

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
ĐẾM SỐ LƯỢNG XE QUA LẠI TRÊN MỘT
ĐOẠN ĐƯỜNG

Giảng viên hướng dẫn:

PGS.TS Lê Đình Duy

Th.S Phạm Nguyên Trường An

Lớp: CS114.N11.KHCL

Sinh viên thực hiện:

Lê Phước Trung - 20522069

Trương Đăng Nghĩa - 20520775

Phạm Kiên- 20520723

LỜI CẢM ƠN

Xin gửi lời cảm ơn sâu sắc đến thầy Lê Đình Duy và thầy Phạm Nguyễn Trường An đã tạo điều kiện cho chúng em có cơ hội được làm đồ án. Chỉ trong vòng 15 tuần nhưng nhờ sự chỉ dẫn nhiệt tình của thầy, chúng em đã tiếp thu được những kiến thức quan trọng để có thể thực hiện đồ án, chỉ ra những lỗi sai, để bạn em nhận ra những lỗi hỏng, sự thiếu sót của bản thân về kiến thức, quy trình làm một đồ án để bạn em có thể kịp thời khắc phục, build được một model gần như hoàn chỉnh.

Cũng xin cảm ơn thầy cô trong khoa Khoa học máy tính đã nhiệt tình hỗ trợ, tạo điều kiện cho chúng em làm bài báo cáo này.

LỜI CẢM ƠN.....	2
MỤC LỤC.....	3
PHẦN 1: GIỚI THIỆU.....	4
1.1 Ngữ cảnh bài toán.....	4
1.2 Input và Output của bài toán	4
1.3 Ứng dụng.....	4
PHẦN 2: XÂY DỰNG BỘ DỮ LIỆU	5
2.1 Cách xây dựng bộ dữ liệu.....	5
2.2 Tiền xử lý dữ liệu	6
2.3 Gán nhãn dữ liệu	6
PHẦN 3: MÔ HÌNH VÀ THUẬT TOÁN.....	7
3.1 Mô hình YOLOv8.....	7
3.1.1 Quá trình training.....	8
3.1.2 Đánh giá mô hình.....	10
3.2 Thuật toán Deep SORT.....	11
3.3 Giải quyết bài	12
PHẦN 4: KẾT QUẢ.....	13
PHẦN 5: KẾT LUẬN	14
TÀI LIỆU THAM KHẢO	14

PHẦN 1: GIỚI THIỆU

1.1 Ngữ cảnh bài toán:

Với việc tai nạn giao thông xảy ra càng nhiều và sự phát triển của hệ thống giao thông đặc biệt là ở những thành phố lớn, việc cài đặt ứng dụng đếm số lượng xe là cần thiết để có thể điều chỉnh trạng thái giao thông.

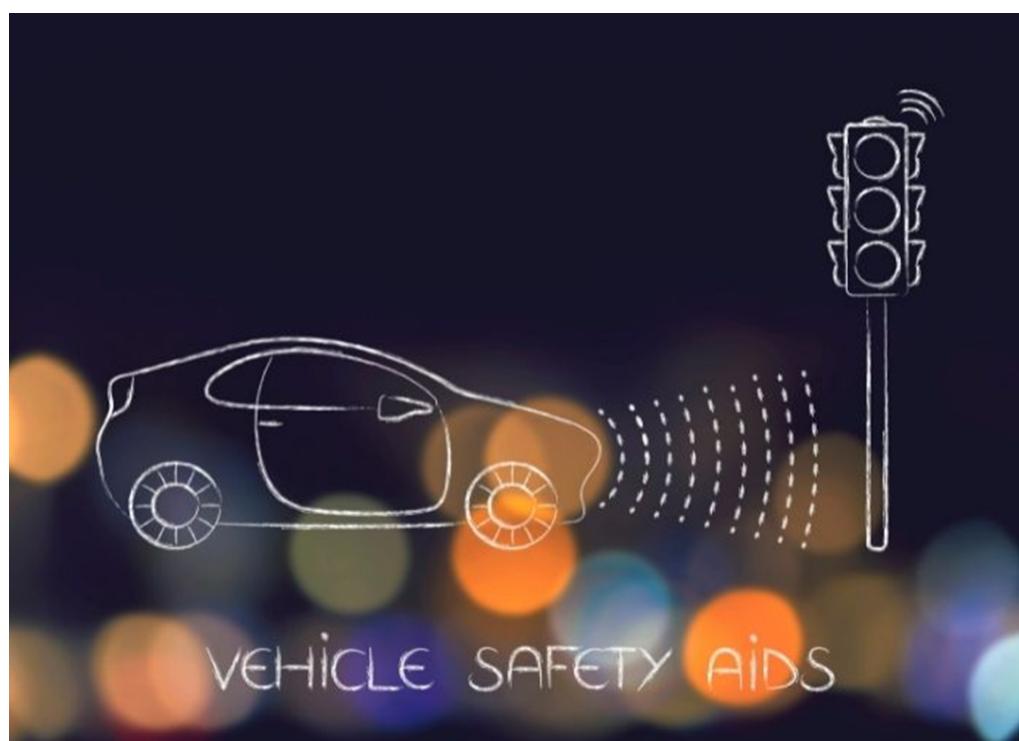
1.2 Input và Output:

