename}', dòng: '{line}'")
        else:
            updated_lines.append(line.strip()) # Giữ nguyên dòng trống
```

Mục đích của đoạn code: Chuyển đổi ID lớp trong các file nhãn YOLO (.txt) và tạo lại file classes.txt để đồng bộ với class của Dataset CoCo mà Yolo đã train nhằm đồng bộ

# Quá trình thực hiện

## Tải và Giải nén Dataset (Trong môi trường Colab)

- Kết nối Google Drive: Mount Drive để truy cập các file .zip dataset.
- Định nghĩa đường dẫn: Xác định vị trí của các file yolo\_dataset.zip và rcnn\_dataset.zip trên Drive.
- Tạo thư mục đích: Tạo các thư mục tạm thời trong Colab (/content/yolo\_temp\_extract, /content/rcnn\_temp\_extract) để giải nén.
- Giải nén: Sử dụng lệnh !unzip để giải nén dataset YOLO và R-CNN vào các thư mục tương ứng.
- Cập nhật Đường dẫn Gốc: Định nghĩa các biến YOLO\_CUSTOM\_DATASET\_ROOT và RCNN\_CUSTOM\_DATASET\_ROOT trở đến thư mục chứa ảnh và nhãn đã giải nén.

# Đánh giá Mô hình YOLOv8

# Giới thiệu YOLOv8:

Mô hình phát hiện đối tượng thế hệ mới nhất của dòng YOLO.

