

**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN**

**KHOA KHOA HỌC MÁY TÍNH**



**MÁY HỌC**

**BÁO CÁO ĐỒ ÁN**

**ĐẾM VÀ PHÂN LOẠI PHƯƠNG TIỆN GIAO THÔNG**

**Giảng viên hướng dẫn:** PGS.TS Lê Đình Duy

Th.S Phạm Nguyễn Trường An

**Lớp:** CS114.N11.KHCL

**Sinh viên thực hiện:** Đỗ Đức Thịnh - 20520780

Nguyễn Phan Quốc Thiện - 20520775

Nguyễn Cao Quốc - 20520723

Thành phố Hồ Chí Minh, tháng 2 năm 2023

# MỤC LỤC

<b>MỤC LỤC.....</b>	<b>i</b>
<b>DANH MỤC HÌNH ẢNH.....</b>	<b>iii</b>
<b>DANH MỤC TỪ VIẾT TẮT .....</b>	<b>iv</b>
<b>CHƯƠNG 1: GIỚI THIỆU.....</b>	<b>1</b>
1.1 Ngữ cảnh bài toán.....	1
1.2 Input và Output của bài toán .....	1
1.3 Ứng dụng .....	1
<b>CHƯƠNG 2: BỘ DỮ LIỆU .....</b>	<b>2</b>
2.1 Cách xây dựng bộ dữ liệu.....	2
2.2 Tiền xử lí dữ liệu .....	3
2.3 Phân tích dữ liệu.....	3
2.4 Gán nhãn dữ liệu .....	9
<b>CHƯƠNG 3: MÔ HÌNH VÀ THUẬT TOÁN.....</b>	<b>10</b>
3.1 Mô hình YOLOv5 .....	10
3.1.1 Quá trình training .....	11
3.1.2 Đánh giá mô hình.....	12
3.1.3 Kết quả và so sánh .....	14
3.2 Thuật toán Deep SORT .....	18
3.3 Kết hợp YOLOv5 và Deep SORT để đếm xe .....	18
<b>CHƯƠNG 4: TRIỂN KHAI THÀNH WEB VỚI STREAMLIT.....</b>	<b>21</b>
4.1 Tinh chỉnh tham số.....	21
4.2 Kết quả đầu ra.....	22
<b>CHƯƠNG 5: PHÂN TÍCH LỖI VÀ KẾT LUẬN .....</b>	<b>23</b>
5.1 Phân tích lỗi.....	23
5.2 Kết luận và hướng phát triển .....	24

<b>TÀI LIỆU THAM KHẢO .....</b>	<b>25</b>
---------------------------------	-----------

## DANH MỤC HÌNH ẢNH

Hình 2-1: Resize hình ảnh.....	3
Hình 2-2: Các ảnh tự thu thập.....	4
Hình 2-3: Các ảnh lấy từ nguồn ngoài .....	5
Hình 2-4: Các ảnh mẫu được gán nhãn.....	9
Hình 3-1: Kiến trúc YOLOv5 .....	10
Hình 3-2: Công thức IoU .....	12
Hình 3-3: Kết quả YOLOv5 của data nguồn ngoài .....	14
Hình 3-4: Kết quả YOLOv5 của data tự tìm .....	14
Hình 3-5: Kết quả YOLOv5 của toàn bộ data .....	14
Hình 3-6: Confusion matrix của toàn bộ data.....	15
Hình 3-7: Hình Precision-Recall Curve trên toàn bộ data.....	16
Hình 3-8: Ví dụ predict và ground truth .....	17
Hình 3-9: Thời gian xử lý của YOLOv5 và Deep SORT .....	20
Hình 4-1: Các cài đặt tham số của mô hình .....	21
Hình 4-2: Kết quả chạy của mô hình trên web .....	22
Hình 5-1: Một số trường hợp nhận diện sai .....	23

## DANH MỤC TỪ VIẾT TẮT

STT	Từ viết tắt	Diễn giải tiếng Anh	Diễn giải tiếng Việt
1	IoU	Intersection over Union	Giao chia hợp
2	mAP	Mean Average Precision	Độ chính xác trung bình
3	YOLO	You Only Look Once	
4	FPS	Frames Per Second	Khung hình trên giây
5	BB	Bounding box	

# CHƯƠNG 1: GIỚI THIỆU

## 1.1 Ngữ cảnh bài toán

Bài toán “đếm và phân loại số lượng phương tiện giao thông” trong máy học là một ứng dụng quan trọng của công nghệ trí tuệ nhân tạo. Đây là một bài toán phức tạp, yêu cầu thu thập và xử lý dữ liệu liên tục về các phương tiện giao thông đang di chuyển trên đường.

Bài toán này đòi hỏi các thuật toán phân loại và nhận diện hình ảnh chính xác, có thể phân biệt giữa các loại phương tiện giao thông khác nhau, bao gồm xe hơi, xe tải, xe buýt, xe máy và các phương tiện khác. Đồng thời, việc đếm số lượng phương tiện cũng là một yếu tố quan trọng để đảm bảo an toàn giao thông và quản lý thông tin về lưu lượng giao thông trên đường.

## 1.2 Input và Output của bài toán

Input của bài toán là video chứa các phương tiện giao thông đang lưu thông trên đường.

Output của bài toán gồm các thông tin sau:

- Vị trí của các phương tiện video.
- Loại phương tiện giao thông: xe hơi, xe tải, xe bus, xe máy.
- Số lượng phương tiện giao thông trong video.

## 1.3 Ứng dụng

Ứng dụng của bài toán này có thể được áp dụng trong nhiều lĩnh vực khác nhau, bao gồm giám sát giao thông, và nhiều ứng dụng khác. Tuy nhiên, để đạt được kết quả chính xác và đáng tin cậy, việc thu thập và xử lý dữ liệu phải được thực hiện đúng cách và sử dụng các kỹ thuật phân tích dữ liệu phù hợp. Trong một số trường hợp đặc biệt như trời mưa, sương mù hay kẹt xe mô hình vẫn chưa đem lại kết quả tốt.

## CHƯƠNG 2: BỘ DỮ LIỆU

### 2.1 Cách xây dựng bộ dữ liệu

Bộ dữ liệu được xây dựng dựa trên hai phương pháp: Tự xây dựng và lấy từ nguồn trên internet.

#### **Bộ dữ liệu tự xây dựng**

*Dữ liệu quay thực tế:*

- Cách thu thập: sử dụng camera điện thoại quay tại các tuyến đường có cầu vượt dành cho người đi bộ. Sau đó tiến hành cắt thủ công ảnh từ video và gán nhãn.
- Địa điểm: cầu vượt bến xe An Sương, cầu vượt quốc lộ 1A, cầu vượt Suối Tiên, cầu vượt ĐH Kinh tế - Luật, cầu vượt ngã 4 ga, cầu vượt KCN Sóng Thần.
- Thời gian: buổi sáng, nhìn rõ được các phương tiện đang lưu thông.
- Góc quay: từ trên cầu nhìn xuống.

*Số ít dữ liệu từ các video trên internet hoặc các camera giao thông:*

- Cách thu thập: Cắt ảnh từ video youtube hoặc các ứng dụng có camera giao thông như: Camera Giao Thông Sài Gòn và gán nhãn.
- Thời gian: trời sáng, thấy được các phương tiện đang lưu thông.
- Góc quay: từ trên nhìn xuống, quay hai chiều xe tới và đi.

#### **Bộ dữ liệu từ nguồn ngoài:**

- Cách thu thập: nhóm lấy từ 2 nguồn chính là Github và RoboFlow, được gán nhãn sẵn của 2 tác giả Maryam Boneh và Mark Basov.