Input: Một đoạn video ngắn ghi lại hoạt động giao thông xe cộ qua lại trong 1 khoảng thời gian.

Output: Một con số thể hiện số lượng xe mỗi loại và tổng số lượng phương tiện trong thời lượng video

1.3 Ứng dụng:

Bài toán này thường được ứng dụng để giám sát tình hình giao thông, có thể được sử dụng ở quy mô lớn, có thể dùng để phát hiện tai nạn giao thông bằng cctv camera, điều chỉnh lưu lượng giao thông đặc biệt vào giờ cao điểm ở nhiều nơi khác nhau, dữ liệu từ bài toán này còn có thể được sử dụng cho việc xây dựng cơ sở hạ tầng dựa vào.



PHẦN 2: BỘ DỮ LIỆU

2.1: Cách thức xây dựng bộ dữ liệu:

- Tự xây dựng dữ liệu, nhóm quay lại video giao thông ở 2 đoạn đường Song Hành và Võ Văn Ngân, mỗi đoạn khoảng 2-3 videos, tổng thời lượng khoảng 27 phút, 60FPS.

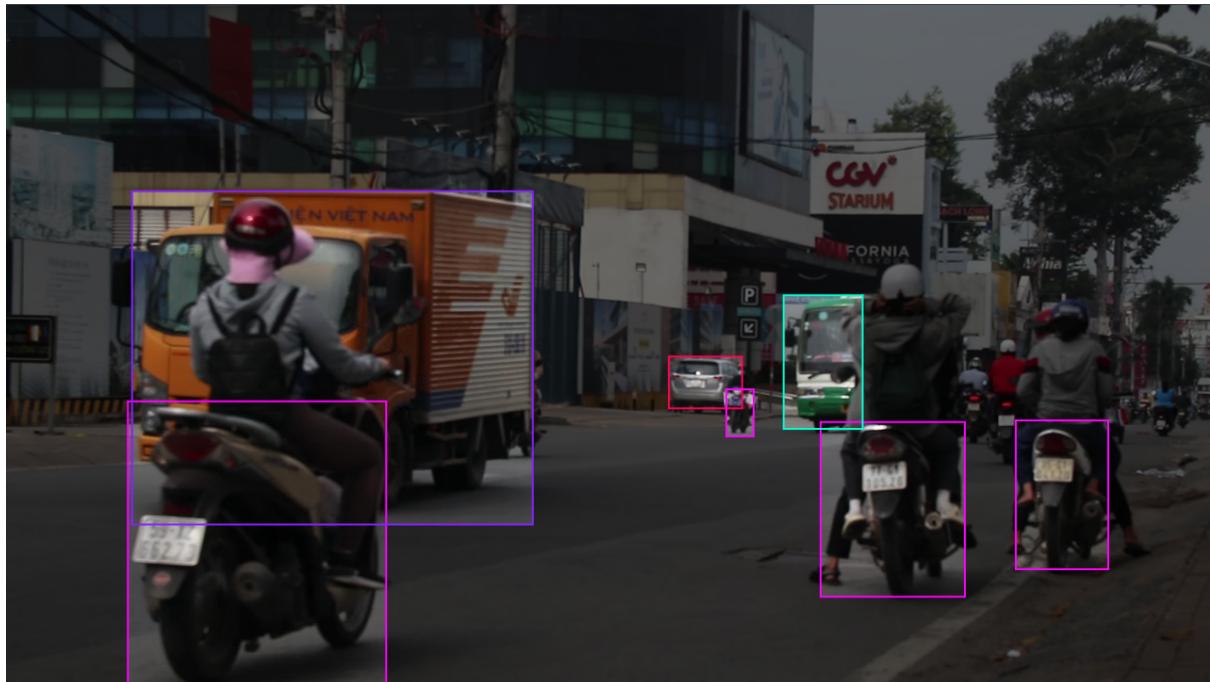


-Tách video thành từng frame, cứ mỗi 1s thì sẽ lấy 1 frame (video với tốc độ 60fps tức cứ 1 giây là chạy được 60 frame), tổng là 1511 tấm ảnh. Có thể sử dụng code để trích xuất frame sử dụng thư viện openCV:

```