Được biết đến với tốc độ và độ chính xác cân bằng.

Là "one-stage detector": dự đoán bounding box và class cùng lúc.

Các bước thực hiện:

Cài đặt ultralytics: Thư viện cần thiết để sử dụng YOLOv8

Tải mô hình YOLOv8 pre-trained: Sử dụng YOLO('yolov8n.pt') để tải một phiên bản mô hình đã được huấn luyện trên dataset lớn (COCO).

Định nghĩa các lớp tùy chỉnh: Chỉ định danh sách 8 lớp giao thông của chúng ta.

Tạo file dataset.yaml:

Chạy Đánh giá (model.val()):

# Các chỉ số đánh giá quan trọng:

mAP@0.5 (mean Average Precision at IoU=0.5 - Pascal VOC metric): Thường được dùng để đánh giá chung hiệu suất phát hiện.

mAP@0.5:0.95 (COCO metric): Giá trị mAP trung bình trên nhiều ngưỡng IoU, đánh giá độ chính xác của cả vị trí và phân loại.

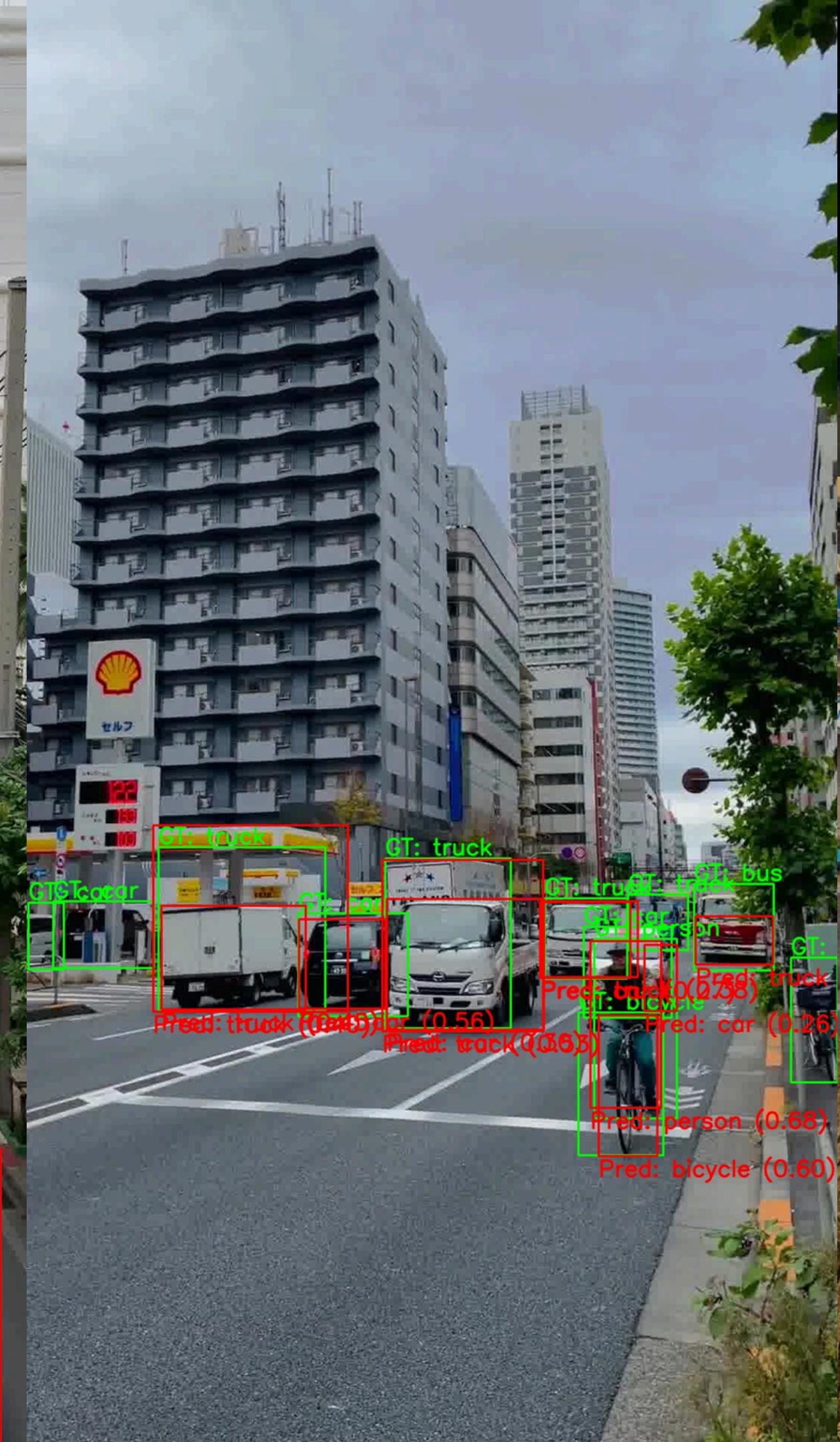
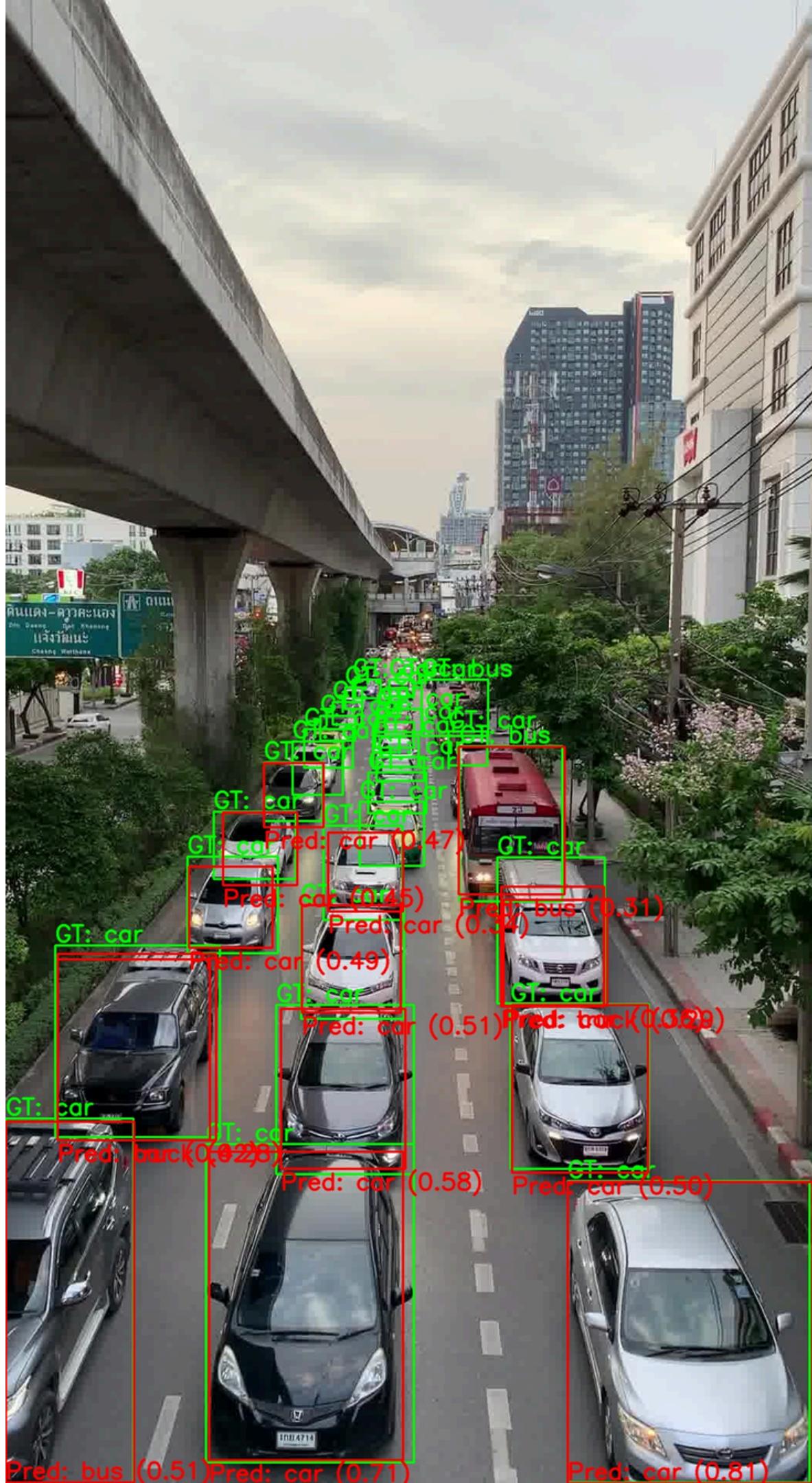
Precision: Tỷ lệ số lượng dự đoán đúng trên tổng số dự đoán.

Recall: Tỷ lệ số lượng đối tượng được phát hiện đúng trên tổng số đối tượng thực tế.

