

Vehicle Counting AI

Báo cáo đồ án Nhập môn Trí tuệ nhân tạo

Sinh viên thực hiện:

Nguyễn Minh Anh - 23001495

Nguyễn Trung Kiên - 23001530

Nguyễn Thế Quang - 23001549

Trần Đăng Tài - 23001558

Nguyễn Doãn Toàn - 23001564

Giảng viên: TS. Hoàng Anh Đức

Khoa Toán - Cơ - Tin học

Trường Đại học Khoa học Tự nhiên - ĐHQGHN

Hà Nội, ngày 8 tháng 12 năm 2025

Nội dung trình bày

1 Giới thiệu

2 Cơ sở lý thuyết

3 Kết quả thực nghiệm

4 Kết luận

Giới thiệu bài toán

1.1 Tổng quan Hệ thống

HỆ THỐNG GIAO THÔNG THÔNG MINH

Giám sát lưu lượng thời gian thực & Trợ lý luật giao thông ảo

INPUT

- Video camera (CCTV)
- Câu hỏi văn bản (Text)

CORE AI

- Computer Vision (YOLO)
- NLP (RAG / LLM)

OUTPUT

- Dashboard realtime
- Tư vấn luật giao thông tự động

Tính năng nổi bật:

- **Giám sát thông minh:** Đếm xe, phân loại phương tiện, theo dõi mật độ, phát hiện tắc đường.
- **Trợ lý pháp lý ảo:** Trả lời chính xác câu hỏi về luật, kèm trích dẫn văn bản pháp quy.

1.2 Bài toán đặt ra & Thách thức

● Bối cảnh thực tế:

- Nhu cầu cấp thiết về dữ liệu lưu lượng để quy hoạch và cảnh báo ùn tắc tại các đô thị lớn (Hà Nội, TP.HCM).
- **Hạn chế giải pháp cũ:** Đêm thủ công tốn kém nhân lực; Cảm biến vật lý chi phí lắp đặt/bảo trì cao.
- **Giải pháp AI:** Tận dụng hạ tầng Camera giám sát có sẵn → Tiết kiệm, độ chính xác cao.

● Các thách thức chính của đề tài:

- **Nhận diện đa dạng:** Phân loại chính xác 4 loại xe trong điều kiện thời tiết/ánh sáng/góc quay phức tạp tại VN.
- **Chính xác dữ liệu:** Đếm đúng qua vùng ROI, xử lý bài toán phương tiện bị che khuất và xuất hiện nhiều lần (tránh đếm trùng).
- **Hiệu năng Realtime:** Xử lý song song nhiều Camera đồng thời với độ trễ thấp.
- **Triển khai:** Xây dựng hệ thống Full-stack hoàn chỉnh, ổn định.

Cơ sở lý thuyết

2.1 Phương pháp

2.1.1 Lý thuyết về Mạng Nơ-ron Tích chập

Tổng quan YOLOv8m

YOLOv8m về bản chất là một mạng CNN sâu, được cấu thành từ các khối xây dựng cơ bản nhằm chuyển đổi dữ liệu hình ảnh thô thành các đặc trưng có ý nghĩa.

Các thành phần cốt lõi:

- **Lớp Tích chập:** Đóng vai trò là bộ trích xuất đặc trưng. Phát hiện từ các đặc trưng đơn giản (cạnh, góc, màu sắc) đến đặc trưng phức tạp (bánh xe, khuôn mặt).
- **Lớp Gộp:** Sử dụng kỹ thuật Max Pooling để giảm kích thước dữ liệu.
 - Giảm tham số tính toán, chống quá khớp (Overfitting).
 - Giữ đặc trưng quan trọng, đảm bảo tính bất biến với dịch chuyển nhỏ.
- **Hàm kích hoạt (SiLU):** Định nghĩa bởi công thức $f(x) = x \cdot \sigma(x)$. Đảm bảo tính phi tuyến tính, giúp mạng học được các dữ liệu phức tạp mà biến đổi tuyến tính không làm được.

2.1.2 Kiến trúc chi tiết YOLOv8m

Hệ thống gồm 3 khối chức năng chính:

1. Backbone (Trích xuất đặc trưng)

- **CSPDarknet + C2f Module:** Chia luồng tín hiệu thành hai nhánh (tính toán sâu & đi tắt) giúp làm giàu gradient.
- **SPPF:** Mở rộng trường tiếp nhận nắm bắt được ngữ cảnh toàn cảnh của vật thể.

2. Neck (Hợp nhất đặc trưng)

- **PANet:** Trộn lẫn đặc trưng ngữ nghĩa (tầng sâu) và vị trí (tầng nông).
- Tăng khả năng nhận diện các vật thể ở nhiều kích thước khác nhau.

3. Head (Dự đoán)

- **Decoupled Head:** Tách riêng nhánh Phân loại và Hồi quy hộp.
- **Anchor-free Detection:** Dự đoán trực tiếp tâm vật thể, không dùng khung neo định sẵn.