import cv2
cap = cv2.VideoCapture('Link dẫn đến video')
success, image = cap.read()
count = 1
frame=0
while success:
    if (frame%60==0):
        cv2.imwrite('/content/datasets/image_%d.jpg' % count, image)
        print('saved image', count)
        count+=1
    success, image = cap.read()
    frame+=1
```

2.2: Tiền xử lý dữ liệu

-Sử dụng roboflow để gán nhãn với thứ tự và 4 class tương ứng: 0-Bus, 1-Car, 2-Motorbike và 3-Truck. Bộ dataset được chia thành 2 bộ train/val với tỉ lệ 80% / 20%, tương ứng train:1155 tấm và val:356 tấm. Trong đó, tổng số lượng object là 11018: bus:211, car:1242, moto:8753, truck:812.



2.3: Gán nhãn dữ liệu:

Nhóm đặt ra các quy tắc để trách sự sai lệch trong việc gán nhãn:

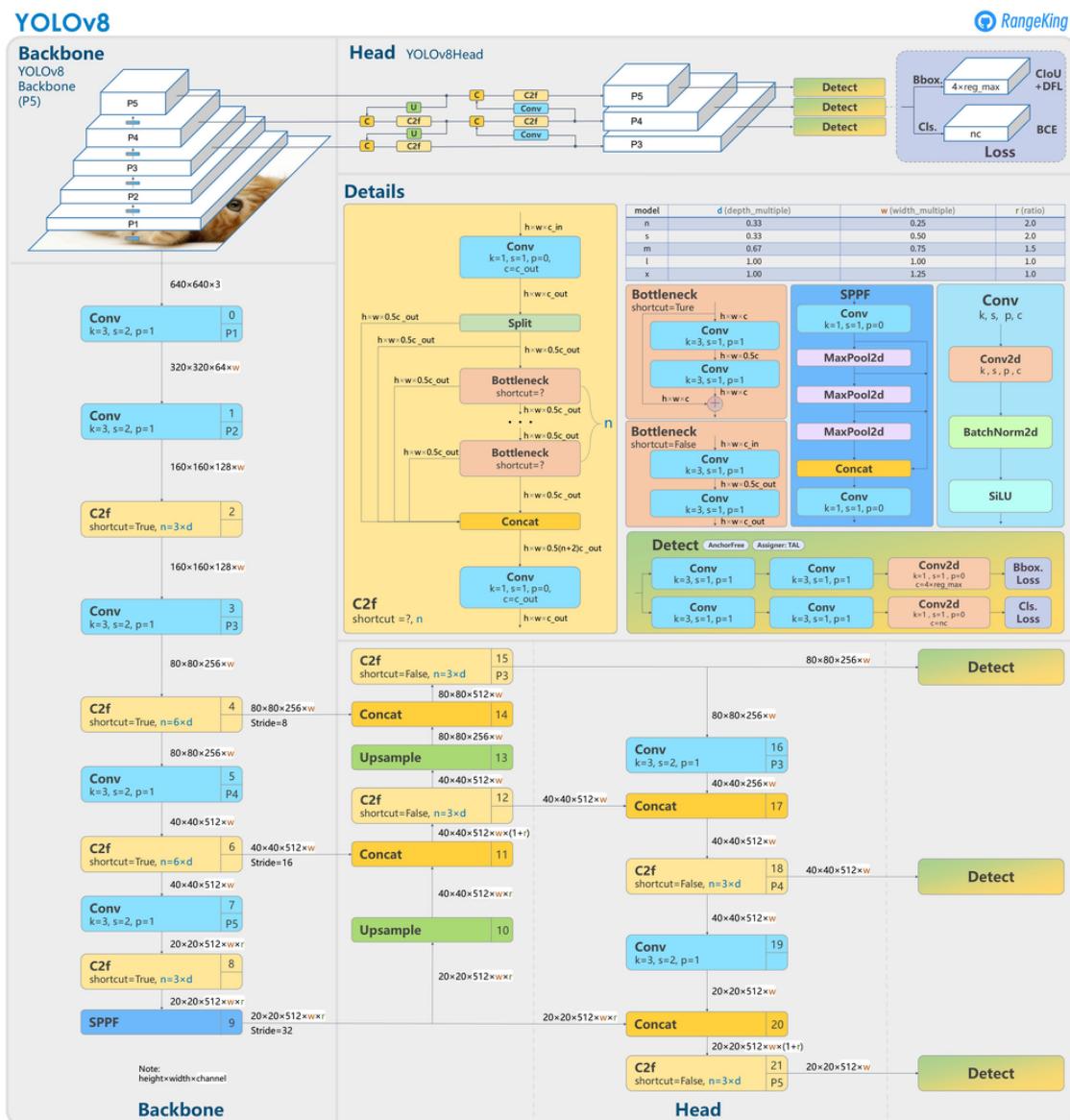
- Các bounding box phải bao sát vật thể.
- Đối với motorbike thì không gán nhãn bao gồm cả người lái xe.
- Những xe có bề ngoài giống truck hoặc bus như xe cứu hỏa, cần cẩu thì sẽ không gán nhãn.
- Không gán label cho những object bị che khuất toàn phần hay chỉ lộ 1 phần nhỏ.

