

**TRƯỜNG ĐẠI HỌC GIAO THÔNG VẬN TẢI
PHÂN HIỆU TẠI TP. HỒ CHÍ MINH
BỘ MÔN CÔNG NGHỆ THÔNG TIN**



BÁO CÁO ĐÒ ÁN TỐT NGHIỆP

ĐỀ TÀI

Nghiên cứu bài toán object detection và phát triển hệ thống giám sát giao thông tích hợp mô hình YOLO để phát hiện tai nạn giao thông tại Việt Nam

Giảng viên hướng dẫn : ThS. TRẦN PHONG NHÃ

Sinh viên thực hiện : HOÀNG GIA KIỆT

Lớp : CÔNG NGHỆ THÔNG TIN

Khoa : 62

Tp. Hồ Chí Minh, năm 2025

**TRƯỜNG ĐẠI HỌC GIAO THÔNG VẬN TẢI
PHÂN HIỆU TẠI TP. HỒ CHÍ MINH
BỘ MÔN CÔNG NGHỆ THÔNG TIN**



BÁO CÁO ĐÒ ÁN TỐT NGHIỆP

ĐỀ TÀI

Nghiên cứu bài toán object detection và phát triển hệ thống giám sát giao thông tích hợp mô hình YOLO để phát hiện tai nạn giao thông tại Việt Nam

Giảng viên hướng dẫn : ThS. TRẦN PHONG NHÃ

Sinh viên thực hiện : HOÀNG GIA KIỆT

Mã sinh viên : 6251071049

Lớp : CÔNG NGHỆ THÔNG TIN

Khoa : 62

Tp. Hồ Chí Minh, năm 2025

NHIỆM VỤ THIẾT KẾ TỐT NGHIỆP

BỘ MÔN: CÔNG NGHỆ THÔNG TIN

-----***-----

Mã sinh viên: 6251071049

Họ tên SV: Hoàng Gia Kiệt

Khoa: 62

Lớp: Công nghệ thông tin

1. Tên đề tài

“Nghiên cứu bài toán Object Detection và phát triển hệ thống giám sát giao thông tích hợp mô hình YOLO để phát hiện tai nạn giao thông tại Việt Nam”

2. Mục đích, yêu cầu

a) Mục đích

Hiểu được các khái niệm của Computer Vision, Object Detection

Hiểu được cách huấn luyện mô hình YOLO

Xây dựng bộ dữ liệu tai nạn giao thông tại Việt Nam

Xây dựng hệ thống giám sát giao thông được tích hợp mô hình để phát hiện tai nạn giao thông thời gian thực

b) Yêu cầu

Yêu cầu chức năng:

Mô hình phát hiện được tai nạn giao thông

Lưu trữ thông tin mô hình phát hiện tai nạn

Xem và tìm kiếm các lịch sử phát hiện

Thông báo tai nạn giao thông phát hiện được

Yêu cầu phi chức năng:

Hệ thống xử lý nhanh, chính xác

Giao diện thân thiện với người dùng (nếu có)

3. Nội dung và phạm vi đề tài

a) Nội dung

Tìm hiểu về mô hình YOLO

Quy trình thu thập dữ liệu và huấn luyện mô hình Deep Learning

Phân tích hệ thống giám sát thời gian thực

b) Phạm vi

Tập trung phát triển bộ dữ liệu tai nạn giao thông tại Việt Nam

Hệ thống hướng tới các cơ quan giám sát giao thông

Hệ thống giám sát phát hiện giao thông thời gian thực

4. Công nghệ, công cụ và ngôn ngữ lập trình

Công nghệ sử dụng: YOLO, Apache Kafka, Django...

Công cụ: PyCharm, Visual Studio Code, LabelImg, Docker...

Ngôn ngữ lập trình: Python, Shell

5. Các kết quả chính dự kiến sẽ đạt được và ứng dụng

Quyển báo cáo đồ án tốt nghiệp

Bộ dữ liệu tai nạn giao thông tại Việt Nam

Mô hình phát hiện tai nạn giao thông tại Việt Nam

Hệ thống giám sát phát hiện tai nạn giao thông thời gian thực

6. Giảng viên và cán bộ hướng dẫn

Họ và tên: ThS. Trần Phong Nhã.

Đơn vị công tác: Bộ môn Công nghệ thông tin Phân hiệu Trường Đại học Giao thông Vận tải Phân hiệu tại TP. Hồ Chí Minh.

Điện thoại: 0906761014

Email: tpnha@utc2.edu.vn

Ngày ... Tháng ... năm 2025
Trưởng BM Công nghệ thông tin

Đã giao nhiệm vụ TKTN
Giảng viên hướng dẫn

ThS. Trần Phong Nhã

ThS. Trần Phong Nhã

Đã nhận nhiệm vụ TKTN

Sinh viên: Hoàng Gia Kiệt

Ký tên:

Điện thoại: 0784265174

Email: kiethoang101.utc2@gmail.com

LỜI CẢM ƠN

Lời đầu tiên em xin gửi lời cảm ơn đến Quý thầy cô Bộ môn Công nghệ thông tin Trường Đại học Giao thông Vận tải Phân hiệu tại TP. Hồ Chí Minh đã tận tình giảng dạy, truyền đạt những kiến thức và kỹ năng trong suốt những năm học qua.

Em xin gửi lời cảm ơn chân thành đến ThS. Trần Phong Nhã – Giảng viên hướng dẫn em thực hiện đồ án này. Thầy luôn tận tình chỉ dẫn, hỗ trợ em không chỉ về kiến thức chuyên môn mà còn chia sẻ nhiều kinh nghiệm hữu ích trong công việc và cuộc sống. Nhờ sự giúp đỡ và động viên của thầy, em đã có thể hoàn thành tốt đồ án tốt nghiệp này.

Bên cạnh đó, em xin gửi lời cảm ơn thân thương đến gia đình và bạn bè – những người luôn ở bên động viên, ủng hộ em trong suốt thời gian thực hiện đồ án. Cảm ơn bạn Bùi Tấn Phát (CNTT – K62), em Phan Công Trí và Nguyễn Quốc Việt (CNTT – K64) đã nhiệt tình hỗ trợ trong quá trình thực hiện; Cảm ơn anh Nguyễn Đông Đông và anh Nguyễn Đình Huy (FTel) đã dành thời gian tư vấn, chia sẻ những kiến thức kỹ thuật quý báu, góp phần giúp em hoàn thành tốt đồ án này.

Do thời gian thực hiện và kinh nghiệm còn hạn chế, đồ án tốt nghiệp này chắc chắn sẽ khó tránh khỏi những sai sót. Em rất mong nhận được những góp ý quý báu từ các thầy cô để có thể hoàn thiện hơn, bổ sung kiến thức và nâng cao năng lực phục vụ cho công việc thực tiễn sau này.

Xin chân thành cảm ơn!

TP. Hồ Chí Minh, ngày ... tháng ... năm 2025

Sinh viên thực hiện

Hoàng Gia Kiệt

NHẬN XÉT CỦA GIẢNG VIÊN HƯỚNG DẪN

TP. Hồ Chí Minh, ngày ... tháng ... năm 2025

Giảng viên hướng dẫn

ThS. Trần Phong Nhã

MỤC LỤC

NHIỆM VỤ THIẾT KẾ TỐT NGHIỆP.....	i
LỜI CẢM ƠN	iv
NHẬN XÉT CỦA GIẢNG VIÊN HƯỚNG DẪN.....	v
MỤC LỤC	vi
DANH MỤC VIẾT TẮT	x
DANH MỤC BẢNG BIỂU	xiii
DANH MỤC HÌNH ẢNH	xiv
MỞ ĐẦU.....	1
1. Lý do chọn đề tài.....	1
2. Mục tiêu, nội dung, phương pháp nghiên cứu đề tài	2
2.1. Mục tiêu nghiên cứu.....	2
2.2. Nội dung nghiên cứu	3
2.3. Phương pháp nghiên cứu	3
3. Đối tượng và phạm vi nghiên cứu	3
3.1. Đối tượng nghiên cứu.....	3
3.2. Phạm vi nghiên cứu	3
4. Cấu trúc cuốn báo cáo.....	3
CHƯƠNG 1: CƠ SỞ LÝ THUYẾT.....	5
1.1. Tổng quan về hệ thống giao thông thông minh (ITS)	5
1.1.1. Định nghĩa và mục tiêu	5
1.1.2. Các thành phần chính của ITS.....	5
1.2. Thị giác máy tính và phát hiện đối tượng	6
1.2.1. Khái niệm cơ bản về thị giác máy tính.....	6
1.2.2. Tổng quan về Object Detection.....	7
1.2.3. Các chỉ số đánh giá hiệu suất trong Object Detection.....	8

1.3. Tổng quan về mô hình YOLO.....	10
1.3.1. Lịch sử và sự phát triển của YOLO.....	10
1.3.2. Cách hoạt động của YOLO	10
1.3.3. Kiến trúc của YOLO 11	12
1.3.4. Hiệu suất và khả năng mở rộng	13
1.4. Lập trình song song.....	13
1.4.1. Khái niệm và tầm quan trọng	13
1.4.2. Các mô hình lập trình song song phổ biến	14
1.4.3. Các công cụ và API (OpenMP, MPI).....	15
1.5. Công nghệ Apache Ray	16
1.5.1. Giới thiệu về Apache Ray	16
1.5.2. Lợi ích và tính năng nổi bật của Apache Ray	16
1.5.3. Kiến trúc Apache Ray	17
1.5.4. Các thư viện AI của Apache Ray	19
1.6. Hệ quản trị cơ sở dữ liệu.....	20
1.6.1. MySQL	20
1.6.2. MongoDB	21
1.7. Giao thức RTMP (Real – Time Messaging Protocol)	22
1.7.1. Định nghĩa và Lịch sử phát triển của RTMP.....	22
1.7.2. Kiến trúc và các thành phần chính	22
1.8. Hệ thống Apache Kafka	23
1.8.1. Giới thiệu về Apache Kafka	23
1.8.2. Kiến trúc	24
1.9. Framework Django	25
1.9.1. Kiến trúc của Django.....	25
1.9.2. Các tính năng chính	26

CHƯƠNG 2: PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG	27
2.1. Tổng quan về hệ thống	27
2.2. Các container trong hệ thống	28
2.2.1. Camera system.....	28
2.2.2. AI system.....	29
2.2.3. Message broker system	29
2.2.4. Application system	30
2.3. Chuẩn bị dữ liệu huấn luyện	31
2.3.1. Thu thập dữ liệu.....	31
2.3.2. Tiền xử lý dữ liệu	32
2.3.3. Gán nhãn dữ liệu	34
CHƯƠNG 3: TRIỂN KHAI VÀ KẾT QUẢ THỰC NGHIỆM	36
3.1. Huấn luyện và đánh giá mô hình	36
3.1.1. Huấn luyện mô hình	36
3.1.2. Đánh giá kết quả	36
3.1.3. Đánh giá Confusion Matrix của các biến thể	38
3.1.4. Kết quả phát hiện của mô hình.....	41
3.2. Giao diện hệ thống quản lý giao thông	47
3.3. Giao diện hệ thống AI	51
KẾT LUẬN VÀ KIẾN NGHỊ.....	55
1. Kết quả đạt được	55
2. Hạn chế	55
3. Hướng phát triển	56
PHỤ LỤC	57
Phụ lục 1: Kết quả đánh giá mô hình sau 3 lần huấn luyện	57
Phụ lục 2: Một số hình ảnh trực quan đánh giá của mô hình sau 3 lần huấn luyện	58

Phụ lục 3: Tập dữ liệu và mã nguồn dự án	61
Phụ lục 4: Hạ tầng triển khai trên Google Cloud Platform	61
TÀI LIỆU THAM KHẢO.....	63

DANH MỤC VIẾT TẮT

STT	Viết tắt	Ý nghĩa	Điễn giải
1	AI	Artificial Intelligence	Trí tuệ nhân tạo
2	AP	Average Precision	Độ chính xác trung bình
3	API	Application Programming Interface	Giao diện lập trình ứng dụng
4	BSON	Binary JSON	
5	C2f	CSP Bottleneck with 2 convolutions and fusion	CSP với 2 lớp tích chập và tích hợp
6	C2PSA	Convolutional block with Parallel Spatial Attention	
7	C3k2	Cross Stage Partial with kernel size 2	
8	CNN	Convolutional Neural Network	Mạng nơ-ron tích chập
9	CPU	Central Processing Unit	Bộ xử lý trung tâm
10	CRUD	Create, Read, Update, Delete	Tạo, đọc, cập nhật và xóa
11	CSDL	Cơ sở dữ liệu	
12	CSRF	Cross Site Request Forgery	Giả mạo yêu cầu liên trang
13	CUDA	Compute Unified Device Architecture	Kiến trúc thiết bị tính toán hợp nhất
14	CV	Computer Vision	Thị giác máy tính
15	GCP	Google Cloud Platform	Nền tảng dịch vụ đám mây của Google
16	GCS	Global Control Store / Global Control Service	Dịch vụ điều khiển toàn cầu
17	GPU	Graphics Processing Unit	Bộ xử lý đồ họa
18	HPC	High Performance Computing	Điện toán hiệu năng cao
19	HTTP	Hypertext Transfer Protocol	Giao thức truyền tải siêu văn bản
20	IETF	The Internet Engineering Task Force	Lực lượng đặc nhiệm kỹ thuật Internet

STT	Viết tắt	Ý nghĩa	Điễn giải
21	IoT	Internet of Things	Internet vạn vật
22	IoU	Intersection over Union	Độ chồng lấp
23	IP	Internet Protocol	Giao thức Internet
24	ITS	Intelligent Transportation System	Hệ thống giao thông thông min
25	JSON	Javascript Object Notation	
26	mAP	Mean Average Precision	
27	MB	Megabyte	
28	ML	Machine Learning	Học máy
29	MPI	Message Passing Interface	Giao thức truyền thông điệp
30	MVT	Model – View – Template	
31	NMS	Non-Max Suppression	Loại bỏ tối đa không cần thiết
32	NoSQL	None Structured Query Language	Ngôn ngữ truy vấn phi cấu trúc
33	OpenMP	Open Multi-Processing	
34	ORM	Object-Relational Mapping	Ánh xạ quan hệ đối tượng
35	RAM	Random Access Memory	Bộ nhớ truy cập ngẫu nhiên
36	R-CNN	Region Based Convolutional Neural Networks	Mạng nơ-ron tích chập dựa trên vùng
37	RDBMS	Relational Database Management System	Hệ thống quản trị cơ sở dữ liệu quan hệ
38	REST	Representational State Transfer	Chuyển giao trạng thái biểu diễn
39	RTMP	Real – Time Messaging Protocol	Giao thức nhắn tin thời gian thực
40	RTSP	Real-Time Streaming Protocol	Giao thức truyền tin thời gian thực

STT	Viết tắt	Ý nghĩa	Điễn giải
41	SPPF	Spatial Pyramid Pooling - Fast	
42	SQL	Structured Query Language	Ngôn ngữ truy vấn có cấu trúc
43	SSPL	Server Side Public License	Giấy phép công cộng phía máy chủ
44	TCP	Transmission Control Protocol	Giao thức điều khiển truyền vận
45	URL	Uniform Resource Locator	Hệ thống định vị tài nguyên thống nhất

DANH MỤC BẢNG BIỂU

Bảng 2.1. Số lượng ảnh trên mỗi tập dữ liệu.....	33
Bảng 2.2. Các nhãn được sử dụng để gán cho ảnh của tập dữ liệu	34
Bảng 3.1. Thông số của máy tính được sử dụng để huấn luyện.....	36
Bảng 3.2. Kết quả đánh giá trung bình các biến thể YOLOv11 sau 3 lần huấn luyện	37
Bảng 3.3 Kết quả phát hiện của các biến thể phiên bản YOLO 11.....	41
Bảng 0.1. Kết quả đánh giá các biến thể của YOLOv11 lần huấn luyện 1	57
Bảng 0.2. Kết quả đánh giá các biến thể của YOLOv11 lần huấn luyện 2.....	57
Bảng 0.3. Kết quả đánh giá các biến thể của YOLOv11 lần huấn luyện 3	58

DANH MỤC HÌNH ẢNH

Hình 1.1. Hình ảnh minh họa về thị giác máy tính	6
Hình 1.2. Hình ảnh minh họa về Object Detection	7
Hình 1.3. Hình ảnh trực quan về IoU	8
Hình 1.4. Kiến trúc YOLO 11	12
Hình 1.5. Minh họa đa luồng và đa tiến trình	14
Hình 1.6. Minh họa một cụm Ray	18
Hình 1.7. Sơ đồ minh họa luồng RTMP	23
Hình 1.8. Minh họa kiến trúc của Kafka	24
Hình 1.9. Minh họa luồng của MVT Django	25
Hình 2.1. Sơ đồ container minh họa cho hệ thống	27
Hình 2.2. Minh họa cụm hệ thống camera	28
Hình 2.3. Minh họa cụm hệ thống AI	29
Hình 2.4. Minh họa cụm hệ thống message broker	29
Hình 2.5. Minh họa cụm hệ thống Application	30
Hình 2.6. Mô tả quy trình xử lý dữ liệu	32
Hình 2.7. Ví dụ một số tình huống tai nạn giao thông từ tập dữ liệu	34
Hình 2.8. (a) Hiển thị tổng quan; (b) Nhãn “Accident”; (c) Nhãn “Motorcycle”; (d) Nhãn “Person”	35
Hình 3.1. Confusion Matrix của YOLOv11 Nano	39
Hình 3.2. Confusion Matrix của YOLOv11 Small	39
Hình 3.3. Confusion Matrix của YOLOv11 Medium	40
Hình 3.4. Confusion Matrix của YOLOv11 Large	40
Hình 3.5. Confusion Matrix của YOLOv11 X-Large	41
Hình 3.6. Giao diện danh sách lịch sử cảnh báo được hệ thống AI phát hiện	47
Hình 3.7. Giao diện xem chi tiết dữ liệu thông tin phát hiện tai nạn	47

Hình 3.8. Giao diện danh sách tai nạn được ghi nhận.....	48
Hình 3.9. Giao diện xuất danh sách tai nạn.....	48
Hình 3.10. Nội dung danh sách tai nạn sau khi xuất excel	49
Hình 3.11. Biểu đồ thống kê dựa trên dữ liệu xuất excel.....	50
Hình 3.12. Giao diện xóa thông tin ghi nhận tai nạn	50
Hình 3.13. Giao diện tạo thông tin ghi nhận tai nạn	51
Hình 3.14. Giao diện chỉnh sửa thông tin ghi nhận tai nạn	51
Hình 3.15. Giao diện hiển thị danh sách các luồng camera	51
Hình 3.16. Giao diện thêm mới luồng camera	52
Hình 3.17. Giao diện chỉnh sửa luồng camera	52
Hình 3.18. Giao diện gửi dữ liệu nhận diện tai nạn mẫu	53
Hình 3.19. Giao diện tổng quan Ray	53
Hình 3.20. Giao diện các job của Ray	54
Hình 3.21. Giao diện các cụm Ray.....	54
Hình 0.1. Kết quả mAP50 nhãn “Accident” các biến thể YOLOv11 3 lần huấn luyện	58
Hình 0.2. Kết quả mAP50 tất cả nhãn của các biến thể YOLOv11 3 lần huấn luyện	59
Hình 0.3. Kết quả Average IoU nhãn “Accident” các biến thể YOLOv11 3 lần huấn luyện.....	59
Hình 0.4. Kết quả Average IoU tất cả nhãn của các biến thể YOLOv11 3 lần huấn luyện	60
Hình 0.5. Kết quả FPS các biến thể YOLOv11 3 lần huấn luyện.....	60
Hình 0.6. Kết quả Latency các biến thể YOLOv11 3 lần huấn luyện.....	61

MỞ ĐẦU

1. Lý do chọn đề tài

Tai nạn giao thông là một trong những nguyên nhân hàng đầu gây tử vong tại Việt Nam, với số liệu thống kê cho thấy khoảng 11.628 ca tử vong vào năm 2023 theo Statista. Trong 8 tháng đầu năm 2024, số vụ tai nạn và số người bị thương có xu hướng tăng so với cùng kỳ năm trước, mặc dù số người chết đã giảm. Thậm chí, chỉ trong 6 tháng đầu năm 2024, đã có hơn 5.200 người thiệt mạng vì tai nạn giao thông. Dù đã có những nỗ lực trong việc cải thiện tình hình, số liệu thống kê vẫn cho thấy mức độ nghiêm trọng của vấn đề này. Đáng chú ý, trong hai tháng đầu năm 2025, đã có sự sụt giảm đáng kể về số vụ tai nạn, số người chết và bị thương so với cùng kỳ năm 2024, điều này có thể là kết quả của việc tăng cường các biện pháp kiểm soát và xử phạt. Tuy nhiên, nhìn chung, tai nạn giao thông vẫn là một vấn đề nhức nhối, đòi hỏi các giải pháp hiệu quả và bền vững.

Với những thách thức nghiêm trọng từ tai nạn giao thông tại Việt Nam, việc triển khai công nghệ trí tuệ nhân tạo (AI), đặc biệt là thị giác máy tính, để giải quyết vấn đề này là vô cùng cấp bách. Các phương pháp giám sát và phát hiện tai nạn truyền thống thường kém hiệu quả, tốn thời gian và không cung cấp dữ liệu tức thời. AI và thị giác máy tính sẽ tự động hóa quá trình này, cho phép các cơ quan chức năng phản ứng nhanh hơn và giảm thiểu hậu quả. Hơn nữa, phân tích dữ liệu giao thông bằng AI còn có thể giúp dự đoán các điểm đen tai nạn, từ đó đưa ra các biện pháp phòng ngừa hiệu quả.

Thị giác máy tính và AI có khả năng phân tích lượng lớn dữ liệu hình ảnh và video từ nhiều nguồn. Các mô hình phát hiện đối tượng tiên tiến có thể được huấn luyện để nhận diện chính xác các yếu tố liên quan đến tai nạn, từ phương tiện, con người, hiện trường đến các tình huống tiềm ẩn nguy hiểm. Một hệ thống phát hiện tai nạn giao thông dựa trên thị giác máy tính sẽ cung cấp thông tin nhanh chóng và chính xác cho các cơ quan chức năng, giúp rút ngắn thời gian phản ứng và triển khai cứu hộ kịp thời, giảm thiểu thiệt hại.

Việc lựa chọn mô hình YOLO cho đề tài nghiên cứu này là dựa trên những ưu điểm và đặc tính phù hợp của nó với bối cảnh giao thông phức tạp tại Việt Nam. YOLO (You Only Look Once) nổi bật với tốc độ xử lý vượt trội, nhanh hơn đáng kể so với các mô hình phát hiện đối tượng khác. Khả năng này cho phép YOLO phân tích các luồng video giám sát trong thời gian thực, một yếu tố then chốt để xây dựng một hệ thống có

thể cảnh báo và phản ứng kịp thời khi phát hiện ra một vụ tai nạn đang xảy ra hoặc vừa mới xảy ra. Phiên bản YOLO 11, được giới thiệu vào năm 2024, mang lại các cải tiến như trích xuất đặc trưng tốt hơn, hiệu suất tối ưu hóa, và độ chính xác cao hơn với ít tham số hơn so với các phiên bản trước. Điều này làm cho YOLO 11 trở thành lựa chọn lý tưởng cho hệ thống phát hiện tai nạn giao thông, đặc biệt trong bối cảnh giao thông phức tạp tại Việt Nam, với mật độ xe máy cao và điều kiện giao thông đặc thù.

Đề tài “NGHIÊN CỨU BÀI TOÁN OBJECT DETECTION VÀ PHÁT TRIỂN HỆ THỐNG GIÁM SÁT GIAO THÔNG TÍCH HỢP MÔ HÌNH YOLO ĐỂ PHÁT HIỆN TAI NẠN GIAO THÔNG TẠI VIỆT NAM” rất cấp thiết. Tai nạn giao thông ở Việt Nam vẫn là một thách thức lớn, gây thiệt hại nghiêm trọng. Việc ứng dụng YOLO để phát hiện tai nạn là một hướng đi mới, sáng tạo, góp phần quan trọng giải quyết vấn đề xã hội này. Nghiên cứu sẽ tạo cơ sở khoa học cho việc xây dựng hệ thống phát hiện tai nạn hiệu quả, nâng cao an toàn giao thông, giảm thiểu thiệt hại và mang lại lợi ích cho cộng đồng.

2. Mục tiêu, nội dung, phương pháp nghiên cứu đề tài

2.1. Mục tiêu nghiên cứu

2.1.1. Mục tiêu về mặt lý thuyết

Nghiên cứu mô hình YOLO 11 và tối ưu hóa cho bài toán phát hiện tai nạn giao thông.

Các lý thuyết của Computer Vision, Object Detection, hệ thống xử lý thời gian thực.

Nghiên cứu phát triển hệ thống giám sát giao thông thời gian thực.

Nghiên cứu tích hợp mô hình vào hệ thống giám sát giao thông.

2.1.2. Mục tiêu về mặt thực nghiệm

Triển khai mô hình YOLO 11 và huấn luyện trên tập dữ liệu tai nạn giao thông tại Việt Nam.