Link source:

<https://github.com/MaryamBoneh/Vehicle-Detection>

<https://universe.roboflow.com/mark-basov/vehicles-dataset/>

## 2.2 Tiền xử lí dữ liệu

Về tiền xử lí dữ liệu nhóm sử dụng RoboFlow để gắn nhãn cho các tấm hình được cắt ra từ trong video đã quay đồng thời resize lại ảnh (640\*640) cũng như chuyển tên các nhãn từ kiểu ký tự sang kiểu số nguyên nào (0 - car, 1 - motorcycle, 2 – truck, 3 – bus). Dataset sẽ được chia thành 3 tập train, valid, test với tỉ lệ 8:1:1, tương ứng với 1224:154:153 ảnh, các ảnh được chia ngẫu nhiên bằng RoboFlow.



*Hình 2-1: Resize hình ảnh*

## 2.3 Phân tích dữ liệu

Dữ liệu của nhóm có tổng cộng 1531 tấm ảnh gồm 4 lớp: car, motorcycle, truck, bus. Trong đó có khoảng 559 tấm ảnh là dữ liệu lấy từ nguồn ngoài và 972 tấm ảnh là dữ liệu tự xây dựng của nhóm. Đây là con số ước lượng gần chính xác vì dataset đã được gộp lại nên nhóm gặp khó khăn khi tách vì các ảnh từ nhiều nguồn lẫn lộn với nhau.

### **Bộ dữ liệu tự xây dựng**





*Hình 2-2: Các ảnh tự thu thập*

Các ảnh (a) là ảnh nhóm tự quay trên các cầu vượt đi bộ, thường có mật độ xe lưu thông cao, chủ yếu là xe máy. Các ảnh (b) là ảnh nhóm cắt từ internet và camera giao thông.

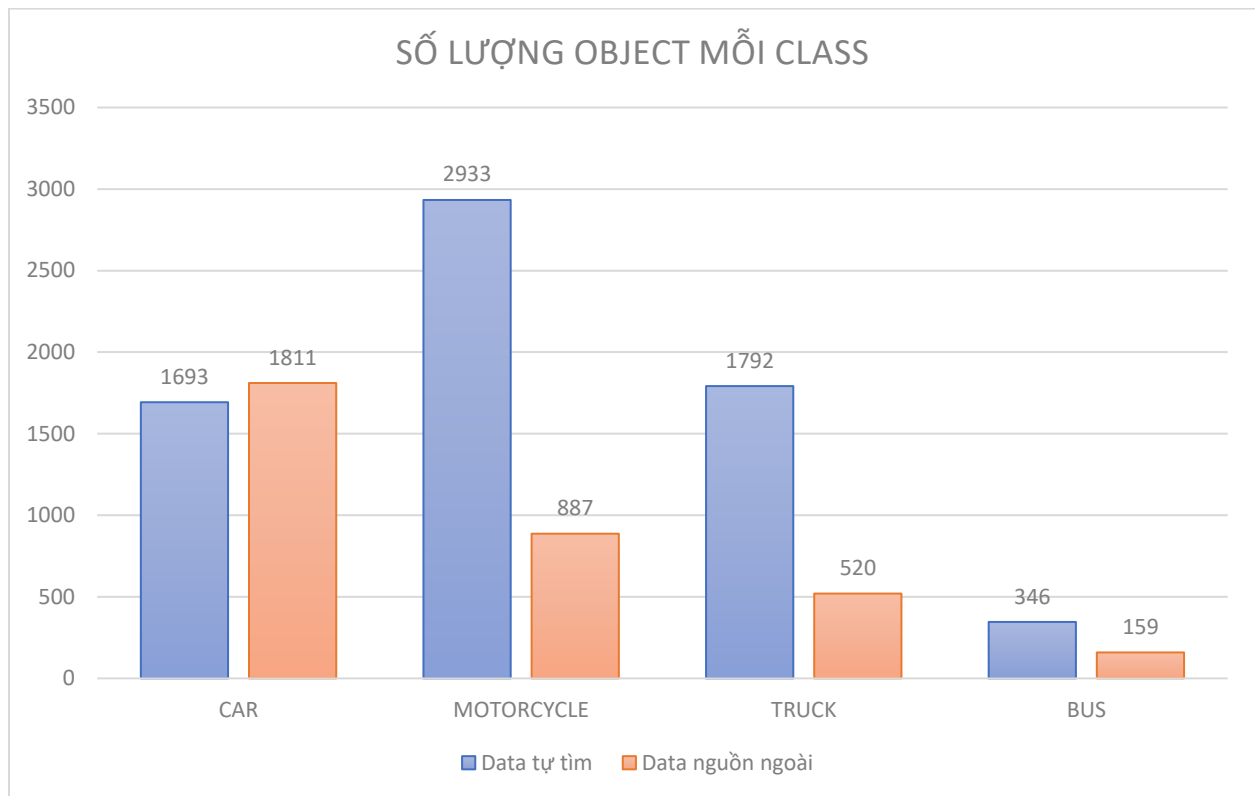
### Dữ liệu ngoài từ nguồn ngoài

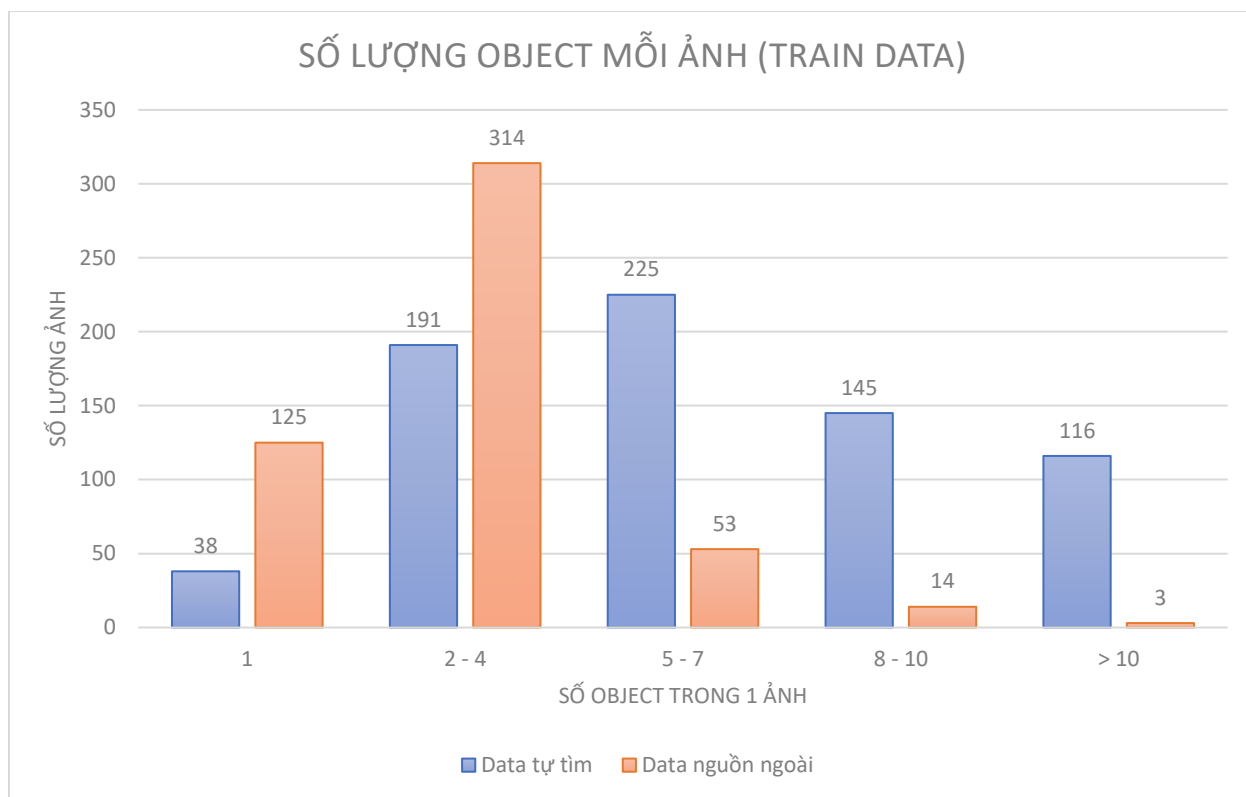


*Hình 2-3: Các ảnh lấy từ nguồn ngoài*

Các ảnh nguồn ngoài có mật độ xe lưu thông không cao và đa phần là xe hơi.

