

**ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN
KHOA KHOA HỌC MÁY TÍNH**



**BÁO CÁO ĐỒ ÁN MÔN HỌC
THỊ GIÁC MÁY TÍNH NÂNG CAO**

Giảng viên hướng dẫn: Ts. Mai Tiến Dũng

Thành viên nhóm:

Thạch Minh Hoàng - 22520477

Nguyễn Quốc Hưng - 22520516

Nguyễn Hữu Đức - 22520270

Thành phố Hồ Chí Minh, ngày 19 tháng 12 năm 2024

Mục lục

I.	Phân công công việc và đánh giá mức độ hoàn thành:	3
II.	Lý do chọn đề tài:	3
III.	Phát biểu bài toán:	4
IV.	Bộ dữ liệu thực nghiệm:	4
V.	Phương pháp thực nghiệm:	7
1.	DETR:	7
1.1	Giới thiệu mô hình:	7
1.2	Cấu trúc của DETR:	8
1.3	LOSS:	9
2.	Faster RCNN:	12
1.1	Giới thiệu mô hình:	12
1.2	Cấu trúc của Faster RCNN:	12
1.3	Faster R-CNN resnet50 fpn từ thư viện Pytorch:	14
1.4	Sử dụng Faster RCNN để thực nghiệm vào bài toán:	14
3.	YOLOv11:	15
VI.	Đánh giá và kết luận kết quả thực nghiệm	17
1.	Độ đo sử dụng:	17
1.1	Average Precision (AP):	17
1.2	Mean Average Precision (mAP50):	20
2.	Kết quả thực nghiệm và kết luận	20
3.	Demo kết quả	21
VIII.	Nguồn tham khảo	24

I. Phân công công việc và đánh giá mức độ hoàn thành:

Thành viên		Mức độ hoàn thành
Thạch Minh Hoàng	Thiết kế, chỉnh sửa, trình bày file báo cáo và file ppt; Thực nghiệm trên bộ dữ liệu với mô hình Faster RCNN, tiền xử lí dữ liệu với các ảnh từ cam05-cam 07	100%
Nguyễn Quốc Hưng	Thiết kế, chỉnh sửa, trình bày file báo cáo và file ppt; Thực nghiệm trên bộ dữ liệu với mô hình DETR, tiền xử lí dữ liệu với các ảnh từ cam08-cam 10	100%
Nguyễn Hữu Đức	Thiết kế, chỉnh sửa, trình bày file báo cáo và file ppt; Thực nghiệm trên bộ dữ liệu với mô hình Yolo, tiền xử lí dữ liệu với các ảnh từ cam01-cam04	100%

*Dữ liệu bao gồm các nhãn camX với X có nghĩa là số thứ tự của camera được lấy ảnh. Tổng tất cả trong bộ dữ liệu có 10 cam.

II. Lý do chọn đề tài:

- TÍNH THỰC TIỄN VÀ ÚNG DỤNG CAO: An toàn giao thông** (ứng dụng trong phát hiện vi phạm giao thông); **Quản lý giao thông thông minh** (quản lý lưu lượng, giảm ùn tắc và tối ưu hóa việc sử dụng cơ sở hạ tầng). **Tối ưu hóa hệ thống tín hiệu đèn giao thông:** Hiện nay TP HCM đang thử nghiệm điều chỉnh thời gian tín hiệu đèn dựa trên mật độ phương tiện để giảm thời gian chờ và tăng lưu lượng phương tiện qua nút giao thông.
- CƠ HỘI NGHIÊN CỨU VÀ PHÁT TRIỂN:** Những kết quả từ đồ án có thể được triển khai hoặc cải tiến để sử dụng trong thực tế hoặc làm nền tảng cho các nghiên cứu sâu hơn.
- TÍNH CẤP THIẾT XÃ HỘI:** Vấn đề giao thông là một trong những thách thức lớn đối với các thành phố hiện đại, đặc biệt ở Việt Nam, nơi giao thông phức tạp và mật độ phương tiện cao. Việc phát triển các mô hình phát hiện phương tiện giao thông có thể đóng góp vào việc giải quyết các vấn đề cấp bách của xã hội.

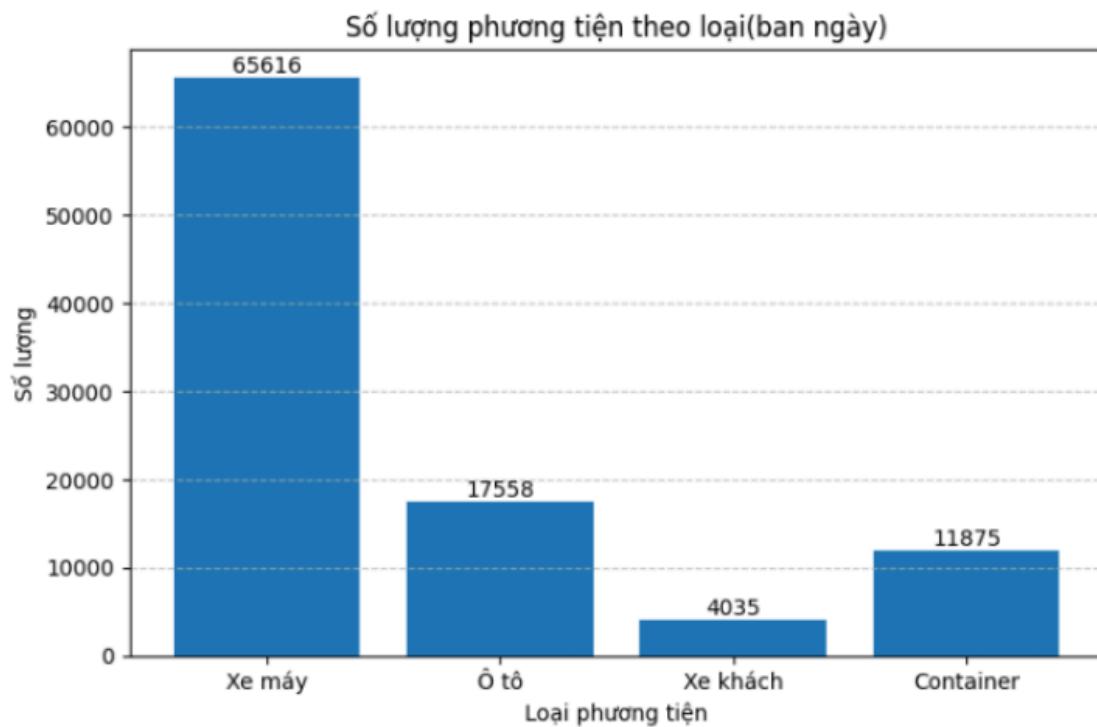
III. Phát biểu bài toán:

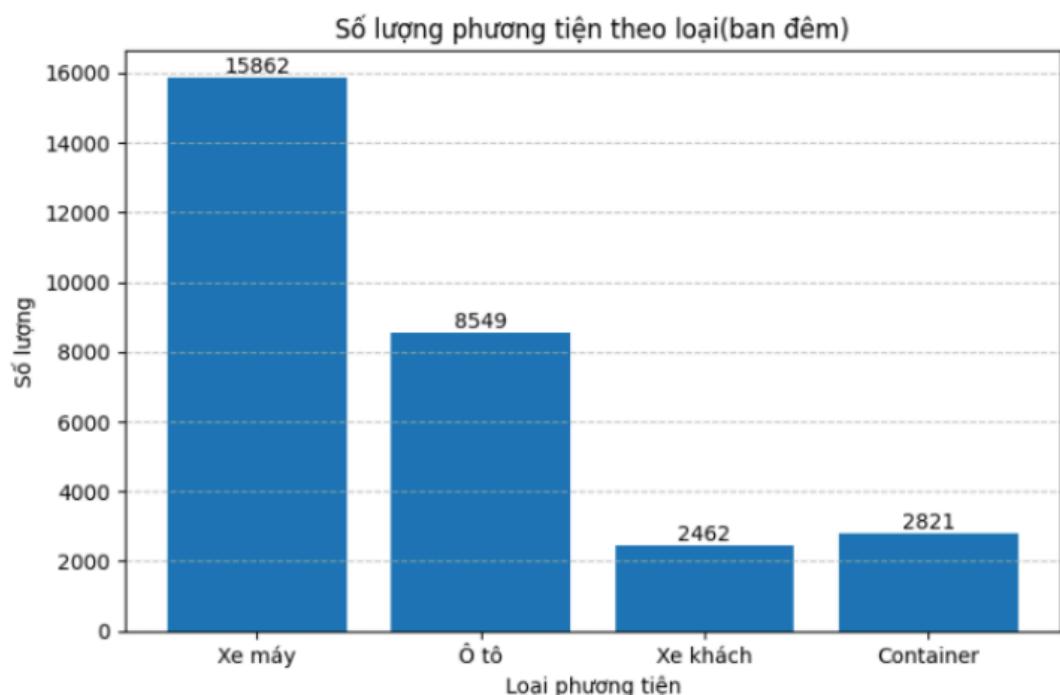
- Phát hiện phương tiện giao thông là bài toán trong lĩnh vực thị giác má
- y tính nhằm xác định vị trí, loại phương tiện và các thông tin liên quan trong ảnh hoặc video. Mục tiêu là phát triển một hệ thống có khả năng xử lý tự động dữ liệu hình ảnh/video để hỗ trợ giám sát, quản lý giao thông hoặc các ứng dụng thực tế khác.
- Dữ liệu đầu vào có thể là ảnh chụp cảnh giao thông, chứa các phương tiện (xe máy, ô tô, xe tải, xe buýt) từ đó đầu ra sẽ là một hoặc nhiều bounding box cho mỗi phương tiện trong ảnh/video, kèm theo nhãn phân loại phương tiện
- Hệ thống cần đạt được các yêu cầu sau:
 - Phát hiện chính xác: Xác định đúng vị trí các phương tiện có trong ảnh hoặc video. Phân loại chính xác: Xác định đúng loại phương tiện giao thông.
 - Tốc độ xử lý nhanh: Đảm bảo hệ thống hoạt động thời gian thực (real-time) hoặc gần thời gian thực đối với video.
 - Độ tin cậy cao: Giảm thiểu sai sót trong phát hiện và phân loại.