Phát triển hệ thống giám sát giao thông đường phố được tích hợp mô hình đã được huấn luyện để phát hiện tai nạn từ video thời gian thực.

Chỉ số mAP đạt trên 90% và FPS trung bình 60 FPS. Phát triển nguyên mẫu hệ thống phát hiện tai nạn có thể triển khai trên các hệ thống camera giám sát đường phố.

2.2. Nội dung nghiên cứu

Thu thập, xây dựng và tiền xử lý một bộ dữ liệu hình ảnh và video chuyên biệt về giao thông và các tình huống tai nạn giao thông tại Việt Nam. Bộ dữ liệu này sẽ được sử dụng để huấn luyện và đánh giá hiệu suất của các mô hình.

Triển khai huấn luyện và đánh giá hiệu suất của mô hình YOLO trên bộ dữ liệu đã được xây dựng.

Xây dựng hệ thống giám sát giao thông được tích hợp mô hình đã được huấn luyện.

2.3. Phương pháp nghiên cứu

Để hoàn thiện được các mục tiêu của đề tài, em đề ra phương pháp nghiên cứu gồm các nội dung sau:

Tìm hiểu về việc thu thập và xử lý dữ liệu

Tìm hiểu việc huấn luyện mô hình YOLO

Tìm hiểu các kỹ thuật xử lý video thời gian thực

Phân tích và xây dựng hệ thống giám sát phát hiện tai nạn giao thông

3. Đối tượng và phạm vi nghiên cứu

3.1. Đối tượng nghiên cứu

Bài toán phát hiện đối tượng

Quy trình huấn luyện mô hình Deep Learning

Các hệ thống giao thông thông minh đang được ứng dụng ở Việt Nam

Các hệ thống streaming dữ liệu

3.2. Phạm vi nghiên cứu

Tìm hiểu việc huấn luyện mô hình phát hiện tai nạn giao thông tại Việt Nam

Tìm hiểu xây dựng hệ thống camera giám sát tích hợp mô hình YOLO để phát hiện tai nạn giao thông

4. Cấu trúc cuốn báo cáo

Mở đầu

Chương 1: Cơ sở lý thuyết

Chương 2: Phân tích và thiết kế hệ thống

Chương 3: Triển khai và kết quả thực nghiệm

Kết luận và kiến nghị

CHƯƠNG 1: CƠ SỞ LÝ THUYẾT

1.1. Tổng quan về hệ thống giao thông thông minh (ITS)

1.1.1. Định nghĩa và mục tiêu

ITS – Intelligent Transportation System là ứng dụng các công nghệ cảm biến, máy tính, điện tử và truyền thông tiên tiến, cùng các chiến lược quản lý một cách tích hợp để cải thiện an toàn và hiệu quả của hệ thống giao thông. Các công nghệ ITS bao gồm công nghệ không dây, điện tử và tự động hóa hiện đại với mục tiêu nâng cao an toàn, hiệu quả và tiện lợi trong giao thông đường bộ. Ngoài ra, ITS còn được coi là một lĩnh vực khoa học và công nghệ được thiết kế để giảm thiểu thời gian di chuyển của người và hàng hóa, đồng thời thúc đẩy an toàn nhờ việc phân bổ hợp lý các nguồn lực sẵn có.

Các mục tiêu chính của việc triển khai ITS là đa diện và hướng tới việc giải quyết các vấn đề cấp bách trong giao thông hiện đại:

Nâng cao an toàn giao thông: Một trong những mục tiêu hàng đầu của ITS là giảm thiểu số vụ tai nạn và mức độ nghiêm trọng của chúng, từ đó giảm thiểu thương vong và thiệt hại về tài sản.

Cải thiện hiệu quả giao thông: ITS tối ưu hóa luồng giao thông, giảm ùn tắc, rút ngắn thời gian di chuyển và giảm tiêu thụ nhiên liệu. Các hệ thống này giúp quản lý tắc nghẽn và giảm thiểu tác động của chúng một cách hiệu quả.

Tăng cường tiện lợi: Cung cấp thông tin giao thông theo thời gian thực, hỗ trợ lập kế hoạch lộ trình tối ưu và tăng cường trải nghiệm di chuyển cho người tham gia giao thông.

Tối ưu hóa việc sử dụng cơ sở hạ tầng hiện có: ITS giúp khai thác tối đa năng lực của mạng lưới đường bộ mà không cần đầu tư lớn vào cơ sở hạ tầng mới, cải thiện đáng kể năng lực giao thông thực tế của mạng lưới đường bộ. [1]

1.1.2. Các thành phần chính của ITS

Hệ thống quản lý giao thông (Traffic Management System): Sử dụng cảm biến, camera, và thuật toán để điều khiển lưu lượng giao thông, ví dụ như điều chỉnh thời gian đèn giao thông dựa trên mật độ xe cộ.

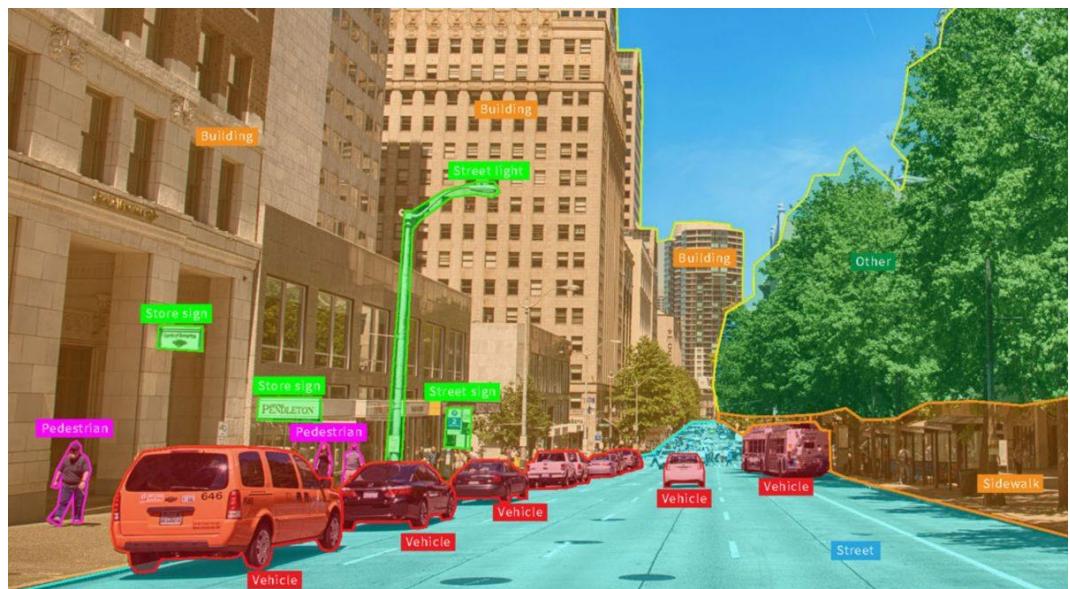
Hệ thống thông tin cho người tham gia giao thông (Traveler Information System): Cung cấp thông tin thời gian thực về tình trạng giao thông, thời tiết, hoặc các sự kiện đặc biệt.

Hệ thống thu phí tự động (Electronic Toll Collection): Cho phép thu phí mà không cần dừng xe, giúp giảm ùn tắc tại các trạm thu phí.

Hệ thống giám sát giao thông (Traffic Surveillance System): Sử dụng camera và cảm biến để ghi nhận và phân tích các vi phạm giao thông.

1.2. Thị giác máy tính và phát hiện đối tượng

1.2.1. Khái niệm cơ bản về thị giác máy tính



Hình 1.1. Hình ảnh minh họa về thị giác máy tính

Thị giác Máy tính (Computer Vision - CV) là một lĩnh vực nghiên cứu khoa học liên ngành tập trung vào việc cho phép máy tính “hiểu” và “diễn giải” thế giới trực quan từ hình ảnh hoặc video. Mục tiêu của CV là tái tạo, tự động hóa và phân tích các nhiệm vụ mà hệ thống thị giác con người có thể thực hiện. Điều này bao gồm khả năng nhận diện các đối tượng, phân tích chuyển động.

Các nhiệm vụ cơ bản trong Thị giác Máy tính rất đa dạng và thường được xây dựng dựa trên nhau:

Phân loại hình ảnh (Image Classification): Gán một nhãn duy nhất cho toàn bộ hình ảnh, ví dụ: “có xe” hoặc “không có xe”.

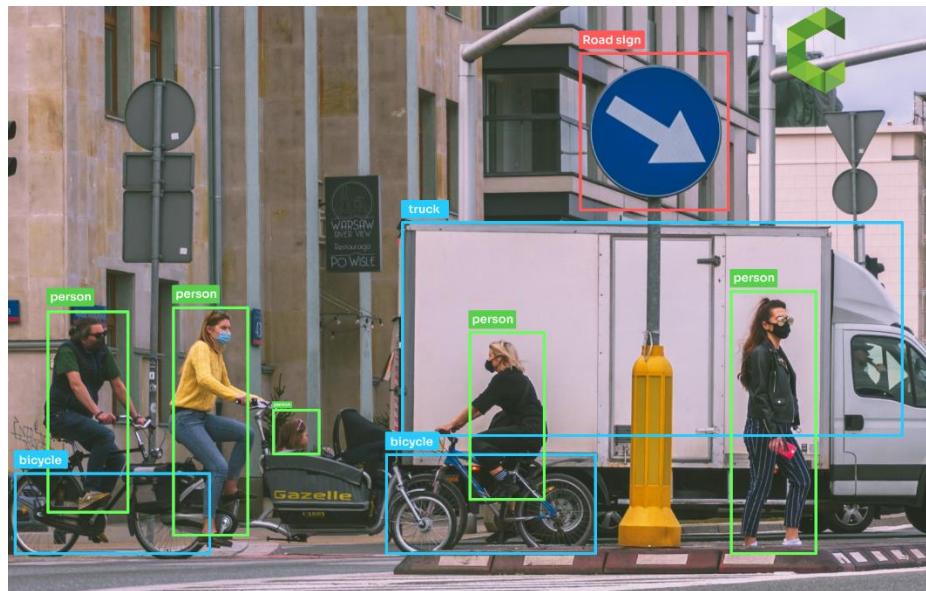
Phát hiện đối tượng (Object Detection): Không chỉ xác định đối tượng có trong ảnh mà còn định vị chính xác vị trí của chúng bằng cách vẽ một hộp giới hạn (bounding box) xung quanh mỗi đối tượng.

Phân đoạn ảnh (Image Segmentation): Chia hình ảnh thành các vùng hoặc đối tượng khác nhau ở cấp độ pixel, cung cấp thông tin chi tiết hơn về hình dạng và ranh giới của đối tượng.

Ước lượng tư thế (Pose Estimation): Phát hiện các điểm chính cụ thể trên cơ thể người hoặc đối tượng để theo dõi chuyển động hoặc tư thế.

Theo dõi đối tượng (Object Tracking): Theo dõi đường đi của một đối tượng cụ thể qua một chuỗi các khung hình video. [2]

1.2.2. Tổng quan về Object Detection



Hình 1.2. Hình ảnh minh họa về Object Detection

Object Detection (phát hiện đối tượng) là một nhánh quan trọng trong thị giác máy tính, dùng để xác định và định vị các đối tượng cụ thể trong ảnh hoặc video. Khác với phân loại ảnh chỉ xác định lớp của toàn ảnh, Object Detection còn khoanh vùng từng đối tượng bằng bounding box và gán nhãn tương ứng, ví dụ như xe cộ, người đi bộ, hay biển báo giao thông.

Về bản chất, Object Detection kết hợp hai nhiệm vụ: định vị đối tượng (xác định vị trí bằng khung giới hạn) và phân loại đối tượng (gán nhãn cho đối tượng). Mục tiêu là giúp máy tính “nhìn” và hiểu thế giới như con người thông qua việc phát hiện và phân loại các vật thể trong ảnh hoặc video. [2]

Nhiệm vụ chính của Object Detection bao gồm ba yếu tố cốt lõi hoạt động song song để đạt được sự hiểu biết chi tiết và sắc thái về các cảnh quan trực quan:

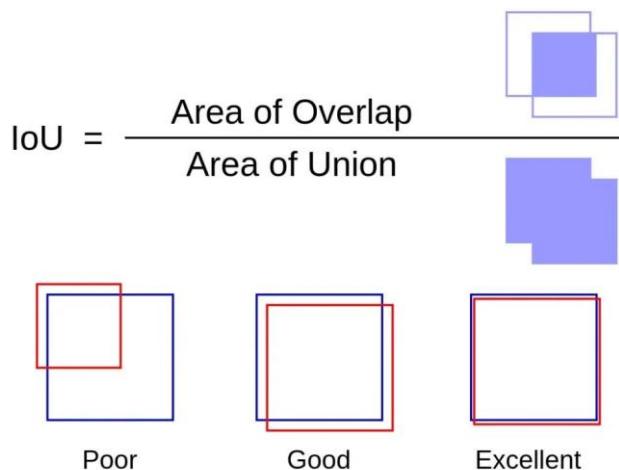
Phân loại (Classification): Mục tiêu chính là xác định và gán một danh mục hoặc nhãn chính xác cho mỗi đối tượng được phát hiện trong hình ảnh. Ví dụ, nhận diện một đối tượng là “xe ô tô”, “xe máy” hoặc “người đi bộ”.

Định vị (Localization): Xác định vị trí cụ thể của đối tượng trong hình ảnh. Điều này thường được thực hiện bằng cách dự đoán tọa độ của một hộp giới hạn (bounding box) hình chữ nhật bao quanh đối tượng.

Điểm tin cậy (Confidence Score): Mỗi phát hiện thường đi kèm với một điểm tin cậy, là một giá trị từ 0 đến 1, biểu thị mức độ chắc chắn của mô hình về sự hiện diện và phân loại chính xác của đối tượng trong hộp giới hạn được dự đoán. [3]

1.2.3. Các chỉ số đánh giá hiệu suất trong Object Detection

1.2.3.1. Intersection over union (IoU)



Hình 1.3. Hình ảnh trực quan về IoU

Đây là độ đo cơ bản đánh giá mức độ trùng khớp giữa hộp giới hạn dự đoán (B_p) và hộp giới hạn thực tế (B_{gt}) của một đối tượng. IoU được tính bằng tỷ lệ giữa diện tích phần giao (intersection) và diện tích phần hợp (union) của hai hộp giới hạn. Giá trị IoU nằm trong khoảng [0,1], với giá trị càng gần 1 thì độ trùng khớp càng cao, cho thấy mô hình định vị đối tượng càng chính xác. IoU thường được sử dụng để xác định một dự đoán là Đúng dương (True Positive – TP) hay Sai dương (False Positive – FP) dựa trên một ngưỡng (threshold) nhất định (ví dụ: $\text{IoU} > 0.5$).

$$\text{IoU} = \frac{\text{Area}(B_p \cap B_{gt})}{\text{Area}(B_p \cup B_{gt})} \quad (1.1)$$

Trong đó:

- Area ($B_p \cap B_{gt}$) là diện tích vùng giao nhau giữa bounding box dự đoán và bounding box thực tế.

- Area ($B_p \cup B_{gt}$) là diện tích vùng hợp bởi hai bounding box.

1.2.3.2. Precision

Precision đo lường tỷ lệ các dự đoán đúng trong tổng số dự đoán được mô hình đưa ra cho lớp đó. Precision cao cho thấy mô hình ít đưa ra các phát hiện sai. Nó được tính bằng tỷ lệ các phát hiện dương tính đúng (True Positive (TP)) trên tổng số các phát hiện dương tính mà mô hình đưa ra (True Positive + False Positive (FP)).

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} = P \quad (1.2)$$

1.2.3.3. Recall

Recall nhấn mạnh khả năng của mô hình trong việc tìm thấy tất cả các đối tượng thực tế có trong ảnh. Recall cao cho thấy mô hình ít bỏ sót các đối tượng. Nó được tính bằng tỷ lệ các phát hiện dương tính đúng (TP) trên tổng số các đối tượng thực tế dương tính (True Positive + False Negative – FN).

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} = R \quad (1.3)$$

1.2.3.4. F1 Score

F1 Score là trung bình điều hòa của Precision và Recall. F1 Score cung cấp một sự cân bằng giữa hai chỉ số này, hữu ích khi cần xem xét cả việc giảm thiểu lỗi phát hiện sai và lỗi bỏ sót. F1 Score có giá trị nằm trong nửa khoảng (0,1]. F1 càng cao, bộ phân lớp càng tốt.

$$F1 = 2 \frac{P \times R}{P + R} \quad (1.4)$$

1.2.3.5. Average Precision (AP)

AP là chỉ số cơ bản cho phát hiện đối tượng, tích hợp Precision, Recall và điểm tin cậy của mô hình trong mỗi phát hiện. Nó được tính riêng cho từng lớp đối tượng và tóm tắt đường cong Precision-Recall thành một giá trị số duy nhất.

$$AP = \int_0^1 P(R)dR \quad (1.5)$$

Trong thực tế, AP thường được tính bằng cách lấy tổng có trọng số của Precision tại các điểm Recall khác nhau. Với P_k và R_k lần lượt là Precision và Recall tại điểm k (chỉ số của từng điểm trên đường cong Precision-Recall, được tạo ra từ các dự đoán của mô hình được sắp xếp theo xác suất (confidence scores)) $N_{classes}$ là tổng số lớp đối tượng, ta có:

$$AP = \sum_{k=1}^N (R_k - R_{k-1})P_k \quad (1.6)$$

1.2.3.6. Mean Average Precision (mAP)

mAP được xây dựng dựa trên khái niệm AP, đặc biệt trong các kịch bản đa lớp. Nó được tính bằng cách lấy trung bình AP trên tất cả các lớp đối tượng. mAP xem xét Precision và Recall cho các ngưỡng IoU và các lớp đối tượng khác nhau, với mAP cao hơn cho thấy hiệu suất tổng thể của mô hình vượt trội. [3]

$$mAP = \frac{1}{N_{classes}} \sum_{i=1}^{N_{classes}} AP_i \quad (1.7)$$

1.3. Tổng quan về mô hình YOLO

1.3.1. Lịch sử và sự phát triển của YOLO

YOLO đã cách mạng hóa lĩnh vực phát hiện đối tượng bằng cách giới thiệu một kiến trúc mạng nơ-ron hợp nhất, đồng thời xử lý cả hồi quy hộp giới hạn và phân loại đối tượng chỉ trong một lần truyền qua mạng. Cách tiếp cận tích hợp này giúp YOLO nhanh hơn và hiệu quả hơn so với các bộ phát hiện đối tượng hai giai đoạn, vốn yêu cầu nhiều bước xử lý tuần tự.

YOLO được giới thiệu lần đầu tiên bởi Joseph Redmon và các cộng sự vào năm 2015. Kể từ đó, mô hình này đã trải qua một quá trình phát triển liên tục với nhiều phiên bản cải tiến, mỗi phiên bản đều nỗ lực nâng cao tốc độ, độ chính xác và hiệu quả tính toán. Các phiên bản chính của YOLO bao gồm YOLOv1, YOLOv2, YOLOv3, YOLOv4, YOLOv5, YOLOv6, YOLOv7, YOLOv8, YOLOv9, YOLOv10, YOLOv11 và YOLOv12. Sự tiến hóa này cho thấy nỗ lực không ngừng trong việc tối ưu hóa mô hình để đáp ứng các yêu cầu ngày càng cao của các ứng dụng thời gian thực. [4]

1.3.2. Cách hoạt động của YOLO

Quá trình hoạt động của YOLO, đặc biệt là các phiên bản sau này, có thể được tóm tắt qua các bước chính sau:

Chia lưới (Grid Division): Ảnh đầu vào được chia thành một lưới các ô (. Mỗi ô trong lưới này chịu trách nhiệm dự đoán sự hiện diện của một đối tượng nếu tâm của đối tượng đó rơi vào ô. Ngoài ra, mỗi ô cũng dự đoán các tọa độ hộp giới hạn và xác suất lớp cho đối tượng đó.

Trích xuất đặc trưng (Feature Extraction): Ảnh được đưa qua một mạng nơ-ron tích chập (CNN) để trích xuất các đặc trưng đa cấp từ ảnh. CNN này đóng vai trò như một bộ xương sống (backbone) để học các biểu diễn hình ảnh có ý nghĩa.

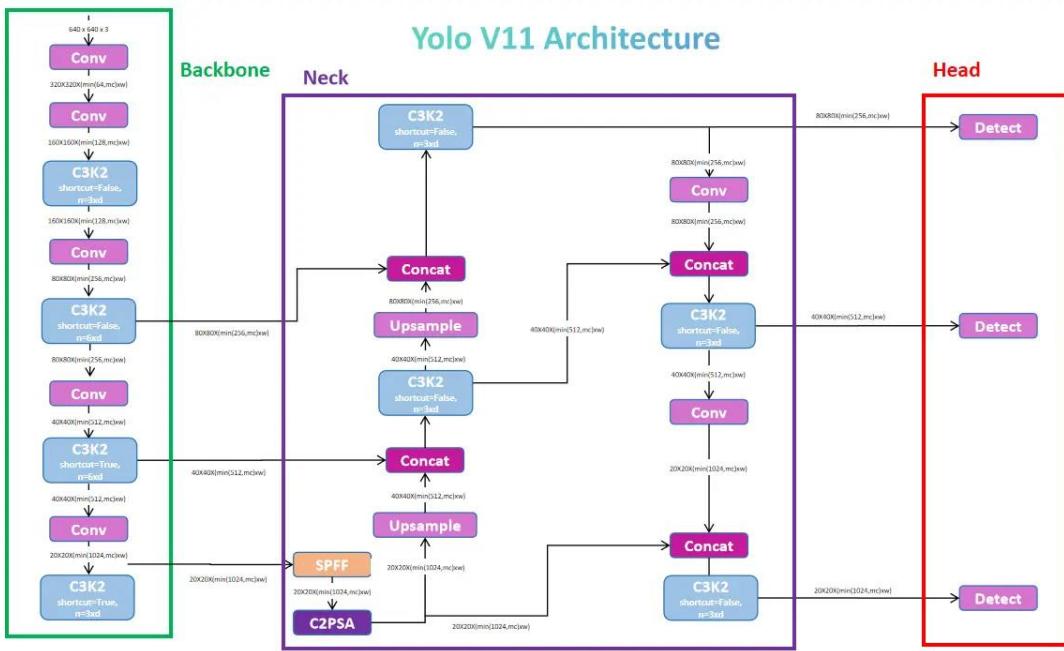
Dự đoán (Prediction): Các đặc trưng được trích xuất sau đó được truyền qua một loạt các lớp kết nối đầy đủ (fully connected layers) hoặc các lớp tích chập khác để dự đoán xác suất lớp và tọa độ hộp giới hạn cho mỗi ô lưới.

Hộp neo (Anchor Boxes): Một cải tiến quan trọng từ YOLOv2 là việc sử dụng các hộp neo (anchor boxes). Đây là các hộp được xác định trước với tỷ lệ khung hình và kích thước khác nhau, giúp mô hình dự đoán tốt hơn các hộp giới hạn phù hợp với các đối tượng có hình dạng và kích thước đa dạng trong ảnh.

Hậu xử lý (Post – processing): Đầu ra thô của mạng là một tập hợp lớn các hộp giới hạn và xác suất lớp cho mỗi ô. Để loại bỏ các dự đoán trùng lặp và chọn ra các phát hiện tốt nhất, một thuật toán hậu xử lý được gọi là triệt tiêu phi cực đại (Non-Max Suppression - NMS) được áp dụng. NMS lọc bỏ các hộp chồng lấn và chỉ giữ lại hộp có điểm tin cậy cao nhất cho mỗi đối tượng.

Kết quả cuối cùng: Sau quá trình hậu xử lý, đầu ra cuối cùng là một tập hợp các hộp giới hạn được dự đoán và nhãn lớp cho mỗi đối tượng được phát hiện trong ảnh. [4]

1.3.3. Kiến trúc của YOLO 11



Hình 1.4. Kiến trúc YOLO 11

YOLOv11 đại diện cho một bước tiến quan trọng trong công nghệ phát hiện đối tượng, được xây dựng và nâng cấp dựa trên nền tảng của YOLOv8 và YOLOv10, vốn được giới thiệu vào đầu năm 2024. Phiên bản này tập trung vào việc cải tiến kiến trúc và tối ưu hóa tham số để đạt được hiệu suất phát hiện vượt trội.

Các đổi mới kiến trúc chính được triển khai trong YOLOv11 góp phần cải thiện hiệu suất theo nhiều cách, bao gồm tăng cường trích xuất đặc trưng:

Khối C3k2 (Cross Stage Partial with kernel size 2): Đây là một khối kiến trúc mới được giới thiệu trong YOLOv11. C3k2 là một phiên bản nhỏ gọn hơn của CSP bottleneck, giúp kiến trúc hiệu quả hơn về số lượng tham số huấn luyện được. Nó cũng cho phép trích xuất đặc trưng nhanh hơn do sử dụng hai phép tích chập nhỏ hơn (kernel size 2) thay vì một phép tích chập lớn, từ đó giảm chi phí tính toán.

SPPF (Spatial Pyramid Pooling - Fast): Thành phần này được tích hợp để tăng cường khả năng trích xuất đặc trưng từ các tỷ lệ khác nhau trong ảnh. SPPF cải thiện hiệu quả tính toán so với các phương pháp SPP truyền thống, giúp mô hình xử lý tốt hơn các đối tượng có kích thước đa dạng.