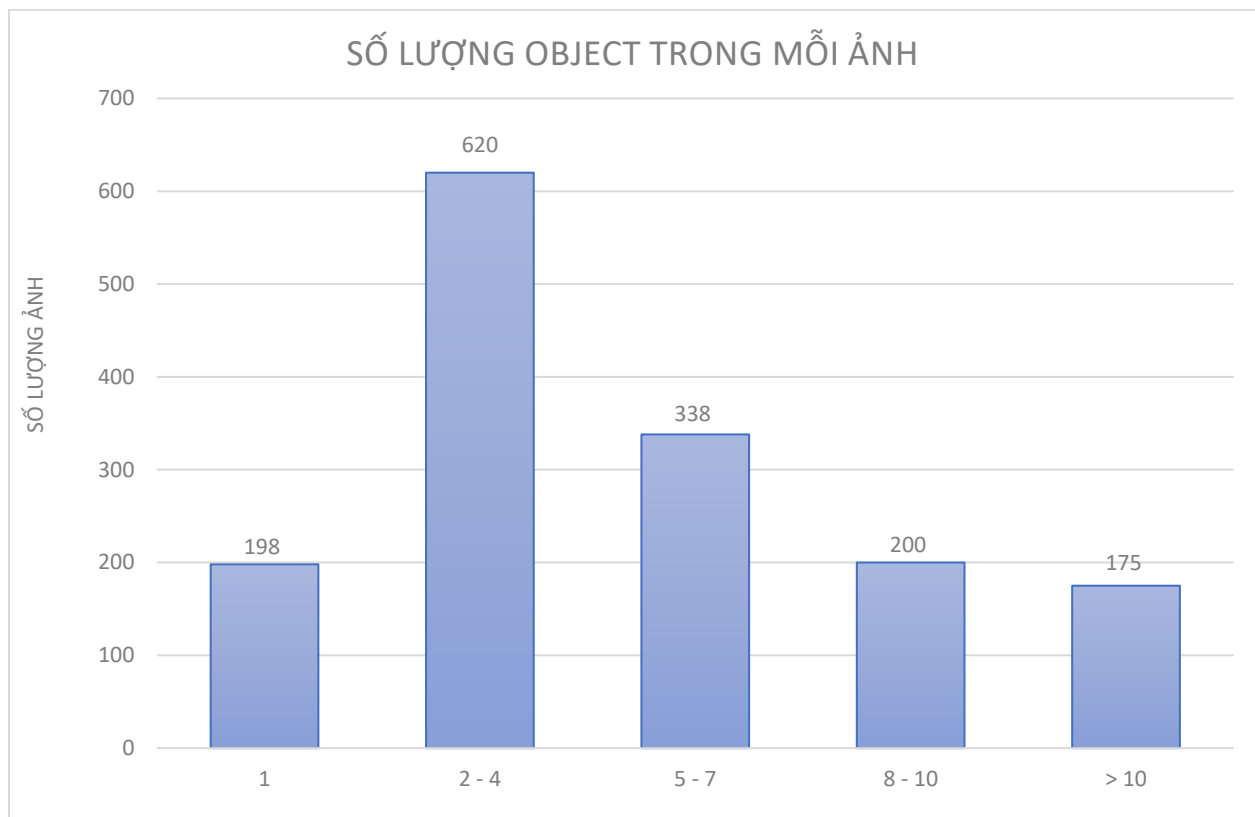
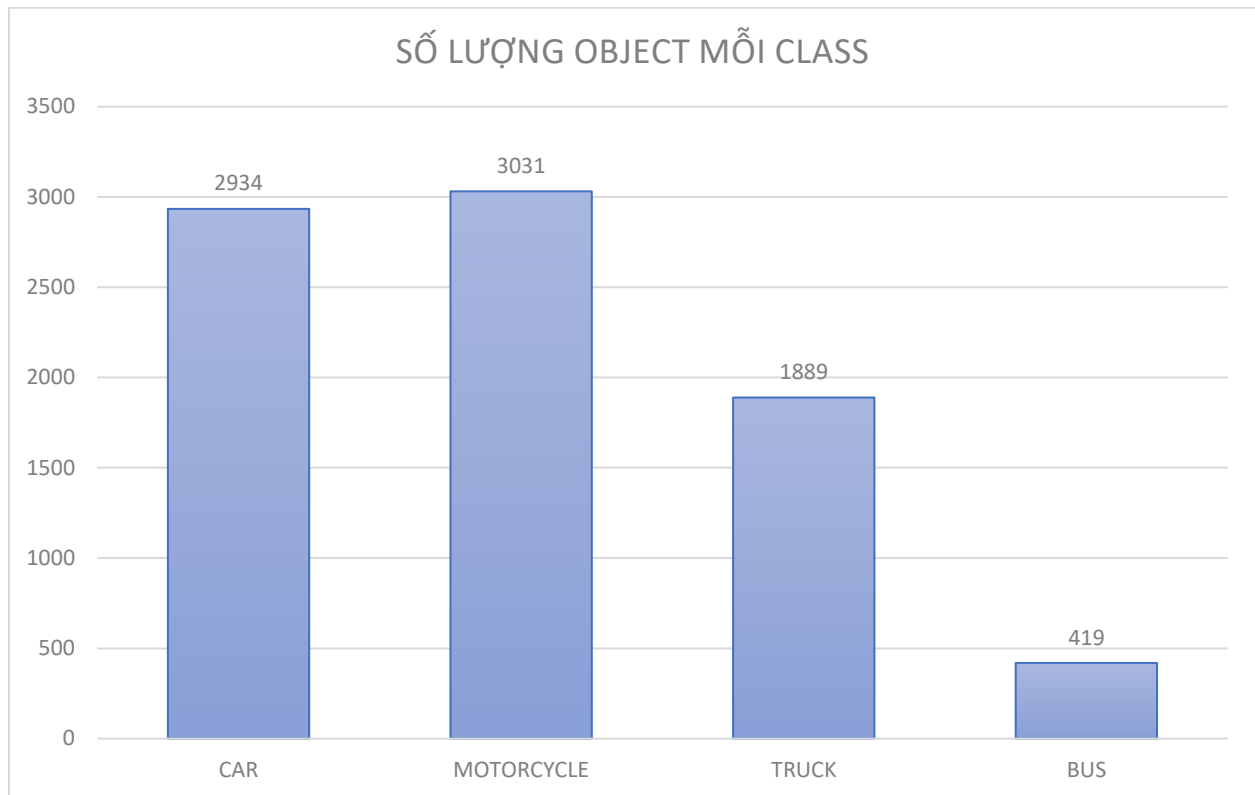
## Biểu đồ số lượng object/class và biểu đồ số lượng object/image





Dựa vào bảng số liệu trên, data nguồn ngoài đa phần có ít object trong 1 ảnh, đa số nằm trong khoảng 1 – 4 object. Đối với data nhóm tự thu thập thì các ảnh có mật độ xe lưu thông cao, đặc biệt là xe máy cho nên 1 ảnh thường sẽ có trên 5 object.