2.1.3 Phương pháp Fine-tuning

Chiến lược Huấn luyện (Full Fine-tuning)

Khởi tạo mô hình từ trọng số **COCO** (yolov8m.pt), sau đó huấn luyện lại tất cả các layers (bao gồm cả Backbone và Head) trên tập dữ liệu phương tiện giao thông Việt Nam.

Kỹ thuật áp dụng

- **No-freeze:** Cập nhật trọng số toàn bộ mạng, không đóng băng layer nào.
- **Learning Rate:** Khởi tạo thấp ($lr = 0.01$) kết hợp *Cosine Annealing*.
- **Mục tiêu:** Tinh chỉnh nhẹ nhàng, tránh phá vỡ các đặc trưng cũ.

Hiệu quả đạt được

- ✓ **Hội tụ nhanh:** Tận dụng điểm khởi đầu tốt từ COCO.
- ✓ **Học sâu:** Tối ưu cả đặc trưng cấp thấp (Backbone) và cấp cao (Head).
- ✓ **Tổng quát hóa:** Kết hợp *Augmentation & Early Stopping* giúp giảm Overfitting trên 1547 ảnh.

2.1.4 Các chỉ số đánh giá hiệu quả

1. Intersection over Union (IoU)

Đo độ chồng lấn giữa dự đoán (B_p) và thực tế (B_{gt}):

$$\text{IoU} = \frac{\text{Diện tích phần giao}}{\text{Diện tích phần hợp}}$$

2. Precision (Độ chính xác)

Tỉ lệ dự đoán đúng trong số các box mô hình đã vẽ:

$$P = \frac{TP}{TP + FP}$$

3. Recall (Độ nhạy)

Tỉ lệ vật thể tìm được trong số vật thể thực tế có trong ảnh:

$$R = \frac{TP}{TP + FN}$$

4. F1-Score

Chỉ số tổng hợp cân bằng giữa Precision và Recall:

$$F1 = 2 \times \frac{P \times R}{P + R}$$

2.1.4 Các chỉ số đánh giá hiệu quả - mAP

5. Mean Average Precision (mAP)

Trung bình AP trên tất cả các lớp (Quan trọng nhất):

- **mAP@0.5:** Độ chính xác trung bình khi ngưỡng IoU chấp nhận là 0.5.
- **mAP@0.5:0.95:** Trung bình cộng của mAP tại các ngưỡng IoU tăng dần từ 0.5 đến 0.95 (bước nhảy 0.05). Chỉ số này đánh giá khắt khe về độ khít của hộp dự đoán.

2.1.5 Dữ liệu và Tiền xử lý

Tổng quan bộ dữ liệu

- **Nguồn:** Roboflow Universe – Vietnamese vehicle Computer Vision Model (1547 ảnh).
- **Đặc điểm:** Đa dạng ánh sáng và môi trường thực tế.
- **4 Lớp mục tiêu:** Car, Motor, Truck, Bus.

Tiền xử lý (Preprocessing)

- **Resize:** Chuẩn hóa về 640×640 pixels.
- **Lọc và gán lại nhãn:** Quy hoạch nhãn về 4 lớp đích.
- **Phân chia dữ liệu:** Chia thành 3 tập con tỷ lệ 80:10:10.

Mẫu dữ liệu thực tế sau khi gán nhãn:



Car



Motor



Truck



Bus

2.1.6 Kết quả huấn luyện mô hình

- **Cấu hình Huấn luyện:**

- **Mô hình:** Fine-tuning YOLOv8m (từ COCO)
- **Tham số:** 87 Epochs, Batch 16, Input 640x640
- **LR:** 0.01 (Cosine annealing)