C2PSA (Convolutional block with Parallel Spatial Attention): Đây là một tiến bộ quan trọng trong YOLOv11, tích hợp các cơ chế chú ý không gian tinh vi. C2PSA cho phép mô hình tập trung hiệu quả hơn vào các vùng quan trọng trong hình ảnh, nâng cao khả năng phát hiện và phân tích đối tượng. Khả năng chú ý được cải thiện này đặc

biết có lợi cho việc xác định các đối tượng phức tạp hoặc bị che khuất một phần, giải quyết một thách thức phổ biến trong các tác vụ phát hiện đối tượng và góp phần vào cải thiện hiệu suất tổng thể của YOLOv11, đặc biệt trong các môi trường thị giác phức tạp. [5]

1.3.4. Hiệu suất và khả năng mở rộng

YOLOv11 thể hiện sự cải thiện hiệu suất đáng kể về độ chính xác trung bình (mAP) và hiệu quả tính toán so với các phiên bản trước.

Cải thiện mAP và hiệu quả tham số: Phiên bản YOLOv11m đạt điểm mAP vượt trội trên tập dữ liệu COCO trong khi sử dụng ít hơn 22% tham số so với YOLOv8m. Điều này chứng tỏ hiệu quả tính toán được cải thiện mà không ảnh hưởng đến độ chính xác, một sự cân bằng rất quan trọng cho các ứng dụng thực tế.

Tốc độ suy luận (Inference Speed): Phiên bản nano của YOLOv11, mặc dù có sự tăng nhẹ về tham số so với phiên bản tiền nhiệm, vẫn thể hiện tốc độ suy luận và số khung hình mỗi giây (FPS) được cải thiện. Sự cải thiện này cho thấy YOLOv11 đạt được sự cân bằng thuận lợi giữa hiệu quả tính toán và độ chính xác phát hiện, làm cho nó lý tưởng cho các ứng dụng thời gian thực.

Khả năng mở rộng: YOLOv11 được thiết kế với một loạt các kích thước mô hình, từ nano đến extra-large, phục vụ các nhu cầu ứng dụng đa dạng. Khả năng mở rộng này cho phép triển khai linh hoạt trong nhiều kịch bản, từ các thiết bị có tài nguyên hạn chế đến môi trường tính toán hiệu năng cao. Phiên bản nano, với tốc độ và hiệu quả được cải thiện, đặc biệt phù hợp cho các ứng dụng thời gian thực trên các thiết bị, nơi tài nguyên tính toán bị giới hạn. Điều này làm cho YOLOv11 trở thành một lựa chọn chiến lược phù hợp cho các hệ thống yêu cầu xử lý theo thời gian thực, nơi yêu cầu về tốc độ và khả năng triển khai trên nhiều loại phần cứng là rất quan trọng. [5]

1.4. Lập trình song song

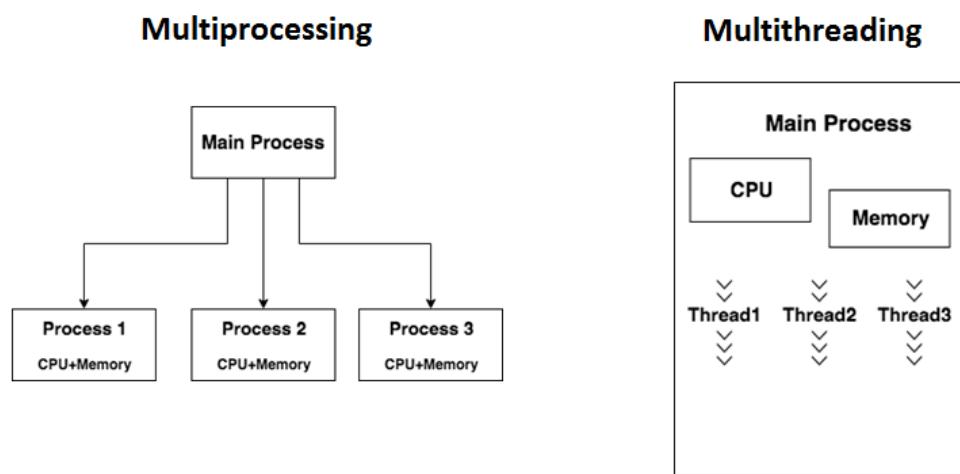
1.4.1. Khái niệm và tầm quan trọng

Lập trình song song là mô hình cho phép nhiều tác vụ hoặc các phần của một tác vụ được thực hiện đồng thời, thay vì tuần tự. Mục tiêu chính là tiết kiệm thời gian bằng cách chia nhỏ nhiệm vụ lớn để xử lý song song. Trong điện toán hiệu năng cao (HPC – High Performance Computing), lập trình song song là nền tảng cốt lõi, giúp tăng tốc độ xử lý chương trình và tối ưu hóa tài nguyên hệ thống, từ đó cải thiện hiệu suất tổng thể.

Sự phát triển của lập trình song song là phản ứng trực tiếp đối với giới hạn vật lý của CPU, cụ thể là “Power wall” (giới hạn về công suất và nhiệt độ) và “Memory wall” (độ trễ khi truy cập bộ nhớ) vào giữa những năm 2004 – 2005. Những giới hạn này đã khiến ngành công nghiệp chuyển hướng sang song song hóa để tiếp tục tăng hiệu năng tính toán. Ngoài ra, lập trình song song còn được ứng dụng rộng rãi trong trò chơi máy tính (render đồ họa 4K/8K) và hệ thống mạng (xử lý yêu cầu mạng, quản lý băng thông). [6]

1.4.2. Các mô hình lập trình song song phổ biến

1.4.2.1. Đa luồng và đa tiến trình



Hình 1.5. Minh họa đa luồng và đa tiến trình

Đa luồng (Multi-threading): Một tiến trình duy nhất chứa nhiều luồng thực thi, chia sẻ cùng không gian bộ nhớ (shared memory). OpenMP là API phổ biến cho đa luồng trên hệ thống bộ nhớ chia sẻ, tận dụng các nhân CPU.

Đa tiến trình (Multi-processing): Chạy nhiều tiến trình độc lập, mỗi tiến trình có không gian bộ nhớ riêng. Giao tiếp giữa chúng thường qua truyền thông điệp. MPI (Message Passing Interface) là API cho phép các tiến trình giao tiếp bằng cách gửi/nhận thông điệp, phù hợp cho hệ thống phân tán (cluster).

1.4.2.2. Tính toán trên GPU

Nguyên lý hoạt động: GPU là phần cứng chuyên biệt với hàng nghìn lõi nhỏ, được thiết kế để xử lý nhiều tác vụ đồng thời (SIMD - Single Instruction, Multiple Data). GPU cực kỳ hiệu quả cho các thuật toán xử lý khói dữ liệu lớn song song.

Vai trò của CUDA: CUDA (Compute Unified Device Architecture) là nền tảng điện toán song song và API độc quyền của NVIDIA. CUDA cho phép phần mềm sử dụng GPU để tăng tốc xử lý đa năng (GPGPU), giúp các thư viện học sâu như TensorFlow và PyTorch tận dụng GPU.

Mô hình lập trình CUDA: CUDA tổ chức các luồng thành cấu trúc phân cấp gồm các khối (block) và lưới (grid). Một kernel (chương trình chạy trên GPU) được thực thi bởi một mảng các luồng CUDA, được nhóm thành các khối và nhiều khối tạo thành một lưới.

Ưu điểm của CUDA: Đơn giản hóa lập trình song song trên GPU, hỗ trợ đọc phân tán, bộ nhớ ảo và hợp nhất, bộ nhớ chia sẻ nhanh giữa các luồng, và cải thiện tốc độ truyền dữ liệu từ/đến GPU.

Nhược điểm của CUDA: Mã nguồn CUDA hiện theo cú pháp C++, có thể gây vấn đề tương thích ngược. Việc sao chép dữ liệu giữa CPU và GPU có thể làm giảm hiệu suất do băng thông và độ trễ.

1.4.3. Các công cụ và API (OpenMP, MPI)

1.4.3.1. MPI

Nguyên tắc: MPI là API cho phép các tiến trình giao tiếp bằng cách gửi/nhận thông điệp, phù hợp cho các hệ thống phân tán (cluster).

Ưu điểm: Giao tiếp hiệu quả giữa các node, kiểm soát tác vụ tốt qua lịch tinh, và các hàm giao tiếp được tối ưu hóa.

Nhược điểm: Không tận dụng bộ nhớ chia sẻ, có thể lãng phí tài nguyên và gây nút chai. Khả năng cân bằng tải và mở rộng kém.

1.4.3.2. OpenMP

Nguyên tắc: OpenMP (Open Multi-Processing) là API cho lập trình song song đa luồng trên hệ thống bộ nhớ chia sẻ, tạo nhiều luồng chạy song song trên các nhân CPU.

Ưu điểm: Tận dụng hiệu quả bộ nhớ chia sẻ, đơn giản hóa lập trình do tự động phân phát nhiệm vụ.

Nhược điểm: Không có chức năng giao tiếp giữa các node, giới hạn khả năng mở rộng trên cụm máy tính phân tán. Đòi hỏi chú trọng đồng bộ hóa.

1.4.3.3. Kết hợp (Hybrid MPI & OpenMP)

Kết hợp MPI với OpenMP là phương pháp phổ biến để tận dụng ưu điểm của cả hai. MPI quản lý giao tiếp giữa các node, OpenMP quản lý song song hóa trên các nhân CPU trong cùng một node.

1.5. Công nghệ Apache Ray

1.5.1. Giới thiệu về Apache Ray

Apache Ray là một framework mã nguồn mở, thống nhất được thiết kế để mở rộng quy mô các ứng dụng AI và Python. Mục đích cốt lõi của Ray là cung cấp một lớp tính toán mạnh mẽ cho xử lý song song và phân tán, giúp người dùng không cần phải hiểu quá sâu về hệ thống phân tán. Ray đơn giản hóa sự phức tạp của việc chạy các quy trình làm việc học máy phân tán từ đầu đến cuối, từ huấn luyện đến suy luận và điều chỉnh siêu tham số.

1.5.2. Lợi ích và tính năng nổi bật của Apache Ray

Khả năng mở rộng: Ray cho phép dễ dàng song song hóa mã từ một CPU đơn lẻ lên môi trường đa lõi, đa GPU, hoặc đa node với thay đổi mã tối thiểu. Nó cung cấp khả năng tự động điều chỉnh quy mô (autoscaling) với các điều khiển tích hợp để điều chỉnh tốc độ mở rộng lên/xuống, cùng với các điều khiển mở rộng theo chương trình.

Khả năng chịu lỗi: Ray cung cấp khả năng chịu lỗi mạnh mẽ thông qua việc kiểm tra điểm dừng (Checkpointing) tích hợp, thử lại tác vụ và khởi động lại actor. Ray sử dụng tái tạo dòng dõi (Lineage Reconstruction) để tính toán lại dữ liệu khi một kho đối tượng trong bộ nhớ bị mất. Điều này giúp các ứng dụng ML có thể hoạt động ổn định trong môi trường phân tán, giảm thiểu rủi ro mất dữ liệu và gián đoạn dịch vụ.

API Pythonic đơn giản: Ray cung cấp một API đơn giản, phổ quát cho việc xây dựng các ứng dụng phân tán. Bất kỳ hàm hoặc lớp Python tùy ý nào cũng có thể được chuyển đổi thành tác vụ hoặc dịch vụ từ xa chỉ bằng cách thêm một decorator (@ray.remote()). Điều này giảm đáng kể rào cản kỹ thuật cho các nhà phát triển Python và nhà khoa học dữ liệu. Việc Ray cung cấp “API Pythonic” là một yếu tố then chốt giúp nó thu hút và trao quyền cho các nhà khoa học dữ liệu và kỹ sư ML, cho phép họ tập trung vào logic thuật toán và mô hình thay vì phải đổi mới với các chi tiết phức tạp của hệ thống phân tán, từ đó tăng tốc đáng kể chu trình phát triển và triển khai các giải pháp AI.

Tích hợp thư viện học máy phong phú: Ray có các tích hợp mạnh mẽ với hầu hết các thư viện ML phổ biến (như PyTorch, TensorFlow, Hugging Face) và các thư viện gốc của Ray cũng kết hợp chúng. Nó cung cấp một bộ công cụ thống nhất và có khả năng mở rộng cho các ứng dụng ML.

Tối ưu hóa AI: Ray được thiết kế đặc biệt để tối ưu hóa và hỗ trợ các tác vụ AI, đặc biệt là các tác vụ sử dụng nhiều GPU như AI tạo sinh. Nó cung cấp hỗ trợ gốc cho GPU và hỗ trợ hạng nhất cho tính toán không đồng bộ (Heterogeneous Computing) để tận dụng sức mạnh của cả CPU và GPU. [7]

1.5.3. Kiến trúc Apache Ray

1.5.3.1. Các Khái niệm Ứng dụng Cốt lõi

Task (Tác vụ): Là một lời gọi hàm từ xa (@ray.remote function). Một tác vụ được thực thi không đồng bộ với bên gọi, trả về một hoặc nhiều **ObjectRef** (futures) để truy xuất giá trị trả về.

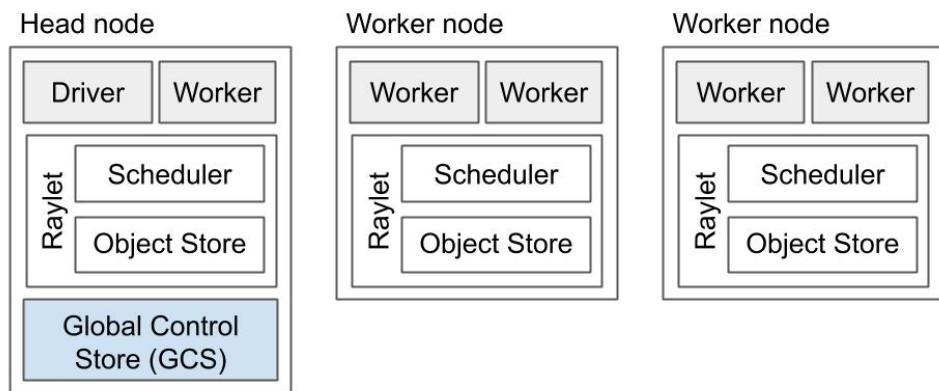
Object (Đối tượng): Là một giá trị ứng dụng không thể thay đổi, được trả về bởi một tác vụ hoặc được tạo thông qua **ray.put**. Một worker có thể tham chiếu đến một đối tượng bằng cách sử dụng ObjectRef.

Actor (Tác nhân): Là một tiến trình worker có trạng thái, được tạo ra từ một lớp @ray.remote. Các tác vụ của actor phải được gửi với một “handle” và có thể sửa đổi trạng thái nội bộ của actor trong quá trình thực thi.

Driver (Trình điều khiển): Là gốc của chương trình, hoặc chương trình “chính”, nơi mã **ray.init()** được chạy.

Job (Công việc): Là tập hợp các tác vụ, đối tượng và actor bắt nguồn (một cách đệ quy) từ cùng một trình điều khiển, cùng với môi trường runtime của chúng. Có một ánh xạ 1:1 giữa trình điều khiển và công việc.

1.5.3.2. Các Thành phần của Cụm Ray



Hình 1.6. Minh họa một cụm Ray

Một cụm Ray bao gồm một hoặc nhiều nút worker (worker node), trong đó một nút worker được chỉ định là nút đầu (head node).

a) Nút worker

Mỗi nút worker bao gồm các tiến trình vật lý sau:

Một hoặc nhiều tiến trình Worker (Worker processes): Chịu trách nhiệm gửi và thực thi tác vụ. Mỗi tiến trình worker được liên kết với một công việc cụ thể và có thể không trạng thái hoặc là một actor. Mỗi worker lưu trữ một bảng sở hữu (Ownership Table) (siêu dữ liệu hệ thống cho các đối tượng mà worker có tham chiếu) và một bộ nhớ trong tiến trình (In-process Store) để lưu trữ các đối tượng nhỏ.

Một Raylet: Quản lý các tài nguyên được chia sẻ trên mỗi nút và được chia sẻ giữa tất cả các công việc đang chạy đồng thời. Raylet có hai thành phần chính chạy trên các luồng riêng biệt:

- **Scheduler (Bộ lập lịch):** Chịu trách nhiệm quản lý tài nguyên, sắp xếp tác vụ và đáp ứng các đối số tác vụ được lưu trữ trong kho đối tượng phân tán. Các bộ lập lịch riêng lẻ trong một cụm tạo thành bộ lập lịch phân tán của Ray.
- **Shared-memory object store (Kho đối tượng bộ nhớ chia sẻ - hay còn gọi là Plasma Object Store):** Chịu trách nhiệm lưu trữ, truyền tải và tràn (Spilling) các đối tượng lớn. Các kho đối tượng riêng lẻ trong một cụm tạo thành kho đối tượng phân tán của Ray.

b) Nút head

Một trong các nút worker được chỉ định là nút đầu. Ngoài các tiến trình worker và raylet, nút đầu còn chứa:

Global Control Service (GCS - Dịch vụ điều khiển toàn cầu): Là mặt phẳng điều khiển cụm của Ray, quản lý siêu dữ liệu cấp cụm như vị trí của các actor, được lưu trữ dưới dạng cặp key – value. GCS quản lý một số hoạt động cấp cụm, bao gồm lập lịch cho các nhóm vị trí (Placement Group) và actor, cũng như xác định tư cách thành viên của nút cụm. Trong Ray 2.0, GCS có khả năng chịu lỗi, cho phép GCS chạy trên bất kỳ và nhiều nút, thay vì chỉ một nút đầu được chỉ định.

Tiến trình Driver (Driver Process): Một tiến trình worker đặc biệt thực thi ứng dụng cấp cao nhất (ví dụ: `__main__` trong Python). Nó có thể gửi tác vụ nhưng không thể tự thực thi bất kỳ tác vụ nào. Mặc dù trình điều khiển có thể chạy trên bất kỳ nút nào, nhưng thường chạy trên nút đầu theo mặc định.

Các dịch vụ cấp cụm khác: Xử lý việc gửi công việc (Job Submission), tự động điều chỉnh quy mô (Autoscaling)... [8]

1.5.4. Các thư viện AI của Apache Ray

Ray Data: Thư viện này cung cấp khả năng tải và biến đổi dữ liệu có khả năng mở rộng, không phụ thuộc vào framework, trên các giai đoạn huấn luyện, điều chỉnh và dự đoán. Nó cho phép xử lý các tập dữ liệu lớn một cách hiệu quả.

Ray Train: Thư viện này hỗ trợ huấn luyện mô hình phân tán đa node và đa lõi với khả năng chịu lỗi được tích hợp, đồng thời tích hợp các thư viện huấn luyện phổ biến như PyTorch và TensorFlow. Ray Train đơn giản hóa việc triển khai huấn luyện học sâu phân tán.

Ray Tune: Thư viện này cung cấp khả năng điều chỉnh các siêu tham số có khả năng mở rộng để tối ưu hóa hiệu suất mô hình. Điều này rất quan trọng để tìm ra các siêu tham số tối ưu và cải thiện độ chính xác cho mô hình.

Ray Serve: Thư viện này cho phép phục vụ mô hình có khả năng mở rộng và lập trình được để triển khai các mô hình cho suy luận trực tuyến, với tính năng microbatching tùy chọn để nâng cao hiệu suất. Ray Serve là thành phần thiết yếu để triển khai mô hình YOLO đã huấn luyện vào môi trường production.

Ray RLlib: Thư viện này dành riêng cho các tác vụ học tăng cường phân tán có khả năng mở rộng. Mặc dù không trực tiếp liên quan đến phát hiện đối tượng, RLlib cho thấy sự đa dạng và khả năng của Ray trong việc hỗ trợ các loại hình tác vụ AI phức tạp khác.

1.6. Hệ quản trị cơ sở dữ liệu

1.6.1. MySQL

MySQL là một mã nguồn mở, hệ thống quản lý cơ sở dữ liệu quan hệ - Relational Database Management System (RDBMS) có hiệu quả trong việc lưu trữ, sắp xếp và truy xuất thông tin. Được phát triển bởi Oracle và dựa trên SQL (Structured Query Language – Ngôn ngữ truy vấn có cấu trúc), được sử dụng phổ biến trong các ứng dụng Web, công mua sắm và ứng dụng dữ liệu lớn.

Những tính năng chính của MySQL:

Mã nguồn mở: MySQL là phần mềm nguồn mở cho phép người dùng tải xuống MySQL và bắt đầu sử dụng nó mà không mất bất kỳ khoản phí nào. Nó có Giấy phép công cộng (GPL) GNU. Nó mang lại sự tự do cho các dự án nguồn mở và làm cho MySQL trở nên phù hợp đối với các doanh nghiệp quan tâm đến ngân sách phát triển.

Hỗ trợ nhiều nền tảng: MySQL hỗ trợ tất cả các nền tảng hệ điều hành chính bao gồm Oracle Linux, Solaris, Ubuntu, SUSE, Debian, Windows và macOS. Nó cũng hỗ trợ các ngôn ngữ và trình điều khiển phổ biến như Perl, Ruby, Go, Rust, C, C++, C# và ODBC.

Cơ sở dữ liệu quan hệ: Dữ liệu trong MySQL được sắp xếp theo mô hình quan hệ và được lưu trữ trong các bảng, mỗi hàng được liên kết với một bảng ghi liên quan. Các phương pháp tiếp cận có cấu trúc đảm bảo rằng dữ liệu này là thống nhất, do đó giúp dễ dàng truy xuất và xử lý thông tin.

Ngôn ngữ truy vấn có cấu trúc: Ngôn ngữ chính được MySQL sử dụng để giao tiếp với CSDL là SQL. Người dùng có thể tạo, đọc, cập nhật hoặc xóa dữ liệu bằng các lệnh SQL để thực hiện công việc phát triển và quản CSDL.

Khả năng mở rộng: MySQL phù hợp cho các ứng dụng cơ sở dữ liệu quy mô nhỏ. Chúng ta có thể mở rộng quy mô theo chiều dọc (thêm nhiều tài nguyên hơn như CPU, Bộ nhớ, Ổ đĩa và I/O) và định cấu hình các cụm để chia tỷ lệ theo chiều ngang cho các yêu cầu ứng dụng của bạn. Các công ty như Facebook và X (Twitter) sử dụng

hệ thống cơ sở dữ liệu MySQL để xử lý khối lượng công việc của người dùng ở quy mô lớn.

Tính linh hoạt: MySQL cho phép các nhà phát triển sử dụng SQL truyền thống hoặc NoSQL. Nó cũng hỗ trợ dữ liệu quan hệ và tài liệu JSON trong cùng một CSDL cũng như trong cùng một ứng dụng.

Hiệu suất: Hiệu suất CSDL là điều cần thiết khi chọn một hệ thống quản lý cơ sở dữ liệu. MySQL bao gồm tối ưu hóa lưu trữ dữ liệu, tăng khả năng mở rộng, truy xuất dữ liệu nâng cao... Nó cung cấp nhiều cơ chế khác nhau cho các chuyên gia CSDL để đảm bảo hiệu suất được tối ưu hóa nhằm tối đa hóa hiệu suất hệ thống.

Cộng đồng và hỗ trợ: MySQL có cộng đồng người dùng và nhà phát triển cơ sở dữ liệu hoạt động tích cực lớn. Chúng ta có thể sử dụng diễn đàn MySQL để thảo luận về các chủ đề và vấn đề nhằm giúp chúng ta khắc phục sự cố. [9]

1.6.2. MongoDB

MongoDB là một CSDL hướng tài liệu mã nguồn mở được thiết kế để lưu trữ dữ liệu quy mô lớn và cho phép chúng ta làm việc với dữ liệu đó hiệu quả. Nó được phân loại theo CSDL NoSQL vì việc lưu trữ và truy xuất dữ liệu trong MongoDB không ở dạng bảng. CSDL MongoDB được phát triển và quản lý bởi MongoDB, Inc. theo SSPL (Giấy phép công cộng phía máy chủ - Server Side Public License) và ban đầu được phát hành vào tháng 2 năm 2009.

Nó cũng cung cấp hỗ trợ trình điều khiển chính thức cho tất cả các ngôn ngữ phổ biến như C, C++, C#, GO, Java, Node.js, Python... Vì vậy, chúng ta có thể tạo một ứng dụng bằng cách sử dụng bất kỳ ngôn ngữ nào trong số này. Ngày nay, có rất nhiều công ty sử dụng MongoDB như Facebook, Nokia, eBay, Adobe, Google... để lưu trữ dữ liệu.

Ưu điểm

Tính linh hoạt: MongoDB là một hệ thống cơ sở dữ liệu phi quan hệ, nó cung cấp khả năng lưu trữ dữ liệu bất cứ khi nào, bất cứ nơi đâu, không cần phải tuân thủ một mô hình quan hệ cụ thể.

Khả năng mở rộng: MongoDB có khả năng mở rộng dễ dàng, nhờ tính năng sharding cho phép phân chia dữ liệu thành nhiều phần và lưu trữ trên nhiều máy chủ.

