NGHIÊN CỨU VÀ ỨNG DỤNG MÔ HÌNH YOLOV8 TRONG PHÁT HIỆN VÀ THỐNG KÊ SỐ LƯỢNG VỊ TRÍ CÒN TRỐNG TRONG BÃI ĐỖ Ô TÔ

Phạm Thị Thanh Thủy*, Kiều Đức Hồng, Trịnh Sơn Trà
Nguyễn Thị Hương, Cao Hạnh Ly, Thái Trung Hiếu
Trường Đại học Tài nguyên và Môi trường Hà Nội

Tóm tắt

Trong những năm gần đây, số lượng ô tô tại các đô thị lớn của Việt Nam tăng lên nhanh chóng. Do đó, thông tin về vị trí các bãi đỗ xe và số lượng chỗ đỗ còn trống là những thông tin rất cần thiết, giúp cho người lái xe nhanh chóng tìm được bãi đỗ xe thích hợp nhất. Hiện nay, có nhiều công nghệ và thuật toán đã được áp dụng để nhận diện và phân loại vật thể, trong số đó không thể không kể đến YOLOv8 bởi đây là công nghệ có khả năng phát hiện đối tượng trong thời gian thực nhờ vào tốc độ xử lý nhanh chóng, kết quả đạt độ chính xác cao, dễ dàng được tích hợp và triển khai trong các hệ thống thực tế. Với các ưu điểm vượt trội của YOLOv8, nghiên cứu đã hướng đến mục tiêu phát hiện các vật thể là ô tô trong bãi đỗ xe bằng công nghệ này. Sau đó đánh giá, phân tích dựa trên dữ liệu xe thu thập được để thống kê số lượng vi trí trong bãi đỗ xe ô tô. Từ kết quả nghiên cứu cho thấy nhận diện ô tô từ hình ảnh đạt độ chính xác 90 % trở lên, nhận diện ô tô qua video (thời gian thực) chỉ đạt 80 %.

Từ khóa: Nhận diện ô tô; Thống kê số lượng; YOLOv8.

Abstract

Research and application of YOLOv8 model in detecting and counting available parking spaces in a car park

Recently, the number of cars in major urban areas of Vietnam has increased rapidly. Therefore, information about the locations of parking lots and the number of available parking spaces is essential, helping drivers quickly find the most suitable parking spot. Many technologies and algorithms have been applied to recognize and classify objects, among which YOLOv8 cannot be overlooked. This technology can detect objects in real-time thanks to its fast processing speed, high accuracy, and ease of integration and deployment in practical systems. With the outstanding advantages of YOLOv8, this research aims to detect cars in parking lots using this technology. Then, it evaluates and analyzes the collected vehicle data to statistically count the available spots in car parking lots. The research results show that image recognition achieves an accuracy of 90 % or more, while car recognition from videos (real-time) only reaches 80 %.

Keywords: Car detection; Quantity statistics; YOLOv8.

Nhận bài: 01/8/2024; Phản biện xong: 19/8/2024; Duyệt đăng: 26/9/2024

***Tác giả liên hệ, Email:** pttthuy.tdbd@hunre.edu.vn

DOI: https://doi.org/10.63064/khtnmt.2024.606

1. Giới thiệu

Nhận diện vật thể là quá trình xác định và phân loại các đối tượng trong hình ảnh, video hoặc trong thời gian thực (real time). Sự kết hợp giữa trí tuệ nhân tạo (AI - Artificial Intelligence) và thị giác máy tính (CV- Computer Vision) có thể được ứng dụng trong nhiều lĩnh vực của đời sống như nhận diện khuôn mặt, phát hiện vật thể, cảnh báo nguy hiểm, giám sát an ninh [10].

Để giải quyết các bài toán ứng dụng trên đã có nhiều thuật toán được công bố [6, 7, 8, 11, 13, 15]. Hiện nay học sâu (Deep Learning) đang là một công nghệ được các nước trên thế giới quan tâm và đầu tư [2]. Điển hình như nghiên cứu của tác giả [9] đã ứng dụng mô hình YOLO và mạng CFNN để phát triển một hệ thống giám sát giao thông thông minh: Ghi lại lưu lượng giao thông và thông tin loại phương tiện trên đường. Phương pháp đề xuất của nhóm tác giả đạt độ chính xác 90,45 % trên bộ dữ liệu công khai của Viện Công nghệ Bắc Kinh [9]. Nghiên cứu của Minar Mahmud Rafi và cộng sự (2022) cũng đã tiến hành phân tích hiệu quả của các mô hình học sâu YOLO để nhận dạng phương tiện trong khu vực Nam Á. Trọng tâm của nghiên cứu là tổng hợp bộ dữ liệu lớn nhất về các phương tiện dành riêng cho khu vực Nam Á và áp dụng bộ dữ liệu này để theo dõi và nhận dạng các phương tiện ngay cả khi đang chuyển động. Để phát triển điều này, nghiên cứu đã tăng các biến thể lớp và số lượng dữ liệu. Nhóm tác giả đã đào tạo các phiên bản khác nhau của mô hình YOLOv5 bằng tập dữ liệu của mình để đo lường mức độ chính xác giữa các mô hình trong việc phát hiện các phương tiện đặc thù. Nếu tính năng phát hiện và theo dõi phương tiện được áp dụng và triển khai trong nguồn cấp dữ liệu camera giao thông trực tiếp, thông tin có thể được sử dụng để tạo ra hệ thống giao thông thông minh có thể điều chỉnh tắc nghẽn và định tuyến bằng cách xác định và phân tách các phương tiện di chuyển nhanh và chậm trên đường. Tại Việt Nam, các nghiên cứu mới nhất phải kể đến là nghiên cứu của Trà Văn Đồng tiến hành so sánh hai thuật toán SSD và YOLO trong phát hiện đối tượng, kết quả chỉ ra rằng thuật toán YOLO cho chỉ số đánh giá (AP - Average Precision) cao hơn với với thuật toán SSD [3]. Theo nhóm tác giả [1] dựa trên thuật toán xác định vật thể của YOLOv4 trong việc nhận diện và phân loại phương tiện giao thông tại Việt Nam, kết quả nghiên cứu đã nhận diện được 05 loại phương tiện gồm: Ô tô, xe buýt, xe tải, xe máy và xe đạp. Với mật độ lưu thông khác nhau (thấp, trung bình và cao) cho thấy tỉ lệ phân loại chính xác từng phương tiện cao nhất khi mật độ lưu thông là thấp nhất và ngược lại [1]. Nghiên cứu của [2] đã sử dụng camera PTZ để thu thập dữ liệu ảnh góc rộng, tiền xử lý hình ảnh, sau đó đưa vào xử lý học sâu bằng thuật toán YOLOv4. Kết quả thử nghiệm cho thấy mô hình có thể hoạt động với tốc độ 45 khung hình/giây, đạt độ chính xác 98 % [2].