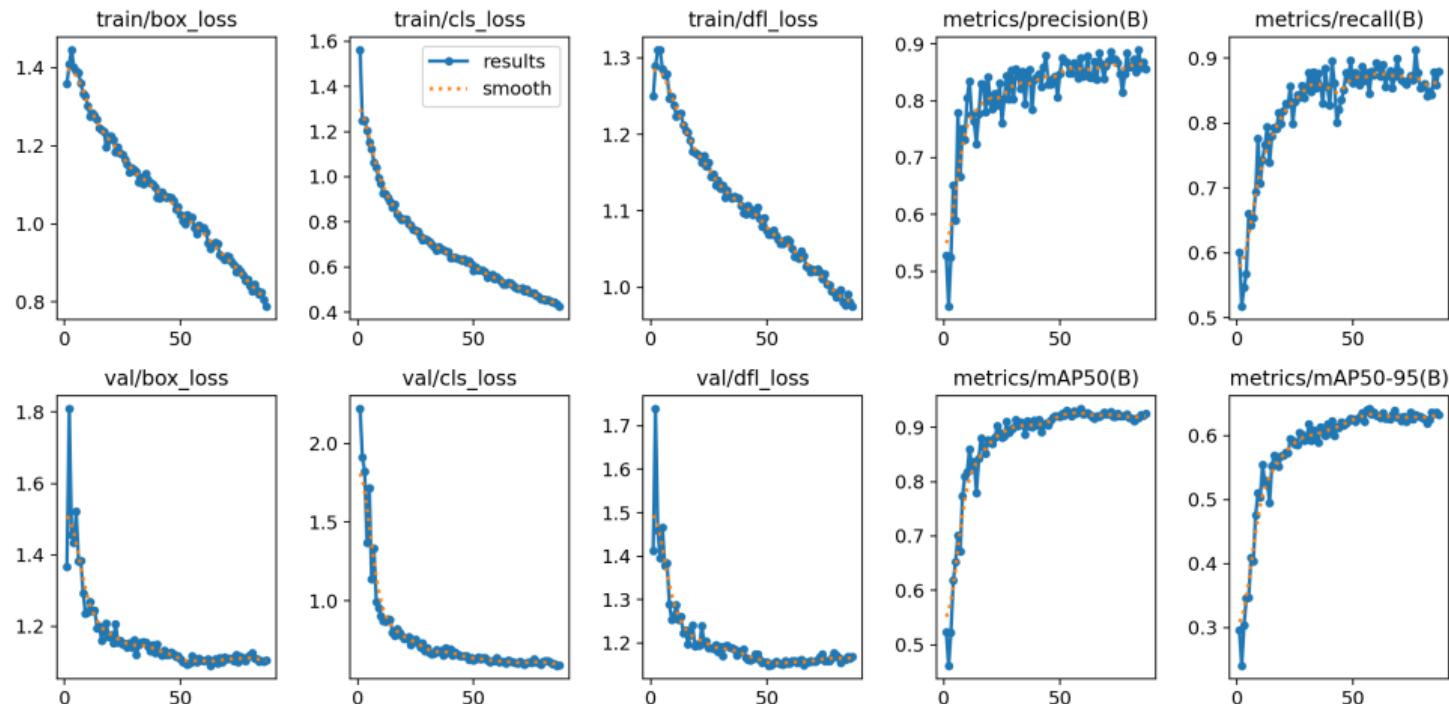
- **Kết quả đánh giá trên tập Test:**

Chỉ số	Giá trị
Precision	85.62%
Recall	87.95%
F1-Score	86.77%
mAP@0.5	92.49%
mAP@0.5:0.95	63.21%

- **Phân tích:**

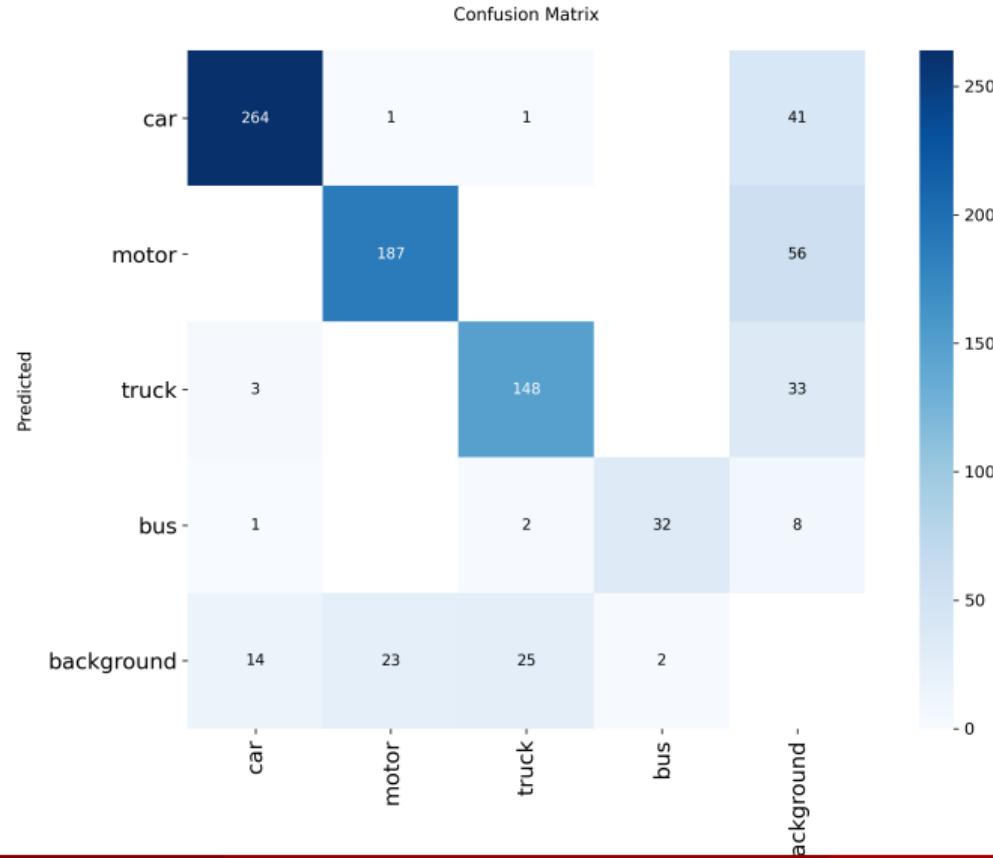
- **Hội tụ:** Loss ổn định sau **30-40 epochs**
- **Confusion Matrix:** Phân loại tốt 4 lớp; nhầm lẫn thấp (Car/Truck)