IV. Bộ dữ liệu thực nghiệm:

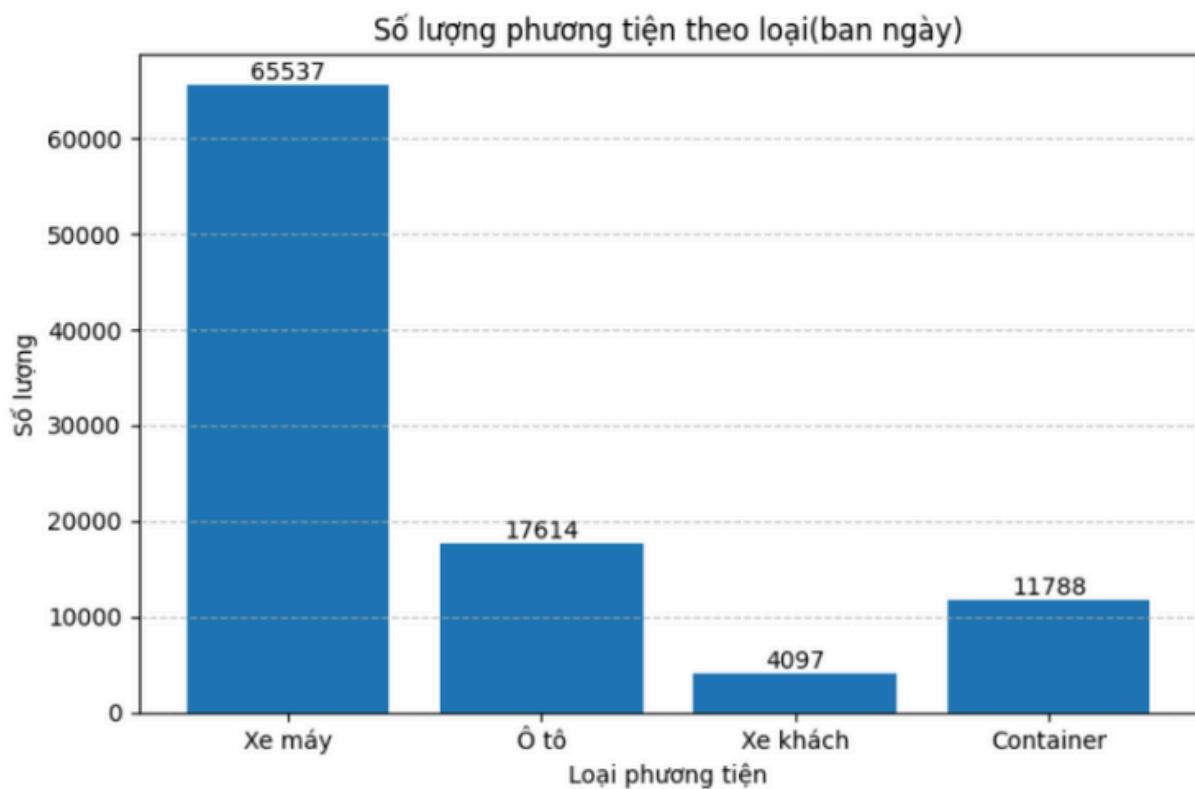
- Bộ dữ liệu được lấy từ cuộc thi SoICT Hackathon 2024 - Traffic Vehicle Detection tập trung vào giải quyết bài toán "nhận diện loại phương tiện giao thông ở Việt Nam".
- Đầu vào cho mô hình là các ảnh thô chưa được gán nhãn. Tệp nhãn là các file định dạng .txt. Mỗi dòng của tệp nhãn chứa thông tin là tên ảnh và nhãn của văn bản chứa trong ảnh đó. Thông tin trong các tệp về bounding boxes ở định dạng YOLO.
- Tập dữ liệu thật có gán nhãn, dùng để huấn luyện mô hình. Tập này gồm 11022 ảnh (6392 ảnh vào ban ngày, 4630 ảnh vào ban đêm).
- Dữ liệu được gán nhãn theo quy tắc:

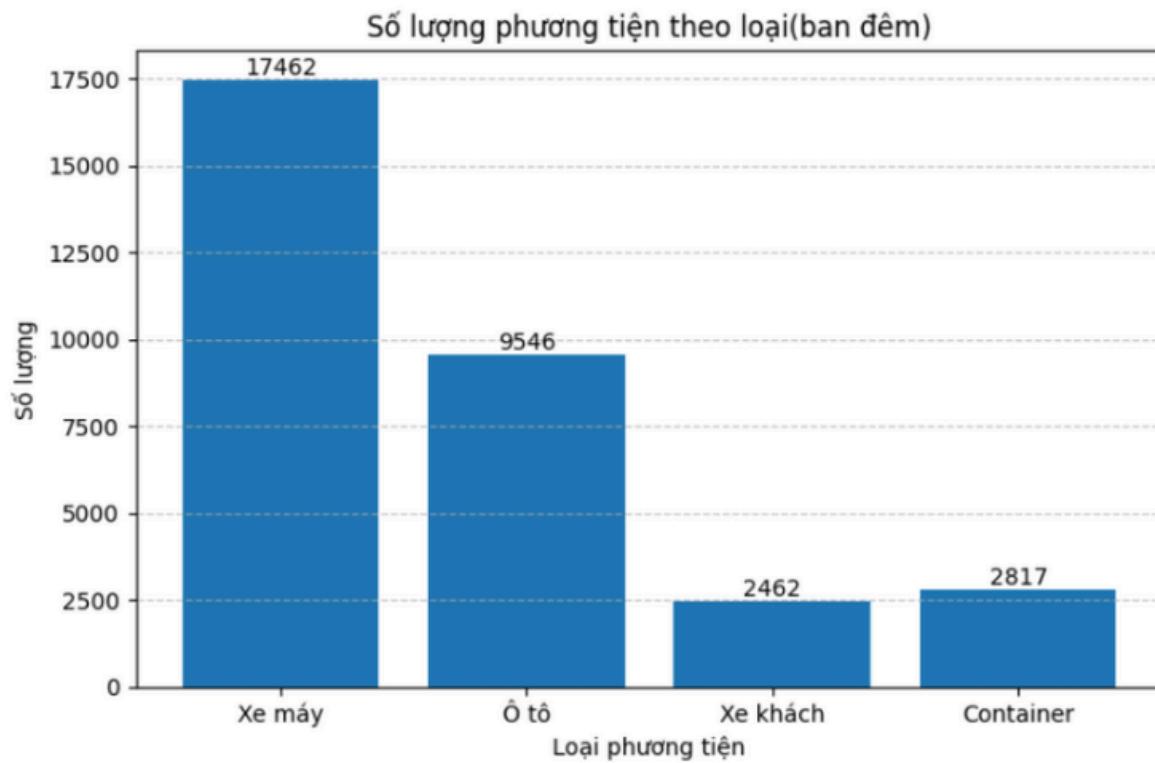
- Nhãn 0: xe máy.
 - Nhãn 1: xe ô tô con.
 - Nhãn 2: Xe vận tải du lịch (xe khách).
 - Nhãn 3: Xe vận tải container.
 - Các ảnh xe ở xa, có điểm ảnh quá mờ hoặc không thể nhìn bằng mắt thường có thể không được được đánh nhãn.
 - Các loại xe bị cải tạo quá 70% so với hình dạng ban đầu (như xe thồ,...) sẽ được đánh nhãn 1.
 - Các loại xe điện (xe đạp điện, máy điện, XanhSM, ...) hoặc xe phân khối lớn sẽ được đưa vào nhóm 1 (Ngoài ra đối với dữ liệu ban đêm nhãn dữ liệu sẽ tương ứng 4->0, 5->1, 6->2, 7->3).
- Xử lý dữ liệu: Tiến hành kiểm tra các ảnh xem có nhãn nào bị gán sai hoặc gán thiếu không để xử lý thêm các bounding box vào hoặc đổi nhãn cho các bounding box, dữ liệu có sự thay đổi như sau:
- Trước khi xử lý, dữ liệu phân bố như sau:





- Sau khi xử lý, dữ liệu phân bố như sau:





- ⇒ Từ việc phân tích dữ liệu như trên, ta có thể thấy số lượng phương tiện xe máy không được gán nhãn vào ban đêm khá lớn, đồng thời cũng có sự phân bố dữ liệu không đều giữa các lớp khi sự chênh lệch giữa lớp xe khách và xe máy là khá lớn

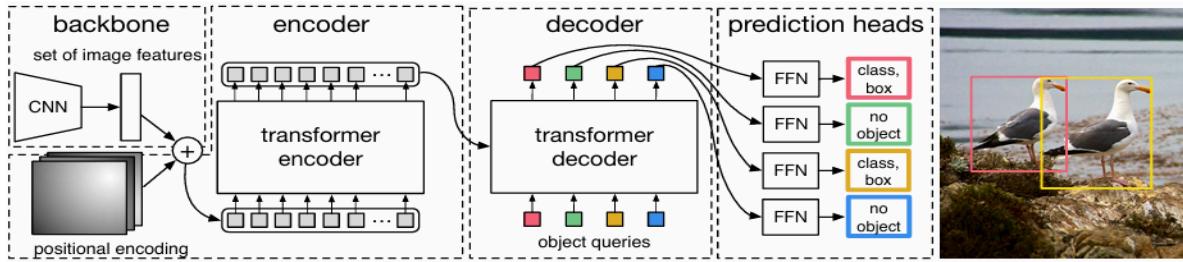
V. Phương pháp thực nghiệm:

1. DETR:

1.1 Giới thiệu mô hình:

- DETR (DEtection TRansformer) là một mô hình phát hiện đối tượng do nhóm nghiên cứu của Facebook AI (Meta AI) phát triển và lần đầu tiên được giới thiệu trong bài báo "End-to-End Object Detection with Transformers" vào năm 2020.
- DETR tận dụng sức mạnh của Transformer - một kiến trúc ban đầu được phát triển cho các bài toán xử lý ngôn ngữ tự nhiên (NLP) - để thực hiện phát hiện đối tượng một cách "end-to-end".

1.2 Cấu trúc của DETR:



Ảnh: Mô tả cấu trúc của DETR

- Backbone: Là một thành phần quan trọng, hoạt động như một bộ trích xuất đặc trưng từ ảnh đầu vào. Backbone thường sử dụng một mạng CNN thông thường (ResNet), với nhiệm vụ giảm độ phân giải của ảnh trong khi giữ lại các đặc trưng quan trọng để phát hiện đối tượng
- Encoder - transformer
 - Giảm chiều kênh đầu vào: Trước khi đưa vào Transformer, số kênh được giảm xuống một giá trị nhỏ hơn để giảm độ phức tạp tính toán. Việc này thường được thực hiện thông qua một convolution 1x1
 - Chuyển đổi không gian thành chuỗi: Transformer yêu cầu đầu vào là một chuỗi. Vì vậy đặc trưng không gian $dxHxW$ được làm phẳng thành một chuỗi có kích thước $dx(HxW)$. Mỗi vị trí không gian (x,y) trong đặc trưng ban đầu trở thành một phần tử trong chuỗi
 - Bổ sung mã hóa vị trí: Transformer không có thông tin vị trí tự nhiên vì Self-Attention chỉ dựa trên mối quan hệ giữa các phần tử. Vì vậy, Positional Encoding được thêm vào để giúp mô hình nhận biết vị trí không gian
 - Cấu trúc của mỗi lớp encoder:
 - Multi-Head Self-Attention (MHA): Cho phép mỗi phần tử trong chuỗi nhìn toàn bộ các phần tử khác để học mối quan hệ toàn cục. Kết quả là mỗi phần tử đầu ra có thông tin tổng hợp từ toàn bộ các phần tử đầu vào.

- Feed-Forward Network (FFN): Là một mạng nơ-ron hai lớp với hàm kích hoạt ReLU dùng để xử lý phi tuyến và tăng khả năng biểu diễn.
- Residual Connection: Thêm đầu vào ban đầu vào đầu ra để tránh mất thông tin
- Layer Norm: Ổn định quá trình huấn luyện
- Decoder - transformer:
 - Self-Attention: Mỗi đầu vào trong Decoder tương tác với tất cả các đầu vào khác để học mối quan hệ giữa chúng giúp cho Decoder hiểu được mối liên kết giữa các object queries
 - Cross-Attention: Kết hợp đầu ra từ Encoder với các object queries giúp các object queries thu thập thông tin cần thiết từ đặc trưng ảnh để dự đoán đối tượng.
 - Feed-Forward Network (FFN): là một mạng nơ-ron hai lớp với hàm kích hoạt ReLU dùng để xử lý phi tuyến để tăng cường khả năng biểu diễn và tạo ra embedding hoàn thiện
 - Residual Connection + LayerNorm: Có tác dụng giữ lại thông tin ban đầu, cải thiện khả năng hội tụ và ổn định đầu ra.
- Feed-Forward network: Là một mạng nơ-ron đa tầng được áp dụng độc lập cho từng đầu ra của Object Queries từ Decoder Dùng để dự đoán nhãn lớp và bounding box.

1.3 LOSS

Trong DETR, Loss được thiết kế để tối ưu hóa hai khía cạnh chính:

1. Nhãn lớp (Class Labels): Xác định loại đối tượng (hoặc "no object").
2. Bounding Boxes: Dự đoán vị trí và kích thước của các hộp giới hạn (Bounding Boxes).

Loss tổng hợp được tính dựa trên kết quả Hungarian Matching, đảm bảo ánh xạ giữa dự đoán và ground truth là duy nhất.

a. Tổng hợp Loss

Loss trong DETR được định nghĩa như sau:

$$\mathcal{L} = \mathcal{L}_{\text{class}} + \lambda_{\text{box}} \mathcal{L}_{\text{box}} + \lambda_{\text{giou}} \mathcal{L}_{\text{giou}}$$

- $\mathcal{L}_{\text{class}}$: Class Loss (dự đoán nhãn lớp).
- \mathcal{L}_{box} : L_1 Loss (so sánh tọa độ hộp giới hạn).
- $\mathcal{L}_{\text{giou}}$: Generalized IoU Loss (đánh giá mức độ chồng lấp và bao phủ hộp giới hạn).
- $\lambda_{\text{box}}, \lambda_{\text{giou}}$: Trọng số (thường $\lambda_{\text{box}} = \lambda_{\text{giou}} = 1$).

b. Class Loss

- Mục đích: So sánh nhãn lớp dự đoán với ground truth, bao gồm cả lớp đặc biệt "no object" (\emptyset).
- Công thức:

$$\mathcal{L}_{\text{class}} = - \sum_{i=1}^N \sum_{c=1}^{C+1} y_{i,c} \log(p_{i,c})$$

- N : Số lượng Object Queries.
- $C + 1$: Số lớp (bao gồm lớp "no object").
- $y_{i,c}$: Nhãn lớp ground truth (one-hot vector).
- $p_{i,c}$: Xác suất dự đoán của lớp c cho query i .
- Ý nghĩa:
 - Phạt nếu xác suất dự đoán của lớp đúng thấp.
 - Phạt cao hơn nếu dự đoán sai lệch nhiều.

c. Bounding Box Loss

- L1 Loss

a. Mục đích: Tính khoảng cách tuyệt đối giữa tọa độ dự đoán và ground truth.

a. Công thức:

$$\mathcal{L}_{\text{box}} = \frac{1}{N} \sum_{i=1}^N \sum_{j \in \{x,y,w,h\}} |b_{ij} - \hat{b}_{ij}|$$

- b_{ij} : Tọa độ ground truth (x, y, w, h).
- \hat{b}_{ij} : Tọa độ dự đoán (x, y, w, h).

c. Ý nghĩa:

- Giảm sai số trong từng tọa độ dự đoán.