Tốc độ truy xuất nhanh: MongoDB có thể đáp ứng các yêu cầu truy vấn dữ liệu trong thời gian ngắn hơn so với các hệ thống cơ sở dữ liệu quan hệ truyền thống.

Tính khả dụng cao: MongoDB cung cấp tính năng sao lưu và phục hồi dữ liệu, giúp người dùng bảo vệ dữ liệu của mình khỏi những rủi ro.

Dễ sử dụng: MongoDB cung cấp các công cụ quản lý dữ liệu trực quan và dễ sử dụng, giúp người dùng tối ưu hóa hiệu suất và quản lý cơ sở dữ liệu một cách dễ dàng.

Nhược điểm

Cần sử dụng bộ nhớ cao để lưu trữ dữ liệu (data storage).

Không được phép lưu trữ hơn 16MB data trong tài liệu.

Data nesting trong BSON cũng bị hạn chế, bạn không được phép nest data quá 100 cấp độ.

1.7. Giao thức RTMP (Real – Time Messaging Protocol)

1.7.1. Định nghĩa và Lịch sử phát triển của RTMP

RTMP là một giao thức truyền thông dựa trên TCP, được thiết kế để truyền tải dữ liệu, âm thanh và video hai chiều. Nó được tạo ra để truyền tải âm thanh, video và dữ liệu trực tiếp một cách hiệu quả qua internet.

RTMP bắt đầu vào khoảng năm 2000 với một dự án của Macromedia có tên “Tin Can”, được đặt tên theo sự tương đồng với điện thoại lon thiếc thời thơ ấu. Giao thức này chính thức ra mắt vào tháng 7 năm 2002 cùng với việc Macromedia phát hành Flash Communication Server 1.0. Mục đích ban đầu của RTMP là để truyền tải nội dung từ các máy chủ RTMP đến trình phát video Flash. Mặc dù sự phổ biến của Flash đã suy giảm và HLS (HTTP Live Streaming) dần thay thế RTMP cho việc phân phối luồng trực tiếp đến người dùng cuối, RTMP vẫn giữ một vai trò quan trọng trong quy trình truyền phát trực tiếp thông qua “RTMP ingest” (tiếp nhận RTMP). Trong vai trò này, RTMP được sử dụng để vận chuyển các tệp video từ bộ mã hóa đến nền tảng video trực tuyến. [10]

1.7.2. Kiến trúc và các thành phần chính

RTMP hoạt động bằng cách thiết lập một đường truyền thông giữa client truyền phát RTMP và máy chủ RTMP, cho phép truyền dữ liệu nhanh chóng. Nó thiết lập một kết nối TCP liên tục qua cổng 1935, duy trì truyền dữ liệu liên tục với chi phí tối thiểu. RTMP chia nội dung video thành các đoạn nhỏ hơn (âm thanh thường là 64 byte, video là 128 byte, mặc dù kích thước có thể thay đổi) để tạo điều kiện truyền dữ liệu hiệu quả trong một kết nối duy nhất, giúp cải thiện chất lượng nội dung.

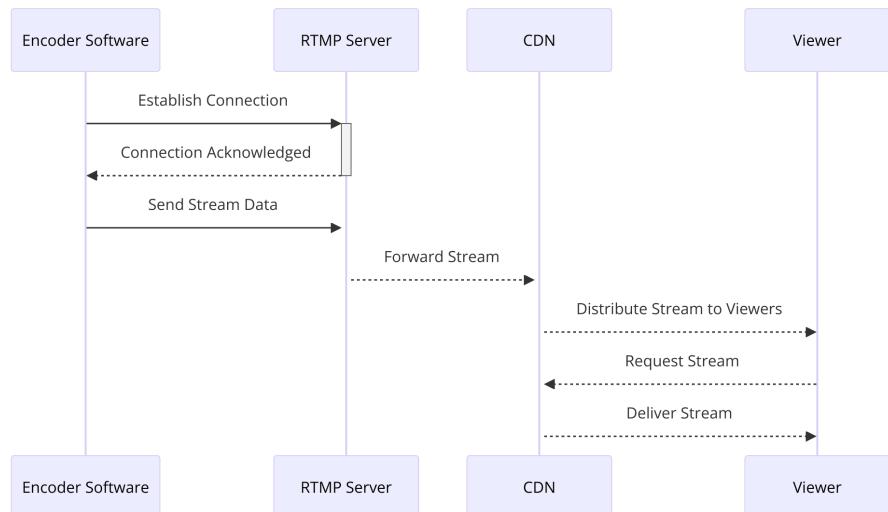
Quá trình thiết lập kết nối RTMP bao gồm ba giai đoạn chính:

Bắt tay (The Handshake): Sau khi kết nối TCP được thiết lập, client và server trao đổi ba gói tin để chỉ ra phiên bản RTMP được yêu cầu và xác nhận việc nhận, từ đó thiết lập kết nối.

Kết nối (The Connection): Trong giai đoạn này, client và server sử dụng mã hóa AMF (Action Message Format) để gửi các tin nhắn liên quan đến “Set Peer Bandwidth” và “Window Acknowledgement Size”, cho phép máy chủ truyền dữ liệu video sau khi các kết nối này được thiết lập.

Luồng (The Stream): Cuối cùng, các lệnh RTMP cụ thể được gửi để truyền dữ liệu video giữa client và server bằng giao thức RTMP.

Một máy chủ RTMP sẵn sàng sản xuất thường bao gồm các thành phần như: Xử lý kết nối, tiếp nhận luồng, ghép kênh luồng & xử lý gói tin, chuyển mã & sao chép (tùy chọn), phân phối luồng & chuyển đổi giao thức, ghi log & giám sát. [11]



Hình 1.7. Sơ đồ minh họa luồng RTMP

1.8. Hệ thống Apache Kafka

1.8.1. Giới thiệu về Apache Kafka

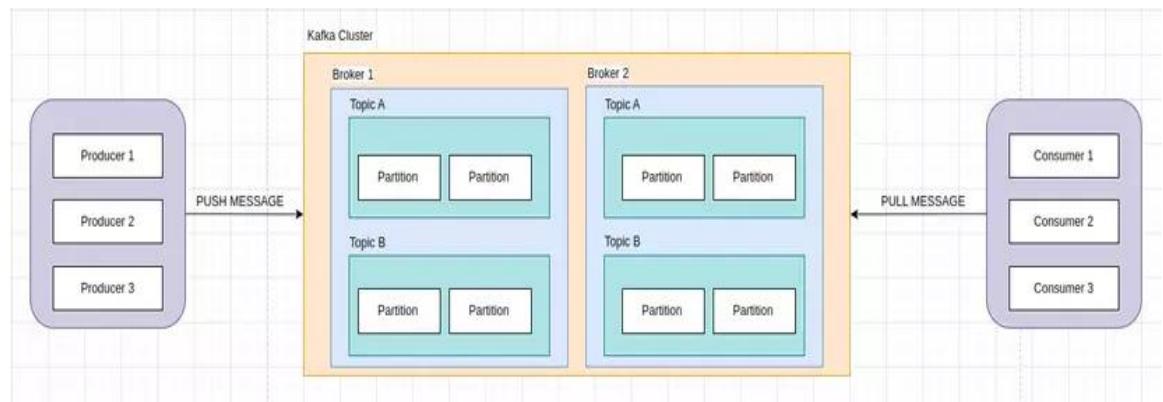
Apache Kafka là một kho dữ liệu phân tán được tối ưu hóa để thu nạp và xử lý dữ liệu truyền phát theo thời gian thực. Dữ liệu truyền phát là dữ liệu được tạo ra liên tục từ hàng nghìn nguồn dữ liệu khác nhau, các nguồn này thường gửi các bản ghi dữ liệu đồng thời. Nền tảng truyền phát cần phải xử lý luồng dữ liệu liên tục này và xử lý dữ liệu theo trình tự và tăng dần.

Kafka cung cấp ba chức năng chính cho người dùng:

- Xuất bản và đăng ký các luồng bản ghi.
- Lưu trữ hiệu quả các luồng bản ghi theo thứ tự tạo bản ghi.
- Xử lý các luồng bản ghi trong thời gian thực.

Kafka chủ yếu được dùng để xây dựng các quy trình dữ liệu truyền phát trong thời gian thực và các ứng dụng thích ứng với luồng dữ liệu đó. Kafka kết hợp nhắn tin, lưu trữ và xử lý luồng nhằm hỗ trợ hoạt động lưu trữ, phân tích cả dữ liệu lịch sử lẫn dữ liệu trong thời gian thực.

1.8.2. Kiến trúc



Hình 1.8. Minh họa kiến trúc của Kafka

Cluster: Một tập hợp các máy chủ Broker bao gồm ít nhất 1 Broker nhưng thường là nhiều Broker hoạt động cùng nhau. Cluster Kafka có vai trò quan trọng trong việc cung cấp tính mở rộng, tính nhất quán và độ tin cậy cho việc xử lý dữ liệu thời gian thực.

Broker: Là thành phần cốt lõi của Apache Kafka, đại diện cho máy chủ xử lý và lưu trữ dữ liệu Kafka. Một cụm Kafka thường bao gồm nhiều Broker và mỗi Broker có thể xử lý Producer (sản xuất) và Consumer (tiêu thụ) dữ liệu. Broker chịu trách nhiệm quản lý và lưu trữ các Partition của các chủ đề Topic.

Topic: Dữ liệu trong Apache Kafka được phân loại thành các Topic (chủ đề). Mỗi Topic là một luồng dữ liệu độc lập và có thể được coi là một danh sách các tin nhắn liên quan. Producer gửi dữ liệu tới các Topic và Consumer đọc dữ liệu từ các Topic. Topic cho phép các ứng dụng chỉ quan tâm đến các loại dữ liệu cụ thể mà nó muốn tiêu thụ.

Partition: Mỗi Topic có thể được chia thành nhiều phân vùng. Partition là một phần nhỏ của chủ đề và đóng vai trò quan trọng trong việc phân tán dữ liệu và tăng hiệu

số lượng. Mỗi Partition được lưu trữ trên một Broker và dữ liệu được đọc và ghi vào từng Partition một.

Producer: Producer là thành phần của Apache Kafka cho phép ứng dụng gửi dữ liệu tới các Topic. Producer sử dụng giao thức Kafka để kết nối với Broker và đưa dữ liệu vào các Partition của các chủ đề.

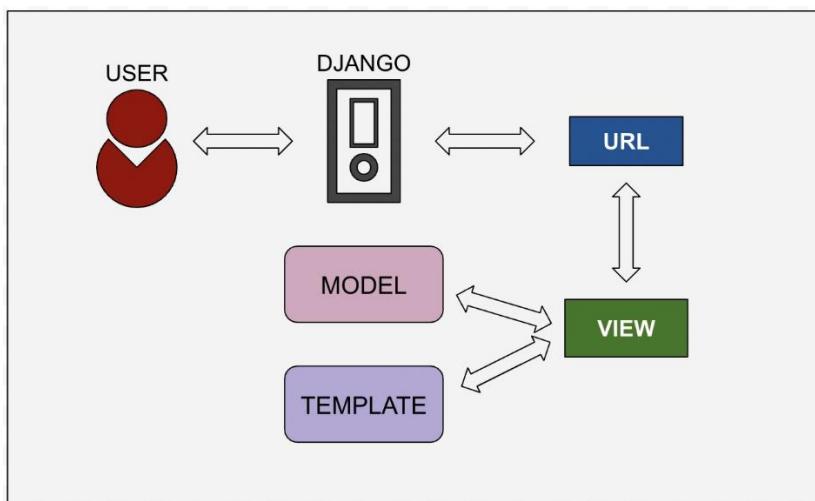
Consumer: Là thành phần cho phép ứng dụng lấy dữ liệu từ các Topic Kafka. Consumer cũng sử dụng giao thức Kafka để kết nối với Broker và đọc dữ liệu từ các Partition. Có thể có nhiều Consumer đọc từ cùng 1 Topic và Kafka đảm bảo rằng mỗi tin nhắn chỉ được đọc bởi 1 Consumer.

ZooKeeper: Là một hệ thống quản lý tập trung được sử dụng để quản lý và duy trì trạng thái của các Broker trong một cụm Kafka. Nó chịu trách nhiệm trong việc theo dõi và quản lý các Broker giúp Kafka hoạt động ổn định và đảm bảo tính nhất quán. [12]

1.9. Framework Django

Django là một web framework Python bậc cao, giúp phát triển nhanh chóng và thiết kế thực tế, sạch sẽ. Django giải quyết các rắc rối trong khi phát triển web, vì vậy lập trình viên chỉ cần tập trung vào việc viết ứng dụng của mình mà không cần phải phát minh lại “bánh xe”.

1.9.1. Kiến trúc của Django



Hình 1.9. Minh họa luồng của MVT Django

Model

Đại diện cho dữ liệu, có thể xem các class là 1 bảng trong CSDL.

Thao tác với CSDL.

View

Xử lý logic của ứng dụng và thao tác với model.

Đóng vai trò như là bộ não của ứng dụng.

Template

Xác định giao diện người dùng hiển thị như thế nào.

Là mẫu cho các trang web.

1.9.2. Các tính năng chính

Giao diện admin: Django cung cấp giao diện admin được tích hợp sẵn để quản lý dữ liệu ứng dụng. Tính năng này sẽ được tự động tạo ra dựa trên model và có thể tùy chỉnh.

ORM: ORM của Django cho phép các nhà phát triển sử dụng Python để tương tác với CSDL thay vì SQL. Hỗ trợ nhiều hệ quản trị CSDL như: MySQL, PostgreSQL, MariaDB, Oracle, SQLite.

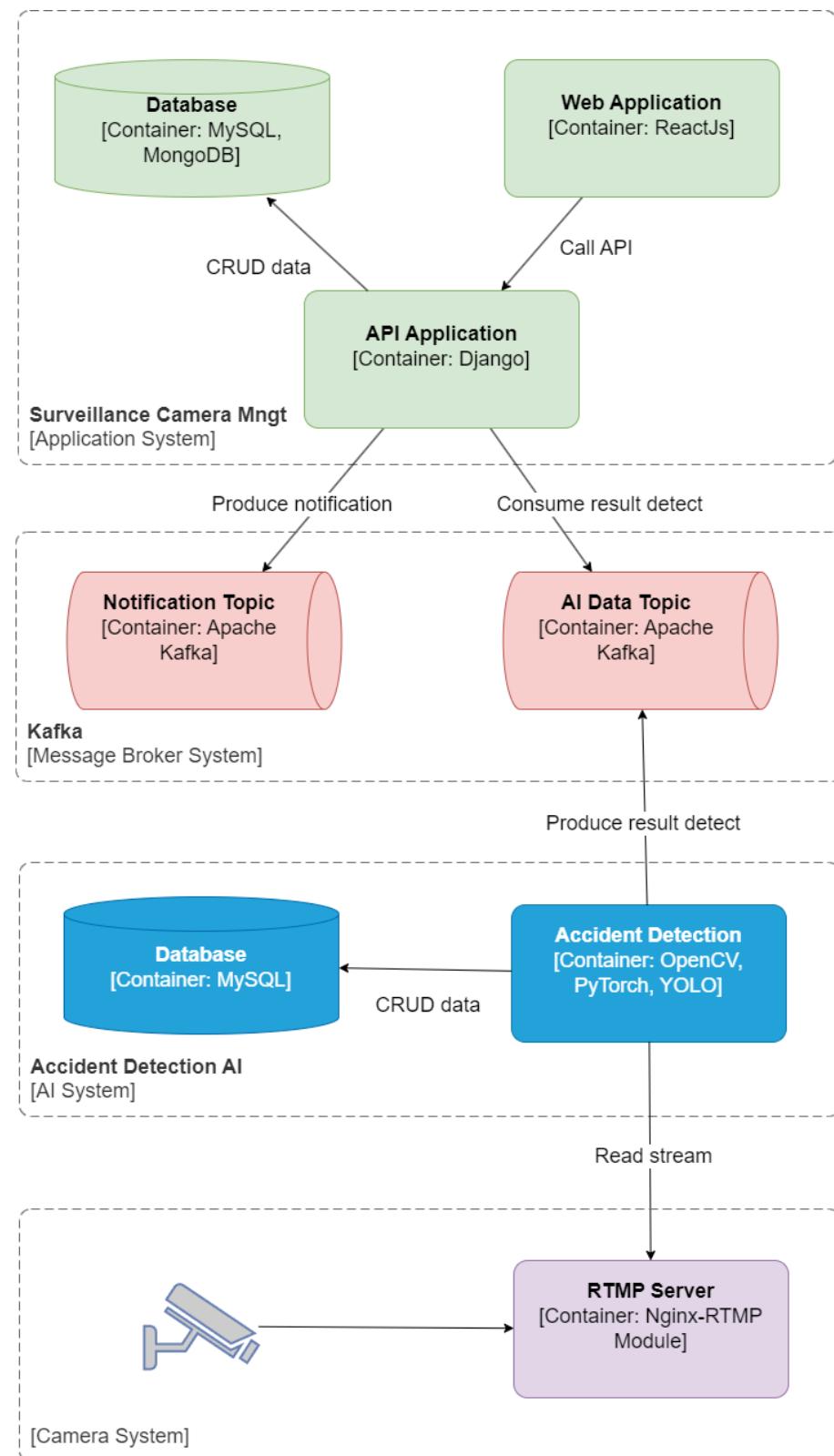
Định tuyến URL: Bộ điều phối URL của Django ánh xạ các mẫu URL đến views, giúp dễ dàng thiết kế các URL sạch và có thể đọc được.

Xử lý form: Django cung cấp tính năng xử lý form mạnh mẽ, bao gồm các tính năng xác thực dữ liệu và bảo mật để bảo vệ chống lại các cuộc tấn công CSRF.

Hệ thống xác thực: Django bao gồm một hệ thống xác thực toàn diện với đăng nhập người dùng, đăng xuất, quản lý mật khẩu và quyền. [13]

CHƯƠNG 2: PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG

2.1. Tổng quan về hệ thống



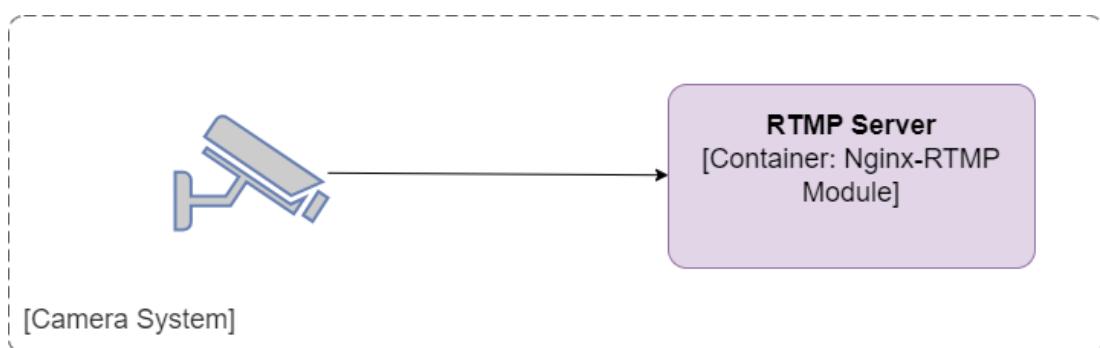
Hình 2.1. Sơ đồ container minh họa cho hệ thống

Luồng xử lý tổng quát:

1. **RTMP Server** phát luồng video từ camera IP thực tế hoặc mô phỏng.
2. **AI (Accident Detection)** đọc luồng camera từ RTMP server, chạy mô hình để phát hiện tai nạn, nếu phát hiện (thỏa mãn ngưỡng cài đặt) gửi kết quả vào topic AI Data.
3. **API Application** consume kết quả phát hiện tai nạn giao thông từ topic, xử lý và cập nhật vào cơ sở dữ liệu. Nếu cần, API Application cũng gửi yêu cầu vào topic Notification để các bên khác có thể nhận thông báo về tai nạn đã được xác nhận bởi cơ quan giám sát.
4. Người dùng ứng dụng web (**Web Application**) gọi API để lấy dữ liệu từ backend và hiển thị trên giao diện.

2.2. Các container trong hệ thống

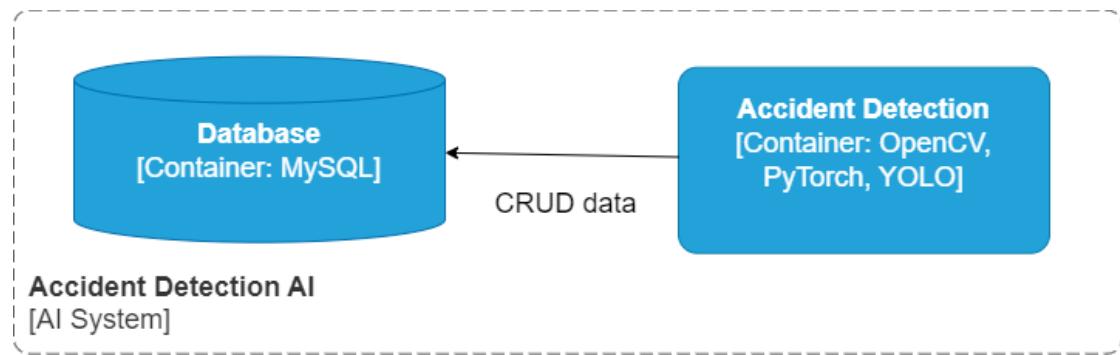
2.2.1. Camera system



Hình 2.2. Minh họa cụm hệ thống camera

RTMP Server (Container: Nginx-RTMP Module) là thành phần tiếp nhận và phân phối các luồng video từ camera giám sát. Nó sử dụng Nginx kết hợp với mô-đun RTMP để lắng nghe và truyền tải các luồng video theo chuẩn giao thức RTMP. Luồng video có thể đến từ camera IP thực tế hoặc từ các tệp video giả lập thông qua công cụ như FFmpeg. RTMP Server đóng vai trò là đầu vào cho hệ thống AI, giúp cung cấp dữ liệu hình ảnh để xử lý phát hiện tai nạn theo thời gian thực.

2.2.2. AI system

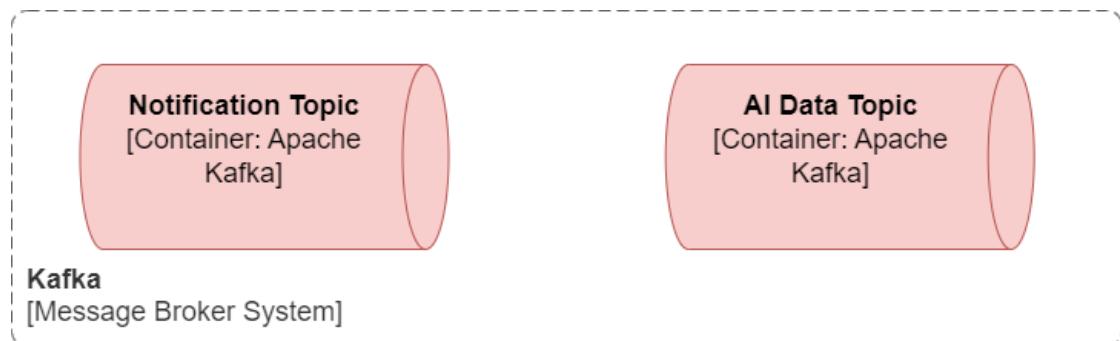


Hình 2.3. Minh họa cụm hệ thống AI

Accident Detection (Container: OpenCV, PyTorch, YOLO) là thành phần xử lý chính trong hệ thống AI, có nhiệm vụ đọc luồng video từ RTMP Server để thực hiện phân tích và phát hiện các sự kiện tai nạn giao thông. Bên trong container này, mô hình được xây dựng dựa trên YOLO kết hợp với OpenCV để xử lý hình ảnh và trích xuất thông tin đối tượng trong khung hình. Sau khi nhận diện được tai nạn, container này sẽ gửi kết quả (gồm thời gian, camera, loại sự kiện, vị trí...) vào Kafka thông qua topic AI Data.

Database (Container: MySQL, MongoDB) trong hệ thống AI đóng vai trò lưu trữ các cấu hình cho hệ thống (nếu có). Đây không phải là cơ sở dữ liệu chính của toàn hệ thống, nó hoạt động độc lập với cơ sở dữ liệu chính trong hệ thống quản lý.