2.1.8 Kết quả huấn luyện mô hình - Biểu đồ tiến trình



Biểu đồ tiến trình huấn luyện mô hình qua các epochs

2.1.9 Kết quả huấn luyện mô hình - Ma trận nhầm lẫn



2.1.9 Phân tích hiệu năng theo từng Lớp

Lớp (Class)	Precision	Recall	mAP@0.5	mAP@0.5:0.95
Bus	90.91%	88.24%	93.31%	72.50%
Car	90.84%	87.88%	92.06%	75.60%
Truck	83.31%	86.93%	88.26%	66.31%
Motor	83.35%	80.64%	85.48%	51.08%

2.2.1 Cơ sở lý thuyết: Kỹ thuật RAG

Khái niệm (Definition)

RAG (Retrieval-Augmented Generation) là kỹ thuật tối ưu hóa kết quả đầu ra của Mô hình ngôn ngữ lớn (LLM) bằng cách truy xuất thông tin từ một **cơ sở tri thức bên ngoài** (External Knowledge Base) trước khi sinh câu trả lời.

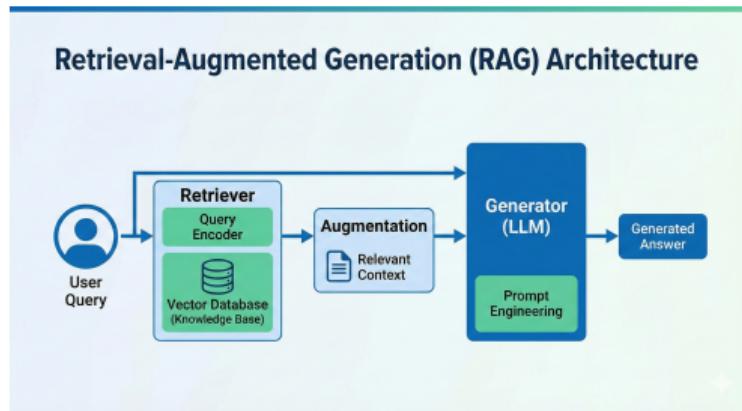
1. Vấn đề của LLM truyền thống

- **Ảo giác (Hallucination):** AI tự bịa ra thông tin sai lệch nhưng văn phong rất thuyết phục.
- **Dữ liệu lỗi thời:** LLM chỉ biết kiến thức đến thời điểm được huấn luyện (Training Cut-off).
- **Thiếu kiến thức chuyên ngành:** Không biết các văn bản luật cụ thể hoặc dữ liệu nội bộ.

2. Cơ chế hoạt động (Mechanism)

- **Retrieval (Truy xuất):** Tìm kiếm các đoạn văn bản liên quan nhất với câu hỏi từ Vector Database.
- **Augmentation (Tăng cường):** Ghép các đoạn văn bản đó vào prompt làm "ngữ cảnh"(Context).
- **Generation (Tạo sinh):** LLM trả lời câu hỏi dựa trên ngữ cảnh vừa được cung cấp.

2.2.2 Kiến trúc hệ thống RAG



Hình: Sơ đồ luồng dữ liệu RAG

Quy trình hoạt động chính:

- **Truy xuất (Retrieval):** Câu hỏi người dùng được vector hóa để tìm kiếm các văn bản luật liên quan nhất trong Vector Database (ChromaDB).
- **Tăng cường & Tạo sinh (Augmentation & Generation):** Các văn bản luật tìm được sẽ đóng vai trò là ngữ cảnh (Context), được gửi kèm với câu hỏi gốc đến LLM (Gemini) để sinh ra câu trả lời chính xác.

2.2.3 Logic đếm số phương tiện

1. Logic Đếm (Counting Logic)

- **Tracking (Theo vết):** Gán và duy trì **ID duy nhất** cho phương tiện qua các khung hình (YOLOv8 Track).
- **Kiểm tra Vùng (ROI Check):** Sử dụng thuật toán *Ray Casting* để xác định vị trí tâm xe (Center Point).
- **Máy trạng thái (State Machine):** Đây là cơ chế chống đếm trùng cốt lõi:
 - **Trạng thái:** Lưu lịch sử vị trí (Trong/Ngoài).
 - **Sự kiện đếm:** Chỉ kích hoạt khi trạng thái chuyển từ **NGOÀI → TRONG**.