- **Generalized IoU Loss**

a. Mục đích: Đánh giá mức độ chồng lấp giữa hộp dự đoán và hộp thực, đồng thời phạt nếu hộp dự đoán không bao phủ tốt hộp thực.

a. Công thức:

$$\mathcal{L}_{\text{giou}} = 1 - \text{IoU} + \frac{\text{Area}(C - U)}{\text{Area}(C)}$$

- IoU : Intersection over Union (phần giao trên phần hợp của hai hộp).
- C : Hộp bao phủ nhỏ nhất chứa cả hộp dự đoán và hộp thực.
- U : Phần hợp của hai hộp.

c. Ý nghĩa:

- Phạt các hộp dự đoán không chỉ dựa trên vùng chồng lấp, mà còn tính đến cách hộp dự đoán bao phủ hộp thực.

d. **Hungarian Matching**

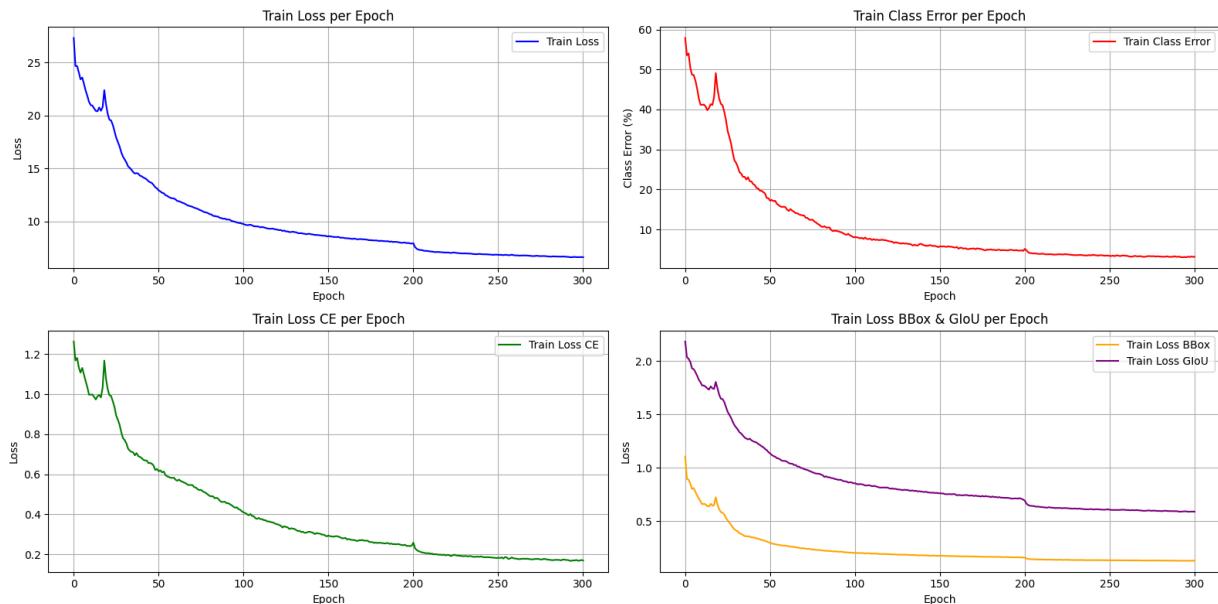
Trước khi tính toán Loss, **Hungarian Matching** được sử dụng để ánh xạ duy nhất giữa các dự đoán và ground truth.

Ý nghĩa:

- Đảm bảo mỗi đối tượng chỉ được ánh xạ bởi một Object Query.
- Tăng hiệu quả tối ưu hóa, tránh tình trạng dự đoán trùng lặp.

1.4 Sử dụng DETR để thực nghiệm bài toán

- Dữ liệu ảnh được reshape lại ở kích thước 800x800, đồng thời định dạng tọa độ bounding box được chuyển từ YOLO sang COCO bằng Roboflow.
- Lựa chọn tham số huấn luyện:
 - o Với các tham số như:
 - epochs 300
 - dataset_file custom
 - num_classes 8
 - num_queries 50
 - batch_size 4



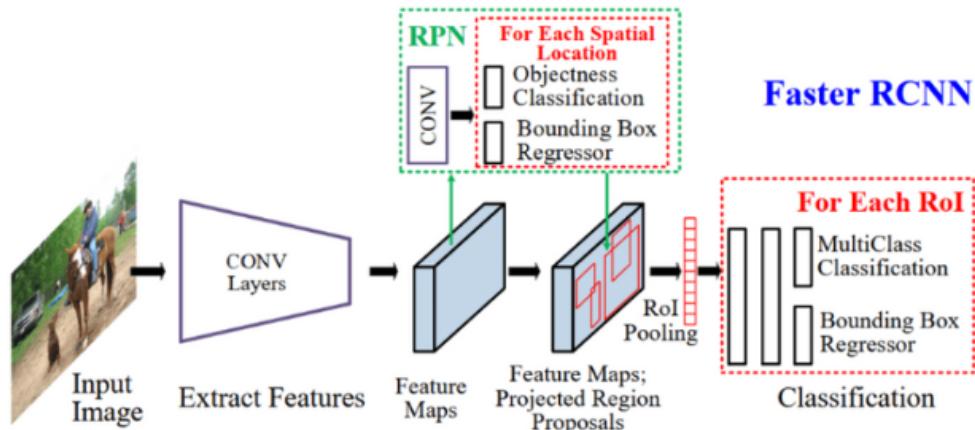
Như ta có thể thấy các loss giảm dần cho thấy nó hội tụ tốt cho việc dự đoán.

2. Faster RCNN:

1.1 Giới thiệu mô hình:

- Faster R-CNN được giới thiệu lần đầu tiên tại hội nghị Neural Information Processing Systems (NeurIPS) vào năm 2015 trong bài báo Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks bởi nhóm tác giả: Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun.
- Faster R-CNN là sự kết hợp của 2 module: RPN (Region Proposal Network) sẽ tạo ra các vùng đề xuất (proposal regions) và module Fast R-CNN detector sẽ sử dụng các vùng đề xuất đó.

1.2 Cấu trúc của Faster RCNN:

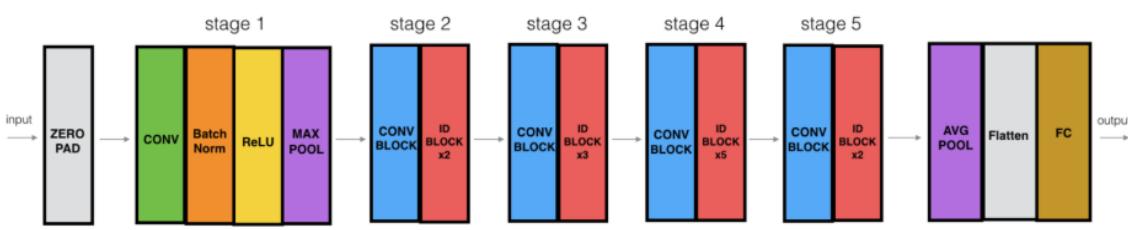


Ảnh: Mô tả cấu trúc của Faster RCNN

Backbone (Convolutional Network)

- Đây là mạng học sâu (thường dùng ResNet, VGG) để trích xuất feature map từ ảnh đầu vào.
- Feature map được sử dụng để giúp RPN thực hiện nhiệm vụ phát hiện đối tượng (Region Proposal). Chất lượng feature map ảnh hưởng trực tiếp đến hiệu quả và độ chính xác của việc đề xuất vùng trong Faster R-CNN.
- Trong bài báo mạng nơ-ron được sử dụng để trích xuất đặc trưng là VGG16, tuy nhiên backbone được sử dụng trong mô hình để giải quyết bài toán lần này là Resnet50 kết hợp fpn

Mạng nơ-ron Resnet50



Ảnh: Mô tả cấu trúc của mạng nơ-ron Resnet50

- Được giới thiệu trong bài báo khoa học nổi tiếng "Deep Residual Learning for Image Recognition", được công bố bởi nhóm tác giả Kaiming He, Xiangyu Zhang, Shaoqing Ren, và Jian Sun tại hội nghị CVPR (Conference on Computer Vision and Pattern Recognition) năm 2016.
- Tuy nhiên lớp FC, Flatten cuối cùng sẽ bị loại bỏ vì mục đích của backbone là để trích xuất feature map, không phải phân loại.

Region Proposal Networks (RPN)

- Tạo các vùng đề xuất (region proposals), tức là các vùng có khả năng chứa các đối tượng. Các vùng này sau đó được sử dụng bởi mạng con phân loại và hồi quy để xác định nhãn và vị trí chính xác của đối tượng.
- Đầu vào của RPN là đặc trưng đầu ra từ backbone (ví dụ: ResNet, VGG16), thường có dạng tensor $H \times W \times C$.