PHẦN 3: MÔ HÌNH VÀ THUẬT TOÁN

3.1 Mô hình YOLOv8

YOLO được viết tắt của You Only Look Once là một model object detection và image segmentation được phát triển bởi Joseph Redmon and Ali Farhadi ở University of Washington. Phiên bản đầu tiên của yolo được ra mắt vào năm 2015 và dần nổi tiếng nhờ vào tốc độ và độ chính xác của nó.

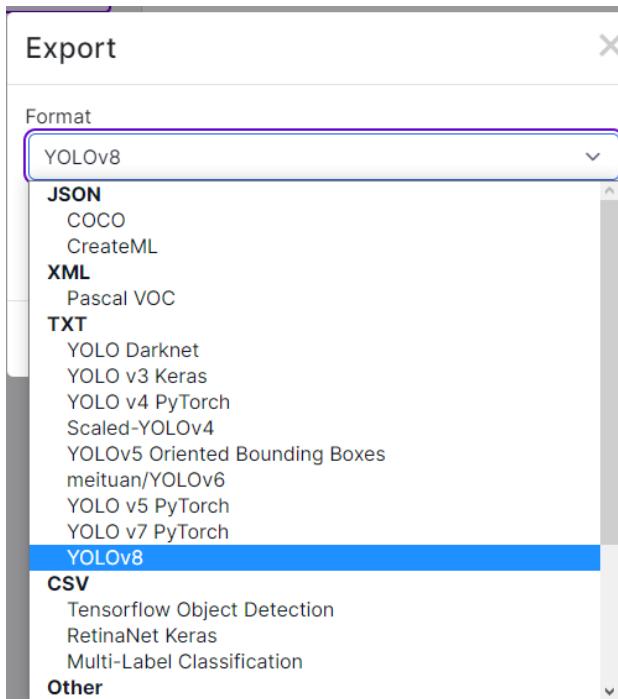
Mô hình YOLOv8 được nhóm tác giả Ultralytics xây dựng, được giới thiệu vào tháng 12 năm 2022 và liên tục cập nhật để phát triển đến nay.



.Hình minh họa cấu trúc YOLOv8

3.1.a: quá trình training:

- Chuẩn bị dataset: Roboflow có hỗ trợ export theo nhiều định dạng khác nhau dành cho từng loại model, ở đây mình dùng model YOLOv8 nên xuất file dạng txt.