# Kết quả mAP (YOLOv8):

mAP@0.5 (Pascal VOC metric): 0.1801  
mAP@0.5:0.95 (COCO metric): 0.1111  
Precision (overall): 0.4853  
Recall (overall): 0.1855

- Hiệu suất tổng thể (mAP): Rất thấp.
- mAP@0.5: 0.1801 (Thấp, cho thấy khả năng phát hiện tổng thể chưa tốt, kể cả với IoU lỏng).
- mAP@0.5:0.95: 0.1111 (Cực kỳ thấp, cho thấy cả vị trí và phân loại đều kém chính xác).
- Độ chính xác (Precision): Tạm chấp nhận được.
- Tổng thể: 0.4853. Khoảng một nửa số dự đoán là đúng, còn lại là sai.
- Độ phủ (Recall): Rất kém.
- Tổng thể: 0.1855. Mô hình bỏ sót phần lớn các đối tượng thực tế trong ảnh.



# Đánh giá Mô hình Faster R-CNN

# Giới thiệu Faster R-CNN:

Mô hình phát hiện đối tượng "two-stage detector" tiên tiến.

Giai đoạn 1:  
Region Proposal Network (RPN) để xuất các vùng có thể chứa đối tượng.

Giai đoạn 2:  
Trích xuất đặc trưng từ các vùng đề xuất và phân loại, tinh chỉnh bounding box.

Thường có độ chính xác cao nhưng tốc độ chậm hơn YOLO.