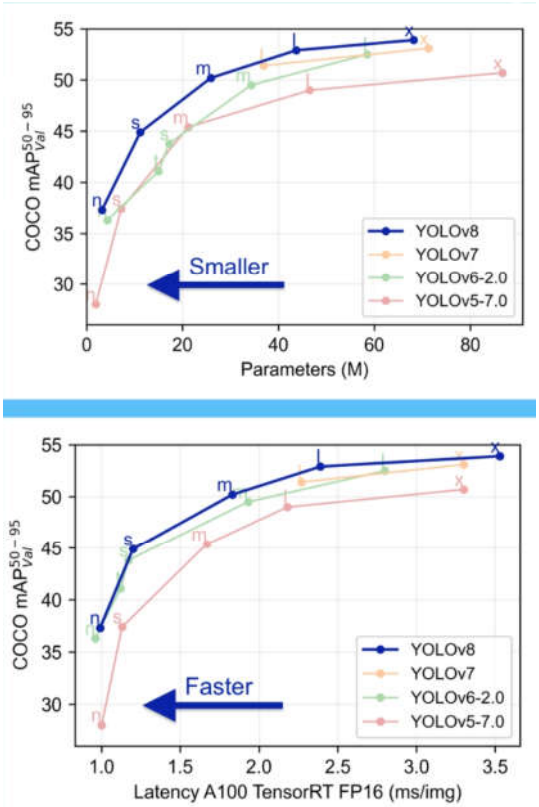
Hiện nay, số lượng ô tô tại các đô thị lớn của Việt Nam đang gia tăng nhanh chóng. Do đó, thông tin về vị trí các bãi đỗ xe và số lượng chỗ đỗ còn trống trở nên rất quan trọng bởi vì các thông tin này sẽ giúp người lái xe tìm được bãi đỗ xe thích hợp một cách nhanh chóng. Trong nghiên cứu này, nhóm tác giả đã sử dụng YOLOv8 để phát hiện vật thể là ô tô và

tính toán số lượng ô tô trong bãi đỗ bởi khả năng phát hiện đối tượng trong thời gian thực nhanh và độ chính xác cao, dễ dàng tích hợp và triển khai trong thực tế.

2. Mô hình YOLOv8

YOLOv8 là phiên bản mới nhất trong dòng mô hình YOLO, YOLOv8 là phiên bản nâng cấp của YOLOv7 và được tích hợp nhiều công nghệ, tính năng (AutoAnchor, Ensemble, Video Supervision, AutoScale). YOLOv8 có những ưu điểm vượt trội như tốc độ và độ chính xác cao, sự linh hoạt khi có thể nhận diện đối tượng và phân loại trên cả GPU và CPU.

YOLOv8 có nhiều tham số hơn so với các phiên bản tiền nhiệm như YOLOv5, nhưng ít tham số hơn so với YOLOv6. Nó cung cấp khoảng 33 % mAP, nhiều hơn cho các mô hình kích thước n và mAP lớn hơn nói chung. YOLOv8 có thời gian suy luận nhanh hơn so với tất cả các phiên bản YOLO khác.



Hình 1: So sánh YOLOv8 với các phiên bản trước đó [5]

Trong YOLOv8, ta có các kích thước mô hình khác nhau như YOLOv8n - nano, s - small, m - medium, l - large và x - extra large.

Bảng 1. So sánh kích thước mô hình

MODEL	SIZE (PIXELS)	MAP^{VAL} 50-95	SPEED CPU ONNX (MS)	SPEED AT100 TENSORTRT (MS)	PARAMS (M)	FLOPS (B)
YOLOv8n	640	37,3	80,4	0,99	3,2	8,7
YOLOv8s	640	44,9	128,4	1,20	11,2	28,6
YOLOv8m	640	50,2	234,7	1,83	25,9	78,9
YOLOv8l	640	52,9	375,2	2,39	43,7	165,2
YOLOv8x	640	53,9	479,1	3,53	68,2	257,8

Nguồn: Hình ảnh từ kho lưu trữ Ultralytics YOLOv8

Kích thước mô hình tương ứng tuyến tính với mAP và nghịch đảo tương ứng với thời gian suy luận. Các mô hình lớn mất nhiều thời gian suy luận để phát hiện đối tượng một cách chính xác với mAP cao hơn. Các mô hình nhỏ có thời gian suy luận nhanh hơn nhưng có mAP tương đối thấp hơn. Các mô hình lớn hơn tốt hơn nếu chúng ta có ít dữ liệu. Các mô hình nhỏ hơn hiệu quả hơn nếu ta có ít không gian (các tình huống biên).

