



APPLICATION OF THE YOLOv26 DEEP LEARNING NETWORK FOR AMBULANCE DETECTION AT TRAFFIC INTERSECTIONS USING RASPBERRY PI 5

Huynh Ngoc Thuan, Pham Nhu Pham^{*}

Gia Lai College, Tran Nhat Duat Street, Dien Hong, Gialai, Vietnam

ARTICLE INFO

TYPE: Research Article

Received: 17/06/2026

Revised: 09/08/2026

Accepted: 03/09/2026

Published online: 15/10/2026

<https://doi.org/10.47869/tcsj.77.8.6>

^{*} *Corresponding author*

Email: phamnhupham@gmail.com

Abstract. Ambulances are priority vehicles tasked with transporting and providing emergency care to patients; they are authorized to run red lights while on duty. However, due to the high speeds at which they often travel, their siren signals are sometimes insufficient for other vehicles to detect them and yield the right-of-way in time, leading to delays or even unfortunate traffic accidents. In this study, the authors propose using the YOLOv26 algorithm to develop an ambulance detection model for use at traffic intersections. The model utilizes YOLOv26n, trained on a dataset of 1,936 ambulance images, and achieves promising metrics: Precision = 0.96, Recall = 0.818, mAP@50 = 0.906, and mAP@50-95 = 0.694. The system was experimentally deployed on a Raspberry Pi 5 device to detect ambulances in video footage displayed on a computer monitor. Experimental results demonstrate that the model offers relatively high detection accuracy with a prediction rate of up to 89% and an average processing speed exceeding 10 FPS. Although the research is currently at the experimental stage, the results indicate its feasibility and potential for application within intelligent transportation systems.

Keywords: Deep learning, smart traffic, model training, Raspberry Pi 5, ambulance, YOLOv26.



ỨNG DỤNG MẠNG HỌC SÂU YOLOv26, ĐỀ XUẤT MÔ HÌNH NHẬN DIỆN XE CỨU THƯƠNG TẠI CÁC NÚT GIAO THÔNG TRÊN THIẾT BỊ RASPBERRY PI 5

Huỳnh Ngọc Thuận, Phạm Như Phẩm*

Trường Cao đẳng Gia Lai, đường Trần Nhật Duật, phường Diên Hồng, Gia Lai, Việt Nam

THÔNG TIN BÀI BÁO

CHUYÊN MỤC: Công trình khoa học

Ngày nhận bài: 17/06/2026

Ngày nhận bài sửa: 09/08/2026

Ngày chấp nhận đăng: 03/09/2026

Ngày xuất bản Online: 15/10/2026

<https://doi.org/10.47869/tcsj.77.8.6>

* Tác giả liên hệ

Email: phamnhupham@gmail.com

Tóm tắt. Xe cứu thương là phương tiện giao thông ưu tiên làm nhiệm vụ vận chuyển và cấp cứu người bệnh trong các tình huống khẩn cấp, được phép vượt đèn đỏ khi thực hiện nhiệm vụ. Tuy nhiên, do thường di chuyển với tốc độ cao, tín hiệu còi ưu tiên không đủ lớn để các phương tiện nhận biết, nhường đường kịp thời, dẫn đến chậm trễ thời gian hoặc thậm chí xảy ra nhiều vụ tai nạn giao thông đáng tiếc. Trong bài nghiên cứu này, nhóm tác giả đề xuất ứng dụng thuật toán YOLOv26 để xây dựng mô hình nhận diện xe cứu thương tại các nút giao thông. Mô hình sử dụng YOLOv26n để huấn luyện tập dữ liệu gồm 1.936 hình ảnh xe cứu thương và đạt được các chỉ số khả quan như Precision = 0,96, Recall = 0,818, mAP@50 = 0,906 và mAP@50-95 = 0,694. Hệ thống được triển khai thử nghiệm trên thiết bị Raspberry Pi 5 và thực hiện nhận diện xe cứu thương qua các đoạn video từ màn hình máy tính. Kết quả thực nghiệm cho thấy mô hình có khả năng nhận diện tương đối chính xác với độ dự đoán lên đến 89%, tốc độ xử lý trung bình đạt trên 10 FPS. Mặc dù nghiên cứu mới dừng ở mức thử nghiệm, nhưng kết quả cho thấy tính khả thi, có khả năng phát triển ứng dụng trong các hệ thống giao thông thông minh.

Từ khóa: học sâu, giao thông thông minh, huấn luyện mô hình, Raspberry Pi 5, xe cứu thương, YOLOv26.

1. GIỚI THIỆU

Sự phát triển mạnh mẽ của thị giác máy tính và trí tuệ nhân tạo (Artificial Intelligence - AI) đang tạo ra những thay đổi đáng kể trong nhiều lĩnh vực của đời sống và sản xuất. Với khả năng học tập từ dữ liệu, phân tích thông tin và đưa ra quyết định tự động, AI ngày càng được ứng dụng rộng rãi nhằm hỗ trợ con người giải quyết các bài toán từ đơn giản đến phức tạp. Trong lĩnh vực giao thông, AI đã trở thành một thành phần quan trọng của các hệ thống giao thông thông minh [1-3] và là chủ đề nhận được nhiều sự quan tâm từ các nhà nghiên cứu. Thông qua việc xử lý dữ liệu theo thời gian thực, các hệ thống AI có thể giám sát lưu lượng phương tiện, phát hiện tình trạng ùn tắc, nhận biết các sự cố giao thông bất thường [4, 5] cũng như hỗ trợ phát hiện các hành vi vi phạm như vượt đèn đỏ, chạy quá tốc độ, không đội mũ bảo hiểm, lấn làn hoặc đi sai phần đường [6, 7], góp phần nâng cao hiệu quả quản lý giao thông, đồng thời giảm khối lượng công việc cho lực lượng chức năng.

Ngoài vai trò giám sát và quản lý, AI còn được nghiên cứu để hỗ trợ trong việc điều khiển giao thông theo hướng tự động thông minh. Dựa trên hình ảnh thu thập từ các camera giám sát, hệ thống có thể đánh giá mật độ phương tiện tại các nút giao thông để điều chỉnh thời gian hoạt động của đèn tín hiệu, góp phần giảm ùn tắc và nâng cao hiệu quả giao thông [8]. Đối với các phương tiện ưu tiên như xe cứu thương, AI có thể nhận diện phương tiện và tự động kích hoạt chế độ ưu tiên đèn xanh [9, 10], để cho xe di chuyển nhanh chóng và an toàn qua các nút giao, rút ngắn thời gian đến cơ sở y tế.

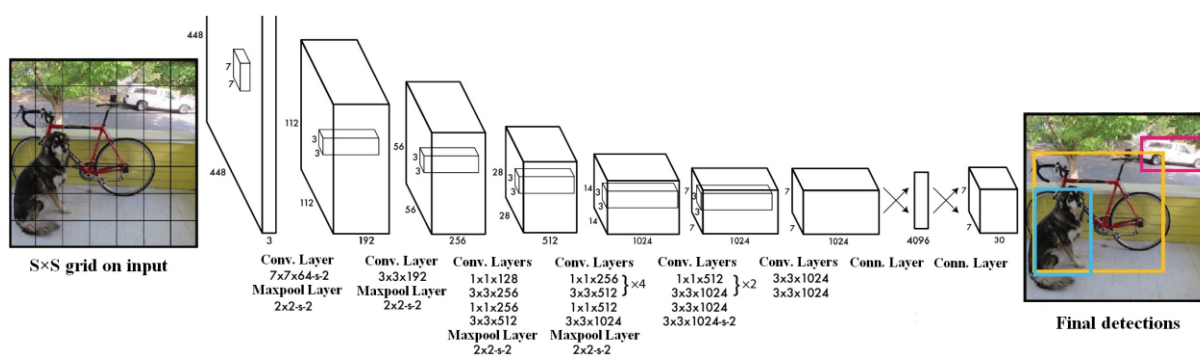
Để thực hiện bài toán phát hiện và nhận diện đối tượng giao thông qua hình ảnh, nhiều mô hình học sâu dựa trên mạng nơ-ron tích chập (CNN) đã được đề xuất như Faster R-CNN, SSD, CenterNet, DETR và YOLO [11, 12]. Mỗi mô hình có những đặc điểm riêng về tốc độ và độ chính xác, tuy nhiên YOLO hiện được đánh giá là một trong những giải pháp nổi bật nhờ khả năng phát hiện đối tượng nhanh, chính xác và phù hợp với các ứng dụng thời gian thực [13]. Chính vì vậy, YOLO đã được nhiều nhà nghiên cứu lựa chọn để giải quyết bài toán nhận diện xe cứu thương. Điển hình như, nghiên cứu của John Paul và cộng sự [14] sử dụng mô hình YOLOv5s để nhận dạng và phân loại xe cứu thương trên bộ dữ liệu gồm 1.840 hình ảnh thu thập tại Philippines. Kết quả cho thấy mô hình đạt giá trị mAP@50 lên tới 93% sau khi áp dụng các kỹ thuật tăng cường dữ liệu và tối ưu siêu tham số huấn luyện. Trong một nghiên cứu khác, Alessandro Mecocci và cộng sự [15] triển khai mô hình YOLOv8 trên nền tảng Jetson Nano để nhận diện xe cứu thương và điều khiển ưu tiên tín hiệu giao thông. Hệ thống đạt tốc độ xử lý khoảng 11,8 FPS với thời gian phản hồi xấp xỉ 0,25 giây, đồng thời vẫn duy trì hiệu quả hoạt động trong điều kiện ánh sáng yếu hoặc thời tiết không thuận lợi. Nghiên cứu [16] đề xuất một hệ thống phát hiện xe cứu thương thời gian thực trên thiết bị biên sử dụng mô hình YOLOv5s kết hợp với nền tảng ONNX Runtime. Mô hình được triển khai trên Raspberry Pi 64-bit, đạt độ chính xác mAP@0,5 là 91,3% với tốc độ suy luận từ 2 đến 5 FPS.