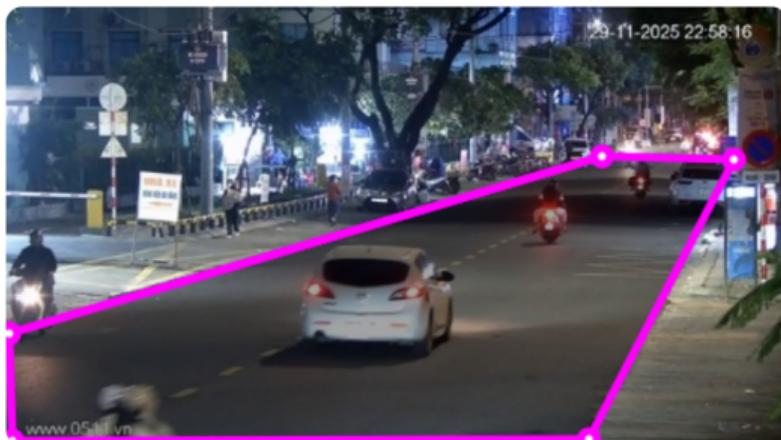
2. Chiến lược Tối ưu (Optimization)

- **Xử lý đầu vào (Input Processing):**
 - **Resize:** Giảm độ phân giải xử lý xuống 854x480 (SD) để tăng tốc độ suy luận.
 - **Skip Frame ($k = 3$):** Chỉ xử lý AI trên mỗi 3 frame, giảm 66% tải CPU.
- **Kiến trúc hệ thống:**
 - **Multiprocessing:** Mỗi Camera chạy trên một tiến trình riêng biệt, tận dụng đa nhân CPU.
 - **Shared Memory:** Truyền dữ liệu giữa các tiến trình mà không gây độ trễ.

2.2.4 Minh họa vùng ROI

Mục đích thiết lập ROI

Định nghĩa khu vực cụ thể để thực hiện đếm xe, giúp **loại bỏ nhiễu** từ vỉa hè, cây cối và các đối tượng không tham gia giao thông.



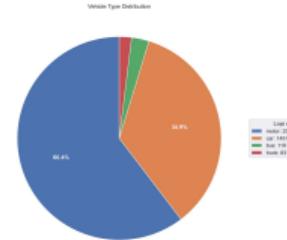
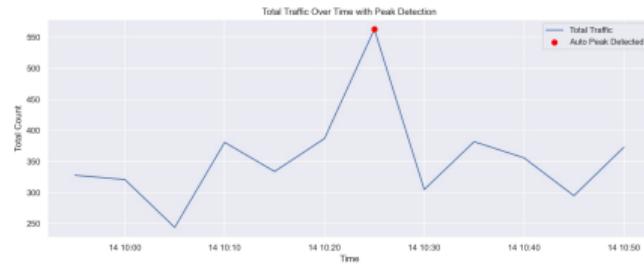
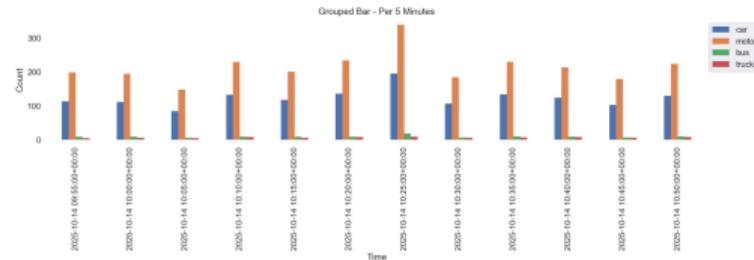
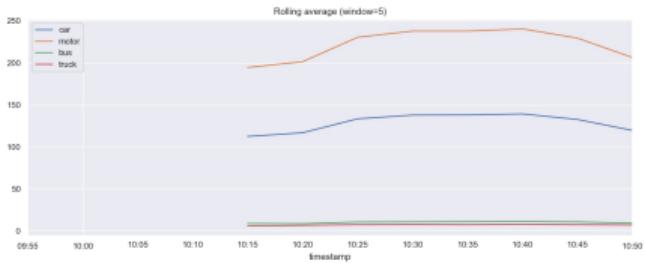
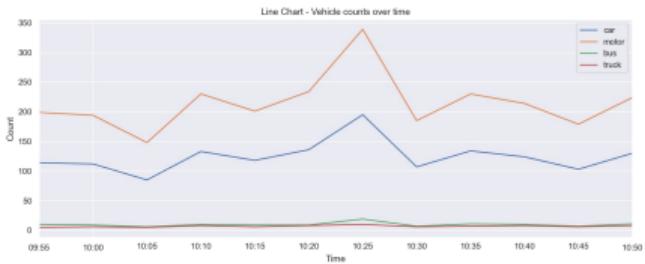
Hình: ROI Camera 01 (Đa giác bao quanh lòng đường)



Hình: ROI Camera 02 (Vùng phát hiện tại nơi khó phân biệt lòng đường)

2.2.5 Phân tích dữ liệu

- **Nguồn Dữ liệu Thô (DB/traffic_logs)**
 - Ghi nhận bởi AnalyzeOnRoadBase (Video Processing) và Background Worker.
 - Cột: timestamp (UTC), count_car, total_vehicles, v.v.
- **Quy trình Xử lý Backend (Python/Pandas)**
 - Đọc trực tiếp từ DB, chuyển Timezone → UTC+7.
 - **Resample (1min/5min)**: Tính lưu lượng thực tế (diff trên total_vehicles).
- **Các Thuật toán & Thông kê Chính**
 - Tính Rolling Mean/Std và Thông kê tóm tắt (histogram, boxplot).
 - **Phát hiện Đỉnh (Peaks)**: Dùng quantile hoặc 3-sigma.
 - Phân tích Tỷ lệ từng loại xe.
- **Đầu ra và Công cụ Hỗ trợ**
 - Kết quả được trả về dưới dạng JSON qua API cho Frontend.
 - Thư mục analysis/ là mô-đun hỗ trợ phân tích offline, không nằm trong luồng server production.