2.1. Kiến trúc của YOLOv8

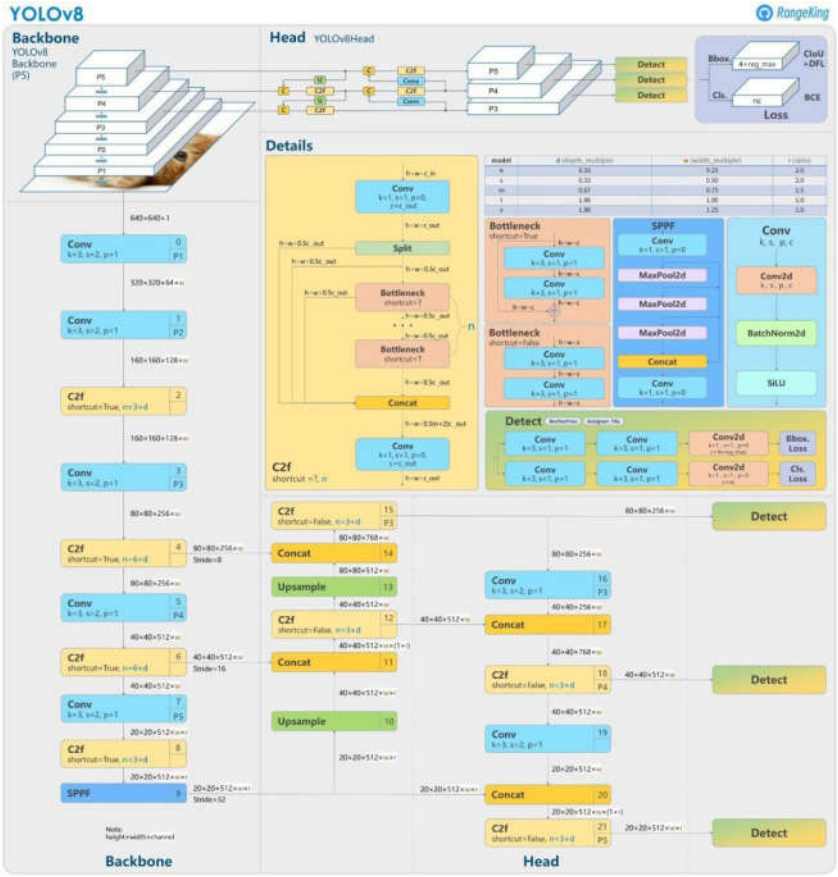
Kiến trúc của YOLOv8 được thiết kế để giải quyết những hạn chế của các phiên

bản YOLO trước đó trong khi vẫn duy trì sự cân bằng giữa tốc độ và độ chính xác.

Mô hình được chia thành ba thành phần chính: Backbone, Neck và Head. Backbone chịu trách nhiệm trích xuất các tính năng từ hình ảnh đầu vào và YOLOv8 sử dụng nhiều Backbone khác nhau, bao gồm CSPDarknet53 và EfficientDet.

Neck có chức năng kết nối giữa Head và Backbone. Neck chịu trách nhiệm dự đoán các hộp giới hạn, lớp đối tượng và điểm tin cậy.

Kiến trúc của YOLOv8 được đặc trưng bởi thiết kế mô-đun, các biến có thể mở rộng, Backbone được cải thiện và các chiến lược đào tạo tiên tiến. Các tính năng này cùng nhau góp phần vào thành công của YOLOv8 trong việc phát hiện đối tượng theo thời gian thực, khiến mô hình này trở thành lựa chọn phổ biến cho các nhà nghiên cứu trong lĩnh vực thị giác máy tính.



Hình 2: Kiến trúc của YOLOv8 [5]

2.2. Ứng dụng của YOLOv8

YOLOv8 có thể được áp dụng cho ba tác vụ quan trọng trong thị giác máy tính:

2.2.1. Phân loại (Classification)

Phân loại là một tác vụ đơn giản với kết quả bao gồm các chỉ số lớp (class index) và các điểm tin cậy (confidence score). Muốn phân loại được đối tượng chỉ cần xác định sự hiện diện của một lớp cụ thể trong hình ảnh đầu vào, mà không cần xác định vị trí chính xác của đối tượng. Thao tác phân loại được thực hiện dễ dàng bằng cách thêm tiền tố -cls vào mô hình YOLOv8 mà người dùng muốn sử dụng. YOLOv8 có các kích thước khác nhau:

YOLOv8n (Nano), YOLOv8s (Small), YOLOv8m (Medium), YOLOv8l (Large) và YOLOv8x (Extra Large). Khả năng phân loại mạnh mẽ của YOLOv8 xuất phát từ việc được huấn luyện trước trên tập dữ liệu ImageNet lớn, chứa hàng triệu hình ảnh.

Với cấu hình YOLOv8n của YOLOv8, tiến hành tạo bộ phân loại như sau:

```
from ultralytics import YOLO
# Load the model
model = YOLO('yolov8n-cls.pt')
# Classification result
result = model('SOURCE_PATH')
```

2.2.2. Nhận dạng đối tượng (Object detection)

Nhận dạng đối tượng là một tác vụ phát triển từ phân loại. Trong tác vụ nhận dạng đối tượng cần xác định các lớp khác nhau có mặt trong hình ảnh và phát hiện vị trí chính xác của chúng. Vị trí của các đối tượng này được thể hiện trực quan thông qua các hộp giới hạn (Bounding Boxes).

Các mô hình YOLOv8 đã được huấn luyện trước trên tập dữ liệu COCO (tập dữ liệu hình ảnh rất lớn) có thể thực hiện nhận dạng đối tượng ngay sau khi được triển khai mà không cần thêm tiền tố nào.

Dưới đây là một ví dụ về việc thực hiện nhận dạng đối tượng trên một hình ảnh:

```
from ultralytics import YOLO
# Load the model
model = YOLO('yolov8n.pt')
# Classification result
result = model('SOURCE_PATH')
```

2.2.3. Phân đoạn (Segmentation)

Phân đoạn nằm ở bước tiếp theo sau nhận dạng đối tượng. Trong nhận dạng đối tượng, vị trí của các đối tượng đã được xác định và ước tính vị trí của chúng thông qua các hộp giới hạn. Phân đoạn nhằm xác định các pixel riêng lẻ thuộc về một đối tượng. Phân đoạn chính xác hơn nhiều so với nhận dạng đối tượng và có tiềm năng ứng dụng rất lớn trong thực tế.