Từ việc khảo sát các công trình liên quan, có thể thấy mô hình YOLO là một giải pháp hiệu quả cho bài toán phát hiện và nhận diện xe cứu thương. Kế thừa những kết quả nghiên cứu trước đây, bài báo này tập trung nghiên cứu và ứng dụng mô hình YOLOv26 để xây dựng hệ thống nhận diện xe cứu thương trên thiết bị Raspberry Pi 5. Mô hình được huấn luyện trên bộ dữ liệu riêng, sau đó triển khai thử nghiệm, đánh giá khả năng nhận diện, tốc độ xử lý và tính khả thi của hệ thống.

2. ĐỐI TƯỢNG VÀ PHƯƠNG PHÁP

2.1. Sơ lược mô hình học sâu YOLO

Mô hình YOLO là một trong những thuật toán phát hiện đối tượng tiêu biểu trong lĩnh vực thị giác máy tính và trí tuệ nhân tạo. Mô hình được xây dựng dựa trên mạng nơ-ron tích chập sâu, là sự kết hợp các lớp tích chập và các lớp kết nối đầy đủ để thực hiện đồng thời nhiệm vụ xác định vị trí và phân loại đối tượng trong ảnh (Hình 1). YOLO được giới thiệu lần đầu bởi Joseph Redmon và cộng sự vào năm 2016 [17], YOLO đã tạo nên bước đột phá quan trọng trong bài toán phát hiện đối tượng nhờ khả năng xử lý nhanh và độ chính xác cao. Khác với các phương pháp truyền thống như R-CNN, Fast R-CNN hay Faster R-CNN phải thực hiện nhiều giai đoạn riêng biệt để xác định vị trí và phân loại đối tượng, còn YOLO tiếp cận theo cơ chế một giai đoạn. Theo đó, mô hình chỉ cần thực hiện một lần suy luận trên toàn bộ ảnh đầu vào thì có thể xác định vị trí và dự đoán được các đối tượng trong ảnh.



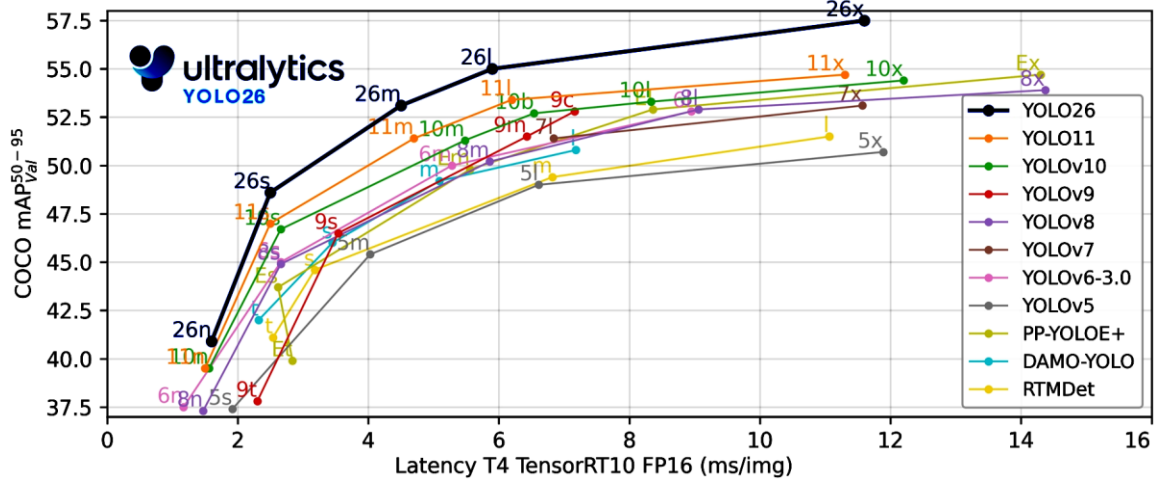
Hình 1. Kiến trúc của mạng YOLO [17, 18].

Nguyên lý hoạt động của YOLO là chia ảnh đầu vào thành một lưới gồm nhiều ô có kích thước ($S \times S$). Mỗi ô lưới chịu trách nhiệm dự đoán các khung giới hạn, độ tin cậy của dự đoán và xác suất thuộc về từng lớp đối tượng. Các kết quả dự đoán sau đó được xử lý bằng thuật toán Non-Maximum Suppression (NMS) để loại bỏ các khung chồng lấp và giữ lại những khung dự đoán có độ tin cậy cao nhất [17,18].

2.2. Mô hình YOLOv26

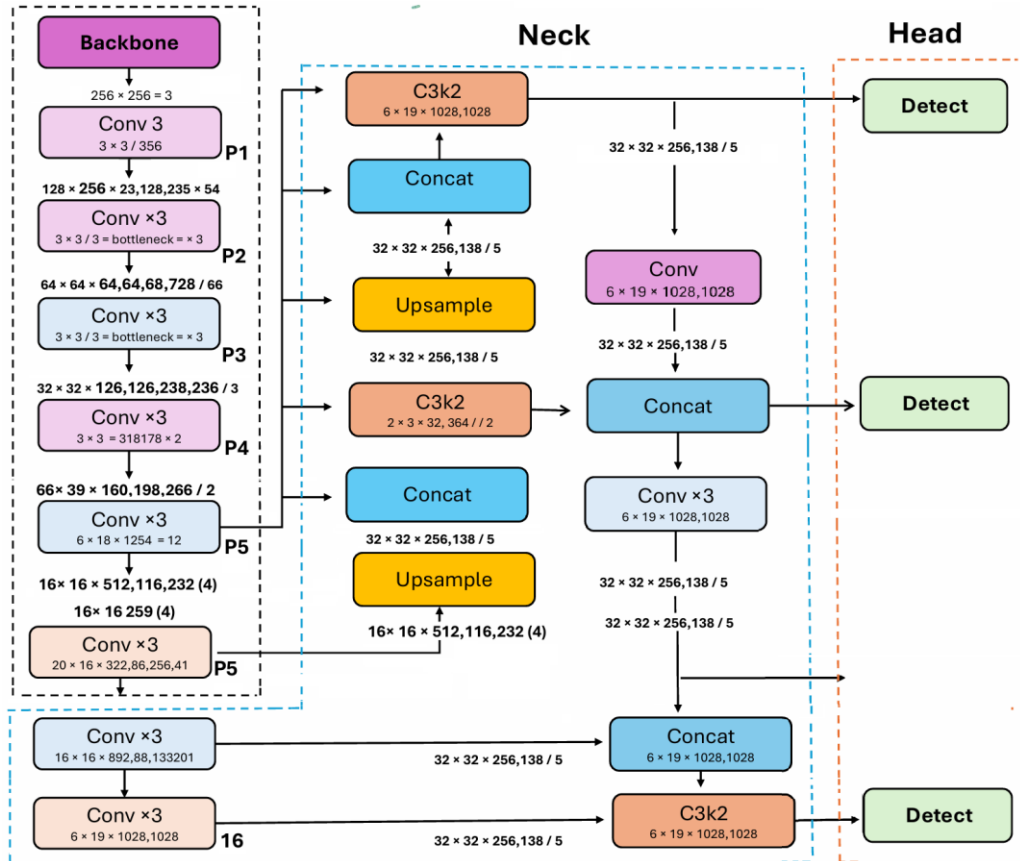
YOLOv26 là thế hệ mới của dòng mô hình phát hiện đối tượng YOLO do Ultralytics công bố năm 2025, được phát triển với mục tiêu nâng cao hiệu năng xử lý thời gian thực trên các nền tảng AI biên (Edge AI) và hệ thống nhúng. Mô hình kế thừa những ưu điểm nổi bật của các phiên bản YOLO trước đây, đồng thời cải tiến về kiến trúc nhằm tăng độ chính xác nhận diện và giảm độ trễ suy luận. Kết quả đánh giá trên bộ dữ liệu COCO (Hình 2) cho thấy các phiên bản YOLOv26 đều đạt hiệu năng vượt trội so với nhiều thế hệ YOLO trước. Đặc biệt, phiên bản YOLOv26x đạt khoảng 57,5% mAP@50-95, cao hơn YOLOv11x, YOLOv10x và YOLOv8x trong cùng điều kiện độ trễ, cho thấy tiềm năng ứng dụng hiệu quả trong các hệ thống phát hiện đối tượng thời gian thực [19, 20].