### Toàn bộ dữ liệu: tổng hợp từ 2 nguồn

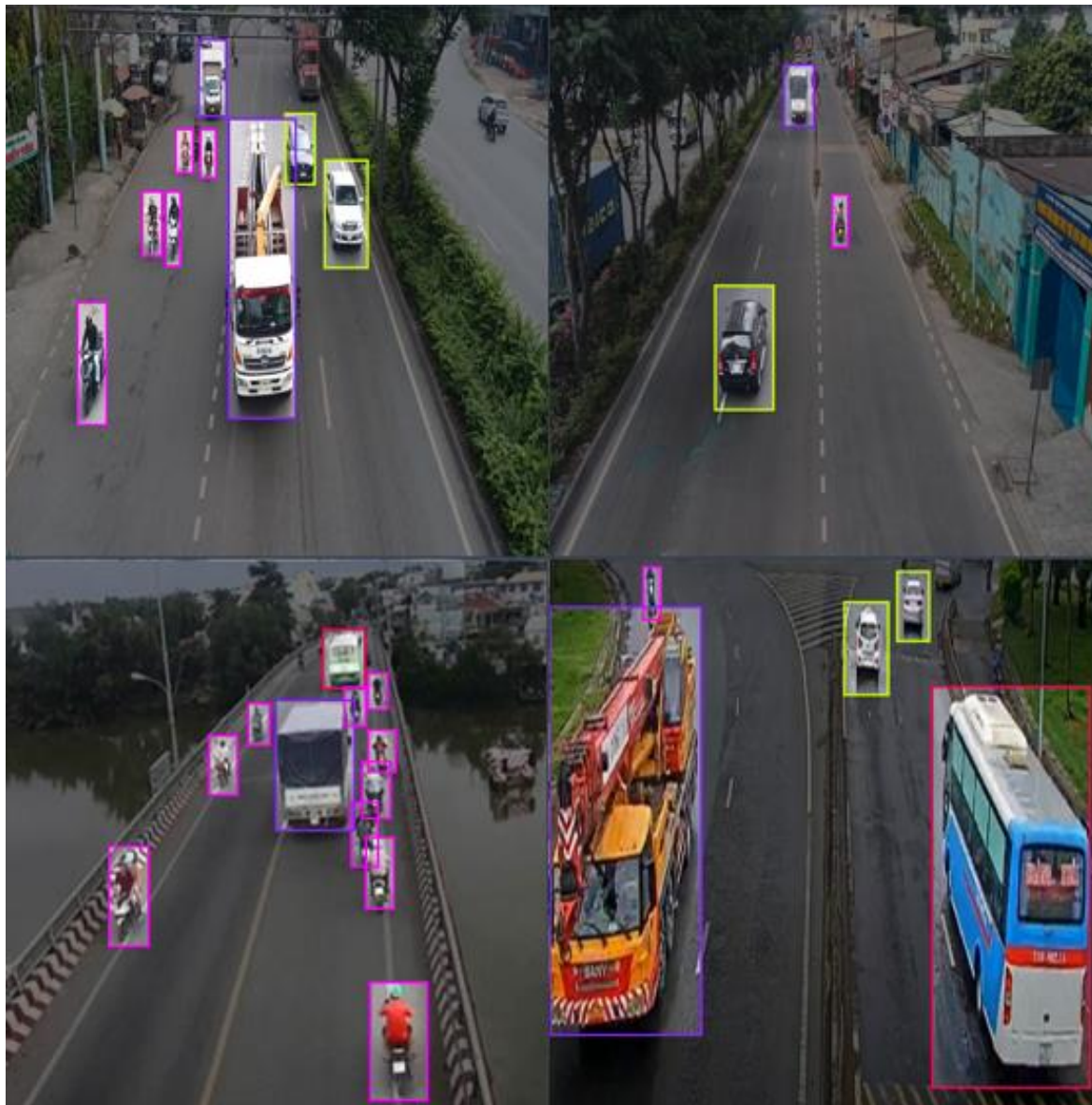




## 2.4 Gán nhãn dữ liệu

Để có sự thống nhất khi gán nhãn, nhóm đặt ra quy tắc gán nhãn:

- Các xe như container, xe cần cẩu được quy về là truck.
- Các bounding box phải cố gắng bao được hết vật thể.
- Khi gán nhãn cho motorcycle là phải gán luôn cho cả người lái xe.



Hình 2-4: Các ảnh mẫu được gán nhãn



### 3.1.1 Quá trình training

1. Chuẩn bị data: dữ liệu được xuất ra gồm có định dạng hình ảnh (.jpg) và định dạng văn bản (.txt) để chứa thông tin các bounding box của các phương tiện được gắn nhãn. Nhóm sử dụng mô hình YOLOv5 nên format của bounding box sẽ như sau:

```
0 0.203125 0.046875 0.134375 0.0875
0 0.359375 0.02890625 0.128125 0.0546875
3 0.76640625 0.07265625 0.45625 0.1375
```

*Hình 2-1: Ví dụ format bounding box của YOLO*

Ví dụ trên có 3 bounding box trong hình, 1 bounding box sẽ gồm: c, x, y, w, h

Chú thích:

- c: bounding box đó thuộc class nào (0 - car, 1 - motorcycle, 2 - truck, 3 - bus)
- x, y: tọa độ tâm của bounding box
- w: chiều rộng bounding box
- h: chiều dài bounding box

2. Tiến hành training: Sử dụng Google Colab và GPU để train nhanh hơn.

```
1 # Train YOLOv5n
2 !python train.py --img 640 --batch 64 --epochs 100 --data custom_data.yaml --weights 'yolov5n.pt'
```

File *custom\_data.yaml* sẽ chứa đường dẫn đến thư mục chứa data

```
train: ../all_dataset/images/train # train images
val: ../all_dataset/images/val # val images
test: ../all_dataset/images/test # test images

# Classes
names:
  0: car
  1: motorcycle
  2: truck
  3: bus
```

YOLOv5 có nhiều loại mô hình với kích thước và số lượng tham số khác nhau, các mô hình lớn như YOLOv5l, YOLOv5x có độ tin cậy cao tuy nhiên phải đánh đổi tốc độ



xử lý. Vì nhóm chạy mô hình trên laptop không có GPU rời cho nên quyết định chọn YOLOv5n – loại mô hình có kích thước nhỏ nhất để đảm bảo tốc độ xử lý.

### 3.1.2 Đánh giá mô hình

- Precision: còn gọi là độ chính xác. Cho biết các xe mô hình dự đoán thật sự đúng bao nhiêu.

$$precision = \frac{\sum TP}{\sum TP + FP}$$


- Recall: còn gọi là độ phủ. Cho biết mô hình có khả năng phát hiện đúng được bao nhiêu xe so với ground truth.

$$recall = \frac{\sum TP}{\sum TP + FN}$$

- F1- score: là trung bình điều hòa giữa precision và recall.

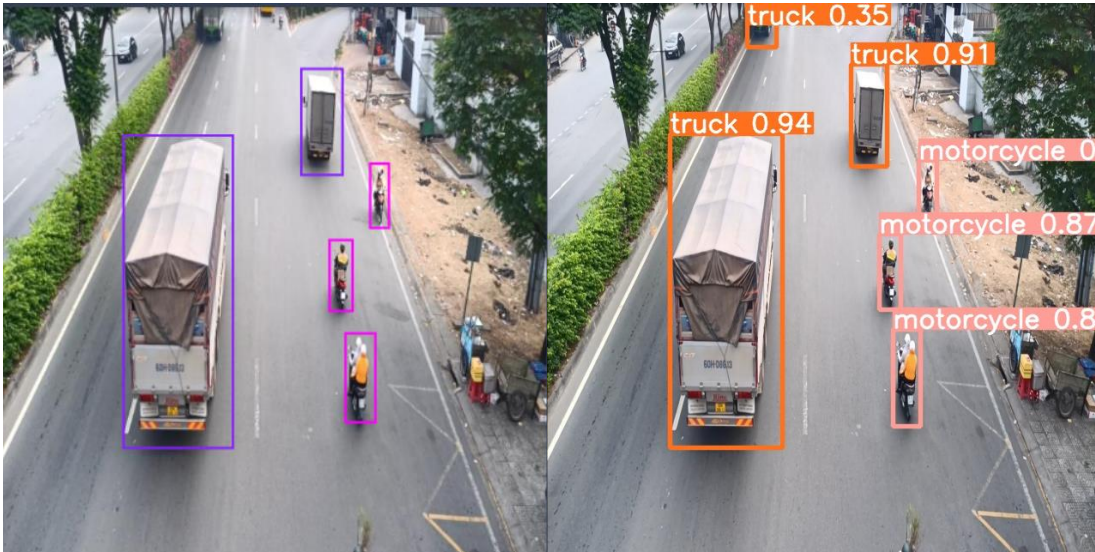
$$F1 - score = \frac{2 * precision * recall}{precision + recall}$$

- mAP: còn gọi là độ chính xác trung bình. Bằng cách tính toán precision và recall ở các True positive, nó cho độ hiệu quả của mô hình khi xếp hạng các bounding box theo độ giảm dần confidence. mAP là trung bình cộng của các AP mỗi class.
- IoU: là độ đo đánh giá các mô hình nhận diện đối tượng. IoU sử dụng để đánh giá độ che lấp lên nhau giữa hai bounding box.