ROI Pooling

- ROI pooling là một dạng của **pooling layer**. Điểm khác so với max pooling hay average pooling là bất kể kích thước của tensor input, ROI pooling luôn cho ra output có kích thước cố định được định nghĩa trước.

Fully Connected Layer & Classifier

- Fully Connected Layer và Classifier nằm ở giai đoạn cuối của pipeline chúng chịu trách nhiệm phân loại đối tượng được phát hiện và dự đoán bounding box regression.

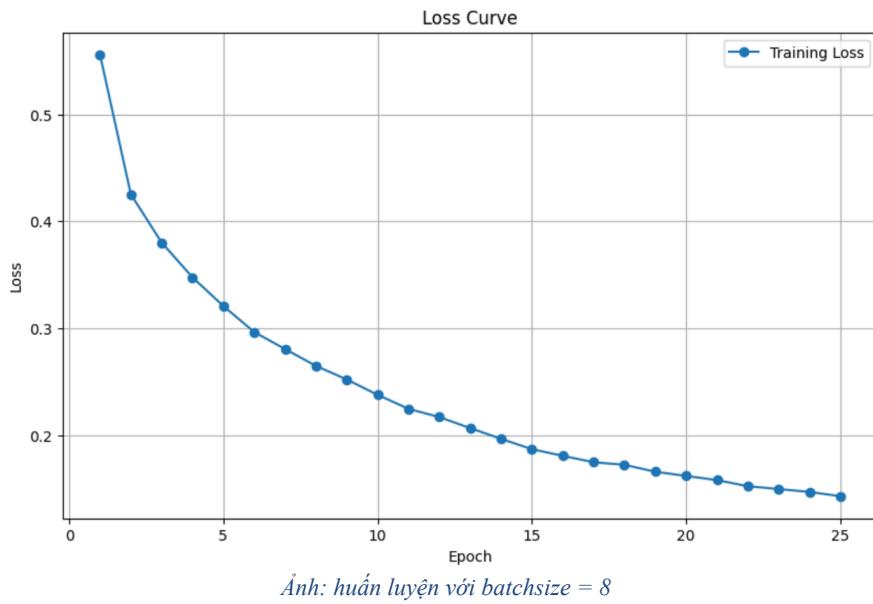
1.3 Faster R-CNN resnet50 fpn từ thư viện Pytorch:

- Ở mô hình này có sự khác biệt so với mô hình gốc ở chỗ ở giai đoạn backbone có thêm một mạng trích xuất đặc trưng là FPN (Feature Pyramid Network).
- FPN là một kiến trúc được sử dụng để tăng khả năng phát hiện đối tượng ở nhiều kích thước khác nhau. Thay vì chỉ sử dụng một feature map duy nhất, FPN tạo ra một loạt các tầng đặc trưng ($P_2, P_3, P_4, P_5, \dots$) tương ứng với các mức độ phân giải giảm dần.
- **Input** là các feature maps từ backbone (ví dụ: ResNet-50) ở các tầng C_k khác nhau: C_2, C_3, C_4, C_5 (từ các block cuối của backbone) (C có kích thước giảm dần khi k tăng). **Output**: Một loạt các feature maps (P_2, P_3, P_4, P_5): Kích thước không gian (H, W) giảm dần, nhưng mỗi tầng có số kênh cố định (thường là 256 kênh).
- Ngoài ra còn có sự khác biệt trong tầng lấy Pooling, ở bài báo 2015, ROI Pooling được sử dụng để trích xuất đặc trưng từ các đầu vào, tuy nhiên mô hình này sử dụng ROI Align có nhiều sự cải tiến so với ROI pooling giúp tăng độ chi tiết khi trích xuất các đặc trưng.

1.4 Sử dụng Faster RCNN để thực nghiệm vào bài toán

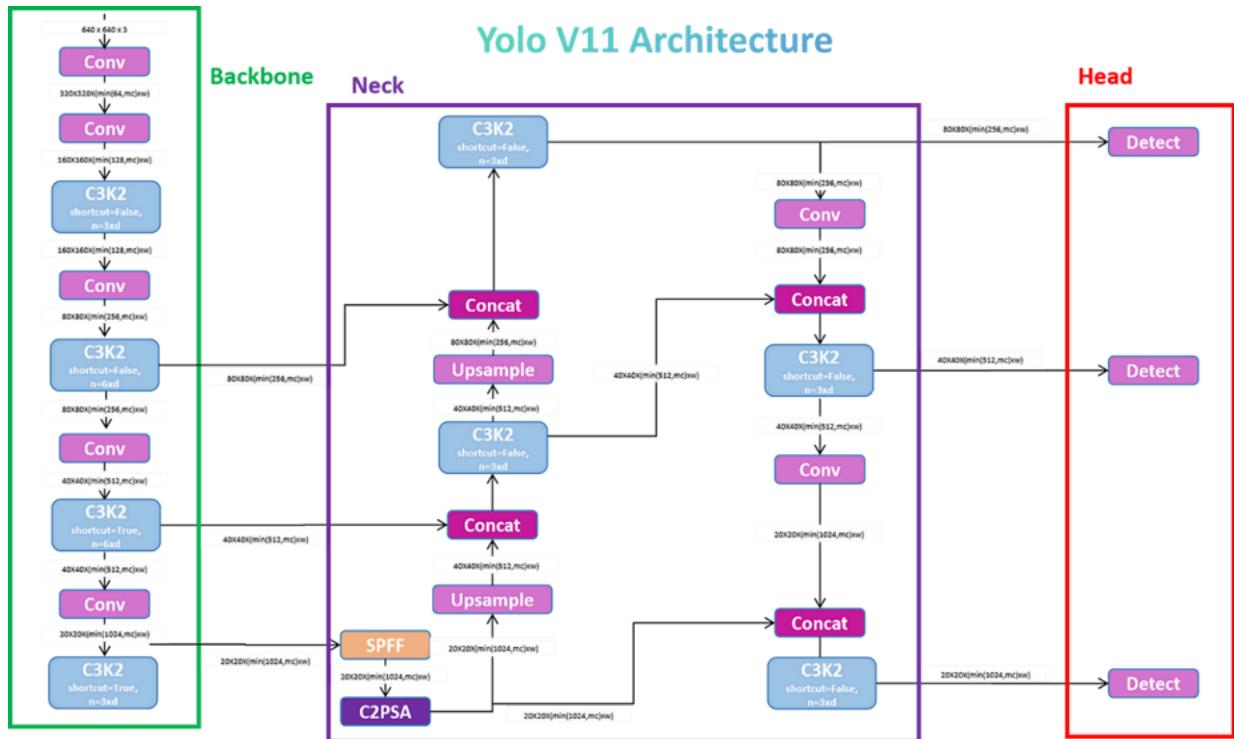
- Dữ liệu ảnh được reshape lại ở kích thước 224×224 , đồng thời định dạng tọa độ bounding box được chuyển từ YOLO sang pascal VOC.
- Tiến hành huấn luyện mô hình với đầu ra của mô hình số lớp (classes) là 9 (8 lớp cho đối tượng phát hiện-foreground và 1 lớp cho nền-background).
- Lựa chọn tham số huấn luyện:
 - Với các tham số như:
 - Optimizer: Adam
 - Epochs = 25
 - Learning rate = $1e-4$
 - Thủ nghiệm batch size với các kích thước: 6, 8, 10, 12, Hầu hết đều có loss giảm sau nhiều lần lặp, tuy nhiên với batchsize là 8 thì đồng

thời test trên 7 epochs khi so với batchsize là 6, 10, và 12 thì batch size là 8 cho kết quả tốt nhất.



3. YOLOv11

- Kiến trúc cơ bản của YOLOv11:



Ảnh: Mô tả cấu trúc của YOLO

Chia thành ba phần chính:

- Backbone
- Backbone có nhiệm vụ trích xuất các đặc trưng cơ bản từ ảnh đầu vào. Các lớp Convolutional (Conv) được sử dụng để thực hiện việc trích xuất các đặc trưng này, bao gồm việc giảm dần độ phân giải của ảnh qua các lớp.
 - Các lớp Conv như C3K2 thực hiện các phép toán chập (convolution) để trích xuất thông tin về hình dạng, kết cấu và các đặc điểm quan trọng khác của đối tượng trong ảnh..

Neck

- Neck có nhiệm vụ kết hợp và xử lý các đặc trưng từ Backbone để tạo ra các đặc trưng có độ phân giải khác nhau, phục vụ cho quá trình phát hiện đối tượng.