YOLOv26 được xây dựng trên kiến trúc gồm ba thành phần chính là Backbone, Neck và Head. Kiến trúc tổng thể của mô hình được trình bày trong Hình 3. Trong đó, Backbone có nhiệm vụ trích xuất đặc trưng từ ảnh đầu vào thông qua các lớp tích chập sâu; Neck thực hiện hợp nhất đặc trưng đa tỷ lệ nhằm tăng khả năng phát hiện đối tượng có kích thước khác nhau; còn Head đảm nhận việc dự đoán vị trí đối tượng, phân loại lớp và tính toán độ tin cậy của từng dự đoán. Toàn bộ quá trình nhận diện được thực hiện theo cơ chế phát hiện một giai đoạn (One-Stage Detector), cho phép mô hình đạt tốc độ suy luận cao trong khi vẫn duy trì độ chính xác tốt [19].



Hình 2. Biểu đồ so sánh hiệu năng của YOLOv26 với các phiên bản YOLO trước [19, 20].

Bên cạnh việc tối ưu kiến trúc tổng thể, YOLOv26 còn được bổ sung nhiều cải tiến quan trọng nhằm nâng cao hiệu quả triển khai thực tế. Một trong những thay đổi nổi bật là cơ chế End-to-End NMS-Free Detection, cho phép mô hình tạo trực tiếp kết quả phát hiện cuối cùng mà không cần sử dụng thuật toán Non-Maximum Suppression (NMS) ở giai đoạn hậu xử lý, nhờ đó, độ trễ suy luận được giảm đáng kể [19, 20].



Hình 3. Sơ đồ kiến trúc cốt lõi của mô hình YOLOv26 [19, 20].

Ngoài ra, YOLOv26 loại bỏ thành phần Distribution Focal Loss (DFL) trong quá trình huấn luyện, góp phần giảm chi phí tính toán và tăng khả năng tương thích khi

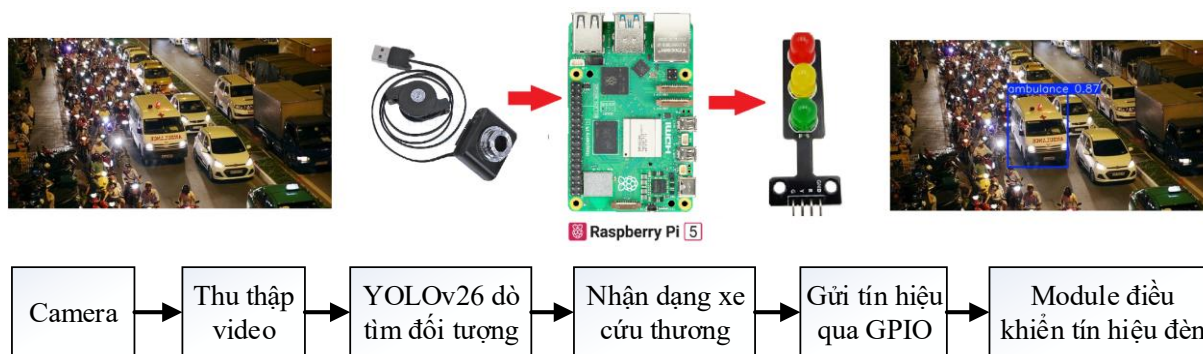
chuyển đổi mô hình sang các nền tảng triển khai khác nhau như ONNX, TensorRT và TFLite. Mô hình cũng tích hợp nhiều kỹ thuật tối ưu mới như ProgLoss (Progressive Loss Balancing) giúp cân bằng hàm mất mát trong quá trình huấn luyện, STAL (Small-Target-Aware Label Assignment) hỗ trợ nâng cao hiệu quả phát hiện các đối tượng kích thước nhỏ, cùng với MuSGD Optimizer nhằm cải thiện tốc độ hội tụ và độ ổn định của mô hình [19, 20].

2.3. Sơ lược về máy tính Raspberry Pi

Raspberry Pi là dòng máy tính nhúng kích thước nhỏ được phát triển bởi Raspberry Pi Foundation nhằm phục vụ các hoạt động nghiên cứu, giáo dục, tự động hóa và phát triển các hệ thống nhúng. Mặc dù có kích thước nhỏ gọn, Raspberry Pi được tích hợp đầy đủ các thành phần của một máy tính hoàn chỉnh như bộ vi xử lý trung tâm, bộ nhớ RAM, các cổng giao tiếp USB, HDMI, Ethernet, kết nối không dây Wi-Fi, Bluetooth và hệ thống chân GPIO. Nhờ chi phí thấp, khả năng mở rộng linh hoạt và mức tiêu thụ điện năng thấp, Raspberry Pi đã trở thành một trong những nền tảng phần cứng phổ biến trong các ứng dụng IoT, robot và hệ thống điều khiển thông minh [21]. Phiên bản Raspberry Pi 5 RAM 8 GB, trang bị bộ vi xử lý Broadcom BCM2712 gồm bốn nhân ARM Cortex-A76 64-bit hoạt động ở xung nhịp tối đa 2,4 GHz. Thiết bị được tích hợp bộ xử lý đồ họa VideoCore VII, hỗ trợ xuất hình ảnh độ phân giải cao và xử lý đa phương tiện hiệu quả [21, 22].

2.4. Thiết kế mô hình nhận diện và điều khiển

Để xây dựng hệ thống nhận diện xe cứu thương và điều khiển tín hiệu đèn giao thông trên nền tảng Raspberry Pi 5, nhóm tác giả đề xuất mô hình hoạt động gồm các khối chức năng chính (Hình 4) như sau:



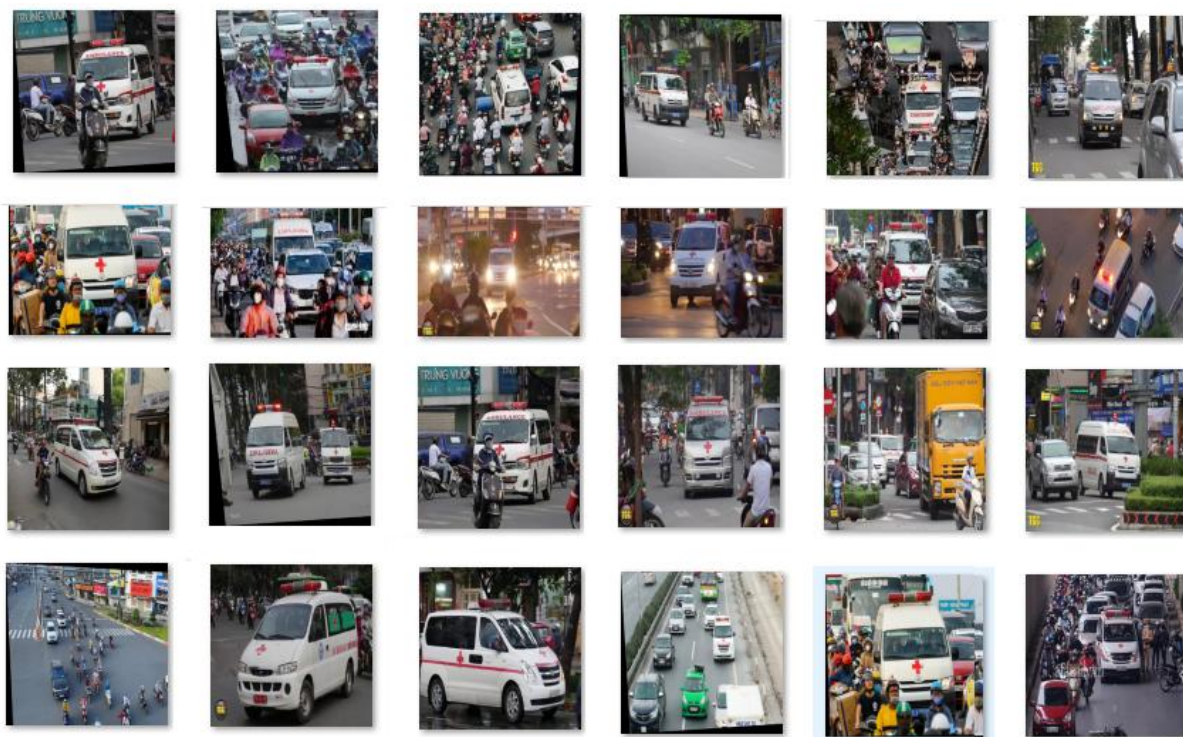
Hình 4. Sơ đồ khối các bước phát hiện, nhận diện và điều khiển đèn tín hiệu giao thông.