2.2.3. Message broker system

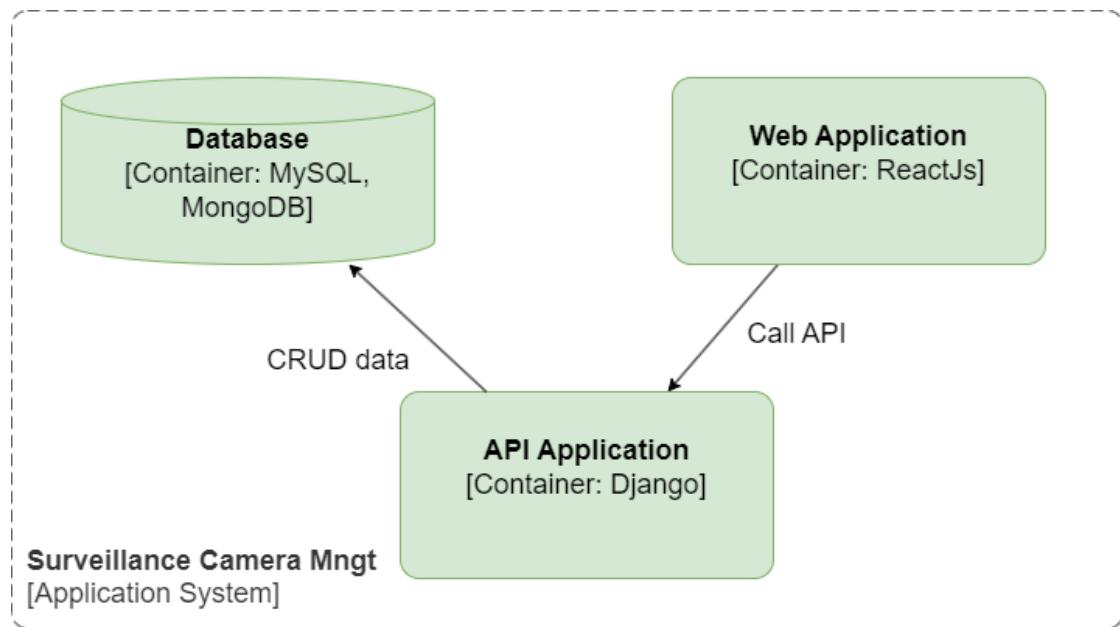


Hình 2.4. Minh họa cụm hệ thống message broker

Notification Topic (Container: Apache Kafka) là một topic tùy chọn trong hệ thống, được thiết kế để gửi các thông báo về sự kiện quan trọng như phát hiện tai nạn, trạng thái bất thường từ camera, hoặc các cảnh báo kỹ thuật đến các bên liên quan như hệ thống giám sát trung tâm, ứng dụng quản lý thành phố.

AI Data Topic (Container: Apache Kafka) là nơi mà container AI gửi kết quả phát hiện tai nạn giao thông. Các thông tin bao gồm camera nào, thời gian xảy ra, loại tai nạn, hình ảnh minh chứng,... sẽ được produce vào topic này. Container Django phía Application system sẽ consume topic này để xử lý và lưu kết quả vào cơ sở dữ liệu chính và hiển thị lên giao diện cho người dùng.

2.2.4. Application system



Hình 2.5. Minh họa cụm hệ thống Application

API Application (Container: Django) là trung tâm điều phối và quản lý dữ liệu trong hệ thống. Nó cung cấp các API để frontend hoặc các hệ thống khác gọi tới nhằm thêm/sửa/xóa dữ liệu. Đồng thời, Django còn tương tác với Kafka để gửi các thông báo vào topic Notification và lắng nghe các kết quả từ hệ thống AI qua topic AI Data để lưu vào cơ sở dữ liệu chính. Nhờ đó, nó giữ vai trò kết nối giữa hệ thống AI và giao diện người dùng.

Web Application (Container: ReactJS) là giao diện người dùng của toàn hệ thống. Nó cho phép người dùng theo dõi trạng thái của các camera, xem lịch sử sự kiện tai nạn đã phát hiện, và thực hiện các tác vụ quản lý như cấu hình, thêm mới hoặc xóa camera. Giao diện này được viết bằng ReactJS và kết nối với backend Django thông qua các API REST để lấy dữ liệu và hiển thị theo thời gian thực.

Database (Container: MySQL, MongoDB) là hệ thống cơ sở dữ liệu trung tâm của toàn bộ hệ thống, trong đó:

- **MySQL** được sử dụng để lưu trữ các thông tin có cấu trúc như: tài khoản người dùng, thông tin ghi nhận tai nạn, và các thông tin khác.
- **MongoDB** được bổ sung nhằm phục vụ lưu trữ các lịch sử phát hiện tai nạn giao thông từ hệ thống AI (dữ liệu phi cấu trúc) như thời gian phát hiện, bounding box, độ tin cậy, nhãn...

2.3. Chuẩn bị dữ liệu huấn luyện

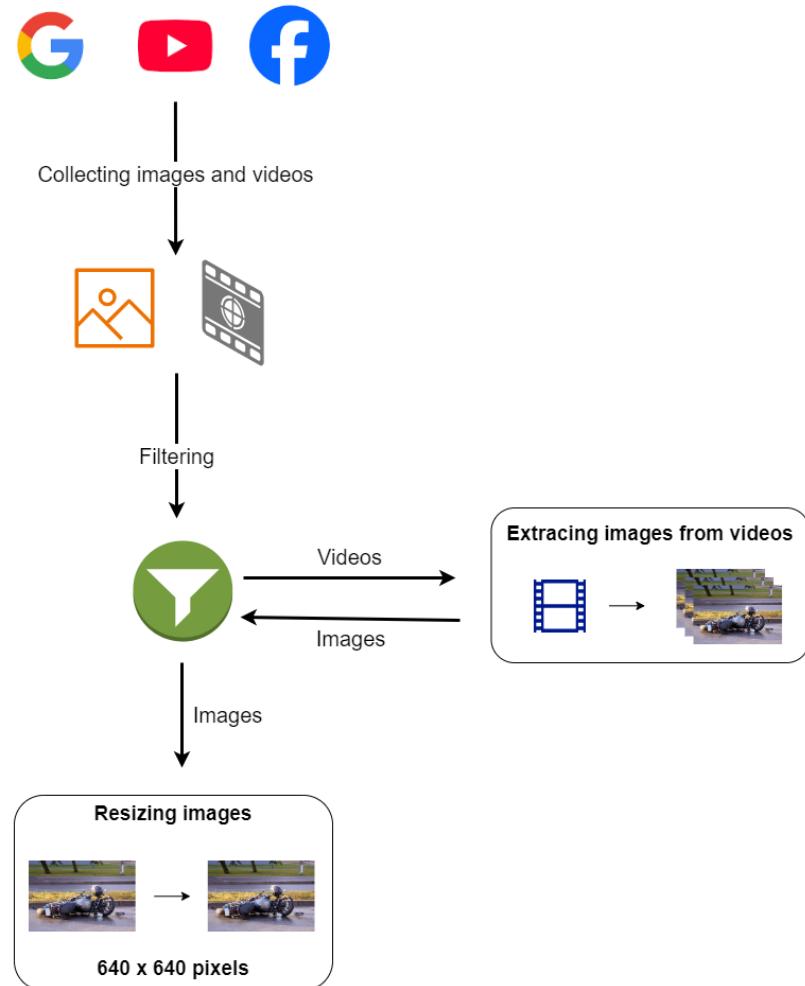
2.3.1. Thu thập dữ liệu

Quá trình thu thập dữ liệu cho đề tài là một công đoạn quan trọng và tốn nhiều thời gian. Để đảm bảo tính thực tiễn và phản ánh đúng đặc thù giao thông tại Việt Nam, tôi đã tập trung thu thập thủ công dữ liệu từ các nguồn công khai trên Internet. Các nền tảng như Google, YouTube và các hội nhóm Facebook là những nguồn em sử dụng để tìm kiếm dữ liệu về tai nạn giao thông.

Do đặc thù nhạy cảm của dữ liệu – bao gồm hình ảnh và video về các vụ tai nạn giao thông, việc tìm kiếm đòi hỏi phải truy cập sâu vào các nhóm kín, trang web ít phổ biến hoặc các diễn đàn chia sẻ nội dung liên quan. Dữ liệu thu thập chủ yếu là các đoạn video và hình ảnh ghi lại những vụ tai nạn thực tế xảy ra trên đường phố Việt Nam, được đăng tải bởi người dân hoặc các cơ quan chức năng.

Sau khi thu thập, em đã xây dựng một quy trình xử lý nhằm chuẩn hóa và đồng bộ dữ liệu đầu vào trước khi đưa vào mô hình học máy. Nhờ tập trung vào các dữ liệu phản ánh bối cảnh thực tế tại Việt Nam, tập dữ liệu thu được có độ tin cậy cao, thể hiện rõ đặc điểm về loại phương tiện, hành vi giao thông và mật độ lưu thông trên đường – những yếu tố có ảnh hưởng lớn đến hiệu quả nhận diện tai nạn giao thông của hệ thống.

2.3.2. Tiền xử lý dữ liệu



Hình 2.6. Mô tả quy trình xử lý dữ liệu

Loại bỏ dữ liệu không đạt yêu cầu: Những hình ảnh và video có chất lượng kém như độ phân giải thấp, hình ảnh bị mờ, hoặc không rõ tình huống tai nạn đều được loại bỏ để tránh ảnh hưởng đến hiệu quả học của mô hình.

Trích xuất khung hình quan trọng từ video: Các video có chứa vụ tai nạn sẽ được xử lý để trích xuất những khung hình quan trọng. Các khung hình được trích xuất sẽ trở thành dữ liệu ảnh đầu vào cho mô hình.

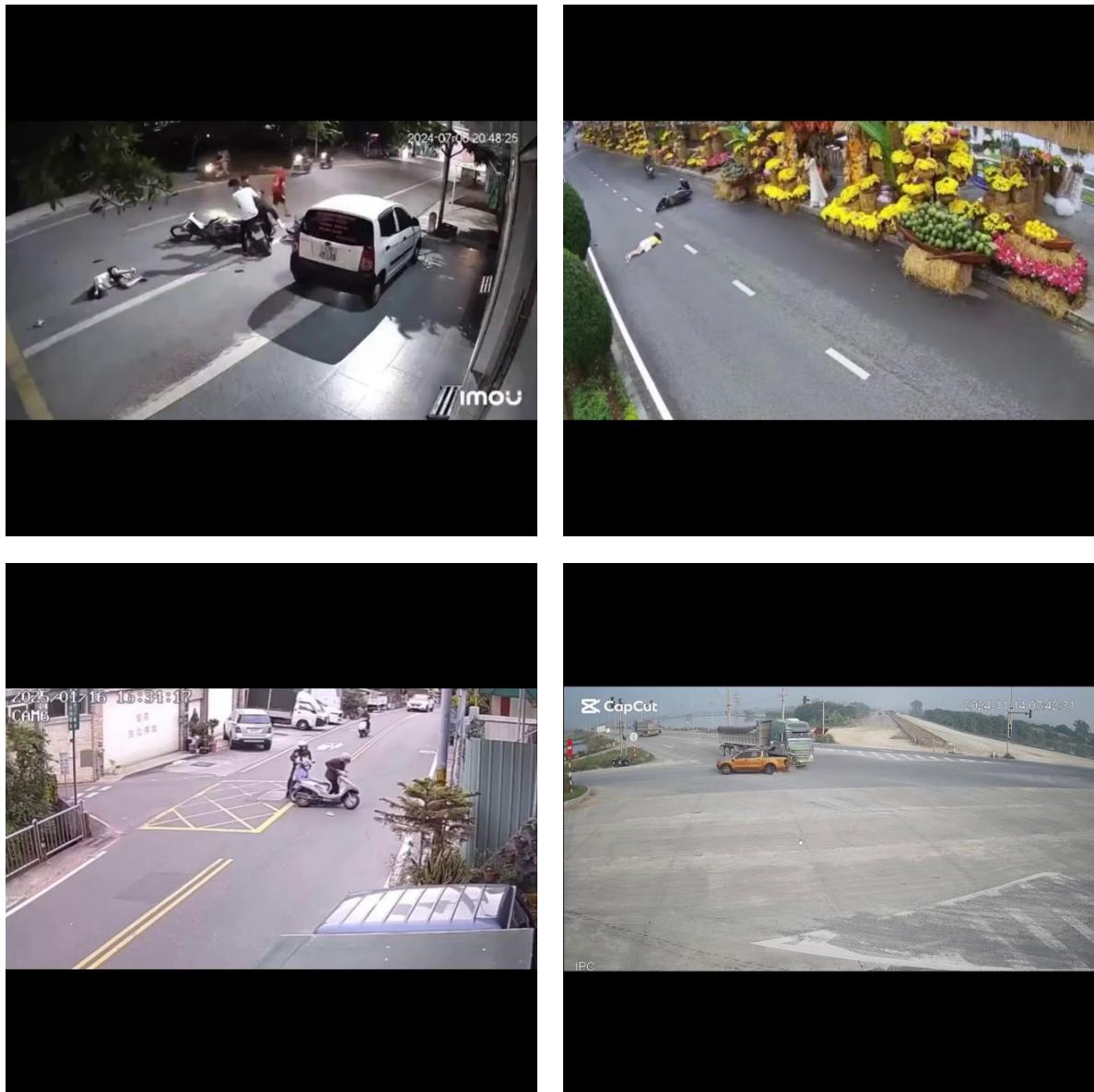
Chuẩn hóa kích thước ảnh: Để đồng bộ đầu vào và tối ưu hiệu suất huấn luyện, tất cả các ảnh đều được resize về kích thước chuẩn là 640×640 pixels. Kích thước này được lựa chọn dựa trên khuyến nghị và thực nghiệm phổ biến đối với các kiến trúc như YOLO, nhằm cân bằng giữa chi tiết hình ảnh và hiệu quả tính toán.

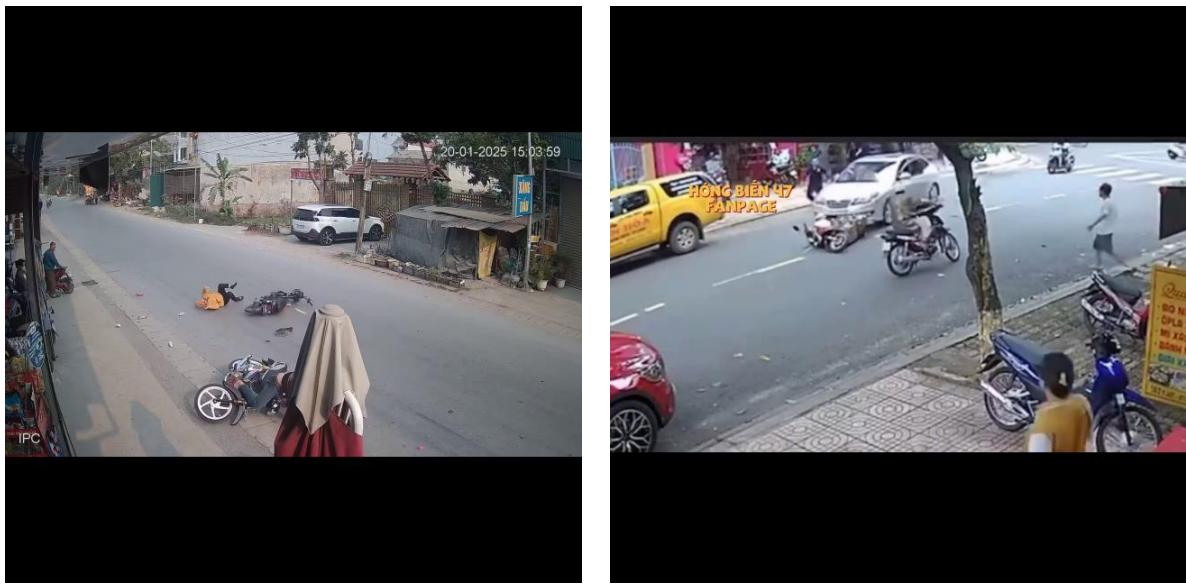
Sau bước tiền xử lý, tập dữ liệu hoàn chỉnh bao gồm 2919 ảnh đã được gán nhãn. Để đảm bảo việc đánh giá mô hình một cách khách quan và tránh hiện tượng quá khớp

(Overfitting), tập dữ liệu này được phân chia thành ba tập con theo tỷ lệ chuẩn: 80% cho tập huấn luyện (Train), 10% cho tập kiểm định (Validation), và 10% cho tập kiểm thử (Test). Cụ thể, số lượng ảnh trong mỗi tập được thể hiện trong bảng sau:

Bảng 2.1. Số lượng ảnh trên mỗi tập dữ liệu

STT	Tên tập dữ liệu	Số lượng ảnh và label tương ứng
1	Train	2335
2	Validation	292
3	Test	292





Hình 2.7. Ví dụ một số tình huống tai nạn giao thông từ tập dữ liệu

Trên đây là một số hình ảnh thu thập được của tập dữ liệu tai nạn giao thông, các hình ảnh được trích xuất từ camera với góc cao đúng với các hệ thống giám sát giao thông trong thực tế.

2.3.3. Gán nhãn dữ liệu

Trong quá trình huấn luyện mô hình nhận diện tai nạn giao thông, việc gán nhãn cho dữ liệu đầu vào đóng vai trò quan trọng nhằm giúp mô hình học sâu (deep learning) hiểu được đối tượng nào cần phát hiện. Dữ liệu hình ảnh được gán nhãn thủ công dựa trên nội dung thực tế trong từng ảnh.

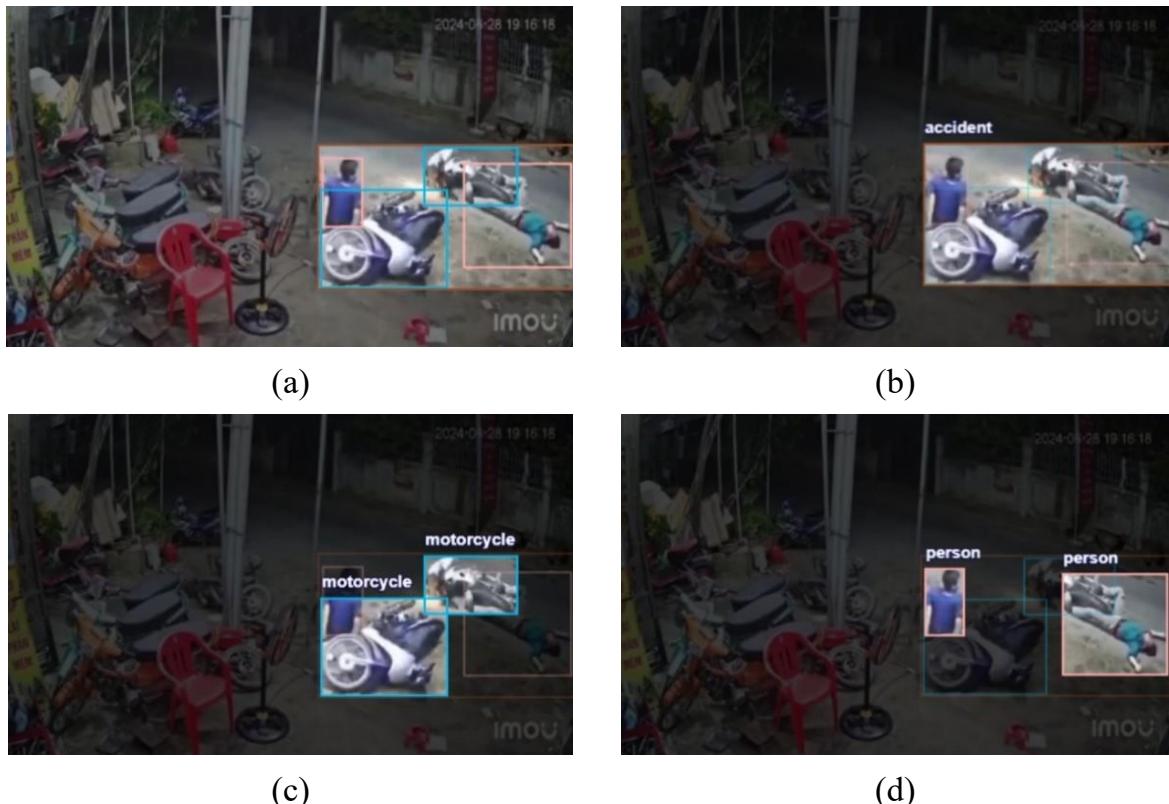
Mỗi ảnh trong tập dữ liệu có thể chứa một hoặc nhiều đối tượng, và các đối tượng này được đánh dấu (bounding box) kèm với tên lớp (label) tương ứng. Việc sử dụng các nhãn cụ thể giúp mô hình YOLO học cách phát hiện chính xác các loại đối tượng quan trọng có thể liên quan đến tai nạn giao thông. Dưới đây là bảng mô tả các nhãn đã sử dụng:

Bảng 2.2. Các nhãn được sử dụng để gán cho ảnh của tập dữ liệu

STT	Nhãn	Giải thích
1	Accident	Tai nạn
2	Bicycle	Xe đạp
3	Bus	Xe buýt
4	Car	Xe ô tô
5	Motorcycle	Xe máy

STT	Nhãn	Giải thích
5	Person	Con người
6	Truck	Xe tải

Để minh họa cho quá trình gán nhãn trong tập dữ liệu, hình dưới đây thể hiện một ảnh được gán nhãn với ba lớp: Accident, Person và Motorcycle. Hình ảnh được trình bày theo các bước cụ thể để giúp người đọc dễ dàng dung từng lớp đối tượng mà mô hình cần học.



Hình 2.8. (a) Hiển thị tổng quan; (b) Nhãn “Accident”; (c) Nhãn “Motorcycle”; (d) Nhãn “Person”

- (a) là ảnh gốc với tất cả các bounding box đã gán nhãn.
- (b), (c), (d) lần lượt hiển thị các đối tượng thuộc từng nhãn riêng biệt nhằm làm rõ cách mỗi nhãn được xác định trong ảnh.

CHƯƠNG 3: TRIỂN KHAI VÀ KẾT QUẢ THỰC NGHIỆM

3.1. Huấn luyện và đánh giá mô hình

3.1.1. Huấn luyện mô hình

Mô hình được huấn luyện trên tập dữ liệu tai nạn giao thông đã xây dựng (bảng 2.1), sử dụng tập Train để học và tập Validation để theo dõi, đánh giá trong quá trình huấn luyện. Môi trường tính toán được sử dụng là một máy laptop với các thông số sau:

Bảng 3.1. Thông số của máy tính được sử dụng để huấn luyện

Hệ điều hành	Windows 11
Bộ vi xử lý (CPU)	Intel Core i7-12700H thế hệ thứ 12
Số lõi CPU	14 lõi
Số luồng xử lý	20 luồng
Tần số tối đa của CPU	2700.00 MHz
Bộ nhớ RAM	16 GB
Card đồ họa (GPU)	NVIDIA GeForce RTX 3050 Ti
Bộ nhớ GPU	4 GB

Các siêu tham số chính cho quá trình huấn luyện được thiết lập như sau:

Số lượng Epoch: 10 – mỗi epoch đại diện cho một lượt duyệt qua toàn bộ tập dữ liệu huấn luyện. Số lượng epoch được giới hạn ở mức 10 trong khuôn khổ thực nghiệm này.

Kích thước lô (Batch Size): 4 – đây là số lượng ảnh được đưa vào mô hình trong một lượt cập nhật trọng số. Kích thước batch size này được chọn dựa trên giới hạn bộ nhớ của GPU.

Mô hình được khởi tạo với trọng số tiền huấn luyện (pre-trained weights) có sẵn của Yolo 11 để tận dụng kiến thức đã học và tăng tốc độ hội tụ cũng như cải thiện hiệu năng cuối cùng.

3.1.2. Đánh giá kết quả

Sau khi hoàn tất quá trình huấn luyện, hiệu năng của mô hình với các biến thể (Nano, Small, Medium, Large, X-Large) được đánh giá một cách định lượng trên tập dữ

liệu Test độc lập. Kết quả đánh giá được thực hiện trên máy laptop với các thông số đã được trình bày ở (bảng 3.1) được tổng hợp như sau:

Bảng 3.2. Kết quả đánh giá trung bình các biến thể YOLOv11 sau 3 lần huấn luyện

Biến thể	Nano	Small	Medium	Large	X-Large
Thông số					
mAP50 (Accident label)	0.9399	0.951	0.936	0.9304	0.8894
mAP50 (All label)	0.7347	0.8454	0.7252	0.7275	0.6876
Average IoU (Accident label)	0.8667	0.8598	0.871	0.8731	0.8518
Average IoU (All label)	0.8424	0.8412	0.8449	0.8472	0.83
FPS	60.3301	52.0596	44.7335	33.8354	21.4596
Latency	0.0166	0.0192	0.0224	0.0295	0.0466

Ghi chú: Xem chi tiết kết quả đánh giá sau 3 lần huấn luyện ở (phụ lục 1)

Chú thích:

- ❖ Chỉ số FPS và Latency được đánh giá với 1000 khung hình / giây.
- ❖ Các giá trị được in đậm thể hiện sự vượt trội của biến thể đó.

Độ chính xác (mAP50)

Ở nhãn “Accident”, tất cả các biến thể đều đạt mAP50 rất cao, với biến thể Small đứng đầu ở mức 0.951, tiếp theo là Nano (0.9399), Medium (0.936) và Large (0.9304). Mặc dù biến thể X-Large là biến thể lớn nhất, nhưng lại có mAP50 thấp nhất (0.8894), cho thấy hiệu quả phát hiện tai nạn không tương xứng với kích thước mô hình.

Tuy nhiên, khi xét trên toàn bộ các nhãn, sự khác biệt trở nên rõ ràng hơn. Biến thể Small đứng đầu với mAP50 là 0.8454, vượt trội so với Nano (0.7347) và đáng kể so với Medium (0.7252), Large (0.7275) và đặc biệt là X-Large (0.6876). Điều này cho thấy biến thể Small không chỉ mạnh về phát hiện tai nạn mà còn tổng thể tốt hơn ở tất cả các lớp.