Thực hiện phân đoạn với YOLOv8 dễ dàng bằng cách thêm tiền tố *-seg*.

```
from ultralytics import YOLO
# Load the model
model = YOLO('yolov8n-seg.pt')
# Classification result
result = model('SOURCE_PATH')
```

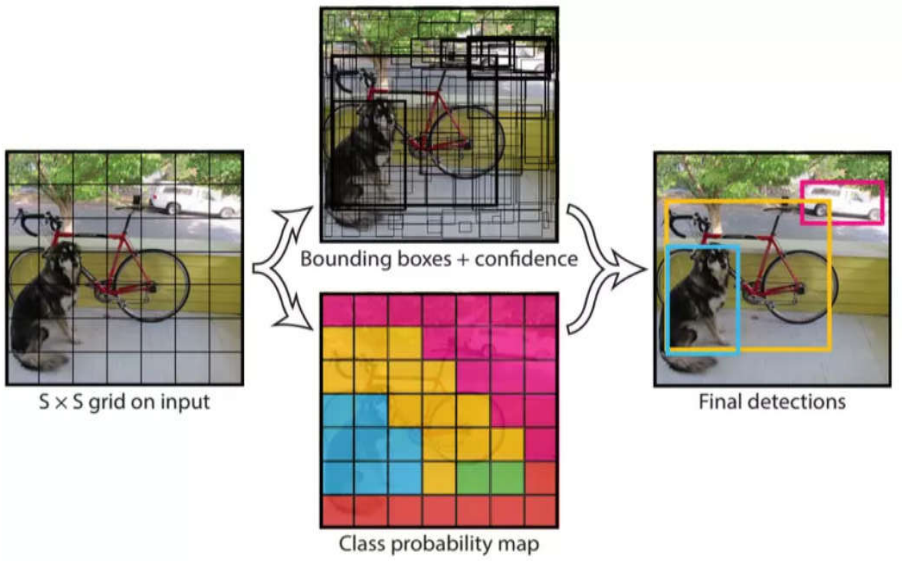
3. Phương pháp nhận diện đối tượng

Mạng YOLO là một thuật toán phát hiện đối tượng qua một lần nhìn, sử dụng một mạng nơ ron tích chập (CNN) duy nhất để thực hiện xử lý hình ảnh và có thể trực tiếp tính toán kết quả phân loại và tọa độ vị trí của các đối tượng trong khung hình được đưa vào tính toán. Với việc sử dụng định vị và phân loại đối tượng end-to-end đã làm tăng tốc độ tính toán lên đáng kể qua các phiên bản và phiên bản YOLOv8 là phiên bản mới nhất tính đến thời điểm năm 2023.

3.1. Cách YOLO hoạt động

Đầu vào của mô hình là một ảnh, mô hình sẽ nhận dạng ảnh đó có đối tượng nào hay không, sau đó sẽ xác định tọa độ của đối tượng trong bức ảnh. Ảnh đầu vào được chia thành thành $S \times S$, ô thường thì sẽ là 3×3, 7×7, 9×9,…việc chia ô này có ảnh hưởng tới việc mô hình phát hiện đối tượng bởi vì tâm của Bounding Box nằm ở ô nào thì ô đó sẽ chứa đối tượng, cho dù đối tượng có thể ở các ô khác thì

cũng sẽ trả về là 0.



Hình 3: Hình ảnh minh họa cho các bước nhận diện đối tượng [16]

Với dữ liệu đầu vào là một ảnh, đầu ra mô hình là một ma trận 03 chiều có kích thước $S \times S \times (5 \times B + C)$, với số lượng tham số mỗi ô là $(5 \times B + C)$, với B và C lần lượt là số lượng Box và Class mà mỗi ô cần dự đoán. Ví dụ, với hình ảnh trên chia thành 7×7 ô, mỗi ô cần dự đoán 02 bounding box và 03 đối tượng (object): Con chó, ô tô, xe đạp thì đầu ra là $7 \times 7 \times (5 \times 2 + 3)$ hay $7 \times 7 \times 13$, mỗi ô sẽ có 13 tham số, kết quả trả về $(7 \times 7 \times 2 = 98)$ bounding box.

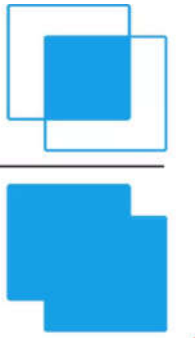
Dự đoán mỗi bounding box gồm 05 thành phần: (x, y, w, h, prediction), với (x, y) là tọa độ tâm của bounding box, (w, h) lần lượt là chiều rộng và chiều cao của bounding box, prediction được định nghĩa $P(Object) * IoU(pred, truth)$.

Ở Hình 3 cho thấy mỗi ô sẽ có 13 tham số, cụ thể như sau: Tham số 1 sẽ chỉ ra ô đó có chứa đối tượng nào hay không *P(Object)*; Tham số 2, 3, 4, 5 sẽ trả về x, y, w, h của Box1; Tham số 6, 7, 8, 9, 10 tương tự sẽ Box2; Tham số 11, 12, 13 lần lượt là xác suất ô đó có chứa object1(P(chó|object), object2(P (ô tô|object)), object3(P(xe đạp|object)).

➢ *Hàm tính IoU (Intersection over Union)*

IoU là hàm đánh giá độ chính xác của object detector trên tập dữ liệu cụ thể. IoU được tính bằng:

$$IoU = \frac{\text{Area of Overlap}}{\text{Area of Union}}$$



Trong đó, Area of Overlap là diện tích phần giao nhau giữa predicted bounding box với grouth - truth bouding box, còn Area of Union là diện tích phần hợp giữa predicted bounding box với grouth - truth bounding box. Những bounding box được đánh nhãn bằng tay trong tập traing set và test set. Nếu IoU> 0,5 thì prediction được đánh giá là tốt.