- Khối Camera thu nhận hình ảnh: Thực hiện đọc hình ảnh tại nút giao thông, cung cấp dữ liệu đầu vào cho hệ thống.
- Khối YOLOv26 dò tìm và nhận diện: Tiền xử lý ảnh và sử dụng mô hình YOLOv26 để phát hiện các đối tượng xuất hiện trong, xác định xe cứu thương.
- Khối GPIO gửi tín hiệu điều khiển: Raspberry Pi 5 xuất tín hiệu qua chân GPIO để điều khiển module điều khiển bật đèn xanh.

Trong các khối chức năng, khối YOLOv26 đóng vai trò trung tâm, đảm nhận nhiệm vụ phát hiện và nhận diện xe cứu thương. Để xây dựng và triển khai mô hình, nhóm tác giả phải trải qua các bước: (1) thu thập dữ liệu hình ảnh xe cứu thương; (2) tiền xử lý và gán nhãn dữ liệu; (3) huấn luyện mô hình theo thuật toán YOLOv26; (4) đánh giá hiệu năng mô hình qua các chỉ số huấn luyện Precision, Recall, F1-Score, mAP và (5) cuối cùng chạy thực nghiệm trên nền tảng Raspberry Pi 5 và nhận xét.

2.4.1. Xây dựng dữ liệu

Bộ dữ liệu được xây dựng theo cấu trúc chuẩn của mô hình YOLO, gồm ba thành phần chính là thư mục chứa ảnh (images), thư mục chứa nhãn (labels) và tệp cấu hình dữ liệu (data.yaml). Thư mục images chứa các ảnh định dạng JPG với kích thước chuẩn hóa 640×640 pixel. Các ảnh trong thư mục này được chia thành hai tập con gồm train và val, trong đó tập train chiếm 80% phục vụ huấn luyện mô hình, còn tập val chiếm 20% dùng để đánh giá hiệu suất sau huấn luyện.



Hình 5. Ảnh minh họa của bộ dữ liệu phát hiện xe cứu thương.

Trong nghiên cứu này, nhóm tác giả sử dụng bộ dữ liệu gồm 1.936 hình ảnh xe cứu thương tại Việt Nam (Hình 5), được chia thành 1.556 ảnh cho tập huấn luyện (80%) và 380 ảnh cho tập đánh giá (20%). Dữ liệu này được nhóm tác giả thu thập từ nhiều nguồn khác nhau, một phần từ nền tảng Roboflow Universe [23], một phần hình ảnh được trích xuất từ các video, các website và các ảnh do nhóm tác giả tự thu thập. Số lượng hình ảnh trong bộ dữ liệu tương đối đa dạng, có khả năng phản ánh đầy đủ các điều kiện trong thực tế, như nhiều góc quan sát, nhiều khoảng cách thu nhận, điều kiện chiếu sáng và bối cảnh giao thông khác nhau. Điều này giúp mô hình học được các đặc trưng của xe cứu thương trong nhiều ngữ cảnh, nâng cao khả năng tổng quát hóa khi triển khai thực tế.

Thư mục labels cũng tương tự như images, chứa thư mục train và val, lưu trữ các tệp nhãn định dạng .txt, có tên tương ứng với từng ảnh trong thư mục images. Mỗi tệp chứa thông tin về đối tượng trong ảnh theo định dạng YOLO gồm năm tham số: class_id, x_center, y_center, width và height. Các thông tin này được tạo ra thông qua quá trình gán nhãn bằng phần mềm MakeSense. Tệp data.yaml được đặt tại thư mục gốc của bộ dữ liệu, đóng vai trò là tệp cấu hình trung tâm. Tệp này dùng để khai báo đường dẫn đến các thư mục train và val, xác định số lượng lớp đối tượng, tên lớp và các tham số cấu hình quá trình huấn luyện.

2.4.2. Huấn luyện mô hình

Huấn luyện mô hình YOLOv26 là quá trình giúp mạng học sâu học được các đặc trưng của đối tượng trong ảnh nhằm thực hiện phát hiện và nhận diện chính xác xe cứu thương trong các tình huống giao thông thực tế. Quá trình huấn luyện được thực hiện dựa trên phương pháp chuyển giao học (transfer learning), sử dụng trọng số khởi tạo từ mô hình đã được huấn luyện trước (pretrained) để rút ngắn thời gian hội tụ và nâng cao hiệu quả nhận diện. Để đảm bảo hiệu suất huấn luyện và giảm thời gian xử lý, hệ thống yêu cầu môi trường tính toán lớn, đặc biệt là bộ xử lý đồ họa (GPU). Vì vậy, nhóm nghiên cứu lựa chọn nền tảng Google Colab để thực hiện huấn luyện các mô hình với các tham số huấn luyện chính được trình bày ở Bảng 1.

Bảng 1. Các tham số huấn luyện mô hình YOLOv26.

Tham số	Giá trị	Tham số	Giá trị	Tham số	Giá trị
Epochs	100	Optimizer	SGD	box	7,5
Imgsz	640	Learning rate	0,01	cls	0,5
Batch size	16	Momentum	0,937	dfl	1,5

Việc lựa chọn các tham số ở Bảng 1 dựa trên cơ sở sự cân bằng giữa khả năng hội tụ, độ chính xác và tài nguyên tính toán. Cụ thể, số epoch (100) đảm bảo mô hình học đủ sâu nhưng tránh overfitting, kích thước ảnh đầu vào (640) tối ưu hóa giữa độ chính xác phát hiện và tốc độ xử lý. Batch size (16) cùng với optimizer SGD và learning rate 0,01 giúp quá trình huấn luyện ổn định và tăng khả năng tổng quát hóa. Tham số momentum (0,937) cải thiện tốc độ hội tụ và giảm dao động gradient. Ngoài ra, các hệ số loss như box (7,5), cls (0,5) và dfl (1,5) được điều chỉnh nhằm ưu tiên độ chính xác định vị và cải thiện dự đoán bounding box.

Để đánh giá hiệu quả của mô hình phát hiện đối tượng YOLO, nghiên cứu sử dụng bốn chỉ số chính gồm mAP (Mean Average Precision), Precision, Recall và F1-Score, được thể hiện ở các công thức (1), (2), (3) và (4) [15]. Các chỉ số này phản ánh khả năng phát hiện chính xác, mức độ bao phủ đối tượng và sự cân bằng giữa độ chính xác với độ nhạy của mô hình.

$$mAP = \frac{1}{N} \sum_{i=1}^N AP_i \quad (1)$$

$$precision = \frac{TP}{TP + FP} \quad (2)$$

$$recall = \frac{TP}{TP + FN} \quad (3)$$

$$F_1 = \frac{2 \times precision \times recall}{precision + recall} \quad (4)$$

2.4.3. Kết quả huấn luyện mô hình

Sau khi hoàn thành quá trình huấn luyện, mô hình YOLOv26 sinh ra hai tệp trọng số chính là “best.pt” và “last.pt”, chứa toàn bộ các tham số đã học của mạng, bao gồm kiến trúc mô hình, trọng số các lớp tích chập, các siêu tham số tối ưu hóa và trạng thái của mô hình tại các giai đoạn huấn luyện. Trong đó, tệp “best.pt” lưu trữ trạng thái có hiệu năng tốt nhất trên

tập kiểm định, được xác định dựa trên giá trị mAP cao nhất và được sử dụng để triển khai thực nghiệm. Tập “last.pt” lưu trạng thái của mô hình tại epoch cuối cùng, thường được sử dụng cho mục đích tiếp tục tinh chỉnh hoặc huấn luyện bổ sung khi cần thiết. Bên cạnh các tệp trọng số, hệ thống còn cung cấp các chỉ số đánh giá quan trọng như Precision, Recall, mAP@50 và mAP@50-95 để phục vụ việc đánh giá, phân tích mô hình.

Để đánh giá, so sánh hiệu quả nhận diện giữa các phiên bản của mô hình YOLOv26, nhóm tác giả thực hiện huấn luyện trong 100 epochs cho từng phiên bản với điều kiện đầu vào giống nhau. Kết quả các chỉ số đánh giá chính của quá trình huấn luyện đạt được thể hiện trong Bảng 2.