- Các lớp như Upsample và Concat thực hiện việc kết hợp các đặc trưng với nhau. Upsample giúp tăng kích thước của các đặc trưng để cải thiện độ phân giải, trong khi Concat kết hợp các đặc trưng từ các lớp khác nhau để làm giàu thêm thông tin.

- C3K2 tiếp tục thực hiện các phép toán chập, giúp trích xuất các đặc trưng ở mức độ cao hơn.

- Head

- Head là phần cuối cùng của mô hình, nơi thực hiện việc phát hiện đối tượng. Các lớp Detect có nhiệm vụ nhận các đặc trưng đã qua xử lý và đưa ra dự đoán về vị trí và loại đối tượng trong ảnh.

- Mỗi lớp Detect có thể dự đoán các bounding box và các nhãn lớp tương ứng cho từng đối tượng trong ảnh.

VI. Đánh giá và kết luận kết quả thực nghiệm.

1. Độ đo sử dụng:

1.1 Average Precision (AP):

- Là một độ đo phổ biến được dùng để đánh giá hiệu suất của các mô hình phân loại nhị phân, phân loại đa lớp, và phát hiện vật thể.
- Average precision phần diện tích dưới precision-recall curve sau đó tính toán nội suy tại 11 điểm (từ 0 đến 1, mỗi bước cách nhau 0.1).
- Trong bài toán này, ta chỉ xét đến các trường hợp sau của bounding boxes:
 - True Positive khi: Confident score lớn hơn confident threshold (tức là đã dự đoán đúng đối tượng cần dự đoán và Chỉ số IoU giữa Predict bounding box và ground truth bounding box lớn hơn IoU threshold).
 - False Positive khi: model phát hiện vật thể đó với một confident score rất cao, nhưng thực chất không có ground truth bounding box

nào tại đó, vì thế IoU sẽ là 0; hoặc IoU nhỏ hơn threshold; hoặc predict bounding box khớp hoàn toàn với ground truth bounding box, tuy nhiên nhãn dự đoán sai.

- False Negative khi: Trong ảnh thực chất là có vật thể, tuy nhiên model không thể phát hiện ra nó.
- Precision và recall được tính theo công thức:

$$\text{Precision} = \frac{\text{Số True Positive (TP)}}{\text{Tổng số dự đoán là Positive (TP+FP)}}$$

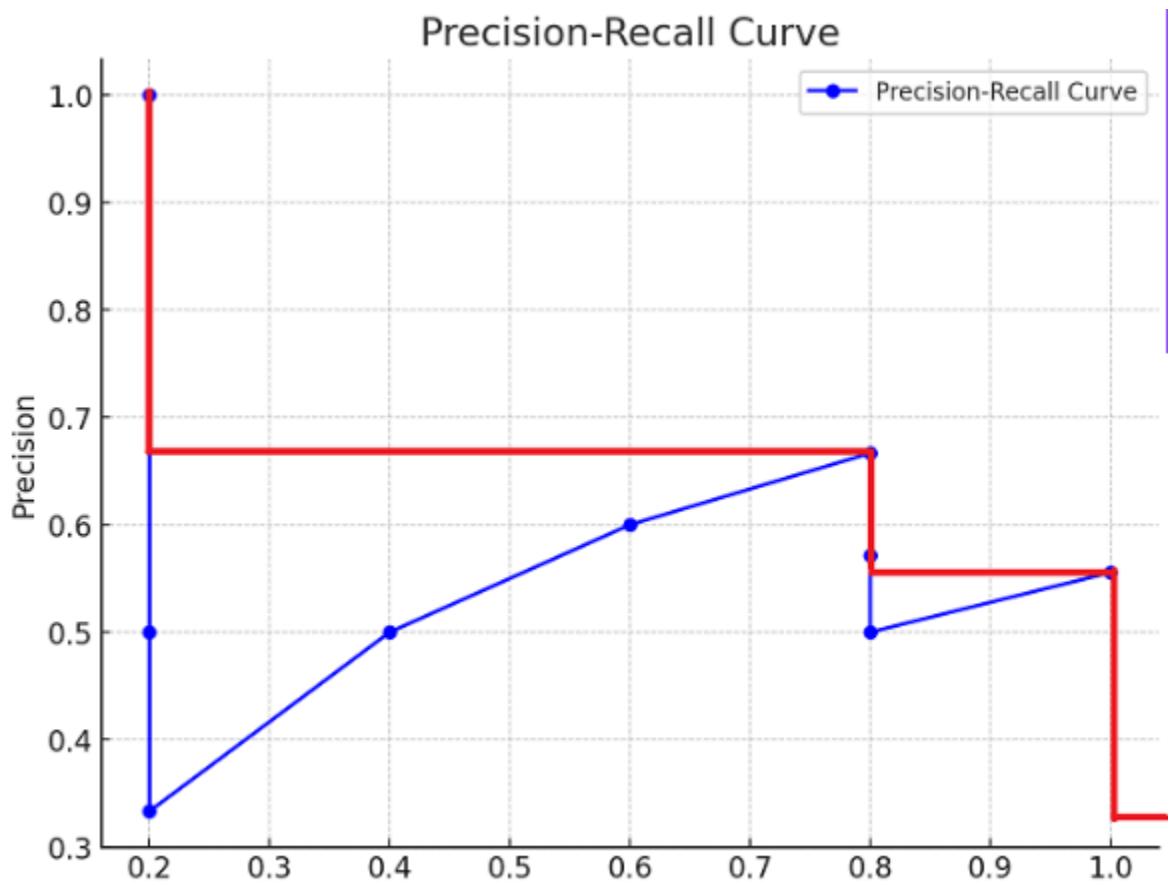
$$\text{Recall} = \frac{\text{Số True Positive(TP)}}{\text{Tổng số thực tế của Positive}}$$

- Giả sử ta có dữ liệu như bảng sau, tổng số dữ liệu positive thực tế là 5:

TP/FP	Accumulated TP	Accumulated FP	Precision	Recall
TP	1	0	1/1	1/5
FP	1	1	1/(1+1)	1/5
FP	1	2	1/(1+2)	1/5
TP	2	2	2/(2+2)	2/5
TP	3	2	3/(3+2)	3/5
TP	4	2	4/(4+2)	4/5
FP	4	3	4/(4+3)	4/5
FP	4	4	4/(4+4)	4/5
TP	5	4	5/(5+4)	5/5

Ảnh: Minh họa từng bước tính toán cho Precision và Recall theo True Positive tích lũy và False Positive tích lũy

- Vẽ Precision-Recall Curve, ta có ảnh sau:



Ảnh: Minh họa biểu đồ thể hiện Precision-Recall Curve

- Từ việc tính toán nội suy tại 11 điểm như trên theo công thức:

$$AP = \frac{1}{11} \sum_{r \in \{0, 0.1, \dots, 1\}} \rho_{\text{interp}}(r)$$

$$\rho_{\text{interp}} = \max_{\tilde{r}: \tilde{r} \geq r} \rho(\tilde{r})$$

- Ta có kết quả của ví dụ trên như sau:

$$AP = \frac{1}{11} (7 * 0.67 + 0.56 * 2)$$

$$AP = 0.5282$$

- Tương tự, áp dụng cách tính này cho từng lớp của bài toán

1.2 Mean Average Precision (mAP50):

- Độ đo được sử dụng trong bài toán là mAP50. Độ đo này được tính bằng cách trung bình cộng của các giá trị AP các lớp với IoU threshold là 0.5.