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

Hình 3-2: Công thức IoU

Ví dụ tính toán mAP:



Hình bên trái là ground truth và hình bên phải là predict

Với  $mAP50 = \frac{\sum P_r}{m}, r > 0.5$ ;  $mAP50-95 = \frac{\sum P_r}{m}, 0.95 > r > 0.5$ ,  $m$  là số  $P_r$

Các bounding box sẽ xếp hạng theo confidence giảm dần. Correct ? TRUE khi IoU của  $BB_{predict}$  và  $BB_{ground\ truth}$  lớn hơn 1 ngưỡng quy định (thường là 0.5).

	Rank	Correct ?	P	R
Truck	1	TRUE	1	0,50
	2	TRUE	1	1,00
	3	FALSE	0,666667	1,00
	AP50 <sub>truck</sub>			1
	AP50-95 <sub>truck</sub>			0,888889
Motor	1	TRUE	1	0,33
	2	TRUE	1	0,67
	3	TRUE	1	1,00
	AP50 <sub>motor</sub>			1
	AP50-95 <sub>motor</sub>			1
Tổng kết	mAP50=(AP50 <sub>truck</sub> + AP50 <sub>motor</sub> ) / 2			1
	mAP50-95=(AP50-95 <sub>truck</sub> + AP50-95 <sub>motor</sub> ) / 2			0,944444

### 3.1.3 Kết quả và so sánh

Để đánh giá hiệu quả mô hình trên dataset tự tìm và dataset ngoài, nhóm sử dụng chung tập val và test, còn tập train sẽ khác nhau.

Class	Images	Instances	P	R	mAP50	mAP50-95
all	154	950	0.563	0.569	0.557	0.327
car	154	310	0.609	0.858	0.807	0.562
motorcycle	154	373	0.764	0.362	0.491	0.192
truck	154	228	0.734	0.314	0.458	0.256
bus	154	39	0.146	0.744	0.474	0.296

Hình 3-3: Kết quả YOLOv5 của data nguồn ngoài

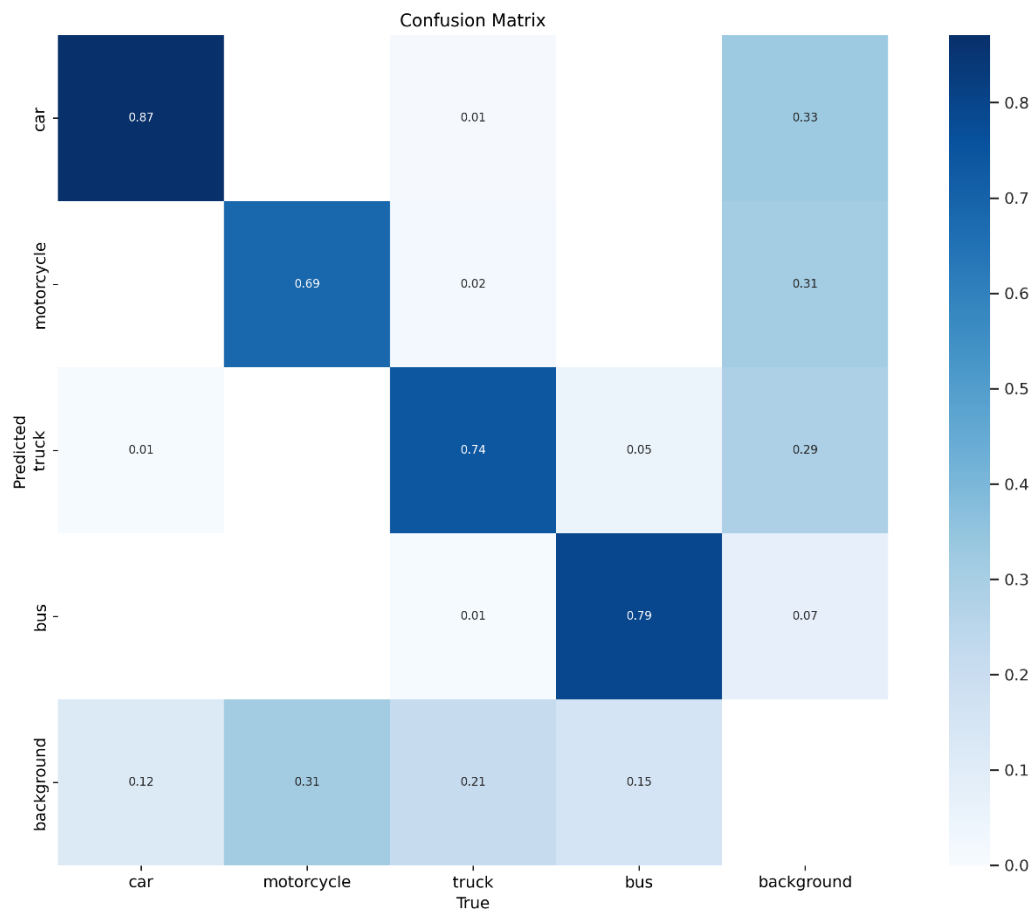
Class	Images	Instances	P	R	mAP50	mAP50-95
all	154	950	0.735	0.703	0.765	0.464
car	154	310	0.875	0.634	0.774	0.47
motorcycle	154	373	0.778	0.662	0.737	0.352
truck	154	228	0.646	0.732	0.732	0.498
bus	154	39	0.643	0.786	0.818	0.537