Bảng 2. Kết quả các chỉ số huấn luyện của các phiên bản YOLOv26.

TT	Các phiên bản YOLOv26	Precision	Recall	mAP@50	mAP@50-95	F1-Score	Parameters	GFLOPs
1	YOLOv26n	0,96	0,818	0,906	0,694	0,883	2.375.031	5,2
2	YOLOv26s	0,951	0,828	0,891	0,696	0,885	9.465.567	20,5
3	YOLOv26m	0,943	0,834	0,899	0,693	0,885	20.350.223	67,8
4	YOLOv26l	0,964	0,84	0,911	0,703	0,898	24.746.511	86,1

Qua số liệu ở Bảng 2 cho thấy tất cả các phiên bản của YOLOv26 đều cho hiệu năng khả quan đối với bài toán phát hiện xe cứu thương, với chỉ số F1-Score dao động từ 0,883 đến 0,898, phản ánh sự cân bằng tốt giữa độ chính xác và khả năng phát hiện đầy đủ đối tượng. Trong số đó, YOLOv26l thể hiện ưu thế nổi bật với các chỉ số Precision, Recall, F1-Score, mAP@50 và mAP@50-95 lần lượt đạt 0,964, 0,840, 0,898, 0,911 và 0,703. Điều này cho thấy mô hình có khả năng nhận diện chính xác và ổn định hơn so với các cấu hình còn lại. Tuy nhiên, kiến trúc này sở hữu số lượng tham số lớn (24.746.511) cùng độ phức tạp tính toán cao (86,1 GFLOPs) kéo theo yêu cầu lớn về tài nguyên phần cứng.

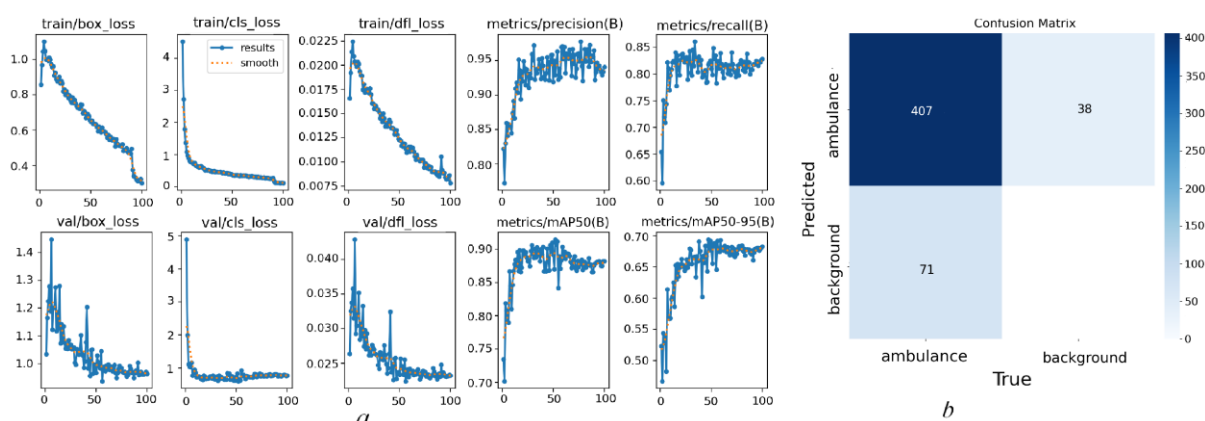
Ngược lại, YOLOv26n là cấu hình gọn nhẹ nhất với 2.375.031 tham số và mức tính toán chỉ 5,2 GFLOPs, phù hợp với các hệ thống yêu cầu tốc độ xử lý cao và tài nguyên hạn chế. Mặc dù các chỉ số mAP và F1-Score thấp hơn so với YOLOv26l, nhưng mô hình vẫn đạt hiệu suất tốt với Precision = 0,960, F1-Score = 0,883 và mAP@50 = 0,906. Hai phiên bản trung gian YOLOv26s và YOLOv26m thể hiện sự cân bằng giữa độ chính xác, F1-Score và độ phức tạp tính toán.

Về khả năng phát hiện xe cứu thương theo ba nhóm kích thước Small (AP_s: diện tích < 1.024 pixel²), Medium (AP_m: 1.024 pixel² < diện tích < 9.216 pixel²) và Large (AP_l: diện tích > 9.216 pixel²) [24], kết quả tại Bảng 3 cho thấy hiệu quả phát hiện có sự khác biệt rõ rệt theo kích thước đối tượng. Đối với đối tượng lớn, các mô hình AP_l đạt từ 0,7150 đến 0,7681, trong đó YOLOv26l đạt cao nhất với 0,7681. Đối với đối tượng trung bình, AP_m dao động từ 0,4148 đến 0,4775 và YOLOv26l vẫn đạt kết quả cao nhất (0,4775). Trong khi đó, khả năng phát hiện đối tượng nhỏ còn hạn chế, với AP_s chỉ từ 0 đến 0,0772 và YOLOv26n đạt giá trị cao nhất (0,0772). Nhìn chung, các phiên bản huấn luyện YOLOv26 cho khả năng phát hiện tốt hơn đối với xe cứu thương có kích thước lớn và trung bình, nhưng hiệu quả giảm đáng kể đối với các đối tượng nhỏ. Điều này cho thấy mô hình còn hạn chế khi xe cứu thương xuất hiện ở khoảng cách xa hoặc chiếm diện tích nhỏ trong khung hình.

Bảng 3. Giá AP (Average Precision) theo kích thước ba nhóm đối tượng Small, Medium, và Large.

TT	Các phiên bản YOLOv26	AP _s	AP _m	AP _l
1	YOLOv26n	0,0772	0,4180	0,7150
2	YOLOv26s	0	0,4148	0,7337
3	YOLOv26m	0,0012	0,4428	0,7559
4	YOLOv26l	0,0125	0,4775	0,7681

Mặc dù đã tiến hành huấn luyện và đánh giá nhiều biến thể của YOLOv26, nhóm tác giả lựa chọn YOLOv26n để triển khai thực nghiệm nhận diện xe cứu thương trên Raspberry Pi 5 nhờ ưu thế về độ phức tạp tính toán thấp, số lượng tham số nhỏ và khả năng đáp ứng yêu cầu xử lý thời gian thực. Các đường cong huấn luyện và ma trận nhầm lẫn (confusion matrix) của mô hình được trình bày trong Hình 6.



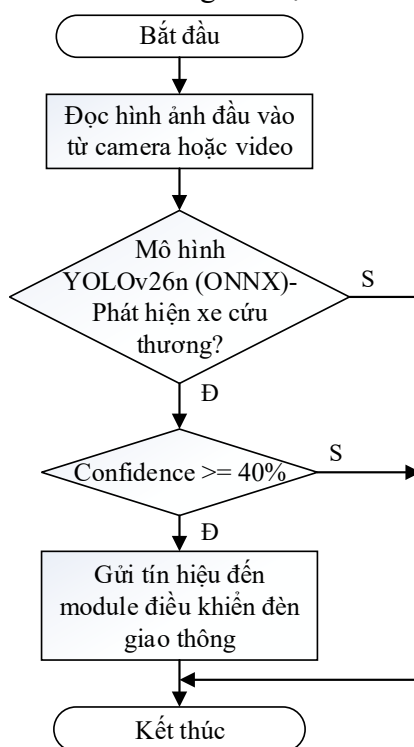
Hình 6. Biểu đồ mô tả hiệu suất huấn luyện (a) và ma trận nhầm lẫn (b) của mô hình.

Hình 6a cho thấy các đường cong hội tụ của mô hình YOLOv26n trong suốt 100 epochs huấn luyện. Các hàm mất mát trên cả tập huấn luyện và kiểm định đều giảm dần và ổn định về cuối quá trình, cho thấy mô hình học tốt đặc trưng dữ liệu và không xuất hiện hiện tượng quá khớp rõ rệt. Các chỉ số Precision, Recall và mAP tăng nhanh trong giai đoạn đầu, sau đó dao động nhẹ và hội tụ ở mức cao. Cụ thể, Precision duy trì quanh mức 0,94–0,96, Recall đạt khoảng 0,81–0,84, trong khi mAP@50 và mAP@50-95 lần lượt ổn định quanh mức 0,89–0,91 và 0,67–0,69. Điều này cho thấy mô hình đạt trạng thái hội tụ ổn định sau khoảng 40–50 epochs và đáp ứng tốt yêu cầu nhận diện xe cứu thương. Ma trận nhầm lẫn ở Hình 6b cho thấy mô hình YOLOv26n đạt TP = 407, FP = 38 và FN = 71. Trong đó, mô hình nhận diện đúng 407 mẫu thuộc lớp “ambulance”, nhận nhầm 38 trường hợp nền (background) thành xe cứu thương và bỏ sót 71 trường hợp xe cứu thương, nhận diện thành nền. Từ các giá trị trên, độ nhạy (Recall) đạt khoảng 85,2%, cho thấy mô hình phát hiện đúng khoảng 85,2% số xe cứu thương trong các mẫu đánh giá, trong đó khoảng 14,8% đối tượng bị bỏ sót. Độ chính xác (Precision) đạt khoảng 91,5%, nghĩa là trong tổng số các trường hợp được mô hình dự đoán là xe cứu thương, thì có khoảng 91,5% là dự đoán đúng và khoảng 8,5% là nhận diện nhầm. Nhìn chung, mô hình có khả năng phát hiện xe cứu thương tương đối tốt, tuy nhiên vẫn còn một số trường hợp bỏ sót hoặc nhận diện nhầm với nền ảnh, đặc biệt khi đối tượng bị che khuất, có kích thước nhỏ, góc quan sát thay đổi hoặc điều kiện ánh sáng không đồng đều.