- Mỗi bộ train/val được chia thành 2 phần là ảnh và label tương ứng với từng ảnh theo format id, x, y, w, h, mỗi dòng là 1 bounding box được gán trong ảnh:

```
2 0.03671875 0.61875 0.06953125 0.1484375
2 0.33359375 0.73046875 0.1484375 0.3046875
2 0.48671875 0.59609375 0.03671875 0.1046875
3 0.69375 0.58828125 0.10390625 0.125
2 0.91953125 0.59921875 0.02421875 0.0328125
2 0.4234375 0.67734375 0.1 0.2
```

Trong đó:

- ID là class của bounding box
- x và y, w và h là vị trí tâm, chiều rộng và cao tương ứng của bounding box.
- Sử dụng model để training:

Việc training đòi hỏi máy có GPU mạnh và cấu hình cao, vì vậy đối với những máy có cấu hình thấp có thể sử dụng google colab có hỗ trợ

GPU của sever dành cho việc training model. Không khuyến khích những máy có GPU mạnh training vì có thể làm giảm tuổi thọ của máy.

```
#Training model
!python train.py model=yolov8n.pt data=dataset/data.yaml epochs=50 imgsze=640
```

Có 5 file pretrain weights của YOLOv8 được train sẵn dựa trên tập dataset của COCO, với mức độ hiệu quả khác nhau theo từng size:

Model	mAP(IoU=0.5)	COCO mAP	COCO mAR(10)
YOLOv8n	43.94	31.74	34.89
YOLOv8s	53.03	39.01	41.04
YOLOv8m	59.43	44.87	45.97
YOLOv8l	62.04	47.37	47.74
YOLOv8x	63.57	48.49	48.50

File data.yaml sẽ bao gồm đường dẫn tới folder chứa data và số lượng, các class tương ứng:

```
train: dataset/train/images
val: dataset/valid/images

nc: 4
names: ['bus', 'car', 'motorbike', 'truck']

roboflow:
  workspace: ml-mvw1s
  project: vehicle-counting-ev2em
  version: 1
  license: CC BY 4.0
  url: https://universe.roboflow.com/ml-mvw1s/vehicle-counting-ev2em/dataset/1
```

Epochs tương ứng với số lần lặp lại và imgsze sẽ resize img cho việc training, mặc định là epochs=50 và imgsze=640.

- Nhận xét training:

- +Việc training càng lâu thì độ chính xác càng được cải thiện, tùy vào bộ dữ liệu training.

+Training đến khi thấy loss bão hòa, không còn thay đổi nhiều thì có thể dừng.

3.1.b: đánh giá mô hình training:

Mô hình được đánh giá bằng những performance metric sau:

Precision:

$$\text{Precision} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Positive}}$$

Recall:

$$\text{Recall} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Negative}}$$

Map:

Mean Average Precision được tính bằng công thức:

$$\text{Mean Average Precision} = \frac{1}{n} \sum_{k=1}^{k=n} AP_k$$

Với True Positive là hình ảnh detect đúng

False Positive là hình ảnh detect sai nội dung

False Negative là hình ảnh detect sai nội dung nhưng là ảnh đúng

n là số lượng các lớp

AP_k là precision trung bình của một lớp k

Model summary: 168 layers, 3006428 parameters, 0 gradients, 8.1 GFLOPs						
Class	Images	Instances	Box(P)	R	mAP50	mAP50-95
all	355	3767	0.621	0.475	0.557	0.302
bus	355	165	0.873	0.544	0.817	0.487
car	355	502	0.667	0.544	0.609	0.359
motorbike	355	2721	0.64	0.682	0.674	0.281
truck	355	379	0.301	0.132	0.129	0.0791

Speed: 3.3ms pre-process, 3.6ms inference, 0.0ms loss, 4.0ms post-process per image