2. Kết quả thực nghiệm và kết luận

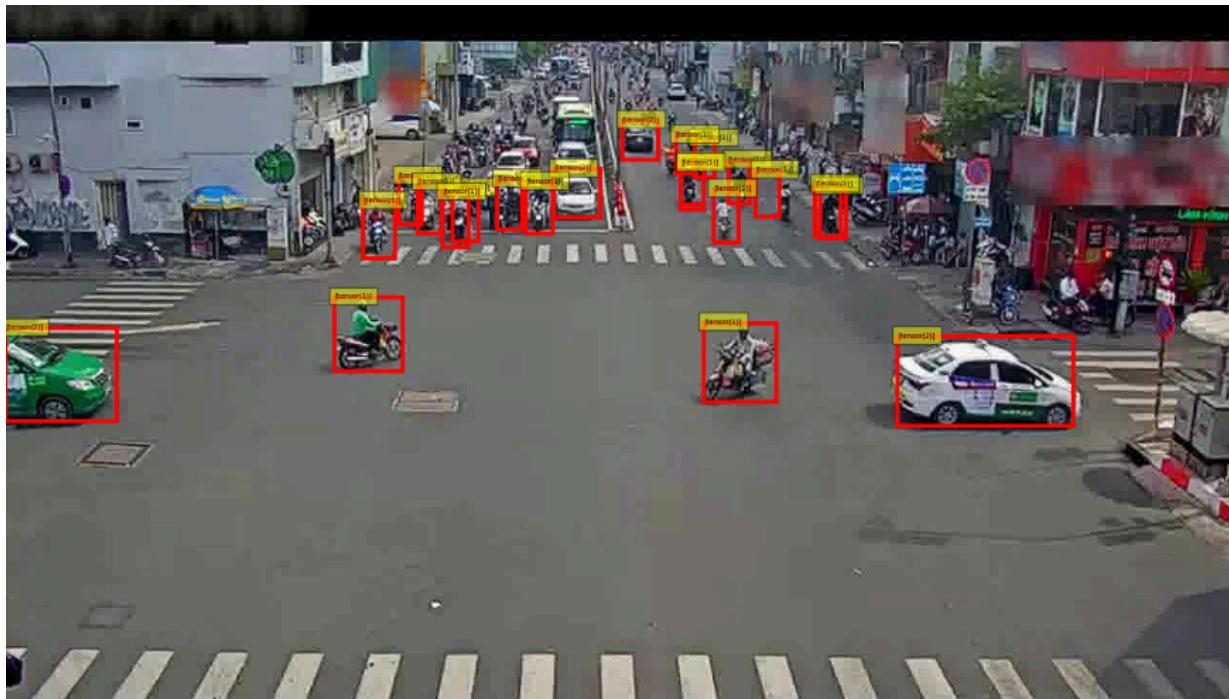
	YOLO	DETR	Faster RCNN
mAP	0.407	0.2362	0.2432
Thời gian dự đoán của mô hình	0.7096	0.2685	0.1204

Bảng thể hiện kết quả thực nghiệm của các mô hình khi được so sánh với nhau

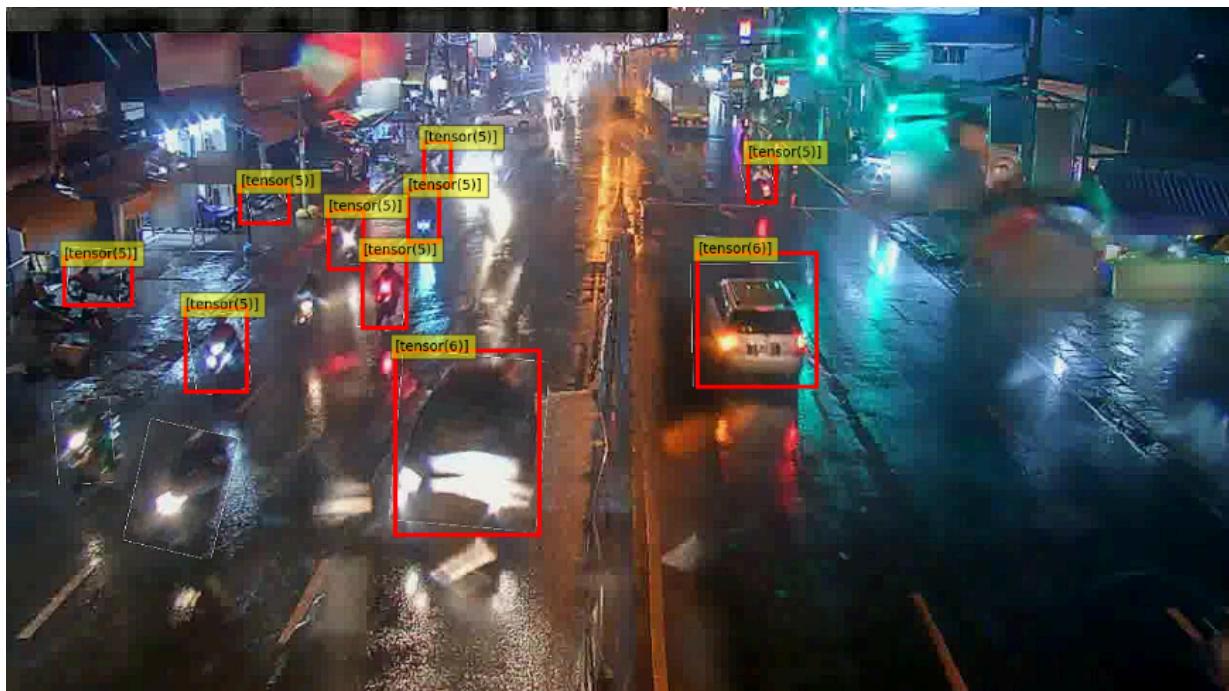
Kết luận:

- **Ưu điểm:** Thông qua kết quả thực nghiệm cho thấy, YOLO hoạt động tốt hơn khi so với DETR hay Faster RCNN, tuy nhiên Faster RCNN có thời gian dự đoán ngắn nhất. Kết quả dự đoán trả về trực quan, có thể chấp nhận đc
- **Nhược điểm:** Trong bài thực nghiệm này tuy đã có các phương pháp tiền xử lý dữ liệu, tuy nhiên vẫn còn thiếu các bước tăng cường để xử lý các ảnh ban đêm, vì thế hiệu suất của các mô hình không được cao, cần có cải tiến thêm.