Mức độ chồng khít (Average IoU)

Chỉ số Average IoU phản ánh độ chồng khít giữa khung dự đoán và thực tế. Nhìn chung, các biến thể của YOLOv11 đều duy trì mức IoU cao và ổn định. Với nhãn

“Accident”, biến thể Large đạt cao nhất (0.8731), tiếp theo là Medium (0.871), Nano (0.8667) và Small (0.8598) và X-Large ở mức thấp hơn (0.8518).

Xét theo tất cả các nhãn, biến thể Large đứng đầu với Average IoU là **0.8472**, trong khi các biến thể còn lại như Medium, Nano và Small đều duy trì ở mức cao và xấp xỉ nhau. Điều này phản ánh khả năng định vị đối tượng chính xác và ổn định của các mô hình YOLOv11.

Tốc độ xử lý

Biến thể Nano vượt trội với tốc độ cao nhất là 60.3301 FPS và độ trễ chỉ 0.0166 giây, cho thấy đây là lựa chọn lý tưởng nếu yêu cầu của hệ thống là cần thời gian xử lý nhanh.

Biến thể Small cũng duy trì hiệu suất ấn tượng với 52.0596 FPS, độ trễ 0.0192 giây, đồng thời có độ chính xác cao nhất, đây là biến thể cân bằng nhất giữa tốc độ và độ chính xác.

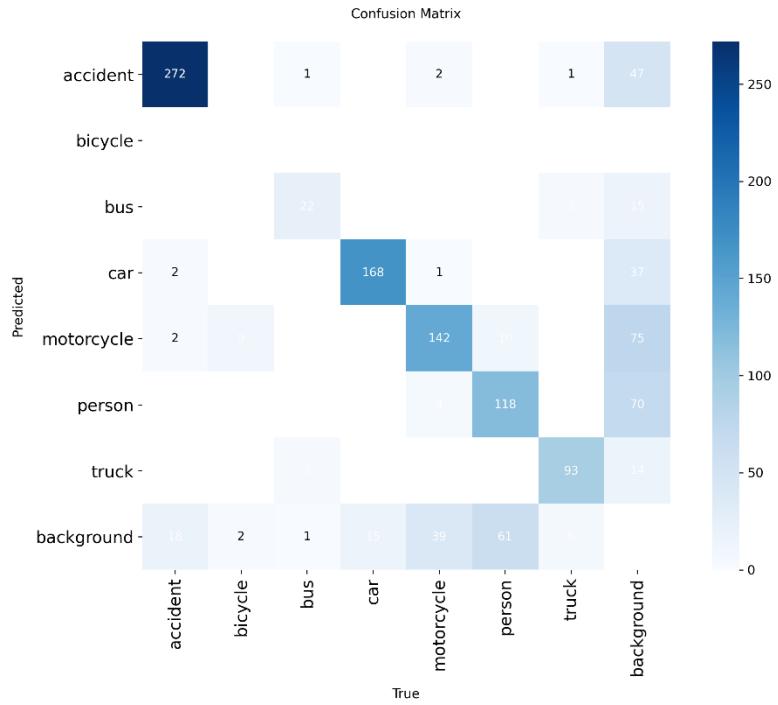
Biến thể Medium (44.7335 FPS), Large (33.8354 FPS) và X-Large (21.4596 FPS) cho tốc độ giảm dần theo kích thước mô hình, đồng thời độ trễ cũng tăng tương ứng (0.0224, 0.0295 và 0.0466 giây). Biến thể X-Large tiếp tục là biến thể có hiệu suất thấp nhất về tốc độ và độ chính xác.

3.1.3. Đánh giá Confusion Matrix của các biến thể

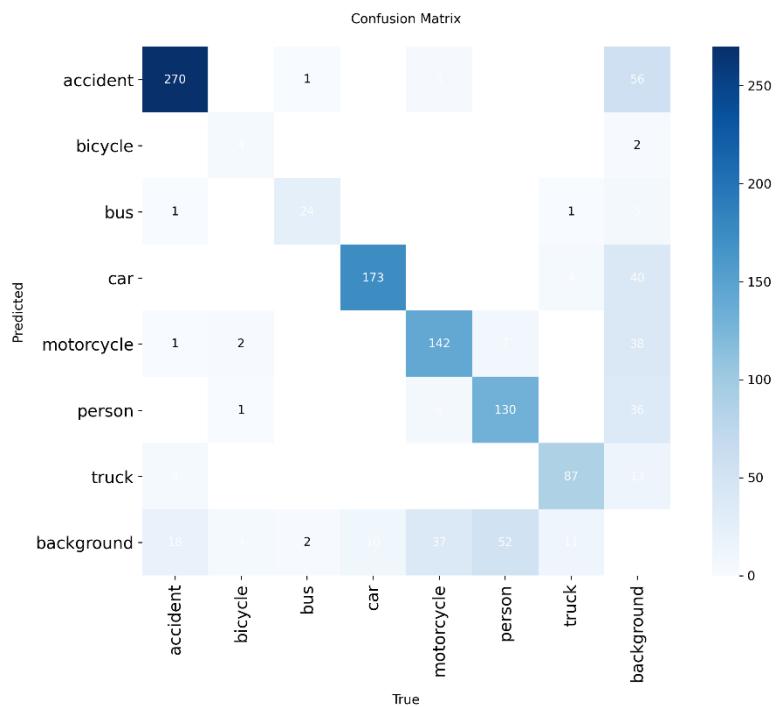
Dưới đây là kết quả Confusion Matrix – Ma trận nhầm lẫn của từng biến thể YOLOv11, được lấy từ một lần huấn luyện do các lần huấn luyện còn lại có kết quả tương đương nhau. Tập dữ liệu kiểm thử gồm 8 lớp: accident, bicycle, bus, car, motorcycle, person, truck và background.

Nhìn chung, các biến thể đều cho kết quả tốt đối với nhãn “accident” với số lượng dự đoán đúng cao (dao động từ 257–272), thể hiện khả năng phát hiện tai nạn ổn định. Tuy nhiên, vẫn có một số nhầm lẫn với “background”.

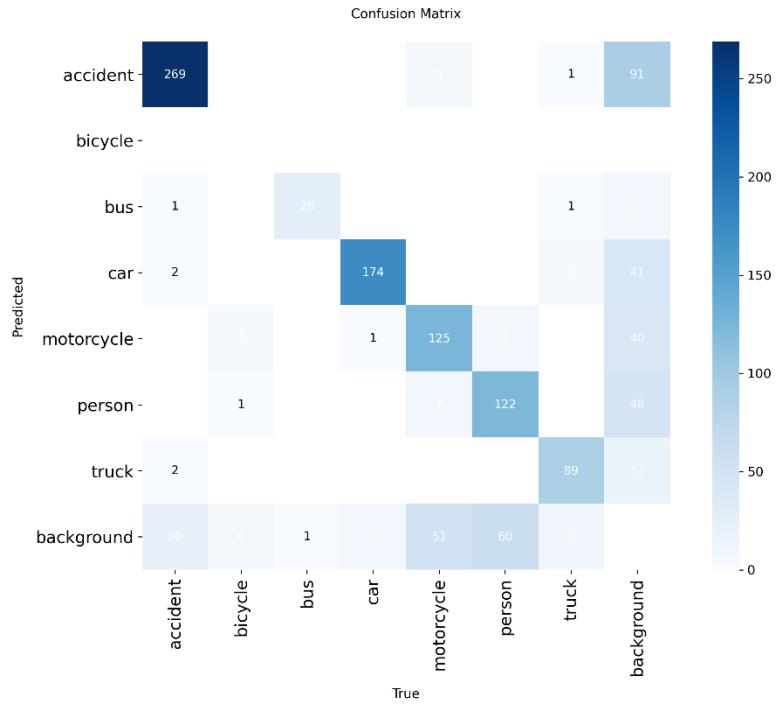
Phần lớn các nhãn đều đạt được số lượng dự đoán đúng đáng kể, và các ô ngoài đường chéo chính (thể hiện nhầm lẫn) đều mờ nhạt, cho thấy độ chính xác cao về mặt phân loại.



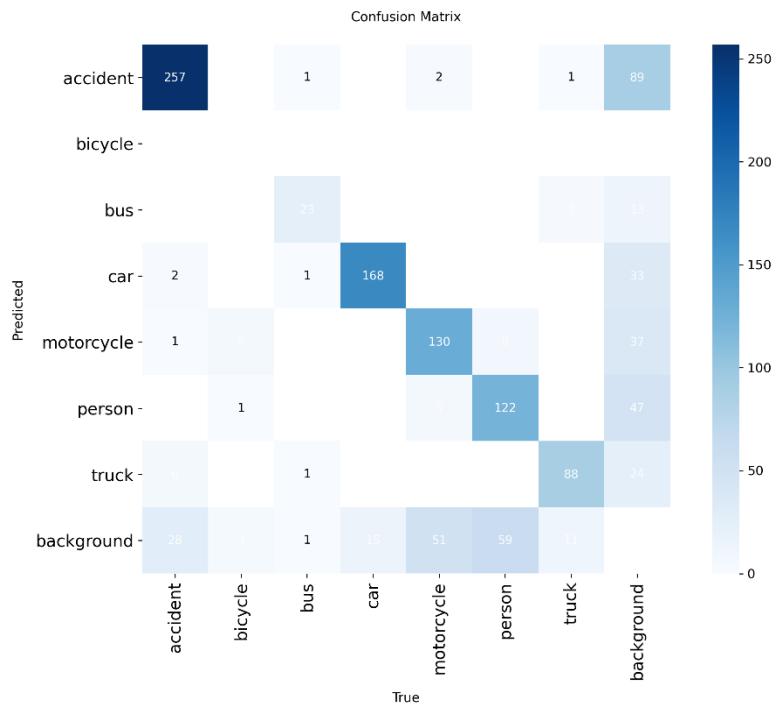
Hình 3.1. Confusion Matrix của YOLOv11 Nano



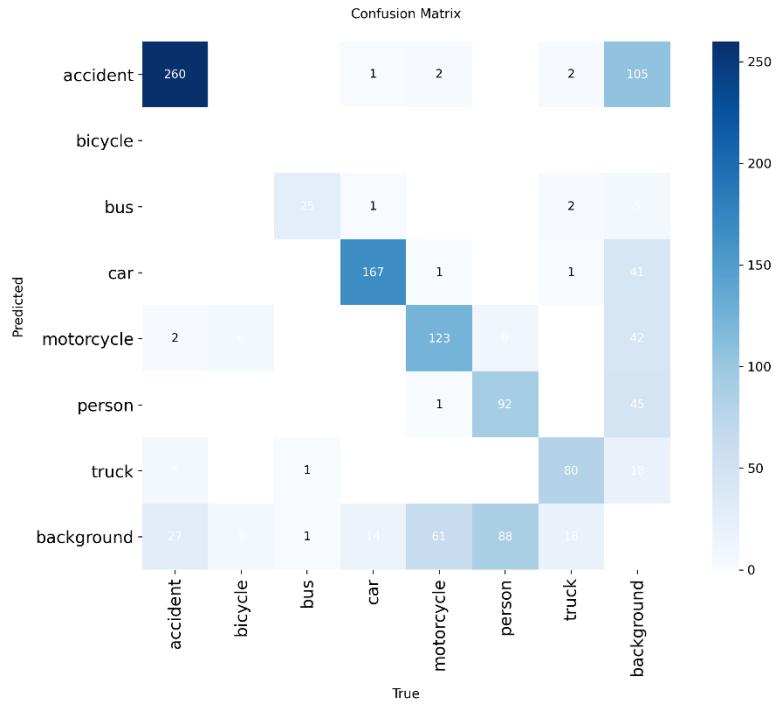
Hình 3.2. Confusion Matrix của YOLOv11 Small



Hình 3.3. Confusion Matrix của YOLOv11 Medium



Hình 3.4. Confusion Matrix của YOLOv11 Large



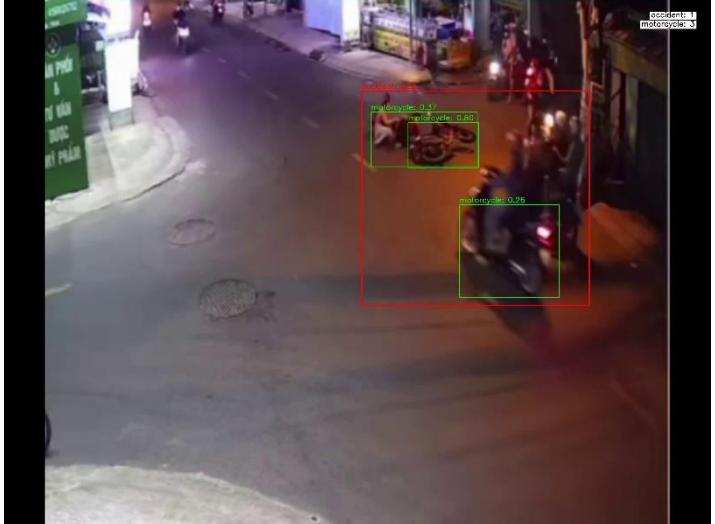
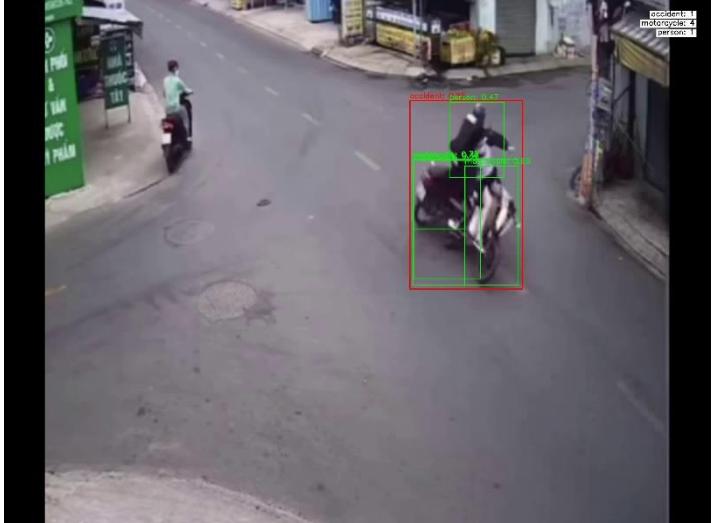
Hình 3.5. Confusion Matrix của YOLOv11 X-Large

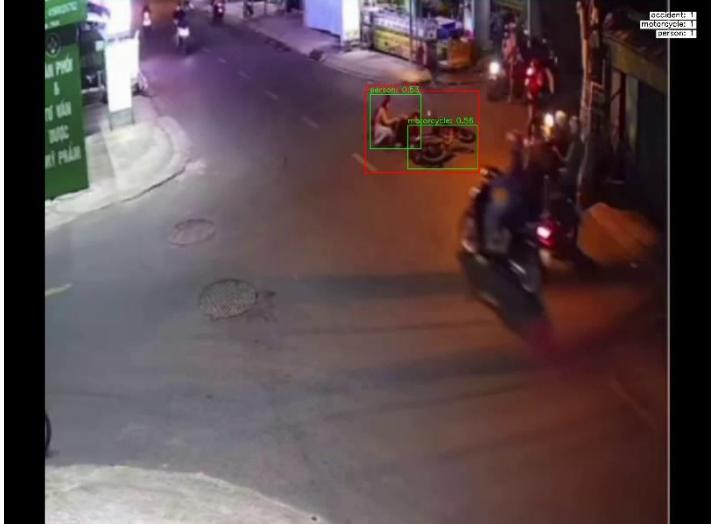
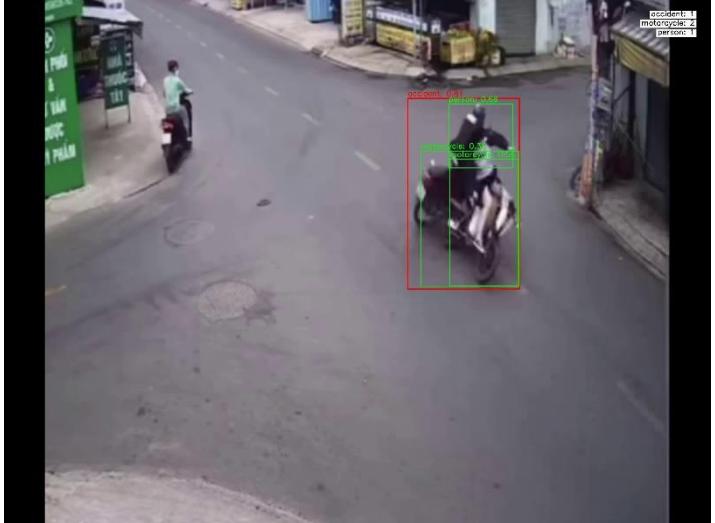
3.1.4. Kết quả phát hiện của mô hình

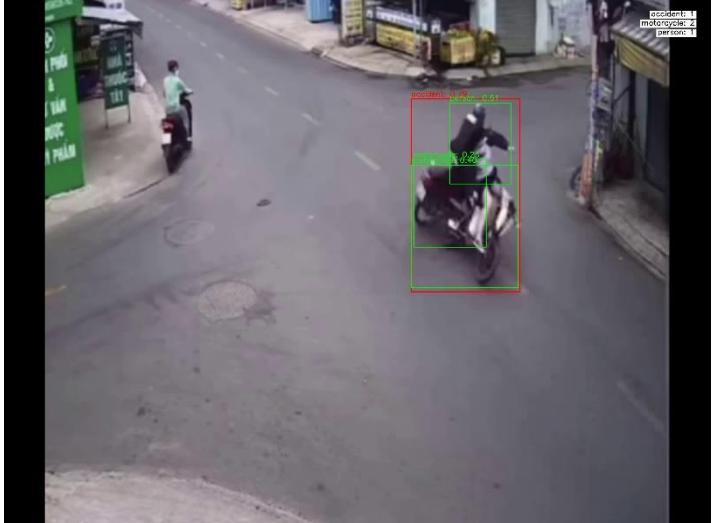
Các hình ảnh ở bên dưới (bảng 3.3) là dự đoán tai nạn giao thông của các biến thể của mô hình YOLO 11 đã được huấn luyện với bộ dữ liệu tai nạn giao thông với pretrain được cung cấp sẵn.

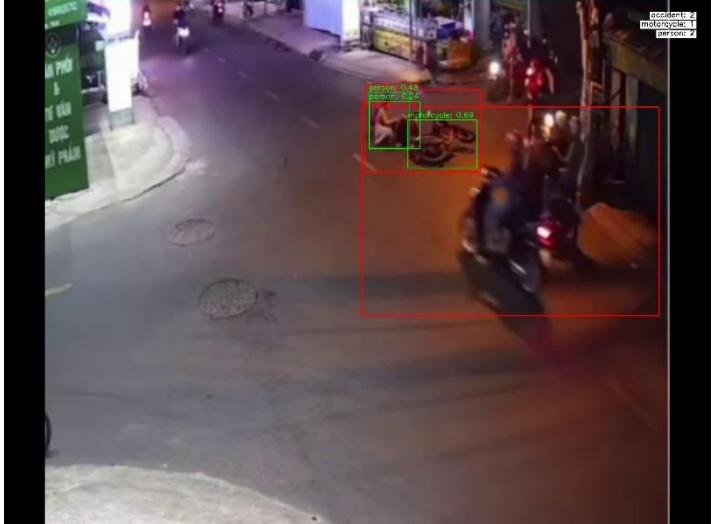
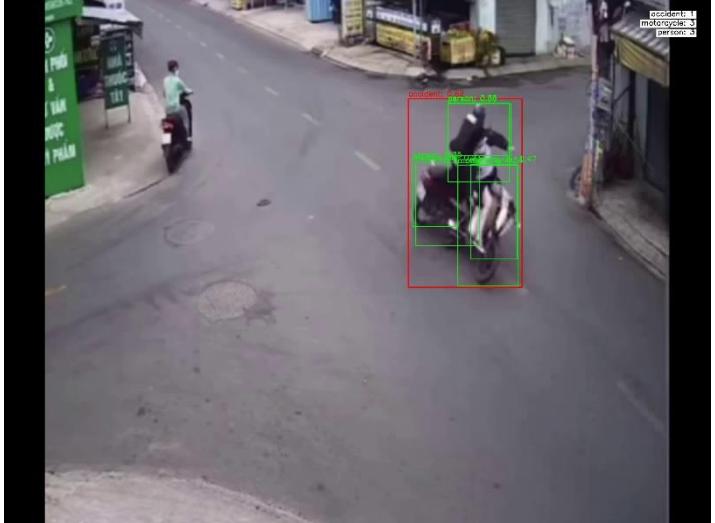
Bảng 3.3 Kết quả phát hiện của các biến thể phiên bản YOLO 11

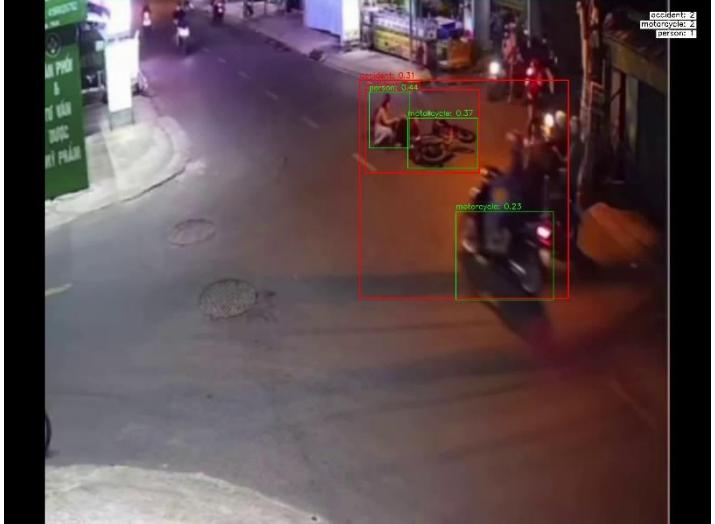
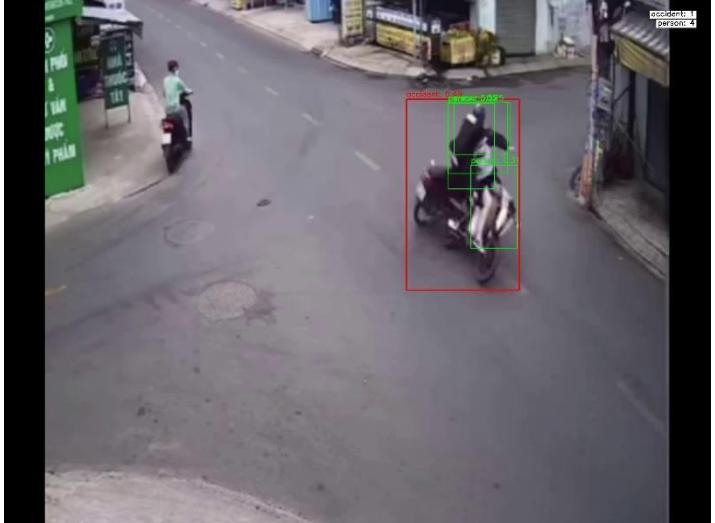
Biến thể	Ảnh nhận diện
Nano	

Biến thể	Ảnh nhận diện
	
	
Small	

Biến thể	Ảnh nhận diện
	
	
Medium	

Biến thể	Ảnh nhận diện
	
	
Large	

Biến thể	Ảnh nhận diện
	
	
X-Large	

Biến thể	Ảnh nhận diện
	
	

3.2. Giao diện hệ thống quản lý giao thông

Lịch sử cảnh báo

- Camera:** qbDa7B44tJmCxO **Địa chỉ stream:** rtmp://35.197.153.95:1935/live/cam1 **Phát hiện:** accident **Score:** 0.79 [Xem chi tiết dữ liệu thô](#)
- Camera:** qbDa7B44tJmCxO **Địa chỉ stream:** rtmp://35.197.153.95:1935/live/cam1 **Phát hiện:** accident **Score:** 0.81 [Xem chi tiết dữ liệu thô](#)
- Camera:** qbDa7B44tJmCxO **Địa chỉ stream:** rtmp://35.197.153.95:1935/live/cam1 **Phát hiện:** accident **Score:** 0.81 [Xem chi tiết dữ liệu thô](#)

Hình 3.6. Giao diện danh sách lịch sử cảnh báo được hệ thống AI phát hiện

Hiển thị danh sách thông tin phát hiện tai nạn giao thông từ các luồng camera được hệ thống AI ghi nhận rồi gửi lên Kafka, từ đó hệ thống quản lý giao thông nhận dữ liệu từ Kafka rồi thông báo lên giao diện cho người dùng và lưu lại vào MongoDB.

Lịch sử cảnh báo

Camera: qbDa7B44tJmCxO **Địa chỉ stream:** rtmp://35.197.153.95:1935/live/cam1 **Phát hiện:** accident **Score:** 0.79 [Xem chi tiết dữ liệu thô](#)

```
{
  "_id": "68568a2de6074c2c45234c70",
  "camera_url": "rtmp://35.197.153.95:1935/live/cam1",
  "camera_serial": "qbDa7B44tJmCxO",
  "detections": {
    "boxes": [
      [
        2190.47921484375,
        40.573883056640625,
        2704.85107421875,
        1639,
        8846435546875.0
      ]
    ]
  }
}
```

Copy

Camera: qbDa7B44tJmCxO **Địa chỉ stream:** rtmp://35.197.153.95:1935/live/cam1 **Phát hiện:** accident **Score:** 0.81 [Xem chi tiết dữ liệu thô](#)

Hình 3.7. Giao diện xem chi tiết dữ liệu thô thông tin phát hiện tai nạn

Hiển thị thông tin chi tiết thông tin phát hiện tai nạn từ hệ thống AI, đây là dữ liệu thô bao gồm các thông tin như: đường dẫn luồng phát của camera, số serial, nhãn, điểm tin cậy...