2.4.4. Lưu đồ thuật toán chương trình

Trong nghiên cứu này, nhóm tác giả sử dụng ngôn ngữ lập trình Python để xây dựng chương trình xử lý và điều khiển hệ thống. Quy trình phát hiện xe cứu thương và điều khiển tín hiệu giao thông được mô tả trong Hình 7. Đầu tiên, camera thu nhận hình ảnh giao thông theo thời gian thực và truyền dữ liệu đến Raspberry Pi 5 để xử lý. Các khung hình sau đó được đưa vào mô hình YOLOv26 nhằm phát hiện và nhận diện xe cứu thương. Nếu không phát hiện đối tượng hoặc độ tin cậy của kết quả nhận diện nhỏ hơn 40%, hệ thống sẽ tiếp tục xử lý các khung hình kế tiếp. Ngược lại, khi phát hiện xe cứu thương với độ tin cậy từ 40% trở lên, Raspberry Pi 5 sẽ xuất tín hiệu điều khiển qua các chân GPIO.

Khi nhận được tín hiệu điều khiển, module điều khiển đèn giao thông sẽ chuyển trạng thái từ đèn đỏ sang đèn xanh để xe cứu thương di chuyển qua nút giao thuận lợi. Sau khi xe cứu thương rời khỏi vùng giao cắt, hệ thống đưa đèn giao thông trở về chu kỳ hoạt động bình thường. Toàn bộ quá trình được thực hiện liên tục trên từng khung hình, đảm bảo khả năng phát hiện, nhận diện và điều khiển theo thời gian thực.



Hình 7. Lưu đồ thuật toán chương trình.

3. THỰC NGHIỆM VÀ ĐÁNH GIÁ

Mô hình thực nghiệm của hệ thống được trình bày ở Hình 8. Trong đó, Raspberry Pi 5 được sử dụng làm bộ xử lý trung tâm, đảm nhiệm chức năng nhận diện xe cứu thương và điều khiển tín hiệu giao thông. Chương trình điều khiển được triển khai trên môi trường lập trình Thonny, đồng thời sử dụng phần mềm VNC để truy cập và điều khiển Raspberry Pi 5 thông qua máy tính cá nhân (PC). Nhằm tối ưu hiệu năng xử lý và giảm tải tài nguyên cho thiết bị biên, mô hình sau khi huấn luyện với định dạng PyTorch (best.pt) được chuyển sang định dạng best.ONNX và triển khai qua nền tảng ONNX Runtime trên Raspberry Pi 5.

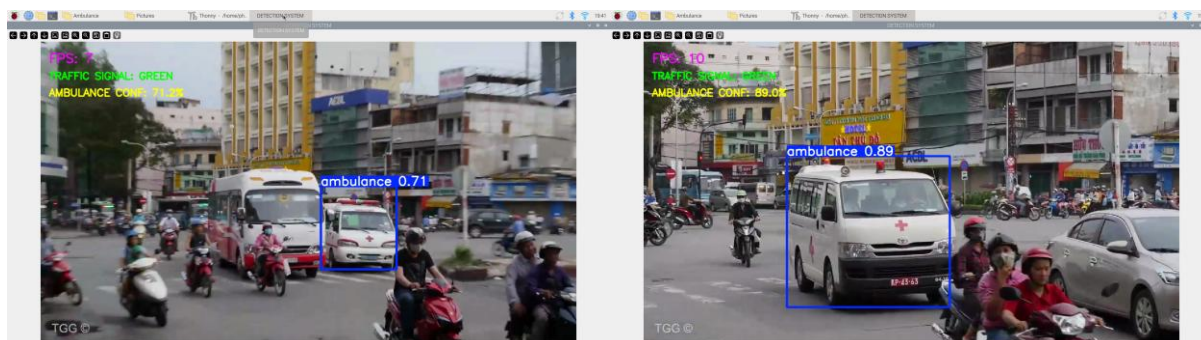


Hình 8. Mô hình hệ thống nhận diện xe cứu thương và điều khiển đèn giao thông.

Quá trình thực nghiệm được xây dựng với hai kịch bản: (1) sử dụng các đoạn video có sẵn làm dữ liệu đầu vào cho mô hình; (2) sử dụng camera để thu nhận hình ảnh từ màn hình máy tính PC và truyền đến Raspberry Pi 5 để xử lý. Kết quả cho thấy hệ thống hoạt động đúng theo ý tưởng thiết kế, có khả năng phát hiện xe cứu thương với độ chính xác tương đối tốt và gửi tín hiệu điều khiển kịp thời đến module điều khiển đèn giao thông. Trong suốt quá trình thử nghiệm, Raspberry Pi 5 vận hành ổn định trong nhiều giờ liên tục, không xuất hiện hiện tượng treo, tự khởi động lại hoặc dừng chương trình. Với nguồn cấp điện ổn định và khả năng tản nhiệt đáp ứng yêu cầu, thiết bị duy trì mức sử dụng tài nguyên và nhiệt độ trong giới hạn cho phép, đảm bảo khả năng hoạt động liên tục của hệ thống nhận diện theo thời gian thực.

3.1. Thực nghiệm nhận diện xe cứu thương từ dữ liệu video

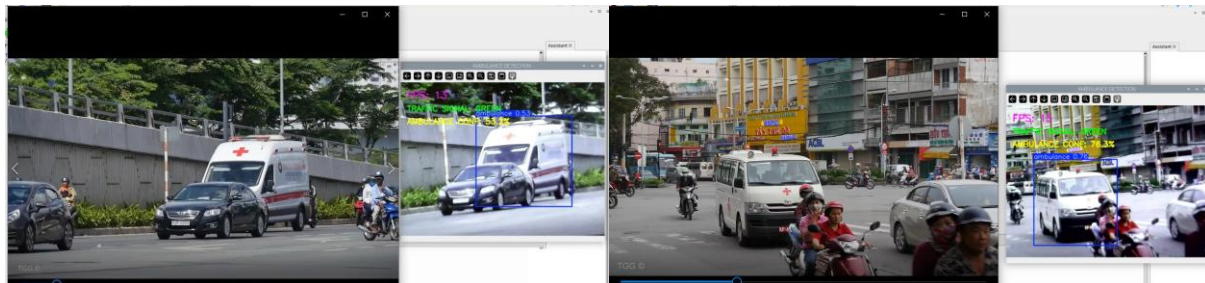
Trong kịch bản thực nghiệm thứ nhất, nhóm tác giả sử dụng các đoạn video giao thông có chứa hình ảnh xe cứu thương làm dữ liệu đầu vào trực tiếp cho mô hình YOLOv26. Hệ thống đèn tín hiệu được giả lập ở trạng thái ban đầu là màu đỏ, đại diện cho tình huống xe cứu thương đang tiếp cận nút giao nhưng chưa được ưu tiên lưu thông. Khi mô hình phát hiện xe cứu thương với độ tin cậy vượt ngưỡng thiết lập, Raspberry Pi 5 gửi tín hiệu điều khiển để chuyển trạng thái đèn giao thông từ màu đỏ sang màu xanh. Kết quả cho thấy mô hình có khả năng nhận diện xe cứu thương tương đối chính xác, với độ tin cậy của dự đoán cao trên từng khung hình và tốc độ xử lý đạt từ 6-12 FPS (Hình 9).



Hình 9. Một số hình ảnh nhận diện xe cứu thương qua video.

3.2. Thực nghiệm nhận diện xe cứu thương thông qua camera

Ở kịch bản thứ hai, nhóm tác giả đánh giá khả năng hoạt động của hệ thống khi sử dụng dữ liệu hình ảnh thu nhận trực tiếp từ camera. Thí nghiệm được bố trí trong điều kiện môi trường ban ngày, ánh sáng trong nhà ổn định. Một đoạn video giao thông có xe cứu thương được phát trên màn hình máy tính cá nhân, camera được đặt cách màn hình khoảng 50 cm để thu nhận hình ảnh đầu vào cho hệ thống.