## Các bước thực hiện:

- Tải mô hình Faster R-CNN pre-trained: Sử dụng fasterrcnn\_resnet50\_fpn\_v2 từ torchvision với trọng số mặc định (được huấn luyện trên COCO)
- Tạo một COCOSimpleDataset class để đọc file annotations.json của.
- Duyệt qua dataset, cho mô hình Faster R-CNN dự đoán trên từng ảnh.
- Quan trọng: Ánh xạ lại các ID lớp dự đoán từ COCO gốc của mô hình sang ID 0-7 của bạn trước khi đưa vào metric.
- Tính toán mAP, Precision, Recall: Sử dụng metric.compute() để lấy các chỉ số.

# Kết quả mAP (Faster R-CNN)

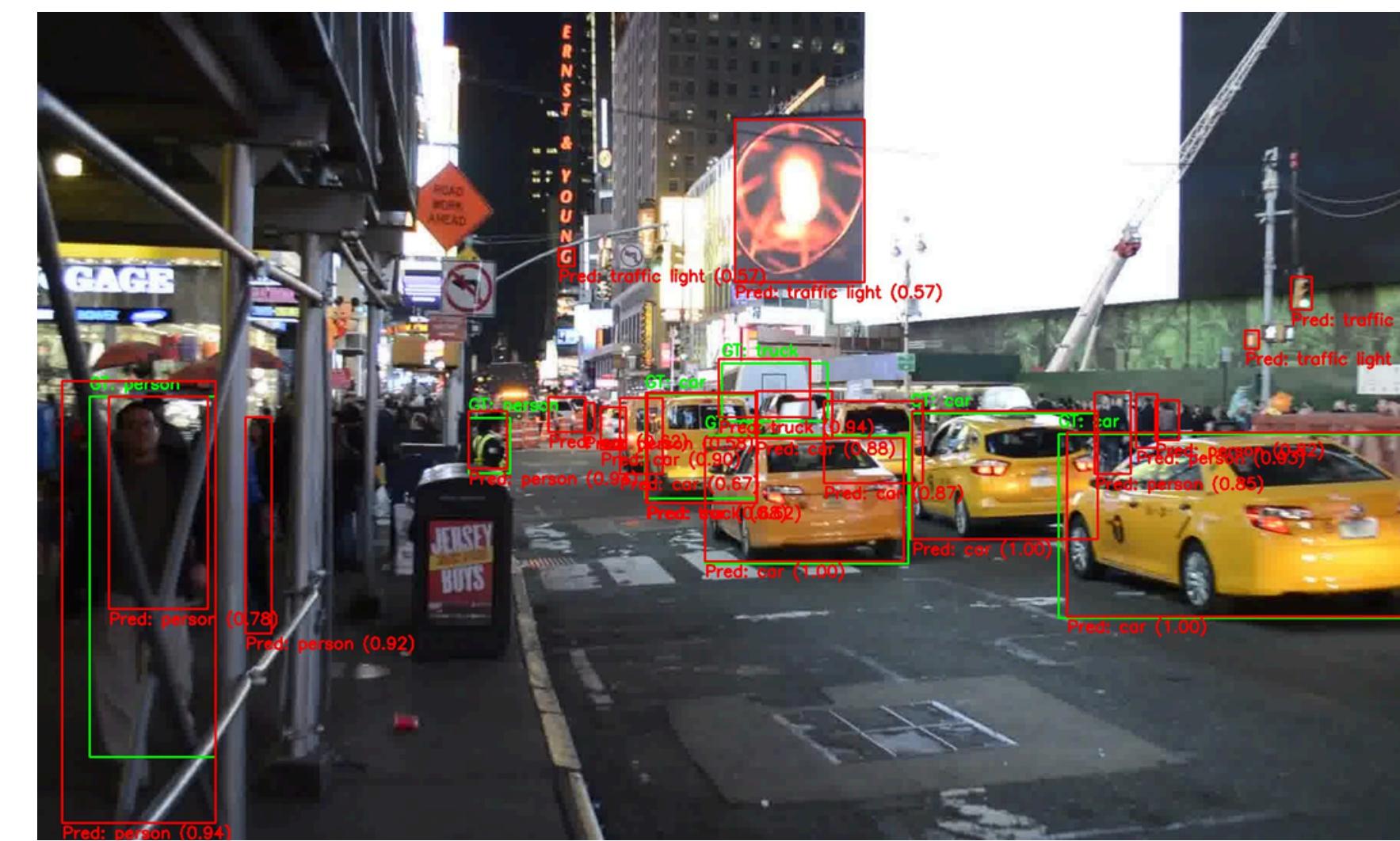
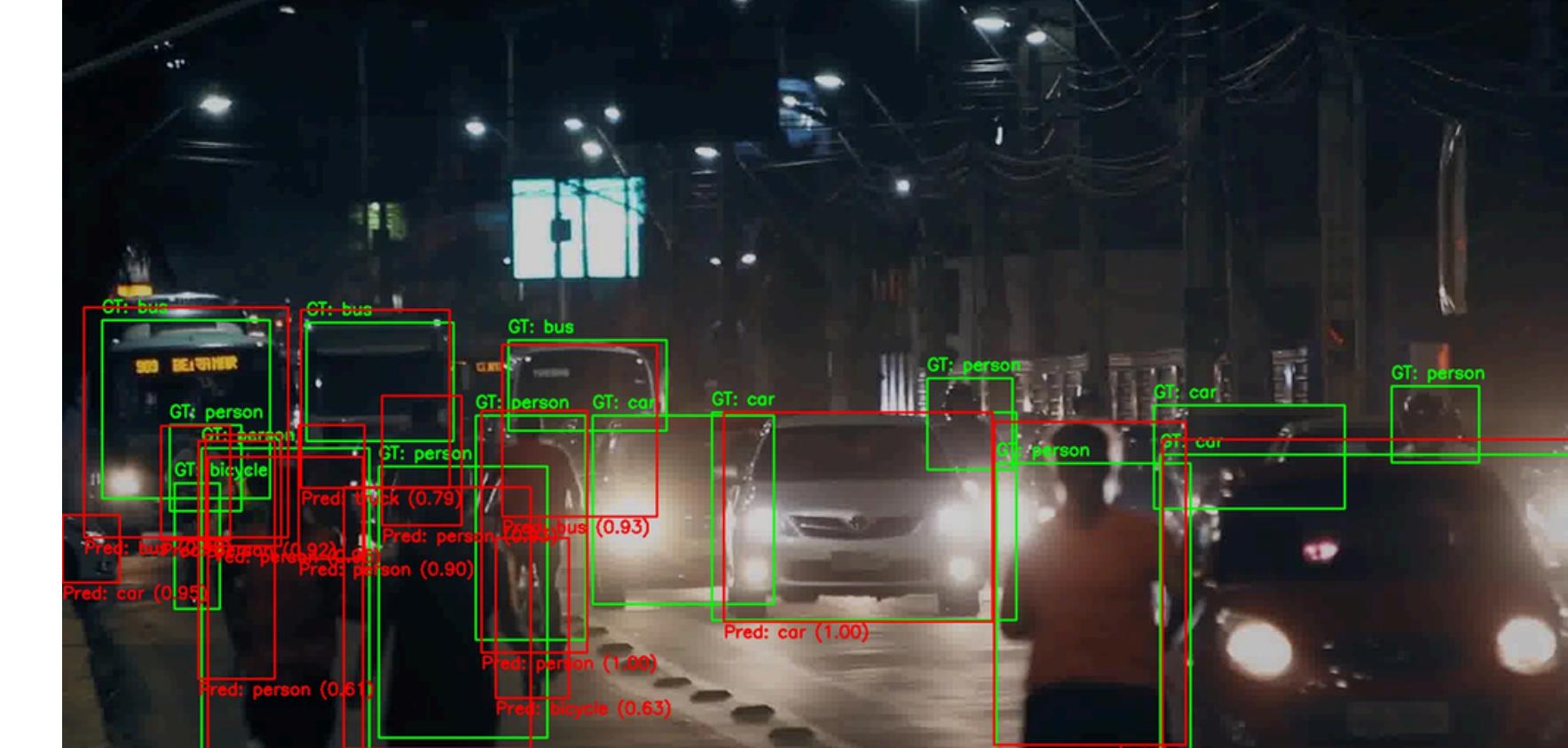