#	Thời gian tạo	Camera	Ảnh chụp	Người xác nhận	Hành động
1	26/06/2025 01:51:41	VWMImxG8OfsV6OHk	Không có ảnh	kiethoang101.dev@gmail.com	<button>Sửa</button> <button>Xóa</button>
2	22/06/2025 16:21:52	VWMImxG8OfsV6OHk	Không có ảnh	kiethoang101.dev@gmail.com	<button>Sửa</button> <button>Xóa</button>
3	21/06/2025 23:58:44	VWMImxG8OfsV6OHk	Không có ảnh	kiethoang101.dev@gmail.com	<button>Sửa</button> <button>Xóa</button>
4	21/06/2025 20:17:01	qbDa7B44tJmCxO	Không có ảnh	kiethoang101.dev@gmail.com	<button>Sửa</button> <button>Xóa</button>
5	21/06/2025 07:06:15	VWMImxG8OfsV6OHk	Không có ảnh	kiethoang101.dev@gmail.com	<button>Sửa</button> <button>Xóa</button>
6	19/06/2025 10:34:30	qbDa7B44tJmCxO	Không có ảnh	kiethoang101.dev@gmail.com	<button>Sửa</button> <button>Xóa</button>
7	18/06/2025 20:05:32	VWMImxG8OfsV6OHk	Không có ảnh	kiethoang101.dev@gmail.com	<button>Sửa</button> <button>Xóa</button>
8	18/06/2025 02:20:03	qbDa7B44tJmCxO	Không có ảnh	kiethoang101.dev@gmail.com	<button>Sửa</button> <button>Xóa</button>
9	13/06/2025 20:08:04	VWMImxG8OfsV6OHk	Không có ảnh	kiethoang101.dev@gmail.com	<button>Sửa</button> <button>Xóa</button>
10	13/06/2025 17:04:42	qbDa7B44tJmCxO	Không có ảnh	kiethoang101.dev@gmail.com	<button>Sửa</button> <button>Xóa</button>

Trang 1 / 5 [Tiếp »](#)

Hình 3.8. Giao diện danh sách tai nạn được ghi nhận

Trang hiển thị danh sách các tai nạn đã được người dùng ghi nhận bao gồm các thông tin như: Thời gian ghi nhận, số serial, ảnh chụp của vụ tai nạn từ hệ thống AI, email của người xác nhận.

#	Thời gian tạo	Camera	Ảnh chụp	Người xác nhận	Hành động
1	26/06/2025 01:51:41	VWMImxG8OfsV6OHk	Không có ảnh	kiethoang101.dev@gmail.com	<button>Sửa</button> <button>Xóa</button>
2	22/06/2025 16:21:52	VWMImxG8OfsV6OHk	Không có ảnh	kiethoang101.dev@gmail.com	<button>Sửa</button> <button>Xóa</button>
3	21/06/2025 23:58:44	VWMImxG8OfsV6OHk	Không có ảnh	kiethoang101.dev@gmail.com	<button>Sửa</button> <button>Xóa</button>
4	21/06/2025 20:17:01	qbDa7B44tJmCxO	Không có ảnh	kiethoang101.dev@gmail.com	<button>Sửa</button> <button>Xóa</button>
5	21/06/2025 07:06:15	VWMImxG8OfsV6OHk	Không có ảnh	kiethoang101.dev@gmail.com	<button>Sửa</button> <button>Xóa</button>
6	19/06/2025 10:34:30	qbDa7B44tJmCxO	Không có ảnh	kiethoang101.dev@gmail.com	<button>Sửa</button> <button>Xóa</button>
7	18/06/2025 20:05:32	VWMImxG8OfsV6OHk	Không có ảnh	kiethoang101.dev@gmail.com	<button>Sửa</button> <button>Xóa</button>
8	18/06/2025 02:20:03	qbDa7B44tJmCxO	Không có ảnh	kiethoang101.dev@gmail.com	<button>Sửa</button> <button>Xóa</button>
9	13/06/2025 20:08:04	VWMImxG8OfsV6OHk	Không có ảnh	kiethoang101.dev@gmail.com	<button>Sửa</button> <button>Xóa</button>
10	13/06/2025 17:04:42	qbDa7B44tJmCxO	Không có ảnh	kiethoang101.dev@gmail.com	<button>Sửa</button> <button>Xóa</button>

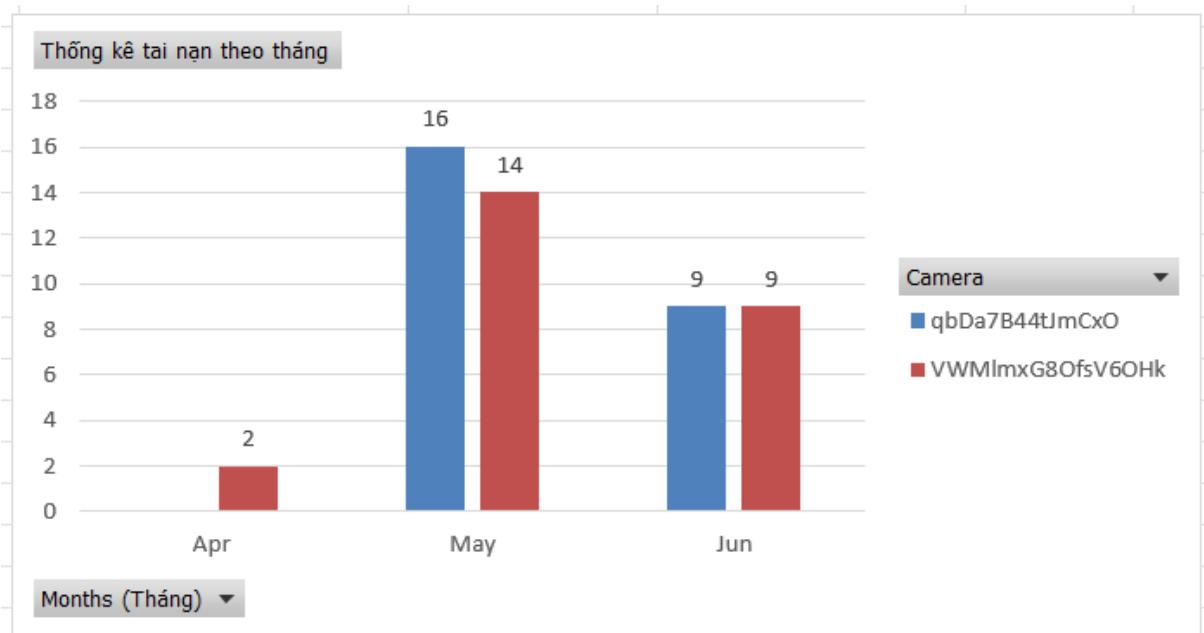
Trang 1 / 5 [Tiếp »](#)

Hình 3.9. Giao diện xuất danh sách tai nạn

Cho phép người dùng chọn khoảng thời gian để xuất dữ liệu ra excel, mặc định thời gian bắt đầu là từ đầu năm hiện tại, thời gian kết thúc là ngày hiện tại.

A	B	C	D	
1	#	Thời gian tạo	Camera Serial	Người xác nhận
2	1	25/06/2025 18:51:41	VWMImxG8OfsV6OHk	kiethoang101.dev@gmail.com
3	2	22/06/2025 09:21:52	VWMImxG8OfsV6OHk	kiethoang101.dev@gmail.com
4	3	21/06/2025 16:58:44	VWMImxG8OfsV6OHk	kiethoang101.dev@gmail.com
5	4	21/06/2025 13:17:01	qbDa7B44tJmCxO	kiethoang101.dev@gmail.com
6	5	21/06/2025 00:06:15	VWMImxG8OfsV6OHk	kiethoang101.dev@gmail.com
7	6	19/06/2025 03:34:30	qbDa7B44tJmCxO	kiethoang101.dev@gmail.com
8	7	18/06/2025 13:05:32	VWMImxG8OfsV6OHk	kiethoang101.dev@gmail.com
9	8	17/06/2025 19:20:03	qbDa7B44tJmCxO	kiethoang101.dev@gmail.com
10	9	13/06/2025 13:08:04	VWMImxG8OfsV6OHk	kiethoang101.dev@gmail.com
11	10	13/06/2025 10:04:42	qbDa7B44tJmCxO	kiethoang101.dev@gmail.com
12	11	11/06/2025 10:21:25	qbDa7B44tJmCxO	kiethoang101.dev@gmail.com
13	12	10/06/2025 12:48:20	VWMImxG8OfsV6OHk	kiethoang101.dev@gmail.com
14	13	07/06/2025 20:08:54	qbDa7B44tJmCxO	kiethoang101.dev@gmail.com
15	14	05/06/2025 15:38:38	qbDa7B44tJmCxO	kiethoang101.dev@gmail.com
16	15	03/06/2025 18:21:14	VWMImxG8OfsV6OHk	kiethoang101.dev@gmail.com
17	16	03/06/2025 09:08:24	qbDa7B44tJmCxO	kiethoang101.dev@gmail.com
18	17	01/06/2025 23:26:52	VWMImxG8OfsV6OHk	kiethoang101.dev@gmail.com
19	18	01/06/2025 01:39:09	qbDa7B44tJmCxO	kiethoang101.dev@gmail.com
20	19	31/05/2025 13:00:24	VWMImxG8OfsV6OHk	kiethoang101.dev@gmail.com
21	20	31/05/2025 06:33:23	VWMImxG8OfsV6OHk	kiethoang101.dev@gmail.com
22	21	30/05/2025 23:21:54	VWMImxG8OfsV6OHk	kiethoang101.dev@gmail.com
23	22	30/05/2025 13:08:04	qbDa7B44tJmCxO	kiethoang101.dev@gmail.com
24	23	29/05/2025 23:07:29	VWMImxG8OfsV6OHk	kiethoang101.dev@gmail.com
25	24	28/05/2025 09:43:24	qbDa7B44tJmCxO	kiethoang101.dev@gmail.com
26	25	25/05/2025 23:03:18	qbDa7B44tJmCxO	kiethoang101.dev@gmail.com
27	26	25/05/2025 16:22:37	VWMImxG8OfsV6OHk	kiethoang101.dev@gmail.com
28	27	25/05/2025 00:00:45	VWMImxG8OfsV6OHk	kiethoang101.dev@gmail.com

Hình 3.10. Nội dung danh sách tai nạn sau khi xuất excel



Hình 3.11. Biểu đồ thống kê dựa trên dữ liệu xuất excel

#	Thời gian tạo	Camera	Ảnh chụp	Người xác nhận	Hành động
1	26/06/2025 01:51:41	VWMLmxG8OfsV6OHk	Không có ảnh	kiethoang101.dev@gmail.com	[Sửa] [Xóa]
2	22/06/2025 16:21:52	VWMLmxG8OfsV6OHk	Không có ảnh	kiethoang101.dev@gmail.com	[Sửa] [Xóa]
3	21/06/2025 23:58:44	VWMLmxG8OfsV6OHk	Không có ảnh	kiethoang101.dev@gmail.com	[Sửa] [Xóa]
4	21/06/2025 20:17:01	qbDa7B44tJmCxO	Không có ảnh	kiethoang101.dev@gmail.com	[Sửa] [Xóa]
5	21/06/2025 07:06:15	VWMLmxG8OfsV6OHk	Không có ảnh	kiethoang101.dev@gmail.com	[Sửa] [Xóa]
6	19/06/2025 10:34:30	qbDa7B44tJmCxO	Không có ảnh	kiethoang101.dev@gmail.com	[Sửa] [Xóa]
7	18/06/2025 20:05:32	VWMLmxG8OfsV6OHk	Không có ảnh	kiethoang101.dev@gmail.com	[Sửa] [Xóa]
8	18/06/2025 02:20:03	qbDa7B44tJmCxO	Không có ảnh	kiethoang101.dev@gmail.com	[Sửa] [Xóa]
9	13/06/2025 20:08:04	VWMLmxG8OfsV6OHk	Không có ảnh	kiethoang101.dev@gmail.com	[Sửa] [Xóa]
10	13/06/2025 17:04:42	qbDa7B44tJmCxO	Không có ảnh	kiethoang101.dev@gmail.com	[Sửa] [Xóa]

Hình 3.12. Giao diện xóa thông tin ghi nhận tai nạn

Hiển thị bảng thông báo xác nhận xóa thông tin ghi nhận tai nạn để tránh người dùng xóa nhầm.

Giám sát giao thông Trang chủ Lịch sử cảnh báo Tai nạn đã được ghi nhận

Thêm tai nạn

Camera serial

Snapshot Choose File No file chosen

Thêm Hủy

Hình 3.13. Giao diện tạo thông tin ghi nhận tai nạn

Giám sát giao thông Trang chủ Lịch sử cảnh báo Tai nạn đã được ghi nhận

Chỉnh sửa tai nạn

Camera serial FDd5dEaeBb

Snapshot Choose File No file chosen

Lưu thay đổi Hủy

Hình 3.14. Giao diện chỉnh sửa thông tin ghi nhận tai nạn

3.3. Giao diện hệ thống AI

Camera Streams

+ Add New Camera Mock Accident Detect

ID	Serial	Location	Stream URL	Status	Action
682a332c-77a2-45f1-83c...	qbDa7B44tJmCxO	Ngã 4 Thủ Đức	rtmp://35.197.153.95:19...	Inactive	<button>Toggle</button> <button>Edit</button> <button>Delete</button>
ecf6a5f3-a228-4e45-a378...	VWMImxG8OfsV6OHk	Vòng xoay Tam Hiệp	rtmp://35.197.153.95:19...	Inactive	<button>Toggle</button> <button>Edit</button> <button>Delete</button>

Hình 3.15. Giao diện hiển thị danh sách các luồng camera

Trang hiển thị danh sách các luồng camera được lưu trữ của hệ thống AI để quan sát phát hiện tai nạn giao thông.

Add New Camera

Serial

Location

Stream URL

Is Active

Save Cancel

Hình 3.16. Giao diện thêm mới luồng camera

Trang để thêm mới một luồng camera, bao gồm: Số serial, địa điểm, đường dẫn luồng phát và trạng thái.

Edit Camera

Serial

Location

Stream URL

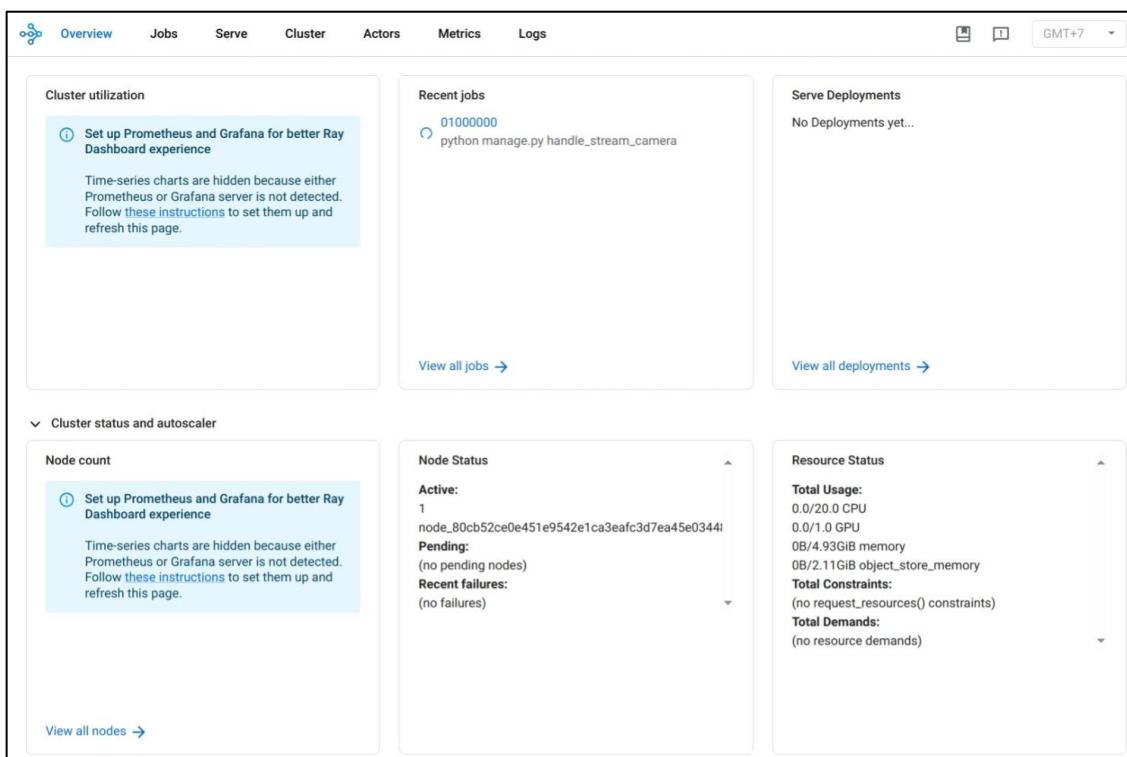
Is Active

Save Cancel

Hình 3.17. Giao diện chỉnh sửa luồng camera

Hình 3.18. Giao diện gửi dữ liệu nhận diện tai nạn mẫu

Trang để gửi dữ liệu phát hiện tai nạn giao thông mẫu để gửi lên Kafka và từ đó hệ thống quản lý giao thông nhận được.



Hình 3.19. Giao diện tổng quan Ray

Trang tổng quan dashboard của Apache Ray, hiển thị các job được chạy gần đây nhất, trạng thái các node, trạng thái tài nguyên...

Job ID	Submission ID	Entrypoint	Status	Status message	Duration	Tasks	Actions	StartTime	EndTime
01000000	-	python manage.py handle_stream_camera	RUNNING	-	1m 15s	CPU Flame Graph (Driver) Memory Profiling (Driver)	Log Stack Trace (Driver) CPU Flame Graph (Driver) Memory Profiling (Driver)	2025/06/26 17:16:40	-

Hình 3.20. Giao diện các job của Ray

Hiển thị thông tin về các job, các thông tin như: trạng thái, thời gian chạy, thời gian bắt đầu, thời gian kết thúc...

Host / Worker Process name	State	State Message	ID	IP / PID	Actions	CPU	Memory	GPU	GRAM
KYK	ALIVE	-	80cb...	21.64.69.49 (Head)	Log	11%	11.02GB/15.32GB(71.9%)	[0]: 22.0%	[0]: 257MiB/4

Hình 3.21. Giao diện các cụm Ray

Hiển thị thông tin về các node trong cụm, các thông tin như: tổng số lượng node, node còn chạy, node đã ngưng, tên hoặc ip của cụm, tài nguyên đang được sử dụng như thế nào.

KẾT LUẬN VÀ KIẾN NGHỊ

1. Kết quả đạt được

Trong quá trình thực hiện đồ án, em có cơ hội để trau dồi và vận dụng những kiến thức từ các môn học: Lập trình hướng đối tượng, Cấu trúc dữ liệu và giải thuật, Kiến trúc và tổ chức máy tính, Cơ sở dữ liệu, Lập trình Web, Mạng máy tính, Lập trình API, Trí tuệ nhân tạo, Học máy cơ bản, Khai phá dữ liệu... Ngoài ra trong quá trình thực hiện, bản thân đã học hỏi thêm được nhiều về Thị giác máy tính, CNN, Thu thập và xử lý dữ liệu, Huấn luyện mô hình học sâu, Framework Django, Kafka...

Từ những kiến thức em đã tìm hiểu và vận dụng trong quá trình thực hiện, em đã xây dựng được một hệ thống giám sát giao thông tích hợp mô hình YOLO để phát hiện tai nạn giao thông. Hệ thống này bao gồm các chức năng như:

- Khả năng nhận diện và phân loại được đối tượng giao thông trong tai nạn
- Phát hiện chính xác và tức thời khi tai nạn xảy ra
- Quản lý các luồng stream của camera
- Xem lịch sử phát hiện tai nạn giao thông từ hệ thống AI
- Quản lý thông tin tai nạn được ghi nhận

Dù khoảng thời gian thực hiện đồ án không dài, nhưng đây là một trải nghiệm vô cùng quý giá, giúp em không chỉ củng cố kiến thức chuyên môn mà còn rèn luyện nhiều kỹ năng quan trọng khác như tìm kiếm, nghiên cứu tài liệu, phân tích và định hình bài toán thực tế để đưa ra giải pháp tối ưu, cách nhìn nhận và xử lý vấn đề, các kỹ năng trình bày báo cáo hợp lý. Song đó, em cũng đã học được cách làm việc với một tinh thần chủ động hơn, đổi mới với khó khăn bằng sự kiên trì và không ngại thử nghiệm. Những kinh nghiệm này chắc chắn sẽ là hành trang vững chắc cho con đường sự nghiệp của em sau này.

Như vậy, với những nỗ lực và kết quả đạt được, em tự tin rằng mình đã hoàn thành cơ bản các nhiệm vụ cốt lõi của đề tài.

2. Hạn chế

Bên cạnh những thành quả đã đạt được, em cũng nhận ra rằng đồ án vẫn còn một số điểm hạn chế. Thời gian thực hiện có hạn và trình độ kiến thức còn nhiều hạn chế và đặc biệt là tài nguyên hệ thống trong quá trình thực hiện, một số khía cạnh chưa thể hoàn thiện như mong đợi.

Bản thân em dành khá nhiều thời gian ban đầu để đọc hiểu và nắm bắt các tài liệu chuyên sâu về công nghệ mới đã phần nào rút ngắn quỹ thời gian dành cho việc tối ưu hóa và hoàn thiện sản phẩm.

Hiệu suất mô hình phát hiện tai nạn vẫn chưa tối ưu trong mọi điều kiện thực tế, còn tỉ lệ sai sót nhất định.

Hệ thống giám sát tổng thể chỉ dừng ở mức nghiên cứu thử nghiệm để có bức tranh tổng quan về hệ thống, chưa có đủ tính năng cần thiết để triển khai thực tế như quản lý dữ liệu lớn, khả năng mở rộng đa camera, hay các lớp bảo mật.

Giới hạn tài nguyên hệ thống cũng ảnh hưởng lớn đến việc tối ưu mô hình và phát triển các tính năng nâng cao.

3. Hướng phát triển

Với những kết quả đã đạt được và các hạn chế nêu trên, bản thân em luôn ấp ủ mong muốn tiếp tục hoàn thiện và phát triển đề tài này. Em hướng đến việc khắc phục các điểm còn hạn chế, tiếp tục nghiên cứu để hệ thống có thể đóng góp nhiều hơn vào thực tiễn.

Trước tiên, để giải quyết các hạn chế em sẽ tập trung nâng cao hiệu suất mô hình phát hiện tai nạn bằng cách nghiên cứu cách gán nhãn tốt hơn, các kỹ thuật tối ưu hóa và đa dạng hóa bộ dữ liệu huấn luyện, nhằm giảm thiểu sai sót và thích nghi tốt hơn với mọi điều kiện thực tế. Đồng thời, em sẽ hoàn thiện các tính năng cốt lõi của hệ thống giám sát để tiến gần hơn đến khả năng triển khai thực tế.

Trong tương lai xa hơn, em thực sự muốn cho đề tài này vươn xa hơn nữa. Tầm nhìn của em là một hệ thống giám sát giao thông thông minh và giúp ích cho xã hội. Không chỉ dừng ở việc phát hiện tai nạn giao thông, với hệ thống này có thể mở rộng để phân tích từ các lịch sử tai nạn giao thông ghi nhận được, từ đó cho ra được những điểm giao thông thường xảy ra tai nạn để có biện pháp khắc phục.

Với tất cả sự nỗ lực, em tin rằng đề tài này sẽ góp phần cải thiện an toàn giao thông, thúc đẩy xã hội phát triển bền vững, từ nghiên cứu thành giá trị thực tiễn thiết thực. Nếu làm được điều đó thì chắc chắn sẽ là niềm hạnh phúc và tự hào lớn nhất trên hành trình phát triển của em.