2.3.1 Kiến trúc tổng thể Frontend

Mục tiêu

Xây dựng giao diện giám sát giao thông thời gian thực:

- Độ trễ thấp, ổn định
- Hiển thị trực quan: video – thống kê – biểu đồ
- Dễ mở rộng theo module

Công nghệ sử dụng

Next.js 14 (App Router), React 18, TailwindCSS, WebSocket, Recharts, Lucide Icons

Luồng hoạt động

- ① Nhận **video** từ WebSocket
- ② Nhận **thống kê realtime**
- ③ Gọi API **dữ liệu lịch sử** (Chart)
- ④ Kết hợp và hiển thị trên Dashboard

2.3.2 Kiến trúc giao diện và phân cấp Component

Các module chính

- **Sidebar**: Cấu hình AI Engine, tham số hệ thống
- **VideoPlayer**: Nhận và hiển thị video WebSocket
- **RealtimeStats**: Thống kê số lượng phương tiện theo thời gian thực
- **Charts**: Line, Area, Histogram, Boxplot, Rolling Avg,...
- **ChatBubble**: Tương tác hỏi–đáp (RAG)

Ưu điểm thiết kế

- Tổ chức theo **component-based**, dễ tái sử dụng
- Hạn chế re-render nhờ tách logic WebSocket khỏi UI
- Dễ thêm camera, thêm biểu đồ mới mà không ảnh hưởng hệ thống

2.4.1 Kiến trúc tổng quan Backend hệ thống

Mô hình Client–Server giám sát giao thông

- **Client (Frontend Web):** Giao diện dashboard giúp người vận hành theo dõi lưu lượng, biểu đồ, khung hình camera thời gian thực.
- **Backend API:** Cung cấp RESTful API cho thống kê, báo cáo lịch sử và WebSocket cho dữ liệu realtime.
- **Worker AI (YOLOv8):** Các tiến trình độc lập nhận video đầu vào, thực hiện nhận dạng—theo vết và đẩy kết quả sang Backend.
- **Cơ sở dữ liệu:** Lưu trữ log giao thông vào cơ sở dữ liệu, metadata nhận dạng và lịch sử tương tác chatbot.

Luồng dữ liệu tổng quát

Camera → Worker YOLOv8 → Shared Memory/Queue → Backend API → DB & Web Dashboard

2.4.2 Cơ sở lý thuyết RESTful API

- Dựa trên giao thức HTTP với các phương thức: GET, POST, PUT, DELETE.
- Các api được xây dựng rõ ràng. Ví dụ:
 - /api/v1/charts/time-series/{camera_id}
 - /api/v1/charts/vehicle-distribution
- Mỗi request là độc lập , server không lưu trạng thái phiên làm việc của client trong HTTP.
- Phù hợp cho truy vấn lịch sử, thống kê, báo cáo, cấu hình hệ thống không yêu cầu realtime tuyệt đối.