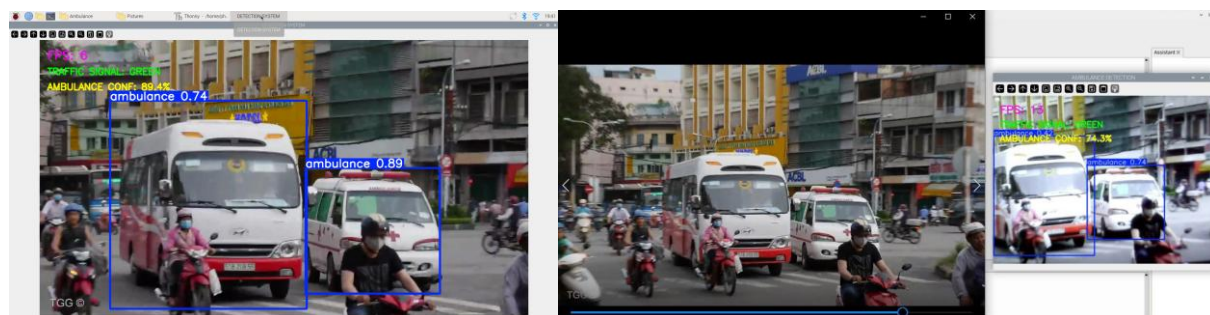


Hình 10. Một số hình ảnh nhận diện xe cứu thương qua camera thu hình ảnh từ màn hình máy tính.

Tương tự kịch bản thực nghiệm thứ nhất, trạng thái ban đầu của đèn tín hiệu được giả lập là màu đỏ. Khi camera ghi nhận hình ảnh xe cứu thương, mô hình YOLOv26 thực hiện nhận diện thành công, tín hiệu điều khiển được gửi, để chuyển đèn sang trạng thái màu xanh. Kết quả cho thấy hệ thống duy trì khả năng nhận diện ổn định với độ tin cậy dự đoán cao, tốc độ xử lý đạt từ 10 - 15 FPS, đáp ứng khả năng xử lý thời gian thực (Hình 10).

3.3. Đánh giá và bình luận kết quả

Kết quả thử nghiệm trên nhiều đoạn video giao thông có sẵn và dữ liệu hình ảnh thu nhận trực tiếp từ camera trên Raspberry Pi 5, cho thấy mô hình có khả năng phát hiện hầu hết các loại xe cứu thương của Việt Nam với độ tin cậy cao. Tốc độ xử lý của hệ thống dao động trong khoảng 6 -15 FPS. Tuy nhiên, khi sử dụng video có sẵn làm dữ liệu đầu vào, tốc độ FPS thấp hơn so với trường hợp nhận dữ liệu trực tiếp từ camera. Nguyên nhân là do Raspberry Pi 5 phải đồng thời thực hiện quá trình đọc, giải mã video và xử lý ảnh, dẫn đến gia tăng tính toán. Ngược lại, với dữ liệu đầu vào từ camera, thiết bị chỉ tập trung thực hiện quá trình suy luận và xử lý ảnh nên đạt tốc độ xử lý cao hơn (10 - 15 FPS).



Hình 11. Một số hình ảnh mô hình nhận dạng nhầm lẫn.

Bên cạnh khả năng nhận diện chính xác, mô hình vẫn tồn tại một số trường hợp nhận dạng nhầm xe ô tô thành xe cứu thương (Hình 11). Nguyên nhân chủ yếu xuất phát từ sự tương đồng về đặc trưng hình ảnh giữa các phương tiện, như các chi tiết có màu đỏ trên thân xe hoặc góc quan sát không thuận lợi. Ngoài ra, do tập dữ liệu huấn luyện còn hạn chế về sự đa dạng của các mẫu xe cứu thương, khiến mô hình chưa học đầy đủ các đặc trưng như màu sắc, ký hiệu dấu cộng y tế hoặc dòng chữ “Ambulance” trên xe.

Nhằm đánh giá hiệu quả và so sánh với các phiên bản YOLO trước, nhóm tác giả sử dụng bộ dữ liệu (Hình 5) và cùng tất cả các tham số cấu hình trong Bảng 1, để huấn luyện các mô hình YOLOv8n, YOLOv10n, YOLOv11n và YOLOv12n. Kết quả trình bày ở Bảng 4 cho thấy YOLOv26n có sự cân bằng tốt giữa độ chính xác và tốc độ xử lý. Cụ thể như: Thông số Precision = 0,960, chỉ thấp hơn nhẹ so với YOLOv12n (0,964), trong khi mAP@50 đạt 0,906, cao hơn YOLOv10n (0,893) và YOLOv8n (0,901). Với mAP@50-95 đạt 0,694, YOLOv26n vẫn duy trì khả năng định vị đối tượng ổn định, chỉ thấp hơn không đáng kể so với YOLOv11n (0,699) và YOLOv12n (0,696). Đặc biệt, về tốc độ suy luận, YOLOv26n đạt 370,3 FPS, cao nhất trong các mô hình được so sánh, vượt qua YOLOv12n (149,3 FPS), YOLOv11n (250 FPS), YOLOv10n (322,6 FPS) và YOLOv8n (232,6 FPS). Kết quả này cho thấy YOLOv26n được tối ưu hiệu quả về khả năng xử lý cũng như độ chính xác.

Bảng 4. Các chỉ số huấn luyện của các phiên bản YOLO trên cùng một điều kiện.

TT	Các phiên bản YOLO	Precision	Recall	mAP@50	mAP@50-95	FPS
1	YOLOv26n	0,96	0,818	0,906	0,694	370,3
2	YOLOv12n	0,964	0,852	0,913	0,696	149,3
3	YOLOv11n	0,938	0,856	0,916	0,699	250,0
4	YOLOv10n	0,943	0,838	0,893	0,683	322,6
5	YOLOv8n	0,938	0,839	0,901	0,686	232,6

Một điểm lưu ý rằng, giá trị 370,3 FPS trong Bảng 4 là tốc độ suy luận của mô hình YOLOv26n được đo trong quá trình đánh giá trên môi trường huấn luyện Google Colab sử dụng GPU NVIDIA Tesla T4 và chỉ phản ánh hiệu năng của mô hình. Trong khi đó, tốc độ 6 - 15 FPS được báo cáo trong phần thực nghiệm là tốc độ xử lý thực của toàn bộ hệ thống khi triển khai trên Raspberry Pi 5, bao gồm các bước thu nhận hình ảnh, tiền xử lý, suy luận, hậu xử lý, hiển thị kết quả và điều khiển đèn giao thông. Vì vậy, hai giá trị đo này được thực trên các phần cứng khác nhau và quy trình xử lý khác nhau nên có sự khác biệt lớn.

Nhìn chung, các kết quả thực nghiệm cho thấy YOLOv26 mang lại những cải tiến đáng kể so với các phiên bản YOLO trước đây nhờ khả năng giảm độ trễ suy luận, nâng cao tốc độ xử lý và duy trì độ chính xác nhận diện ổn định. Sự cải tiến này giúp YOLOv26 trở thành lựa chọn phù hợp cho các ứng dụng phát hiện đối tượng theo thời gian thực trên nền tảng Edge AI và các hệ thống nhúng như Raspberry Pi.

4. KẾT LUẬN

Trong nghiên cứu này, nhóm tác giả đã đề xuất và triển khai thành công hệ thống nhận diện xe cứu thương và điều khiển tín hiệu giao thông dựa trên mô hình học sâu YOLOv26. Bộ dữ liệu huấn luyện gồm 1.936 ảnh xe cứu thương tại Việt Nam được thu thập và xử lý, sau đó tiến hành huấn luyện, tối ưu trên nền tảng Google Colab. Kết quả đánh giá cho thấy mô hình đạt hiệu suất nhận diện tương đối tốt với các chỉ số Precision = 0,960, Recall = 0,818, mAP@50 = 0,906 và mAP@50-95 = 0,694.

Kết quả thực nghiệm trên nền tảng Raspberry Pi 5 cho thấy hệ thống có khả năng phát hiện hầu hết các loại xe cứu thương với độ tin cậy dự đoán khoảng 89% và tốc độ xử lý đạt trên 10 FPS, đáp ứng yêu cầu xử lý thời gian thực cho các ứng dụng giao thông thông minh.

Tuy nhiên, mô hình vẫn xuất hiện một số trường hợp nhận diện nhầm với các phương tiện có đặc điểm hình ảnh tương đồng, đặc biệt là các xe có dải màu đỏ hoặc hình dạng gần giống xe cứu thương.