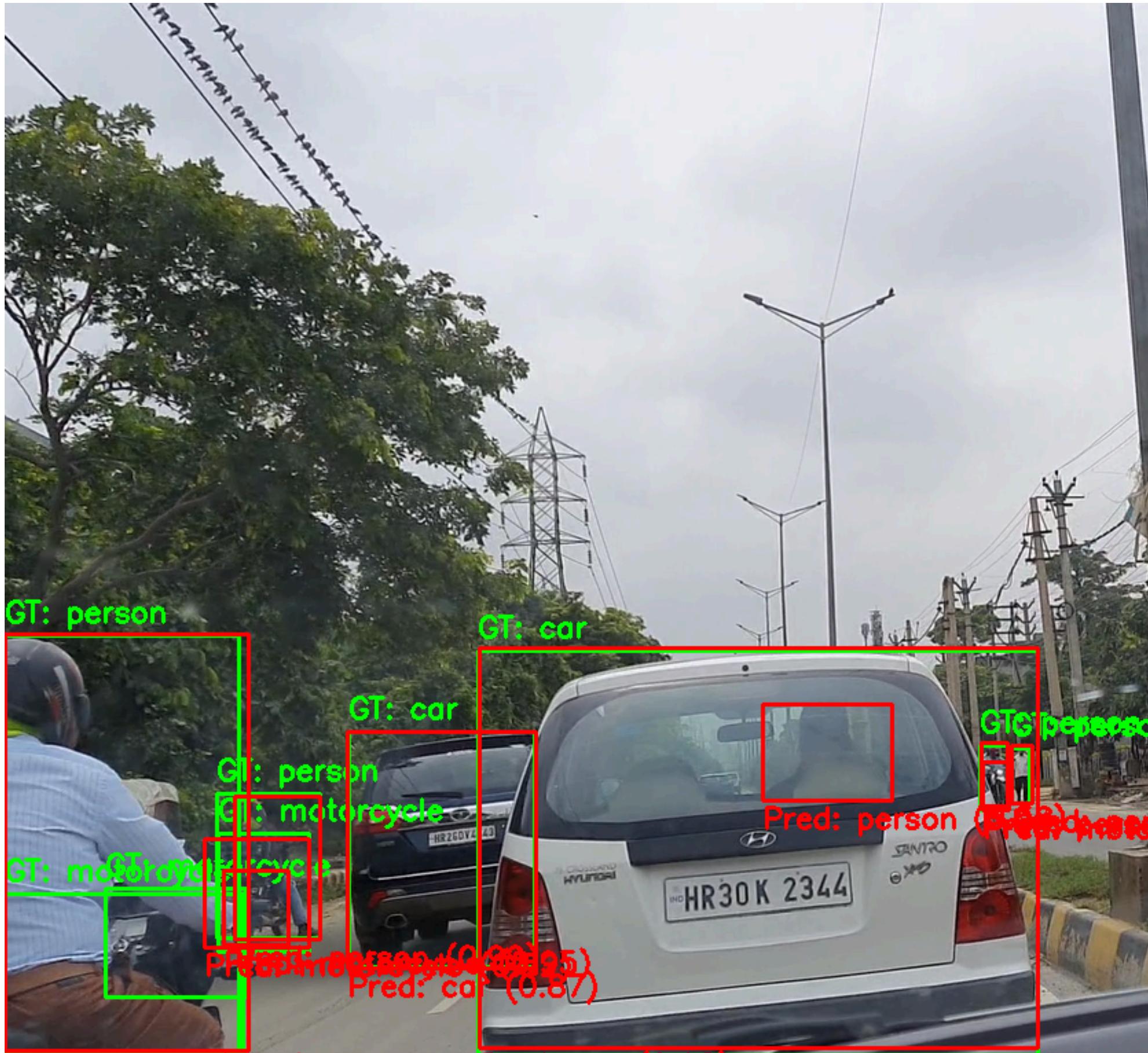
--- Faster R-CNN Evaluation Results ---

mAP@0.5 (Pascal VOC): 0.4967

mAP@0.5:0.95 (COCO): 0.2872

Average Recall (AR) with max 100 detections: 0.4595

- Đánh giá kết quả Faster R-CNN:
- mAP@0.5 (Pascal VOC): 0.4967 Đây là một kết quả khá tốt, gần với ngưỡng 0.5.