Kết quả sau khi train thành công

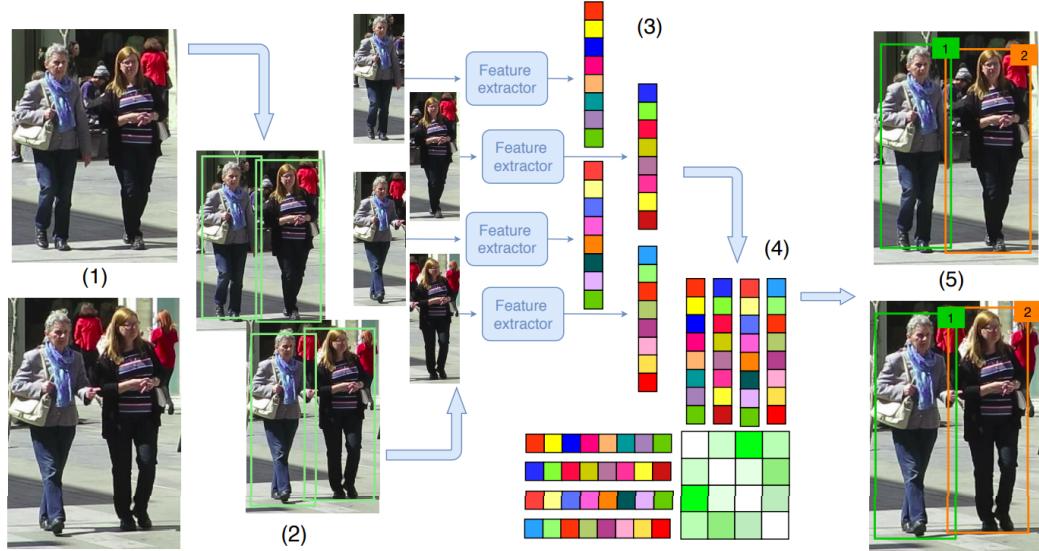
- Precision = 0.621 cho thấy trong các đối tượng mô hình dự đoán, có khoảng 62,1% dự đoán là đúng.
- Recall = 0.475 cho thấy mô hình có khả năng phát hiện khoảng 47.5% số lượng xe có trong ảnh.
- mAP50 = 0.557, cho thấy độ chính xác của mô hình đối với các mức ngưỡng khác nhau của độ tin cậy của phát hiện là 55.7%.

3.2 Mô hình deep sort :

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 Hungarian (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 bày 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.

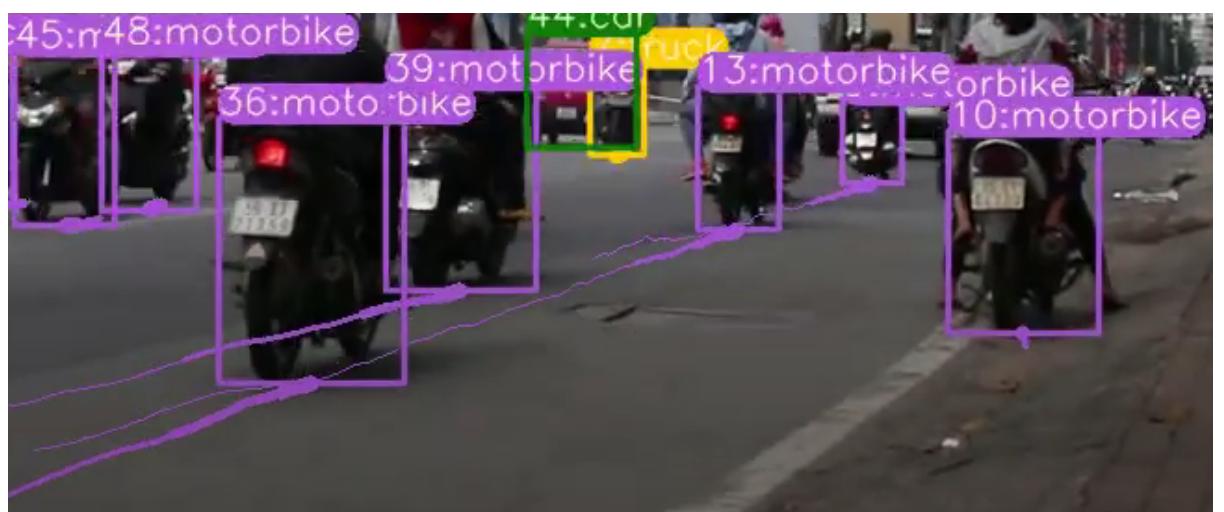


. Hình minh họa cách Deep Sort hoạt động

3.3 Kết hợp YOLOv8 và DeepSORT để đếm xe:

Ý tưởng:

- Sử dụng YOLO để tìm vị trí các phương tiện và tính bounding box của từng phương tiện.
- Mỗi phương tiện sẽ được gán 1 ID để theo dõi (sử dụng số thứ tự xe counting để gán). Trong trường hợp đối tượng bị khuất trong vài frame rồi xuất hiện lại, để tránh việc tính thêm là 1 object mới thì ta sẽ tính thời lượng mà xe biến mất, đặt ra quy ước nếu quá 10 frame thì sẽ bỏ ra khỏi tracking. Việc này được thể hiện bởi đoạn trail nhỏ sau bounding box thể hiện vị trí của xe trong 10 frame trước.