Hình 3-4: Kết quả YOLOv5 của data tự tìm

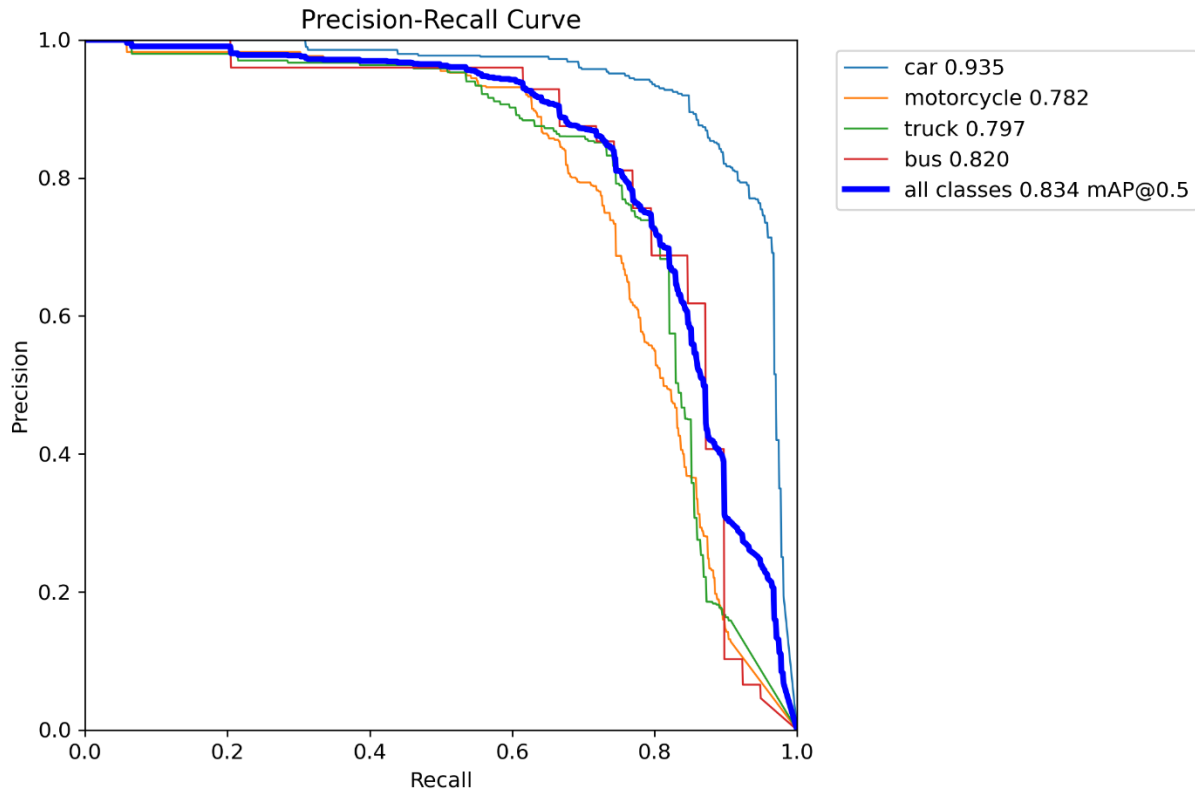
**Một số kết quả của toàn bộ data:**

Class	Images	Instances	P	R	mAP50	mAP50-95
all	154	950	0.866	0.736	0.834	0.541
car	154	310	0.906	0.848	0.935	0.68
motorcycle	154	373	0.9	0.627	0.782	0.37
truck	154	228	0.852	0.715	0.797	0.535
bus	154	39	0.807	0.753	0.82	0.58

Hình 3-5: Kết quả YOLOv5 của toàn bộ data



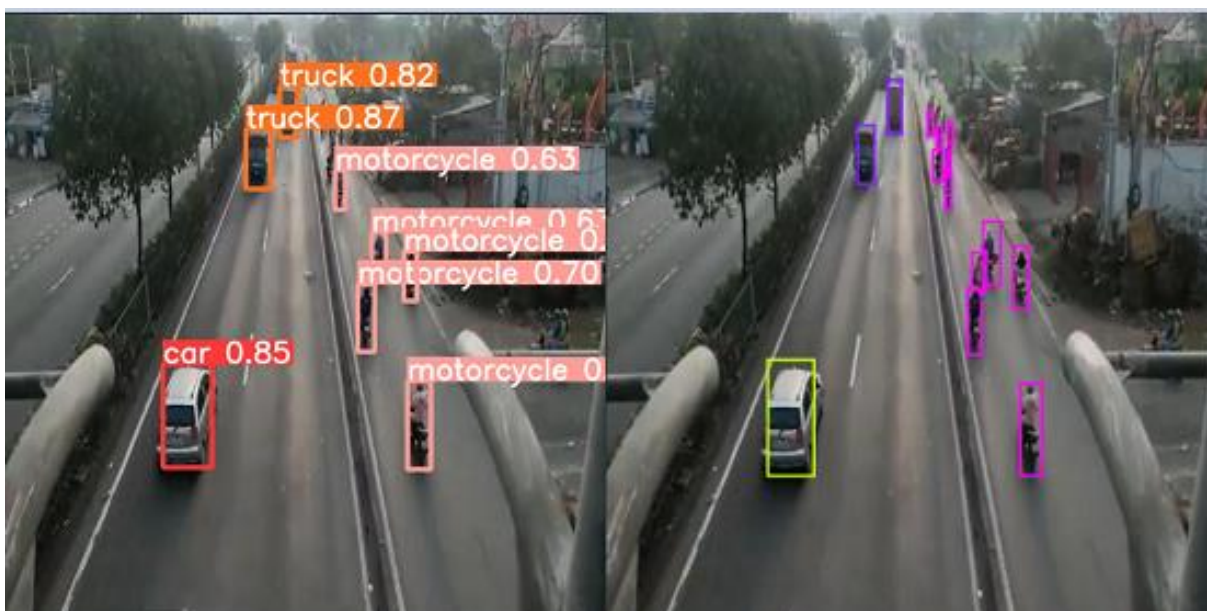
Hình 3-6: Confusion matrix của toàn bộ data



Hình 3-7: Hình Precision-Recall Curve trên toàn bộ data

Phân tích kết quả:

- Data nguồn ngoài: vì dataset mất cân bằng tại class car nên hiệu suất class car tốt nhất khi recall được 0.85 và mAP50 cao hơn hẳn các class khác.
- Data tự tìm: kết quả khá tốt, các mAP50 đều trên 0.7
- Toàn bộ data: kết quả khi kết hợp cả 2 nguồn đem lại kết quả tốt nhất. Ta có thể thấy ở class motorcycle có nhiều ảnh nhất nên precision cao tuy nhiên kích thước của motorcycle nhỏ nên việc phát hiện hết sẽ rất khó, recall chỉ 0.62.



Hình 3-8: Ví dụ predict và ground truth

Nhìn hình predict ta thấy các vật thể to như car và truck được nhận diện đúng và có confidence khá cao, tuy nhiên đối với vật thể nhỏ như motorcycle thì việc phát hiện ra toàn bộ vật thể trong ảnh rất khó và confidence cũng không cao. Đây cũng chính là nhược điểm của mô hình YOLO khi không hiệu quả trong việc phát hiện vật thể nhỏ.

### So sánh với các bài báo

Việc tìm ra bài báo có giống hoàn toàn về bài toán, bộ dữ liệu, phiên bản YOLO là rất khó nên nhóm chỉ so sánh với một vài bài mang tính tương đối.

Tên mô hình	Precision	MAP@0.5
Mô hình của nhóm (YOLOv5n)	86%	83.4%
YOLOv4(custom) [1]	-	82.08%
YOLOv4(activation function: Mish) [2]	-	79%
YOLOv3 [3]	99.15%	-

Link các bài báo:

[1] <https://ieeexplore.ieee.org/document/9431784>

[2] <https://ieeexplore.ieee.org/abstract/document/9287483>

[3] <https://ieeexplore.ieee.org/document/9266010>

### 3.2 Thuật toán Deep SORT

Deep SORT là một thuật toán giúp theo dõi vật thể (Object tracking) trong video. Deep SORT kết hợp giữa 2 phương pháp là detection (phát hiện đối tượng) và Embedding (nhúng đối tượng), được Nicolai Wojke và Alex Bewley phát triển trên SORT, cho phép theo dõi đối tượng một cách chính xác trong các video có độ phân giải cao. Deep SORT được xây dựng nhằm giải quyết các vấn đề thiếu sót liên quan đến số lượng ID switches cao. Hướng giải quyết mà Deep SORT đề xuất dựa trên việc sử dụng deep learning để trích xuất các đặc trưng của đối tượng nhằm tăng độ chính xác trong quá trình liên kết dữ liệu.

Về ý tưởng thuật toán, trong multiple object tracking, đặc biệt là đối với lớp thuật toán tracking-by-detection, có 2 yếu tố chính ảnh hưởng trực tiếp đến performance của việc theo dõi:

- Data Association: Quan tâm đến vấn đề liên kết dữ liệu, cụ thể là tiêu chí để xét và đánh giá nhằm liên kết một detection mới với các track đã được lưu trữ sẵn.
- Track Life Cycle Management: Quan tâm đến việc quản lý vòng đời của một track đã được lưu trữ, bao gồm, khi nào thì khởi tạo track, khi nào thì ngưng theo dõi và xóa track ra khỏi bộ nhớ, ...