- mAP@0.5:0.95 (COCO): 0.2872 Đây là một kết quả tương đối tốt đối với một mô hình pre-trained trên một dataset tùy chỉnh nhỏ
- Average Recall (AR) with max 100 detections: 0.4595 Chỉ số Average Recall là khá tốt. Nó có nghĩa là mô hình có khả năng thu hồi (phát hiện) được gần một nửa số đối tượng thực tế có trong ảnh.



# So sánh Kết quả YOLOv8 và Faster R-CNN

Chỉ số / Đặc điểm	YOLOv8 (Pre-trained)	Faster R-CNN (Pre-trained)	Giải thích
mAP@0.5 (Pascal VOC)	0.1801 (Thấp)	0.4967 (Khá tốt)	Điểm chính xác chung (IoU > 50%). <b>Faster R-CNN vượt trội hơn nhiều.</b>
mAP@0.5:0.95 (COCO)	0.1111 (Rất thấp)	0.2872 (Tương đối tốt)	Điểm chính xác vị trí và phân loại chặt chẽ hơn. <b>Faster R-CNN vượt trội hơn nhiều.</b>
Recall (khả năng phát hiện)	0.1855 (Rất kém)	0.4595 (Khá tốt)	Tỷ lệ đối tượng thực tế được phát hiện. <b>Faster R-CNN phát hiện được nhiều hơn đáng kể.</b>

# Nhận xét:

- YOLOv8 thường nhanh hơn, phù hợp cho các ứng dụng thời gian thực.
- Faster R-CNN thường chính xác hơn (đặc biệt là mAP@0.5:0.95), đặc biệt với các đối tượng nhỏ hoặc bị che khuất một phần.
- So sánh trực quan các ảnh demo để thấy sự khác biệt về chất lượng bounding box và số lượng phát hiện.
  - [Dán các ảnh so sánh từ phần demo của bạn vào đây, ảnh có cả GT và Pred box]

Slide 10: Tầm  
quan trọng của  
YOLOv8 và  
Faster R-CNN  
trong Bài toán  
Object  
Detection

# YOLOv8:

Tốc độ: Là lựa chọn hàng đầu cho các ứng dụng yêu cầu xử lý nhanh (ví dụ: giám sát giao thông trực tiếp, hệ thống ADAS trong xe tự lái).

Hiệu quả tài nguyên: Phiên bản n (nano) nhẹ, có thể chạy trên các thiết bị có tài nguyên hạn chế.

Ưu điểm: Khả năng phát hiện nhiều đối tượng trong một khung hình rất nhanh.

# Faster R-CNN:

**Độ chính xác cao:** Đặc biệt với các phiên bản mạnh hơn, Faster R-CNN thường đạt mAP cao hơn, phù hợp cho các ứng dụng cần độ tin cậy cao về vị trí và phân loại đối tượng (ví dụ: nhận dạng biển báo, phân tích tai nạn giao thông).

**Xử lý các trường hợp khó:** Có khả năng xử lý tốt hơn các đối tượng nhỏ, chồng chéo, hoặc bị biến dạng nhờ cơ chế hai giai đoạn.

# Kết luận:

Việc lựa chọn mô hình phụ thuộc vào yêu cầu cụ thể của ứng dụng: ưu tiên tốc độ hay độ chính xác.

Cả hai mô hình đều là những công cụ mạnh mẽ, tạo nên nền tảng vững chắc cho sự phát triển của các hệ thống AI trong lĩnh vực giao thông minh.

# Demo Trực quan