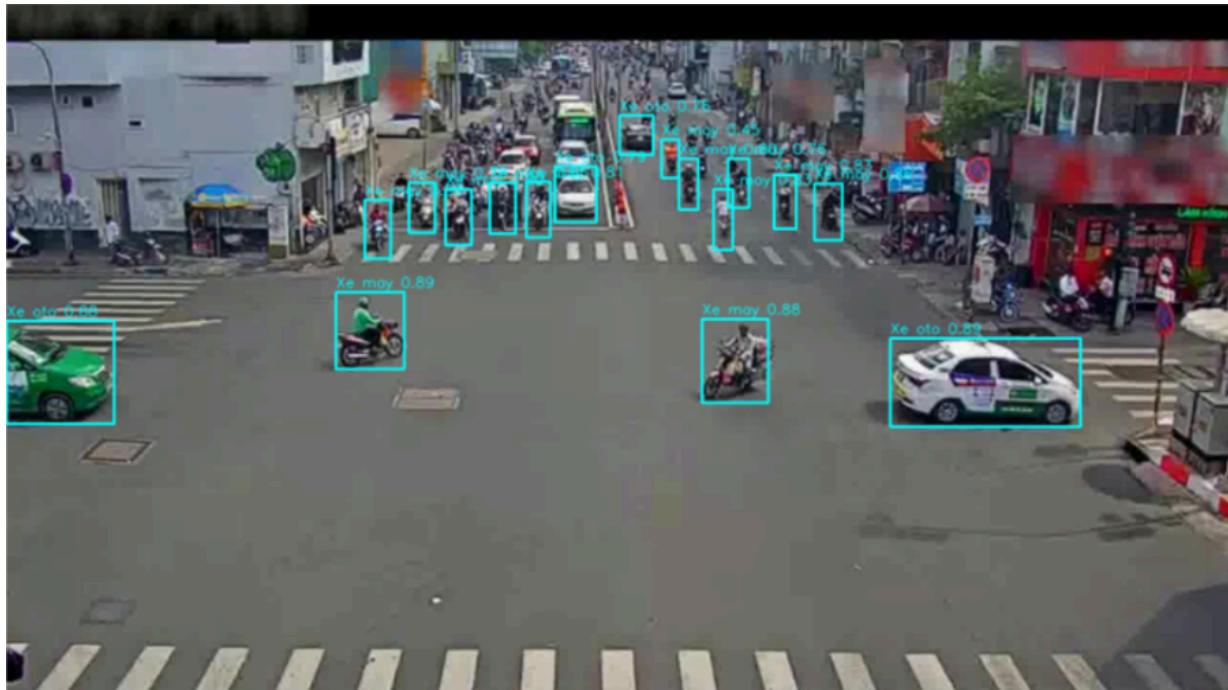
3. Demo kết quả



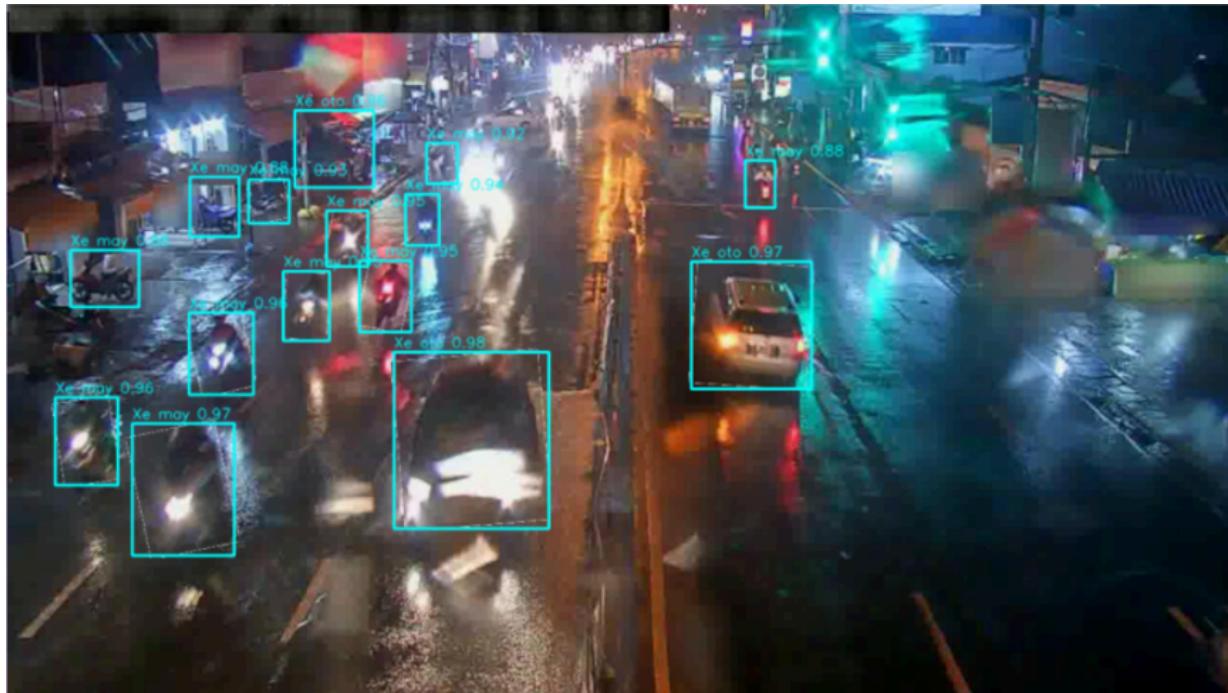
Ảnh: DETR phát hiện phương tiện giao thông vào ban ngày tại đường phố Việt



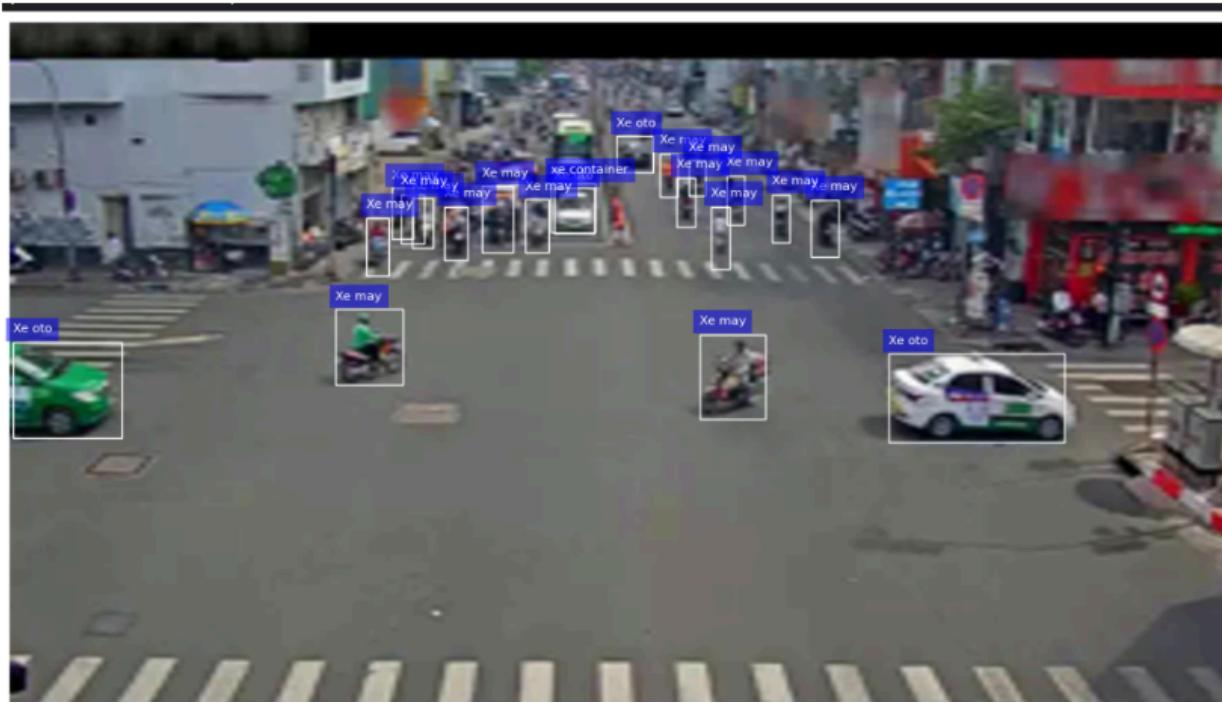
Ảnh: DETR phát hiện phương tiện giao thông vào ban ngày tại đường phố Việt



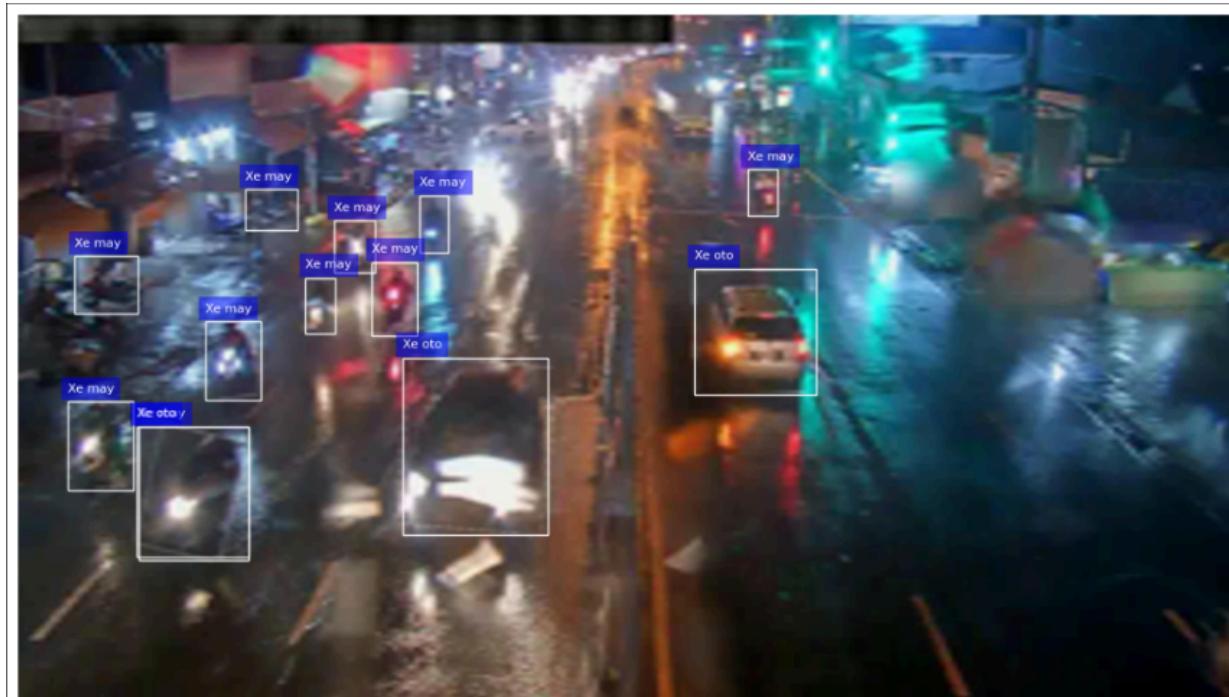
Ảnh: YOLO phát hiện phương tiện giao thông vào ban ngày tại đường phố Việt



Ảnh: YOLO phát hiện phương tiện giao thông vào ban ngày tại đường phố Việt



Ảnh: Faster RCNN phát hiện phương tiện giao thông vào ban ngày tại đường phố Việt



Ảnh: Faster RCNN phát hiện phương tiện giao thông vào ban ngày tại đường phố Việt

VIII. Nguồn tham khảo

- Faster RCNN:
 - o Bài báo “**Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks**” của nhóm tác giả Shaoqing Ren, Kaiming He, Ross Girshick, Jian Sun.
 - o https://pytorch.org/vision/main/models/generated/torchvision_models.detection.fasterrcnn_resnet50_fpn.html
- DETR:
 - o [**End-to-End Object Detection with Transformers**](#)
 - o [**ChatGPT**](#)
- YOLOv11:
 - o [**YOLOv11: An Overview of the Key Architectural Enhancements**](#)