➢ *Hàm lỗi (Loss Function)*

Hàm lỗi trong YOLO được tính trên việc dự đoán và nhãn mô hình. Cụ thể đó là tổng độ lỗi của 03 thành phần sau [14]:

* *Classification loss*: Độ lỗi dự đoán loại nhãn của đối tượng (object)

Hàm lỗi này chỉ tính trên những ô vuông có xuất hiện đối tượng, còn những ô vuông khác không tính đến. Classification loss được tính bằng công thức sau:

$$L_{Classification} = \sum_{i=0}^{S^2} \mathbb{1}_i^{obj} \sum_{c \in Class} (p_i(c) - \hat{p}_i(c))^2 \quad (1)$$

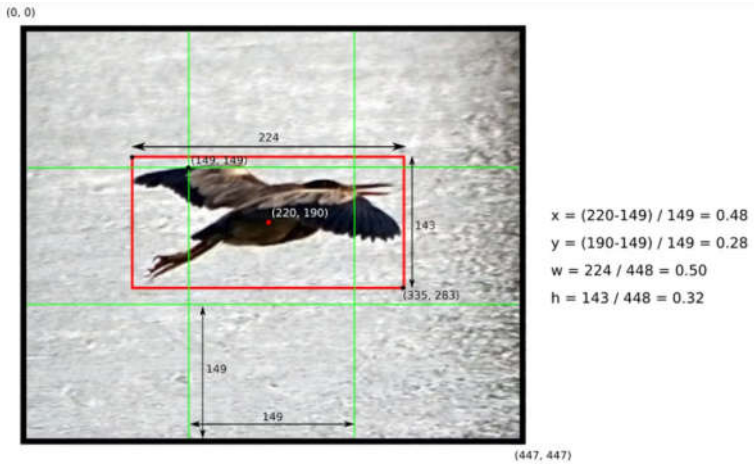
trong đó:

$\mathbb{1}_i^{obj}$: Bằng 1 nếu ô vuông đang xét có đối tượng và ngược lại bằng 0;

$\hat{p}_i(c)$: Xác suất có điều kiện của lớp C tại ô vuông tương ứng mà mô hình dự đoán.

* *Localization loss*: Độ lỗi dự đoán tọa độ tâm, chiều cao, chiều rộng của boundary box (x,y,w,h).

Localization loss là hàm lỗi dùng để tính giá trị lỗi cho boundary box được dự đoán bao gồm: Tọa độ tâm, chiều rộng, chiều cao so với vị trí thực tế từ dữ liệu huấn luyện của mô hình. Không nên tính giá trị hàm lỗi này trực tiếp từ kích thước ảnh thực tế mà cần phải chuẩn hóa về [0, 1] so với tâm của bounding box. Việc chuẩn hóa kích thước này giúp cho mô hình dự đoán nhanh hơn và chính xác hơn so với để giá trị mặc định của ảnh (Hình 4).



Hình 4: Chuẩn hóa kích thước của đối tượng [4]

Giá trị hàm Localization loss được tính trên tổng giá trị lỗi dự đoán toạ độ tâm (x, y) và (w, h) của predicted bounding box với grouth-truth bounding box. Tại mỗi ô có chứa đối tượng, ta chọn 01 boundary box có IoU (Intersect over Union) tốt nhất, rồi sau đó tính độ lỗi theo các boundary box này.

Giá trị hàm lỗi dự đoán tọa độ tâm (x, y) của predicted bounding box và (x̂, ŷ) là tọa độ tâm của truth bounding box được tính như sau:

$$\lambda_{Coord} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{obj} (x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 \quad (2)$$

Giá trị hàm lỗi dự đoán (w,h) của predicted bounding box so với truth bounding box được tính như sau:

$$\lambda_{Coord} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{obj} (\sqrt{w_i} - \sqrt{\hat{w}_i})^2 + (\sqrt{h_i} - \sqrt{\hat{h}_i})^2 \quad (3)$$

* *Confidence loss*: Độ lỗi dự đoán bounding box đó chứa đối tượng (object) so với nhãn thực tế tại ô vuông đó.

Confidence loss là độ lỗi giữa dự đoán boundary box đó chứa đối tượng so với nhãn thực tế tại ô vuông đó. Độ lỗi này tính trên cả những ô vuông chứa đối tượng và không chứa đối tượng.

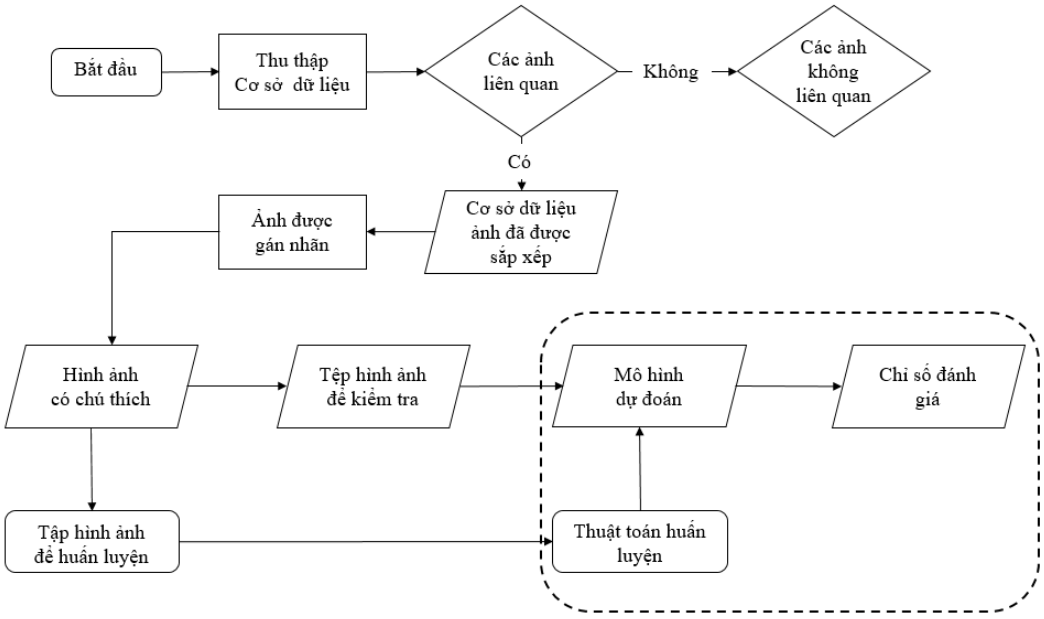
$$L_{Confidence} = \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{obj} (C_i - \widehat{C}_i)^2 + \lambda_{noobject} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{obj} (C_i - \widehat{C}_i)^2 \quad (4)$$

* *Total loss*: Tổng của 03 hàm lỗi trên.

$$L_{Total} = L_{Classification} + L_{Localization} + L_{Confidence} \quad (5)$$

3.2. Quy trình phát hiện đối tượng

Nghiên cứu dựa trên thuật toán xác định vật thể của YOLOv8, quy trình phát hiện phương tiện ô tô được trình bày trong Hình 5.