Trong deep SORT, nhóm tác giả giải quyết vấn đề data association dựa trên thuật toán Hungary (tương tự như SORT), tuy nhiên, việc liên kết không chỉ dựa trên IOU mà còn quan tâm đến các yếu tố khác: khoảng cách của detection và track (xét tính tương quan trong không gian vector) và khoảng cách cosine giữa 2 vector đặc trưng được trích xuất từ detection và track (chi tiết được trình ở các phần sau) - 2 vector đặc trưng của cùng 1 đối tượng sẽ giống nhau hơn là đặc trưng của 2 đối tượng khác nhau.

Hiện tại, nhóm vẫn chưa đánh giá được hiệu quả của thuật toán này.

### 3.3 Kết hợp YOLOv5 và Deep SORT để đếm xe

SORT nói chung và Deep SORT nói riêng đều tập trung vào vấn đề liên kết giữa các detection và track sau khi đã detect được từ frame, do đó, phần object detection có thể là bất cứ mô hình detector. Sau khi có tham khảo một số bài báo đếm xe trên đường, nhóm

lựa chọn dùng YOLOv5 để detect xe và dùng Deep SORT để gán ID cho các xe để theo dõi. Ý tưởng như sau:

1. Sử dụng YOLO để detect các xe ở trên đường, tính toán tâm bounding box.
2. Các xe được YOLO detect sẽ gán 1 ID để theo dõi.
3. Vẽ một vạch kẻ ảo trong khung hình (vạch kẻ thường nằm ở nửa dưới của khung hình), và các xe có tâm bounding box nằm dưới vạch kẻ này sẽ được đếm dựa vào ID. Lí do phải đếm xe ở dưới vạch kẻ để tránh trường hợp có xe khi ở xa mang 1 ID, rồi bị mất dấu, đến khi xuất hiện lại ở gần lại mang thêm 1 ID mới, các xe ở dưới vạch kẻ thường gần với camera nên độ tin cậy khi detect sẽ cao hơn so với ở xa.

```
def count_obj(box, w, h, id, label, line_pos):
    global data_car, data_bus, data_truck, data_motor, already
    center_coordinates = (int(box[0]+(box[2]-box[0])/2) , int(box[1]+(box[3]-
box[1])/2))
    # classify one time per id
    if center_coordinates[1] > (h*line_pos):
        if id not in already:
            already.append(id)
            if label == 'car' and id not in data_car:
                data_car.append(id)
            elif label == 'bus' and id not in data_bus:
                data_bus.append(id)
            elif label == 'truck' and id not in data_truck:
                data_truck.append(id)
            elif label == 'motorcycle' and id not in data_motor:
                data_motor.append(id)
```

Ở đây có sử dụng list *already* là để đảm bảo các ID đã phân loại vào 1 class sẽ không được phân loại vào class khác, vì trong lúc test nhóm phát hiện có trường hợp 1 xe khi ở dưới vạch kẻ ảo vẫn bị phân loại vào 2 class khác nhau. Số lượng ID trong mỗi list cũng chính là số lượng xe đếm được.

Thuật toán này sẽ gán ID dựa vào detection cho nên việc YOLO phát hiện đúng các vật thể đóng vai trò rất quan trọng. Thời gian của Deep SORT theo dõi các vật thể lớn hơn



nhiều so với thời gian YOLO nhận diện, phân loại vật thể. Nếu đoạn đường nhiều xe thì sẽ làm tăng thời gian xử lý của mô hình.

```
video 1/1 (20/610) D:\Python Project\SML\Vehicle_Detection_and_Counting_System\videos\as.mp4: 288x480 2 cars, 8 motorcycles, 3 trucks, Done. YOLO:(0.069s), DeepSort:(0.302s)  
2023-03-02 21:29:31.015 video 1/1 (21/610) D:\Python Project\SML\Vehicle_Detection_and_Counting_System\videos\as.mp4: 288x480 2 cars, 8 motorcycles, 3 trucks, Done. YOLO:(0.071s), DeepSort:(0.332s)  
video 1/1 (21/610) D:\Python Project\SML\Vehicle_Detection_and_Counting_System\videos\as.mp4: 288x480 2 cars, 8 motorcycles, 3 trucks, Done. YOLO:(0.071s), DeepSort:(0.332s)
```

*Hình 3-9: Thời gian xử lý của YOLOv5 và Deep SORT*

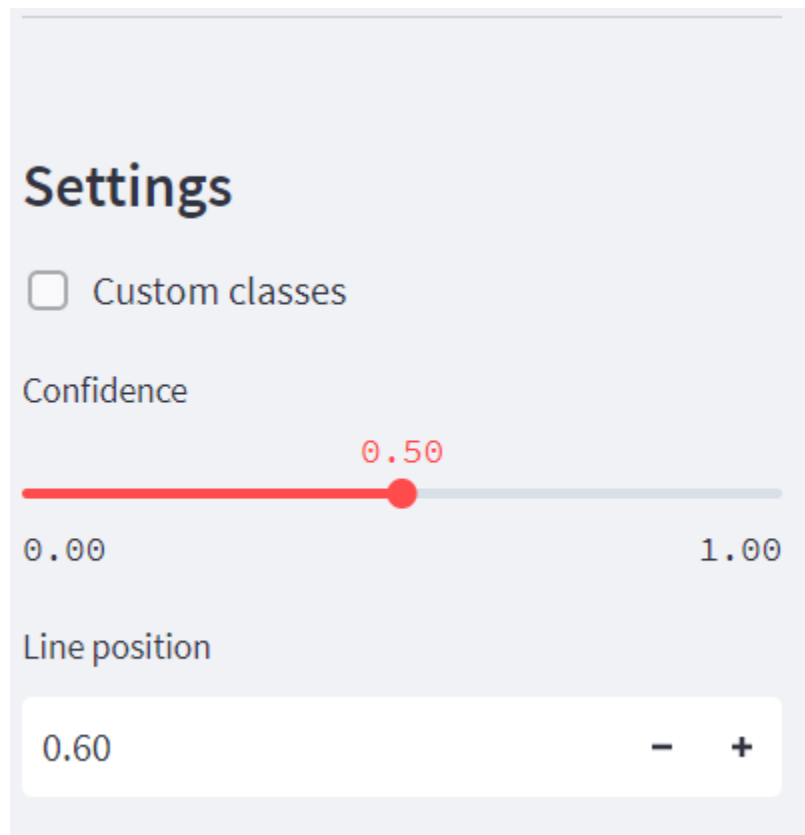
## CHƯƠNG 4: TRIỂN KHAI THÀNH WEB VỚI STREAMLIT

Streamlit là một framework mã nguồn mở cho phép người dùng xây dựng các ứng dụng web để trực tiếp tương tác với các mô hình máy học, dữ liệu và các thư viện khác của python một cách dễ dàng và nhanh chóng. Streamlit hỗ trợ việc thiết kế giao diện người dùng, tạo các thành phần tương tác và trình bày kết quả mô hình một cách trực quan. Với streamlit, người dùng có thể tạo ra các ứng dụng web để giới thiệu và chia sẻ các dự án mô hình và kết quả của mình với những người khác.

### 4.1 Tinh chỉnh tham số

Các tham số có thể điều chỉnh:

- Custom classes: tùy chỉnh các class muốn YOLO detect. Nếu muốn detect hết 4 class thì không cần chọn.
- Confidence: độ tin cậy, xác suất mà vật thể thuộc một class nào đó.
- Line position: vị trí vạch kẻ ảo, các xe nằm dưới vạch kẻ này sẽ được đếm.