- c. Vẽ 1 vạch kẻ ảo nằm ngang hoặc thẳng đứng tương ứng với hướng xe chạy từ North - South (hay S-N) hoặc từ West về East (hay E-W). phương tiện đi ngang qua vạch kẻ ảo thì sẽ tính counting thêm 1.
- d. Và để phân hướng để đếm số xe theo làn đường bên trái và bên phải, ta sẽ có 2 list, khi tính theo trục x, nếu vị trí frame sau lớn hơn vị trí của frame trước (tức xe đi về phía bên phải) thì thêm hướng East vào chuỗi direction_str và ngược lại là thêm hướng West. Tương tự nếu tính theo trục y, vị trí frame sau lớn hơn vị trí frame trước thì xe chạy theo hướng South và ngược lại là chạy theo hướng North. Như vậy có thể phân hướng, tính số lượng từng xe theo 2 hướng làn đường trái phải.

```
def get_direction(point1, point2):
    direction_str = ""

    # calculate y axis direction
    if point1[1] > point2[1]:
        direction_str += "South"
    elif point1[1] < point2[1]:
        direction_str += "North"
    else:
        direction_str += ""

    # calculate x axis direction
    if point1[0] > point2[0]:
        direction_str += "East"
    elif point1[0] < point2[0]:
        direction_str += "West"
    else:
        direction_str += ""

    return direction_str
```

PHẦN 4: KẾT QUẢ

Đánh giá dựa trên độ chính xác (Precision) bằng công thức:

$$Precision(\%) = 100 - Error(\%)$$

$$Error(\%) = \frac{|Estimated - TrueNo|}{TrueNo} X 100$$

Sử dụng bộ dataset GRAM M-30 HD để test, dưới đây là kết quả thu được và so sánh với các tác giả *Yang et al.*, *Abdelwahab* và *Adson M. Santos* của bài báo ở mục tham khảo:

Tác giả	Video test	True No.	Estimated	Precision (%)
Yang et al.	M-30 HD	42	37	88.1
Abdel wahab	M-30 HD	42	42	100
Proposed Model	M-30 HD	42	42	100
Data của nhóm	M-30 HD	42	18	42.8

So sánh kết quả đánh giá với các tác giả trước và của nhóm

PHẦN 5: KẾT LUẬN

Mô hình vẫn có thể thực hiện tương đối ổn chức năng đếm xe cơ bản. Đồng thời vẫn thể hiện lỗi trong việc nhận diện xe, và tập train ở góc nhìn hạn chế nên hiệu suất chưa cao, vẫn chưa thể đánh giá cách kết quả cuối cùng.

Mô hình vẫn có thể phát triển trong tương lai bằng cách làm đa dạng dữ liệu và có thể đạt kết quả chính xác hơn bằng cách tối ưu hóa nhiều góc quay, điều kiện sáng khác nhau, cung cấp thêm nhiều dataset cho các class để làm tăng độ hiệu quả trong việc detect.

TÀI LIỆU THAM KHẢO:

<https://viblo.asia/p/sort-deep-sort-mot-goc-nhin-ve-object-tracking-phan-2-djeZ1m78ZWz> - Thuật toán DeepSORT

<https://github.com/ultralytics/ultralytics> - Model YOLOv8

<https://github.com/MuhammadMoinFaisal/YOLOv8-DeepSORT-Object-Tracking> - Model YOLOv8 DeepSORT for object tracking

<https://api.deepai.org/publication-download-pdf/vehicle-detection-and-counting-using-deep-learning-based-yolo-and-deep-sort-algorithm-for-urban-traffic-management-sy>

[stem](#) - Vehicle Detection and Counting using Deep Learning basedYOLO and Deep SORT Algorithm for Urban Traffic Management System

http://sibgrapi.sid.inpe.br/col/sid.inpe.br/sibgrapi/2020/09.15.17.37/doc/Article%20Sibgrapi_2020__Counting%20Vehicle%20with%20High-Precision_Final_CamaraRead.y.pdf - Counting Vehicle with High-Precision in Brazilian Roads Using YOLOv3 and Deep SORT

<https://gram.web.uah.es/data/datasets/rtm/index.html> - GRAM dataset

<https://www.youtube.com/watch?v=nnSCYdraVrA> - Training on custom dataset tutorial

<https://www.youtube.com/watch?v=9jRRZ-WL698>