2.4.3 Cơ sở lý thuyết WebSocket

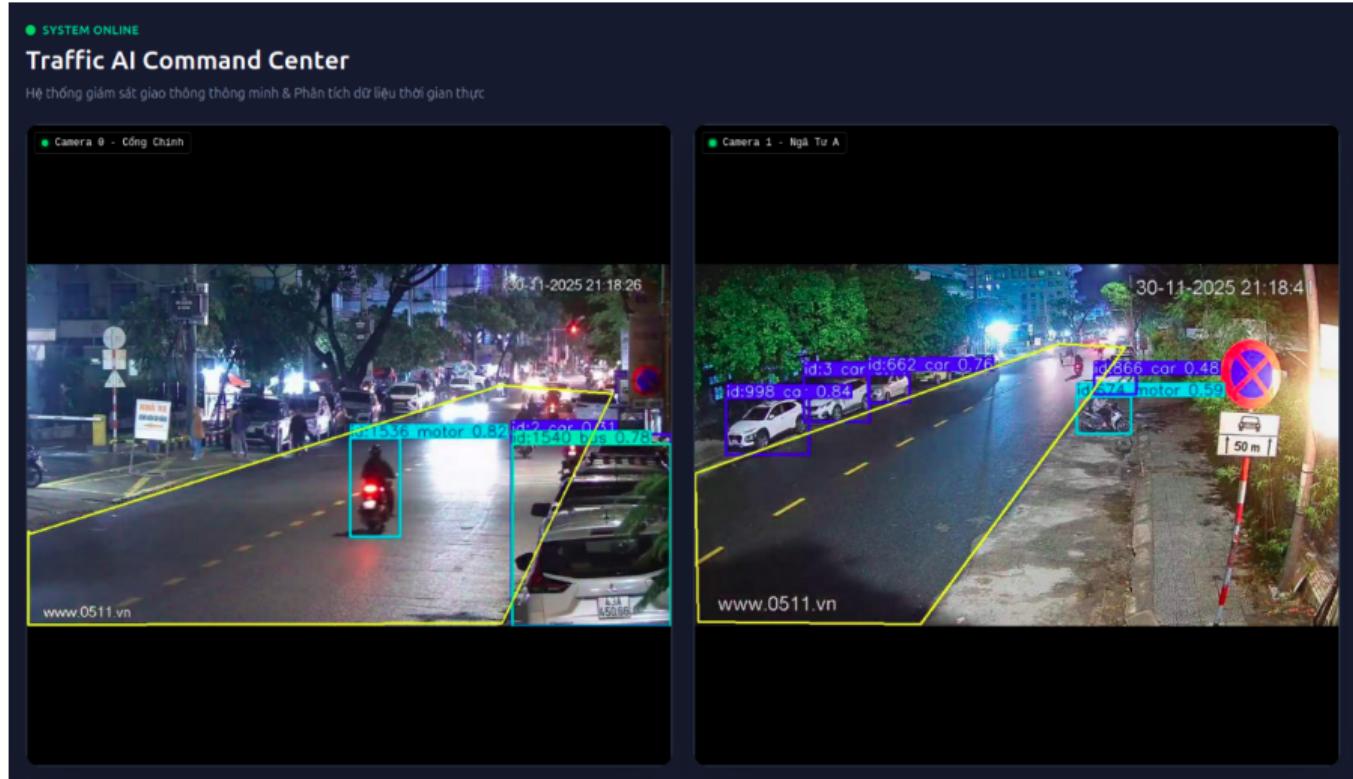
- Là kết nối 2 chiều chạy trên một TCP socket duy nhất.
- Sau khi handshake qua HTTP, kết nối được nâng cấp và giữ mở, cho phép server chủ động đẩy dữ liệu cho client mà không cần client gửi request mới.
- Phù hợp cho các luồng dữ liệu thời gian thực, ví dụ trong hệ thống:
 - Stream khung hình camera: /ws/frames/{camera_id}
 - Stream thống kê đếm xe realtime: /ws/info/{camera_id}

RESTful API cho dữ liệu lịch sử, WebSocket cho dữ liệu thời gian thực.

Kết quả thực nghiệm

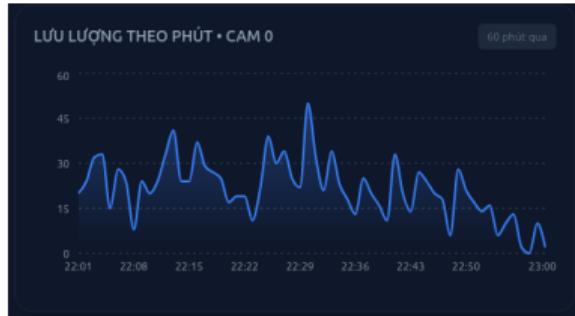
Nguồn dữ liệu thực nghiệm là nguồn camera công khai trực tiếp trên Youtube, khu vực thành phố Đà Nẵng

3.1 Giao diện Giám sát Trung tâm (Main Dashboard)



Hình: Giao diện giám sát đa luồng thời gian thực

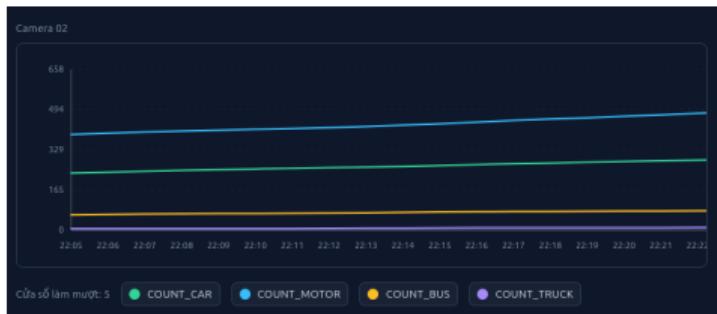
3.2 Kết quả phân tích dữ liệu



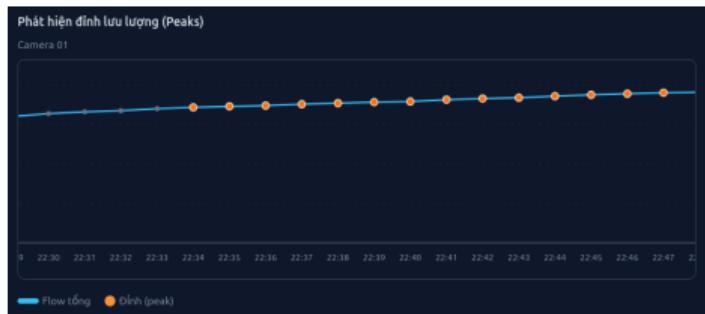
Hình 1: Biểu đồ lưu lượng lũy kế



Hình 3: So sánh số lượng xe theo giờ



Hình 2: Trung bình động (Làm mượt)