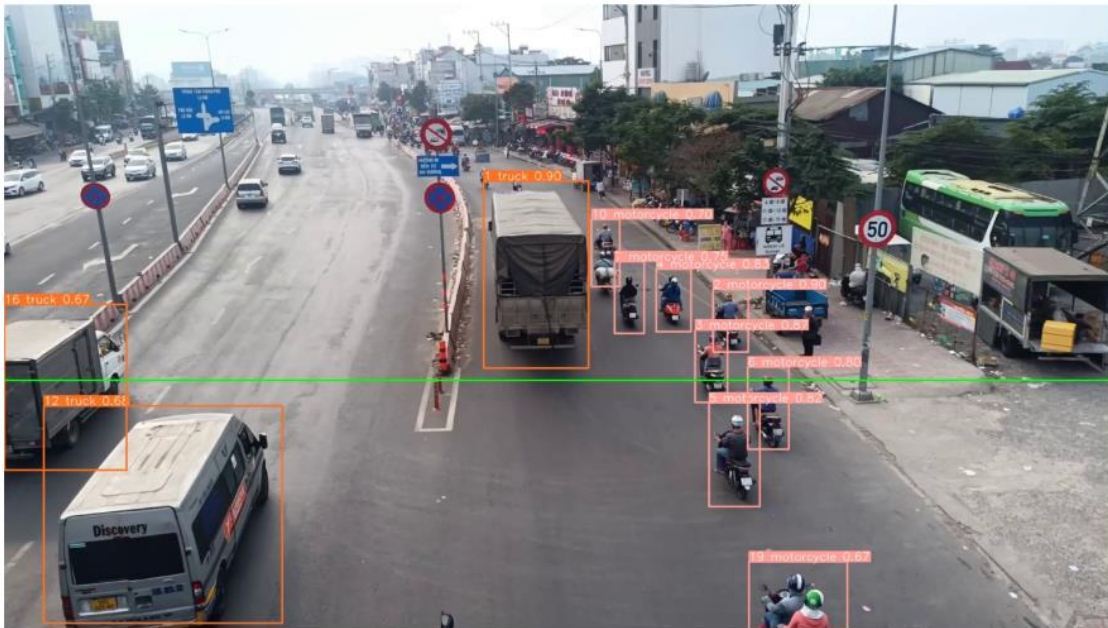
The image shows a 'Settings' panel with three adjustable parameters:

- Custom classes:** A checkbox that is currently unchecked.
- Confidence:** A horizontal slider ranging from 0.00 to 1.00. The slider is currently set to 0.50, indicated by a red dot and the number '0.50' above it.
- Line position:** A numeric input field showing the value '0.60'. To the right of the input are minus and plus buttons for adjustment.

Hình 4-1: Các cài đặt tham số của mô hình

## 4.2 Kết quả đầu ra

Status: Running...



Car	Bus	Truck	Motorcycle
1	0	1	4
FPS			
1.9			

Hình 4-2: Kết quả chạy của mô hình trên web

- Nơi hiển thị video có vạch kẻ ảo, bounding box có các thành phần id, class\_name, confidence\_score.
- Số lượng phương tiện của từng class đếm được.
- FPS cho biết tốc độ xử lý của mô hình. Vì nhóm không có GPU rời nên FPS rất thấp, thường là khoảng 5 FPS, đối với các video có nhiều xe xuất hiện trong khung hình thì mô hình phải xử lý nhiều dẫn đến FPS có thể thấp hơn 5.

## CHƯƠNG 5: PHÂN TÍCH LỖI VÀ KẾT LUẬN

### 5.1 Phân tích lỗi

Trong khi chạy trên một số video khác để kiểm tra hiệu suất mô hình, nhóm nhận thấy một số lỗi, các lỗi này ít xuất hiện và chỉ trong một vài trường hợp đặc biệt.



(a)



(b)



(c)



(d)



(e)

Hình 5-1: Một số trường hợp nhận diện sai

Các trường hợp mà mô hình nhận dạng sai có thể phụ thuộc nhiều yếu tố, nhóm cũng suy nghĩ những giải pháp để cải thiện mô hình:

- (a) Ảnh có 2 motorcycle nhưng nhận diện chỉ có 1 motorcycle, điều này có thể xảy ra khi các xe nằm chồng lên nhau.
- (b) Mô hình vẽ bounding box của car bị dư và bao cả 2 motorcycle ở phía sau, xảy ra khi các xe nằm quá gần nhau.
- (c) Mô hình nhận diện lỗi khi không có truck nào ở đó, confidence chỉ 0.54, bounding box này chỉ xuất hiện trong vài frame và biến mất rất nhanh. Giải pháp có thể đưa ra là tăng ngưỡng confidence hoặc thêm các hình ảnh background không chứa vật thể, theo tác giả YOLOv5 số ảnh background nên chiếm khoảng 10% tổng ảnh dataset, điều này có thể giúp giảm các false positive.
- (d) Mô hình nhận diện sai hoàn toàn khi vật thể không phải là motorcycle.
- (e) Nhận diện sai truck thành bus khi xe ở xa camera, confidence cũng thấp, truck và bus có kích thước và hình dáng khá giống nhau. Trường hợp này có thể tăng ngưỡng confidence.

## 5.2 Kết luận và hướng phát triển

Mô hình có thể vận hành ở mức cơ bản và thực hiện được chức năng đếm xe của nó. Tuy nhiên cũng còn nhiều nhược điểm như: chưa thể đánh giá được quá trình theo dõi, các trường hợp lỗi vẫn còn, hiệu suất không tốt khi gặp các điều kiện khắc nghiệt (trời tối, kẹt xe).

Để mô hình có thể cho ra kết quả tốt hơn trong thực tế thì nhóm có ý định:

- Sẽ tiếp tục nghiên cứu sâu hơn để tối ưu hóa mô hình về mặt tốc độ xử lý, độ chính xác của mô hình, tinh chỉnh các tham số.
- Thu thập thêm các dữ liệu mới để dữ liệu đa dạng hơn.
- Phát triển ứng dụng sao cho có thể đếm được xe ở riêng hai chiều ngược nhau.

## TÀI LIỆU THAM KHẢO

- [1] [Online]. Available: <https://github.com/ultralytics/yolov5/issues/280>. [Accessed 2 3 2023].
- [2] [Online]. Available: <https://github.com/ultralytics/yolov5>. [Accessed 2023 3 2].
- [3] [Online]. Available: <https://viblo.asia/p/sort-deep-sort-mot-goc-nhin-ve-object-tracking-phan-1-Az45bPooZxY>. [Accessed 2 3 2023].
- [4] [Online]. Available: <https://viblo.asia/p/sort-deep-sort-mot-goc-nhin-ve-object-tracking-phan-2-djeZ1m78ZWz>. [Accessed 2 3 2023].
- [5] [Online]. Available: [https://github.com/nwojke/deep\\_sort](https://github.com/nwojke/deep_sort). [Accessed 2 3 2023].
- [6] [Online]. Available: <https://viblo.asia/p/tong-hop-kien-thuc-tu-yolov1-den-yolov5-phan-3-63vKjgJ6Z2R>. [Accessed 2 3 2023].
- [7] [Online]. Available: <https://analyticsindiamag.com/5-object-detection-evaluation-metrics-that-data-scientists-should-know/>. [Accessed 2 3 2023].
- [8] [Online]. Available: <https://docs.streamlit.io/>. [Accessed 2 3 2023].
- [9] [Online]. Available: <https://ieeexplore.ieee.org/document/9431784>. [Accessed 2 3 2023].
- [10] [Online]. Available: <https://ieeexplore.ieee.org/abstract/document/9287483>. [Accessed 2 3 2023].
- [11] [Online]. Available: <https://ieeexplore.ieee.org/document/9266010>. [Accessed 2 3 2023].