Hình 5: Quy trình phát hiện đối tượng

4. Kết quả nghiên cứu

4.1. Thu thập cơ sở dữ liệu

Nhóm nghiên cứu đã thu thập tổng cộng 17.000 bức ảnh từ cơ sở dữ liệu của Tensoflow và các nguồn khác nhau. Tất cả các ảnh đều có kích thước là 600 × 400 pixels, độ phân giải 96 dpi để có thể dễ dàng gán nhãn và giảm đi kích cỡ dung lượng. Để tăng độ mạnh cho mô hình đào tạo, nghiên cứu tiến hành thu thập các ảnh từ nhiều góc độ khác nhau để đảm bảo điều kiện chiếu sáng và mức tương phản là khác nhau. Loại bỏ những ảnh không liên quan đến đối tượng nghiên cứu và sắp xếp lại các ảnh có liên quan đến đối tượng nghiên cứu để phục vụ cho việc xử lý dữ liệu.

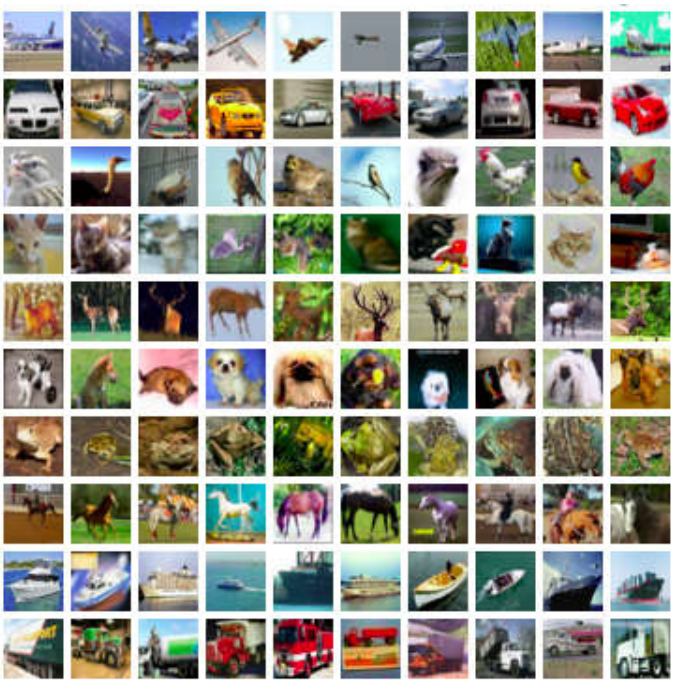
4.2. Gán nhãn đối tượng trên ảnh

Các hình ảnh có liên quan đến đối tượng được chuẩn hóa về kích thước 640 × 640 pixel để phù hợp với mô hình YOLOv8 và tăng tính nhất quán của các mẫu huấn luyện.

Nhóm nghiên cứu sử dụng công cụ MakeAI.Sense để thực hiện việc gán nhãn cho đối tượng trên ảnh. Kết quả sau khi xử lý thu được tập dữ liệu định dạng *.txt có các trường là <*class_id*> <*center_x*> <*center_y*> <*width*> <*height*>. Trong đó *class_id* có giá trị là "0" tương ứng với vật thể là ô tô. "*center_x*" và "*center_y*" lần lượt là tọa độ tâm của bounding box. "*Width*" và "*Height*" là chiều dài và và độ cao của bouding box. Tất cả các chỉ số đều được chuẩn hóa về khoảng giá trị [0, 1].

Hình 6: Hình ảnh xe trước và sau khi gán nhãn

Nghiên cứu còn sử dụng bộ dữ liệu ảnh đã được gán nhãn lấy từ kho dữ liệu ảnh có sẵn của Coco, VOC, ImageNet.



Hình 7: Bộ dữ liệu có sẵn trên MSCoCo

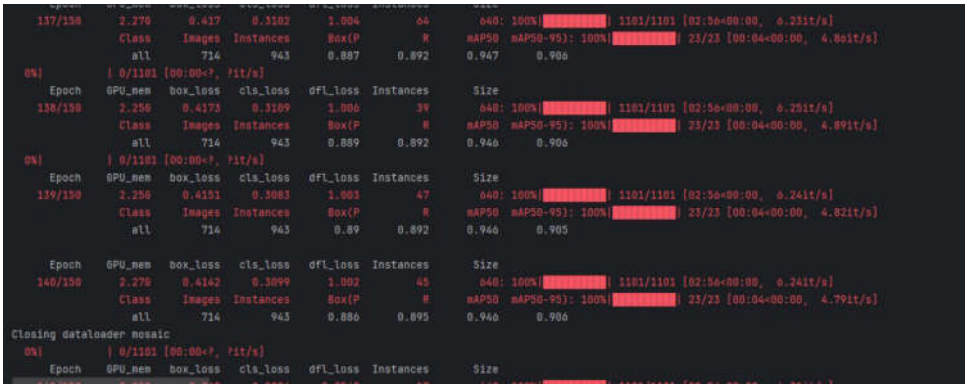
4.3. Huấn luyện mô hình YOLOv8

Chia tập hình ảnh đã được gán nhãn ra thành hai phần: 70 % số ảnh được sử dụng để huấn luyện (train) và 30 % số ảnh còn lại dùng để kiểm tra (Validation - Val).

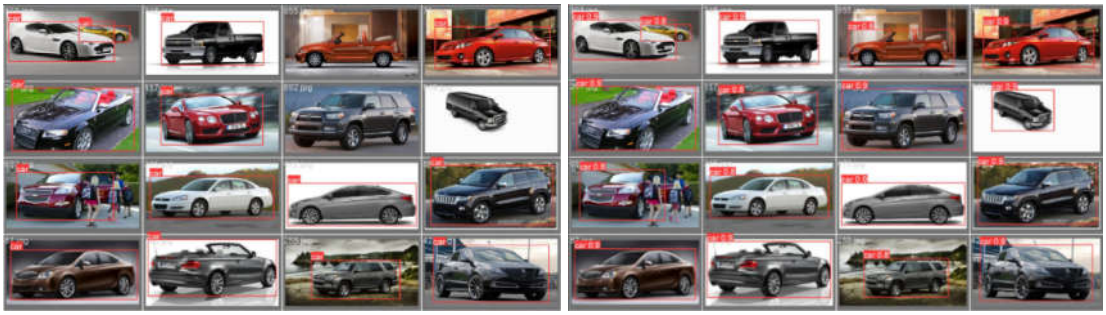
Để thực hiện việc huấn luyện mô hình, nghiên cứu đã thiết lập cài đặt một số phần mềm gồm: CUDA 12.1, Python 3.7.3 và môi trường để có thể thực thi được các dòng lệnh (Pycharm, VS Code). Sử dụng file Pretrain có sẵn của YOLOv8 để huấn luyện, sau đó lấy kết quả file weights/last.pt để thực hiện tiếp nếu như độ chính xác chưa đạt yêu cầu. Kết quả cuối cùng thu được là file được đặt tên là

best.pt, đây sẽ là file cuối cùng dùng để phát hiện đối tượng ô tô.