Mặc dù kết quả nghiên cứu bước đầu đạt được tính khả thi của giải pháp nhưng hệ thống vẫn còn một số hạn chế. Thứ nhất, các thí nghiệm mới được thực hiện trong môi trường mô phỏng nên chưa phản ánh đầy đủ các điều kiện giao thông thực tế. Thứ hai, mô hình chỉ dừng ở việc phát hiện xe cứu thương mà chưa xác định được phương tiện có đang thực hiện nhiệm vụ ưu tiên hay không. Thứ ba, thuật toán điều khiển đèn giao thông còn đơn giản, chưa xem xét các yếu tố như hướng di chuyển, làn đường và các tình huống xung đột tại nút giao.

Trong thời gian tới, nhóm tác giả sẽ tiếp tục hoàn thiện hệ thống theo các hướng nghiên cứu sau: (1) mở rộng và đa dạng hóa tập dữ liệu huấn luyện, bổ sung thêm các mẫu xe cứu thương trong nhiều điều kiện chiếu sáng, góc quan sát, khoảng cách và kiểu dáng khác nhau, kết hợp các kỹ thuật tăng cường dữ liệu (Data Augmentation) và xây dựng tập dữ liệu kiểm thử (test) độc lập nhằm nâng cao khả năng tổng quát hóa và giảm tỷ lệ nhận diện nhầm; (2) tối ưu quá trình triển khai trên các nền tảng Edge AI thông qua các kỹ thuật tăng tốc suy luận như ONNX Runtime và TensorRT, đồng thời phát triển thuật toán điều khiển đèn giao thông thông minh có khả năng xử lý nhiều hướng lưu thông và các tình huống giao thông phức tạp.

TÀI LIỆU THAM KHẢO

- [1]. L. Zemmouchi-Ghomari, Artificial intelligence in intelligent transportation systems, *Journal of Intelligent Manufacturing and Special Equipment*, 6 (2025) 26-42. <https://doi.org/10.1108/JIMSE-11-2024-0035>
- [2]. D. Shokri, C. Larouche, S. Homayouni, Real-Time Moving Vehicle Counting and Speed Estimation Toward Efficient Traffic Surveillance, *IEEE Access*, 13 (2025) 36687-36700. <https://doi.org/10.1109/ACCESS.2025.3540950>
- [3]. S. Dikshit, A. Atiq, M. Shahid, V. Dwivedi, A. Thusu, The Use of Artificial Intelligence to Optimize the Routing of Vehicles and Reduce Traffic Congestion in Urban Areas, *EAI Endorsed Transactions on Energy Web*, 10 (2023) 1-13. <https://doi.org/10.4108/ew.4613>
- [4]. T. X. Kiên, Đ. X. Trường, P. T. Thành, L. M. Dương, Nghiên cứu các kỹ thuật thị giác máy tính ứng dụng theo dõi phương tiện, giám sát giao thông trong thời gian thực, *Tạp chí Khoa học và công nghệ*, 59 (2023) 20-24. <https://doi.org/10.57001/huih5804.2023.103>
- [5]. A. S. Pillai, Traffic management: Implementing AI to optimize Traffic flow and reduce congestion, *Journal of Emerging Technologies and Innovative Research (JETIR)*, 11 (2024) c272-c278. <https://www.jetir.org/view?paper=JETIR2407235>
- [6]. L. Q. Thao, D. D. Cuong, N. T. Anh, P. M. Anh, H. M. Duc, N. Minh, Automatic Traffic Red-Light Violation Detection Using AI, *International Information Engineering Technology Association*, 27 (2022) 75-80. <https://doi.org/10.18280/isi.270109>
- [7]. R. Ithrees, Udhayakumar T., Susvin S., Rohini Priya, Yuvaraj N., AI-Based Intelligent Traffic Management System, *International Journal of Scientific Research and Technology*, 3 (2026) 846-858. <https://doi.org/10.5281/zenodo.19698757>
- [8]. D. K. Thai, N. D. Quang-Anh, N. Quynh-Chi Nguyen, N. T. Thu Hien, V. Quan, D. X. Minh-Duc, N. Van-Ninh, Optimizing Traffic Light Control using YOLOv8 for Real-Time Vehicle Detection and Traffic Density, 2024 9th International Conference on Integrated Circuits, Design, and Verification, (2024) 119-124. <https://doi.org/10.1109/icdv61346.2024.10616901>
- [9]. S. S. Nagtilak, S. D. Sakhalkar, S. S. Kumbhar, A. J. Powar, M. M. Gaonkar, Multimodal Ambulance Detection Using Image Processing and RF433MHz Sensor, *African Journal of Biomedical Research*, 27 (2024) 934-942. <https://doi.org/10.53555/AJBR.v27i6S.7640>

- [10]. Yugenthiran. S, P.N. Shiammala, Smart Traffic Light System With Ambulance Detection Using Yolo, International Journal of Creative and Open Research in Engineering and Management, 2 (2026) 1-7. <https://doi.org/10.55041/ijcope.v2i5.038>
- [11]. N. X. Thắng, Ứng dụng mô hình học sâu thích ứng trong bài toán phát hiện phương tiện giao thông, Tạp chí điện tử Khoa học và Công nghệ Giao thông, Trường Đại học Công nghệ Giao thông vận tải, 3 (2023) 38-46. <https://doi.org/10.58845/jstt.utt.2023.vn.3.2.38-46>
- [12]. V. L. Q. Phương, T. N. M. Thư, P. N. Khang, Hiệu quả các nhóm mô hình học sâu trong bài toán phát hiện phương tiện giao thông, Tạp chí Khoa học Đại học Cần Thơ, 60 (2024) 20-27. <https://doi.org/10.22144/ctujos.2024.240>
- [13]. D. D. Aboyomi, C. Daniel, A Comparative Analysis of Modern Object Detection Algorithms: YOLO vs. SSD vs. Faster R-CNN, ITEJ- Information Technology Engineering Journals, 8 (2023) 96 - 106. <https://doi.org/10.24235/itej.v8i2>
- [14]. J. P. Q. Tomas, S. V. P. C. Feria, J. R. A. Franco, H. C. Laguna, A Computer Vision Approach to Ambulance Classification in the Philippines using YOLOv5 Small, In 2024 The 8th International Conference on Machine Learning and Soft Computing (ICMLSC 2024), (2024) 172 – 177. <https://doi.org/10.1145/3647750.3647777>
- [15]. A. Mecocci, C. Grassi, RTAIAED: A Real-Time Ambulance in an Emergency Detector with a Pyramidal Part-Based Model Composed of MFCCs and YOLOv8, Sensors, 24 (2024) 1-24. <https://doi.org/10.3390/s24072321>
- [16]. A. Khode, T. R. Teja, B. S. C. Goud, D. Goud, T. Vallabhapurapu, Real-Time Ambulance Detection on Resource Constrained Edge Devices Using YOLOv5 and ONNX Runtime, International Journal for Research in Applied Science & Engineering Technology (IJRASET), 14 (2026) 1406-1417. <https://doi.org/10.22214/ijraset.2026.77118>
- [17]. J. Redmon, S. Divvala, R. Girshick, A. Farhadi, You Only Look Once: Unified, Real-Time Object Detection, 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), (2016) 779-788. <https://doi.org/10.1109/CVPR.2016.91>
- [18]. G. Lavanya, S. D. Pande, Enhancing Real-time Object Detection with YOLO Algorithm, EAI Endorsed Transactions on Internet of Things, 10 (2024) 1-9. <https://doi.org/10.4108/eetiot.4541>
- [19]. R. Sapkota, R. H. Cheppally, A. Sharda, M. Karkee, YOLO26: Key Architectural Enhancements and Performance Benchmarking for Real-Time Object Detection, arXiv:2509.25164v5, 2026. <https://doi.org/10.48550/arXiv.2509.25164>
- [20]. P. Hidayatullah, R. Tubagus, YOLO26: A Comprehensive Architecture Overview and Key Improvements, arXiv:2602.14582, 2026. <https://doi.org/10.48550/arXiv.2602.14582>
- [21]. N. Đ. An, Nghiên cứu về Raspberry Pi và ứng dụng trong hệ thống IoT, Tạp chí Thiết bị Giáo dục, 2 (2024) 4-6. <https://vjol.info.vn/tctbgd/issue/view/8125>
- [22]. Raspberry Pi. <https://www.raspberrypi.com/products/raspberry-pi-5/> (truy cập 02/6/2026)
- [23]. Ambulance1 Computer Vision Model. <https://universe.roboflow.com/business-jh3ri/ambulance1-z5bdu/dataset/1> (truy cập 04/5/2026)
- [24]. F. Li, H. Yan, L. Shi, Multi-scale coupled attention for visual object detection, Scientific Reports, 14 (2024) 1-19. <https://doi.org/10.1038/s41598-024-60897-8>