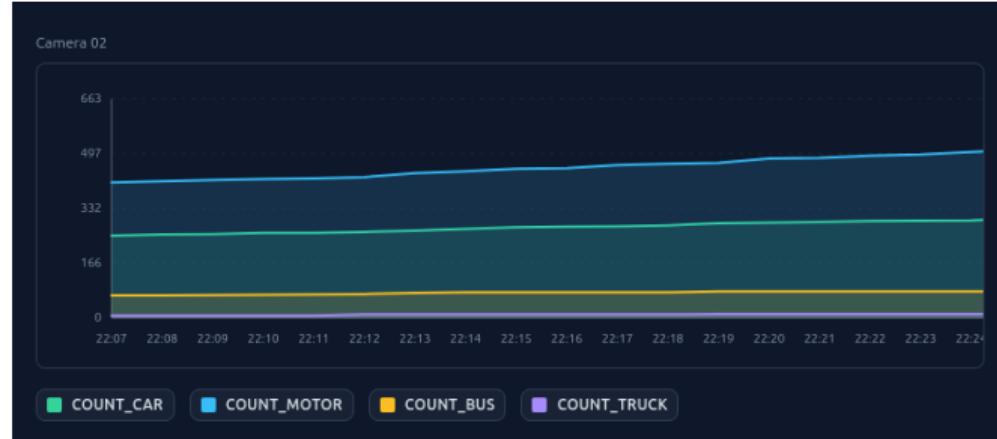


Hình 4: Phát hiện đỉnh lưu lượng (Peak)

3.3 Biểu đồ cơ cấu thành phần



Hình: Tỷ lệ phần trăm các loại xe.



Hình: Phân bố tích lũy lưu lượng theo loại xe.

3.4 Trợ lý ảo Chatbot



Hình: Kịch bản 1: Tra cứu mức phạt vi phạm



Hình: Kịch bản 2: Giải thích luật có trích dẫn nguồn được cung cấp

Kết luận và hướng phát triển

4.1 Kết luận

- **Xây dựng thành công mô hình nhận diện:**

- Ứng dụng phương pháp **Fine-tuning** trên YOLOv8m với dữ liệu giao thông Việt Nam (1547 ảnh).
- Kết quả ấn tượng: mAP@0.5 đạt 92.49%, Precision 85.62%, Recall 87.95%, F1-Score 86.77%, mAP@0.5:0.95 63.21%.

- **Triển khai hệ thống giám sát toàn diện:**

- **Kiến trúc đa tầng:** Tích hợp AI Core, Pipeline phân tích (Hotspot detection), Backend (FastAPI) và Frontend (Next.js).
- **Hiệu năng:** Xử lý Realtime đa camera (10-20 FPS); Độ chính xác đếm xe cao nhờ kỹ thuật Tracking ROI.

- **Đóng góp công nghệ:**

- Chứng minh tính hiệu quả của Fine-tuning đối với bài toán dữ liệu hạn chế.
- Các kỹ thuật tối ưu hóa (Multiprocessing, Frame skipping, Binary-safe I/O) đảm bảo tính ổn định và khả năng mở rộng trong thực tế.

4.2 Hạn chế

- **Chênh lệch dữ liệu:**

- Sự khác biệt lớn về góc quay, ánh sáng giữa tập Training và môi trường thực tế dẫn đến việc bỏ sót phương tiện.

- **Hạn chế với phương tiện tốc độ cao:**

- Thuật toán Tracking có thể mất dấu khi xe di chuyển quá nhanh → Hệ thống gán ID mới, gây ra lỗi đếm trùng.

- **Vấn đề bị che khuất:**

- Độ chính xác giảm khi phương tiện bị che khuất một phần bởi vật thể hoặc các xe khác trong dòng giao thông đồng đúc.

- **Điều kiện môi trường đa dạng:**

- Hiệu năng chưa được tối ưu hoàn toàn trong các điều kiện khắc nghiệt (mưa lớn, sương mù, ban đêm).

4.3 Hướng phát triển

- **Nâng cao dữ liệu và khả năng thích ứng:**

- **Đa dạng hóa dữ liệu:** Bổ sung mẫu ở nhiều góc quay, điều kiện thời tiết (mưa, sương mù) để tăng tính tổng quát.
- **Kỹ thuật:** Áp dụng Domain Adaptation và xử lý ảnh thiếu sáng để cải thiện nhận diện ban đêm.

- **Tối ưu hóa Thuật toán và Hiệu năng:**

- **Tracking:** Nâng cấp lên DeepSORT hoặc các mô hình Deep Learning để bắt phương tiện tốc độ cao chính xác hơn.
- **Edge Computing:** Sử dụng Quantization và Model Pruning để giảm nhẹ mô hình, triển khai trên thiết bị cấu hình thấp.

- **Mở rộng Tính năng và Quy trình vận hành:**

- **Phân tích nâng cao:** Thêm tính năng dự báo lưu lượng, phát hiện tai nạn và vi phạm giao thông.
- **MLOps:** Xây dựng pipeline tự động hóa (Data collection → Retrain → Deploy) để cập nhật mô hình định kỳ.

Cảm ơn Thầy và các bạn đã lắng nghe!