Cấu hình để huấn luyện càng cao thì tốc độ xử lý càng nhanh, nhóm nghiên cứu đã sử dụng cấu hình CPU Intel Core i7-10750H, Ram 16Gb, Card đồ hoạ Nvidia RTX 2060 6Gb Vram. Bên cạnh đó, nghiên cứu còn sử dụng thêm một số web có hỗ trợ GPU để đào tạo mô hình đó là Roboflow, Google Colab, Kaggle.



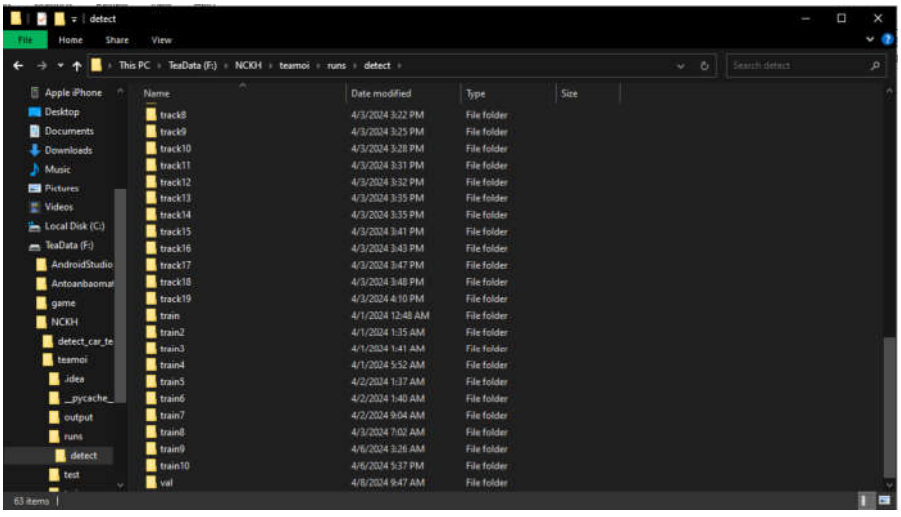
Hình 8: Quá trình huấn luyện dữ liệu trên cấu hình I7 10750H Ram 16Gb - GPU RTX 2060 6Gb



Hình 9: Kiểm tra lại dữ liệu đã gán nhãn

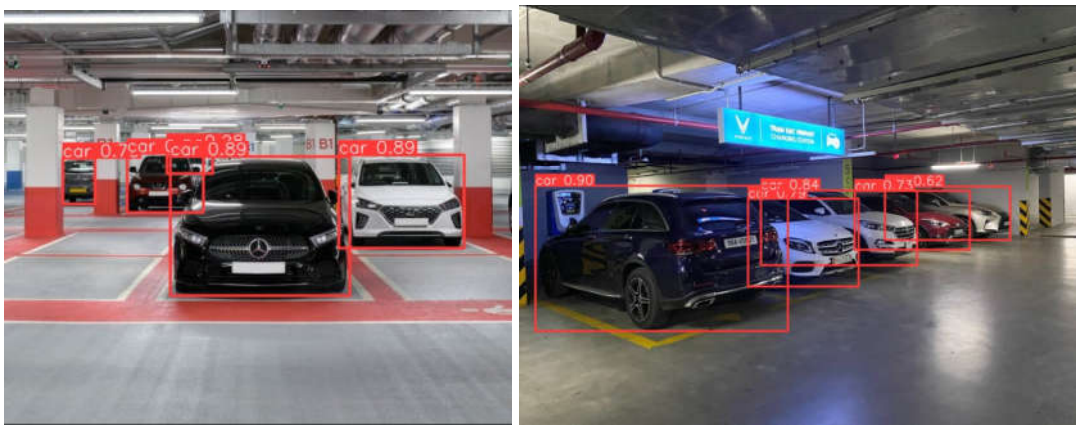
4.4. Đánh giá hiệu suất và kiểm tra mô hình

Kết quả huấn luyện mô hình thu được một thư mục *train* trong đường dẫn *runs/detect*. Trong đó có chứa tất cả các thông tin liên quan tới quá trình huấn luyện trước đó.



Hình 10: Kết quả sau khi huấn luyện

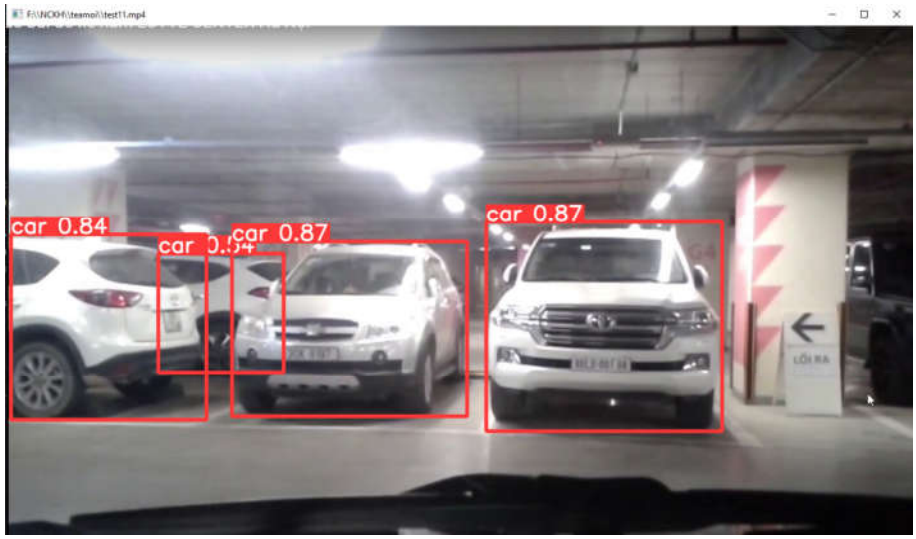
Lấy file best.pt trong thư mục train/weights để thử với các dữ liệu mới (trên hình ảnh, video hoặc là camera trực tiếp). Kết quả thu được như sau:



a) Phát hiện 4 ô tô *b) Phát hiện 5 ô tô*

Hình 11: Kết quả khi chạy mô hình YOLOv8 phát hiện đối tượng trên ảnh

Khi dữ liệu đầu vào là các hình ảnh trên video, tốc độ phát hiện vật thể khá nhanh, tuy nhiên độ chính xác chưa được cao như phát hiện vật thể trên ảnh. Hình 12 có 05 xe ô tô đang đỗ nhưng mô hình chỉ phát hiện được 04 xe ô tô.



Hình 12: Kết quả khi chạy mô hình YOLOv8 phát hiện đối tượng trên video

5. Kết luận

Nghiên cứu đã ứng dụng mô hình YOLOv8 trong phát hiện phương tiện giao thông là ô tô. Kết quả cho thấy, mô hình YOLOv8 nhận diện đối tượng xuất hiện trên ảnh tốt hơn so với đối tượng xuất hiện trên video hay camera. Kết quả của nghiên cứu có thể hỗ trợ công tác quản lý giao thông đường bộ như phát hiện và thống kê phương tiện giao thông là ô tô.

Lời cảm ơn: Nghiên cứu này là sản phẩm của đề tài nghiên cứu khoa học sinh viên Trường Đại học Tài nguyên và Môi trường Hà Nội năm học 2023 - 2024 *"Ứng dụng công nghệ GIS và mô hình YOLOv8 trong lựa chọn vị trí không gian và quản lý bãi đỗ xe thông minh"*.

TÀI LIỆU THAM KHẢO

[1]. Nguyễn Mạnh Cường, Vũ Văn Rực (2021). *Nghiên cứu thuật toán phân loại*

phương tiện giao thông dựa trên thị giác máy tính.

[2]. Lương Công Duẩn, Nguyễn Ngọc Minh (2021). *Ứng dụng học sâu (Deep Learning) cho bài toán nhận diện vật thể lạ trên đường băng*. Tạp chí Khoa học Công nghệ và Truyền thông, 04, 50 - 55.

[3]. Trà Văn Đồng (2020). *So sánh thuật toán SSD v*à YOLO.

[4]. Vũ Hà Anh (2021). *Xây dựng ứng dụng phát hiện lửa dựa trên mô hình học sâu trên Jetson Nano.* Khóa luận tốt nghiệp, Trường Đại học Công nghệ thông tin, Đại học Quốc gia Thành phố Hồ Chí Minh.

[5]. Ardeshir, S et al., (2014). *GIS-assisted object detection and geospatial localization*. Computer Vision - ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6 - 12, 2014, Proceedings, Part VI 13, Springer.

[6]. Futatsumori, S., Morioka, K., Kohmura, A., kada, K.vàYonemoto, N., (2016). *Detection characteristic evaluations of optically-connected wideband 96 GHz millimeter-wave radar for airport surface foreign object debris detection*. In Proceedings of the 41st International Conference on Infrared, Millimeter, and Terahertz waves, Copenhagen, Denmark, 25–30 September 2016, 1-2.

[7]. Li, J., Deng, G., Luo, C., Lin, Q., Yan, Q., Ming, Z., (2016). *A Hybrid Path Planning Method In Unmanned Air/Ground Vehicle (UAV/UGV) Cooperative Systems*. IEEE Trans. Veh. Technol, 65, 9585 - 9596.

[8]. Li, Y.vàXiao, G., (2011). *A new FOD recognition algorithm based on multi-source information fusion and experiment analysis*. Proc. SPIE.

[9]. Cheng-Jian Lin, Jyun-Yu Jhang (2022). *Intelligent traffic-monitoring system based on YOLO and convolutional fuzzy neural networks*. IEEE Access 10, 14120 - 14133.

[10]. Loey, M et al., (2021). *Fighting against COVID-19: A novel deep learning model based on YOLO-v2 with ResNet-50 for medical face mask detection*. Sustainable Cities and Society 65: 102600.

[11]. Mund, J., Zouhar, A., Meyer, L., Fricke, H., Rother, C., (2015). *Performance evaluation of LiDAR point clouds towards automated FOD detection on airport aprons.* In Proceedings of the 5th International Conference on Application and Theory of Automation In Command and Control Systems, Toulouse, France, 30th September - 2nd October 2015, 85 - 94.

[12]. Nguyễn Mạnh Cường, Vũ Văn Rực (2021). *Nghiên cứu thuật toán phân loại phương tiện giao thông dựa trên thị giác máy tính.*

[13]. Ölzen, B., Baykut, S., Tulgar, O., Belgül, A. U., Yalçin, I. K., Sahinkaya, D. S. A., (2017). *Foreign object detection on airport runways by mm-wave FMCW radar.* In Proceedings of the 25th IEEE Signal Processing and Communications Applications Conference, Antalya, Turkey 15 - 18 May 2017, 1 - 4.

[14]. Redmon, J et al., (2016). Y*ou only look once: Unified, real-time object detection*. Proceedings of the IEEE conference on computer vision and pattern recognition.

[15]. Zeitler, A., Lanteri, J., Pichot, C., Migliaccio, C., Feil, P., Menzel, W., (2010). *Folded reflectarrays with shaped beam pattern for foreign object debris detection on runways*. IEEE Trans. Antennas Propag, 58, 3065 - 3068.

[16]. Rosebrock, A., (2018). *YOLO object detection with OpenCV* [Online]. Available: https://www.pyimagesearch.com/2018/11/12/yolo-object-detection-with-opencv/.