PHỤ LỤC

Phụ lục 1: Kết quả đánh giá mô hình sau 3 lần huấn luyện

Bảng 0.1. Kết quả đánh giá các biến thể của YOLOv11 lần huấn luyện 1

Lần 1						
Biến thể	Nano	Small	Medium	Large	X-Large	
Thông số						
mAP50 (Accident label)	0.9399	0.951	0.936	0.9304	0.8894	
mAP50 (All label)	0.7347	0.8454	0.7252	0.7275	0.6876	
Average IoU (Accident label)	0.8667	0.8598	0.871	0.8731	0.8518	
Average IoU (All label)	0.8424	0.8412	0.8449	0.8472	0.83	
FPS	61.0164	53.4893	44.8038	33.6568	21.5722	
Latency	0.0164	0.0187	0.0223	0.0297	0.0464	

Bảng 0.2. Kết quả đánh giá các biến thể của YOLOv11 lần huấn luyện 2

Lần 2						
Biến thể	Nano	Small	Medium	Large	X-Large	
Thông số						
mAP50 (Accident label)	0.9399	0.951	0.936	0.9304	0.8894	
mAP50 (All label)	0.7347	0.8454	0.7252	0.7275	0.6876	
Average IoU (Accident label)	0.8667	0.8598	0.871	0.8731	0.8518	
Average IoU (All label)	0.8424	0.8412	0.8449	0.8472	0.83	
FPS	61.6948	51.1761	44.8775	33.7326	21.6481	
Latency	0.0162	0.0195	0.0223	0.0296	0.0462	

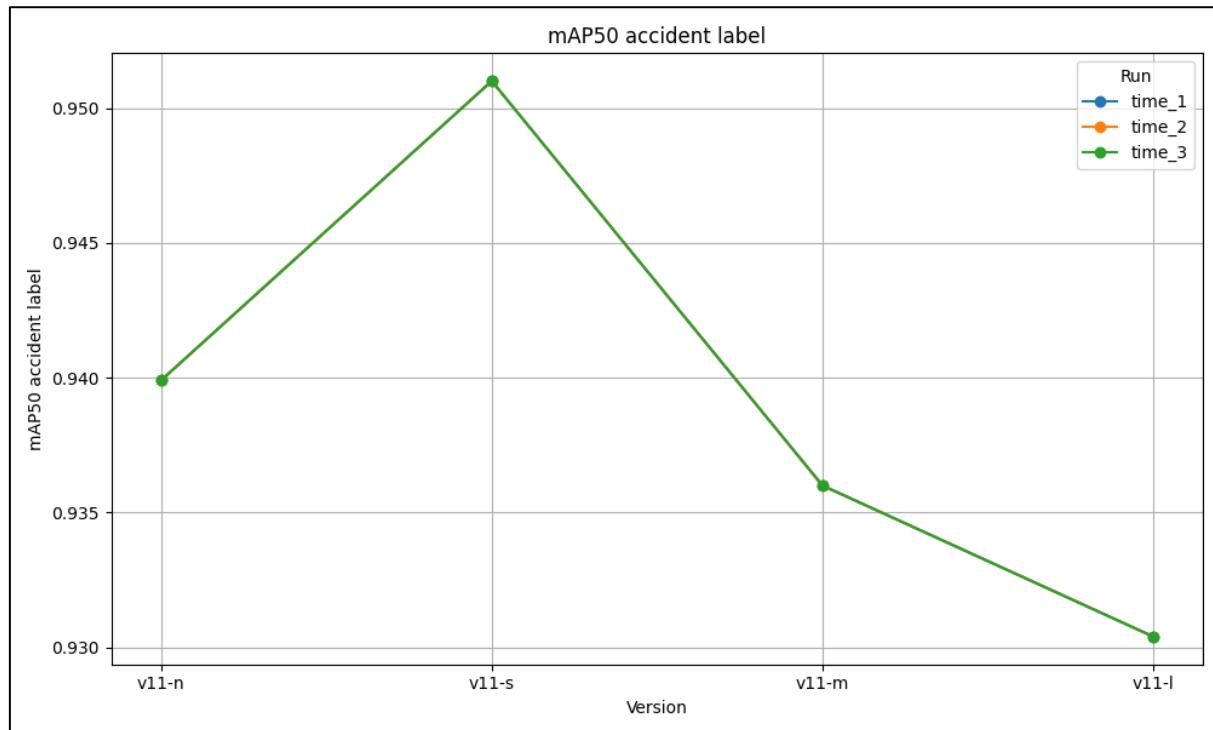
Bảng 0.3. Kết quả đánh giá các biến thể của YOLOv11 lần huấn luyện 3

Lần 3						
Thông số	Biến thể	Nano	Small	Medium	Large	X-Large
mAP50 (Accident label)		0.9399	0.951	0.936	0.9304	0.8894
mAP50 (All label)		0.7347	0.8454	0.7252	0.7275	0.6876
Average IoU (Accident label)		0.8667	0.8598	0.871	0.8731	0.8518
Average IoU (All label)		0.8424	0.8412	0.8449	0.8472	0.83
FPS		58.2792	51.5135	44.5193	34.1168	21.1584
Latency		0.0172	0.0194	0.0225	0.0293	0.0473

Chú thích:

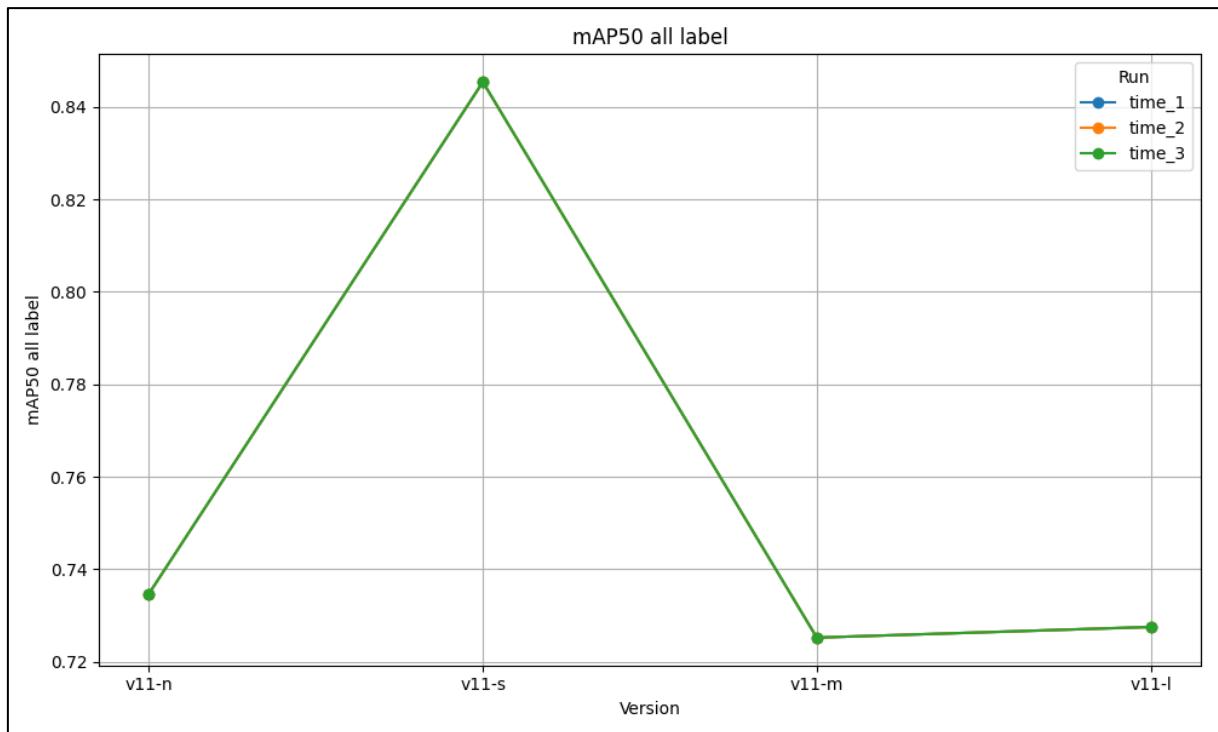
- ❖ Chỉ số FPS và Latency được đánh giá với 1000 khung hình / giây.
- ❖ Các giá trị được in đậm thể hiện sự vượt trội của biến thể đó.

Phụ lục 2: Một số hình ảnh trực quan đánh giá của mô hình sau 3 lần huấn luyện



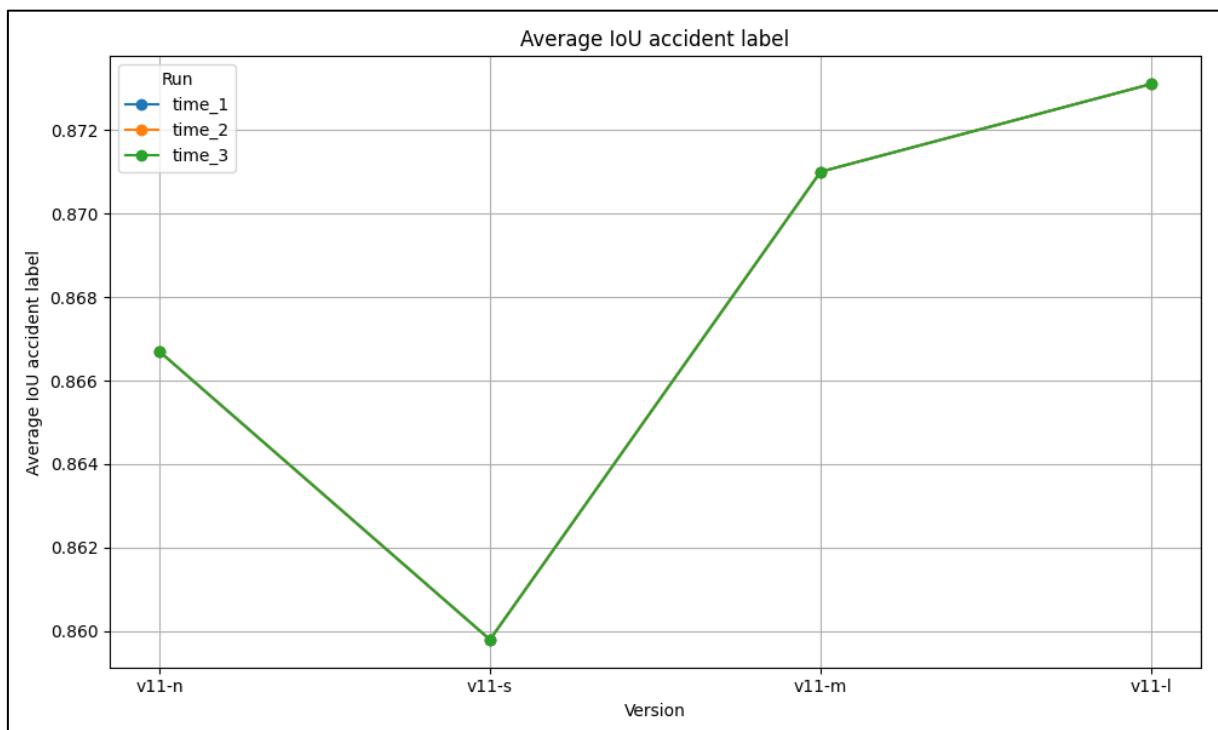
Hình 0.1. Kết quả mAP50 nhãn “Accident” các biến thể YOLOv11 3 lần huấn luyện

Cả 3 lần chạy thì chỉ số mAP50 với nhãn “Accident” đều không thay đổi



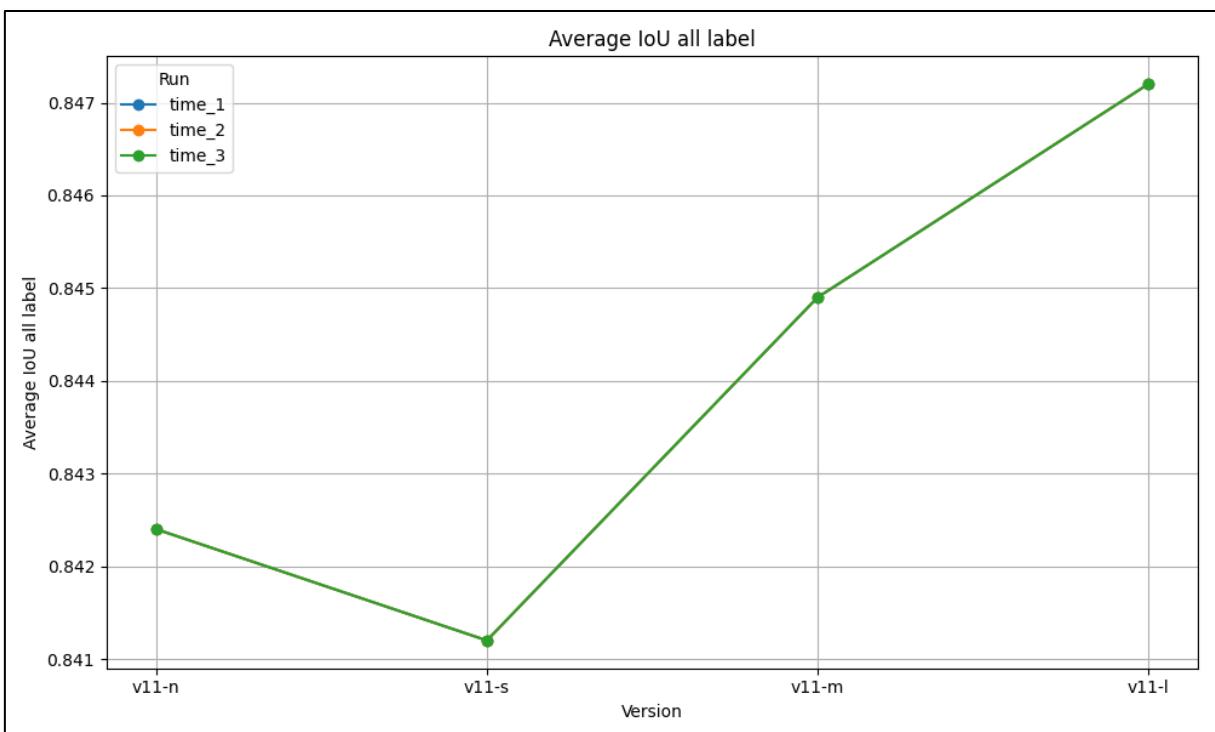
Hình 0.2. Kết quả mAP50 tất cả nhãn của các biến thể YOLOv11 3 lần huấn luyện

Cả 3 lần chạy thì chỉ số mAP50 với tất cả nhãn đều không thay đổi



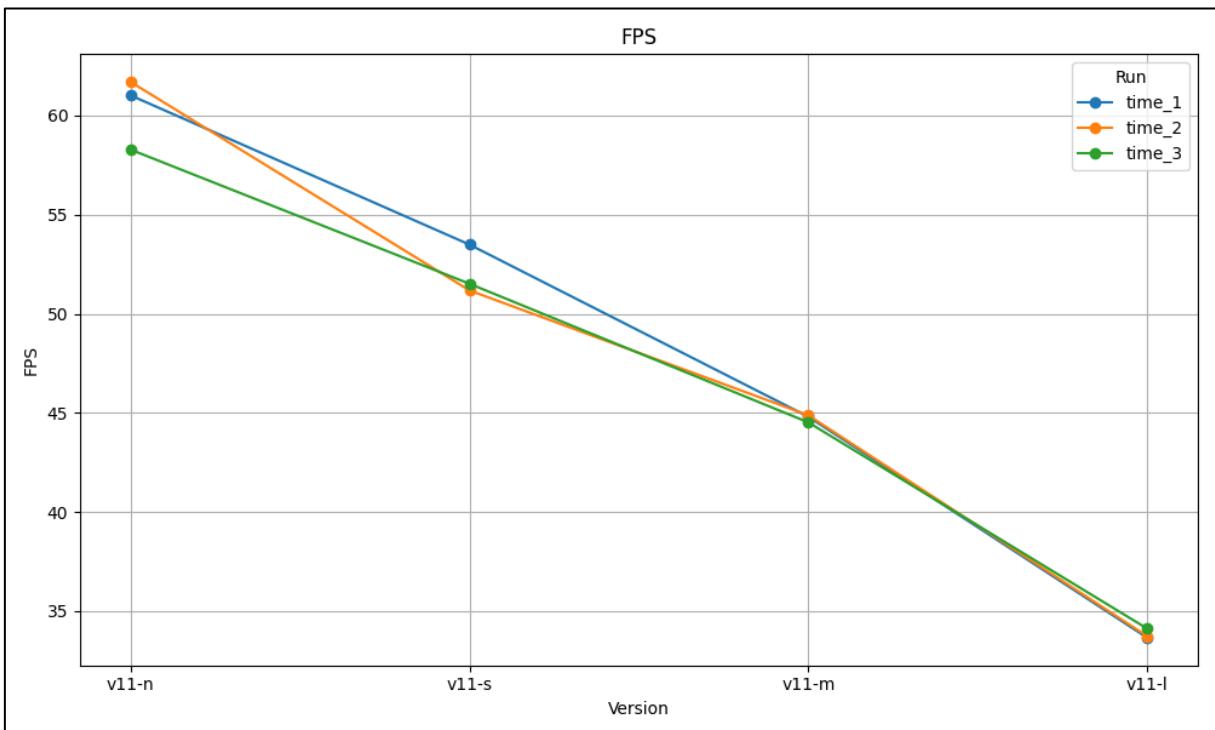
Hình 0.3. Kết quả Average IoU nhãn “Accident” các biến thể YOLOv11 3 lần huấn luyện

Cả 3 lần chạy thì chỉ số Average IoU với nhãn “Accident” đều không thay đổi



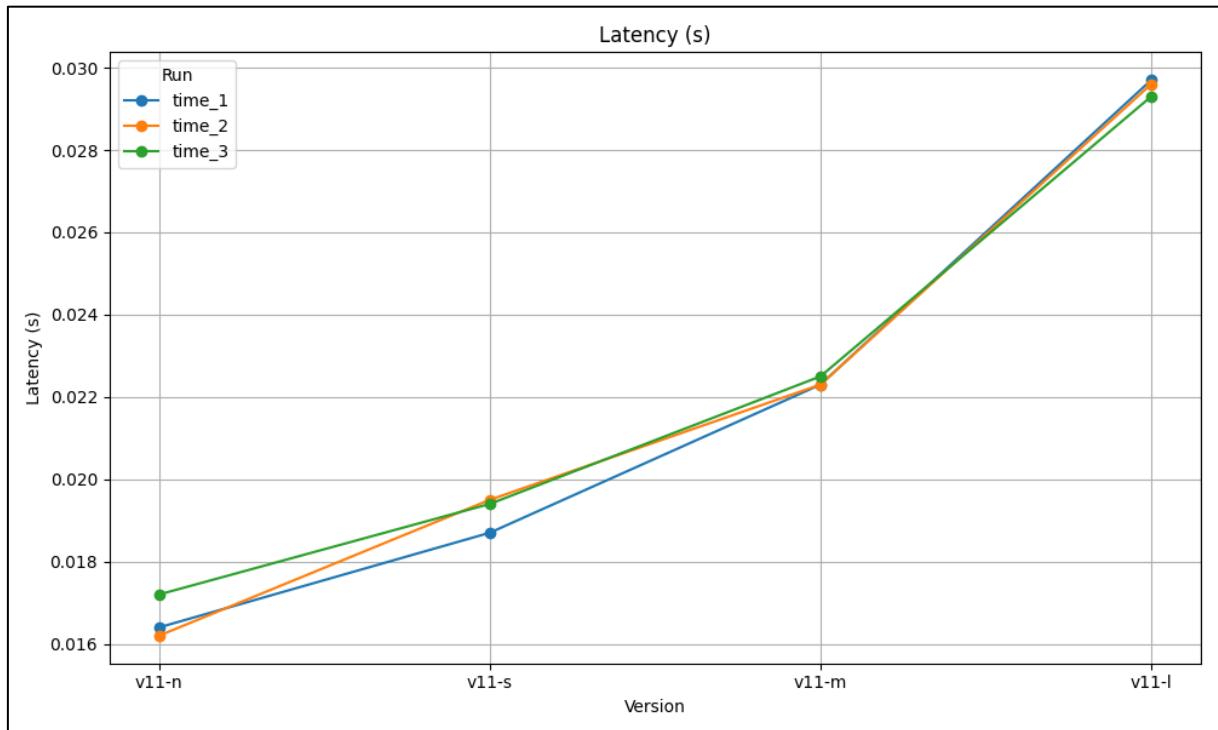
Hình 0.4. Kết quả Average IoU tất cả nhãn của các biến thể YOLOv11 3 lần huấn luyện

Cả 3 lần chạy thì chỉ số Average IoU với tất cả nhãn đều không thay đổi



Hình 0.5. Kết quả FPS các biến thể YOLOv11 3 lần huấn luyện

Sau 3 lần chạy thì chỉ số FPS có thay đổi nhẹ nhưng không quá quan trọng bởi vì tài nguyên hệ thống cấp phát cho mô hình mỗi lần chạy có thể khác nhau.



Hình 0.6. Kết quả Latency các biến thể YOLOv11 3 lần huấn luyện

Sau 3 lần chạy thì chỉ số Latency có thay đổi nhẹ nhưng không quá quan trọng bởi vì tài nguyên hệ thống cấp phát cho mô hình mỗi lần chạy có thể khác nhau.

Phụ lục 3: Tập dữ liệu và mã nguồn dự án

Tập dữ liệu: https://app.roboflow.com/convertyolo2voc/convert_yolo_2_voc/1

Mã nguồn hệ thống: <https://github.com/K1ethoang/Surveillance-Camera-System>

Mã nguồn huấn luyện & đánh giá mô hình:
https://github.com/K1ethoang/Accident_Detect

Phụ lục 4: Hạ tầng triển khai trên Google Cloud Platform

Trong quá trình phát triển hệ thống, em không có đủ tài nguyên để có thể chạy các service của hệ thống. Vì vậy, em đã tận dụng sức mạnh của Google Cloud Platform (GCP) – một nền tảng dịch vụ đám mây mạnh mẽ từ Google.

Cụ thể, em đã sử dụng dịch vụ Google Compute Engine để xây dựng môi trường máy ảo. Trên môi trường này, để đảm bảo các chức năng của hệ thống giám sát hoạt động thông suốt, em đã triển khai và cấu hình nhiều dịch vụ như: Kafka, MySQL, MongoDB, RTMP Server

Việc sử dụng GCP không chỉ giúp em có được môi trường phát triển linh hoạt mà còn cung cấp cái nhìn thực tế về việc triển khai các hệ thống quy mô lớn.

Lần đầu tiên sử dụng thì Google cho chúng ta 300\$ credit và 90 ngày dùng thử miễn phí. Thông tin chi tiết và đăng ký trải nghiệm thử các dịch vụ đám mây của Google: <https://cloud.google.com/free>

TÀI LIỆU THAM KHẢO

- [1] Mandal, R., Mandal, A., Dutta, S., Yusuf Alam, M., Saha, S., Nandi, S., “Framework of Intelligent Transportation System: A Survey,” in *Proc. Int. Conf. Frontiers in Computing and Systems*, 2022.
- [2] Zhao, X., Wang, L., Zhang, Y., “A review of convolutional neural networks in computer vision,” *Artificial Intelligence Review*, tập 57, 2024.
- [3] Liu, L.; Ouyang, W.; Wang, X.; Fieguth, P.; Chen, J.; Liu, X.; Pietikäinen, M., “Deep Learning for Generic Object Detection: A Survey,” *International Journal of Computer Vision*, tập 128, p. 261–318, 2020.
- [4] M. L. Ali, Z. Zhang, “The YOLO Framework: A Comprehensive Review of Evolution, Applications, and Benchmarks in Object Detection,” *Computers*, tập 13, 2024.
- [5] G. Jocher and J. Qiu, “Ultralytics YOLOv11, version 11.0.0,” 2024.
- [6] B. Burns, “What Is Parallel Programming?,” [Trực tuyến]. Available: <https://totalview.io/blog/what-is-parallel-programming>. [Đã truy cập 05 06 2025].
- [7] Ray, “Ray documentation,” [Trực tuyến]. Available: <https://docs.ray.io/en/latest/ray-overview/index.html>. [Đã truy cập 01 06 2025].
- [8] R. Team, “Google doc,” [Trực tuyến]. Available: https://docs.google.com/document/d/1tBw9A4j62ruI5omIJbMxly-la5w4q_TjyJgJL_jN2fI. [Đã truy cập 03 06 2025].
- [9] quest, “What is MySQL? Everything explained,” [Trực tuyến]. Available: <https://www.quest.com/learn/what-is-my-sql.aspx>. [Đã truy cập 30 04 2025].
- [10] E. Krings, “What is RTMPS and Why is it Important to Secure Streaming,” [Trực tuyến]. Available: <https://www.dacast.com/blog/rtmps-streaming/>. [Đã truy cập 05 05 2025].

- [11] M. Carroll, “Real-Time Messaging Protocol (RTMP) Architecture,” [Trực tuyến]. Available: <https://www.pubnub.com/blog/real-time-messaging-protocol-architecture/>. [Đã truy cập 09 05 2025].
- [12] DoVuMinhOanh, “Tổng quan về Apache Kafka - Hệ thống xử lý dữ liệu thời gian thực phân tán,” [Trực tuyến]. Available: <https://viblo.asia/p/tong-quan-ve-apache-kafka-he-thong-xu-ly-du-lieu-thoi-gian-thuc-phan-tan-5OXLA5XkLGr>. [Đã truy cập 07 05 2025].
- [13] K. Baskaran, “Essential Features of Django That Make It a Developer’s Favorite,” [Trực tuyến]. Available: https://medium.com/@kanithkar_baskaran/essential-features-of-django-that-make-it-a-developers-favorite-6c916692c890. [Đã truy cập 23 05 2025].
- [14] geeksforgeeks, “MongoDB - Working and Features,” [Trực tuyến]. Available: <https://www.geeksforgeeks.org/what-is-mongodb-working-and-features/>. [Đã truy cập 13 04 2025].