ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH

Trường Đại học Công nghệ thông tin

ĐỒ ÁN CUỐI KỲ

Đề tài: Phân loại và đếm số lượng phương tiện lưu thông với mô hình YOLO

Môn học: CS114.ML – MACHINE LEARNING

Lớp: CS114.L2x.KHCL

Thành viên:

1. Nguyễn Văn Tài – 19520250
2. Nguyễn Ngọc Trưởng – 19522440
3. Trần Xuân Nhơn – 18521212

Giảng viên: Phạm Nguyễn Trường An – Lê Đình Duy

Chương I: Tổng quan

1. Mô tả bài toán

- Ngữ cảnh ứng dụng: Trong thời đại ngày nay, công nghệ thông tin hầu như đã thâm nhập vào toàn bộ các lĩnh vực đời sống xã hội. Xã hội ngày càng phát triển thì nhu cầu áp dụng các tiến bộ của công nghệ thông tin vào cuộc sống ngày càng cao để giải quyết những vấn đề phức tạp về y tế, giáo dục, giao thông... Ở nước ta những năm gần đây, cùng với sự phát triển của kinh tế xã hội thì số lượng phương tiện giao thông đường bộ cũng tăng lên nhanh chóng đi liền với đó là sự ùn tắc giao thông đường bộ. Thông qua việc xây dựng các thông số của dòng giao thông (số lượng phương tiện, loại phương tiện) theo thời gian thực sẽ được hệ thống trung tâm phân tích, tính toán để có những chiến lược phát triển ở những nút giao thông quan trọng. Mô hình nhận diện và cung cấp thông tin luồng phương tiện tham gia giao thông trong thời gian thực sẽ là tiền đề trong việc phân tích và đưa ra các biện pháp cải cách trong tương lai. Những cải cách phù hợp với thực tại sẽ góp phần giải tỏa ùn tắc cho các nhánh giao thông, giúp giảm thiểu thời gian chờ cho các phương tiện giao thông công cộng.

- Input: frame ảnh giao thông tại một nút giao thông (trích xuất từ camera của Sở giao thông vận tải Hồ Chí Minh)

- Output: nhận diện các loại phương tiện và trích xuất thông tin về luồng giao thông

2. Mô tả tổng quan về dữ liệu

- Dữ liệu được thu thập từ hệ thống camera giao thông của **Sở giao thông vận tải Thành phố Hồ Chí Minh**[1]
- Tổng dữ liệu thu thập được Sự phân bố và độ đa dạng các đối tượng sẽ được trình bày ở chương 3.
- Một số khó khăn trong việc thu thập dữ liệu:

- Đa số các camera trong thành phố điều có tầm nhìn rộng, nhiều phương tiện ở xa khó nhìn nhận diện được là đối tượng gì.

- Độ phân giải của camera thành phố cung cấp khá kém. Kích thước của mỗi frame ảnh thu được tối đa là 800 x 450 chiếm 45% (957 hình), kích thước 512x288 chiếm 43% (913 hình), kích thước 640x360 chiếm 12% (248 hình) so với tổng ảnh thu thập từ bộ dữ liệu.

  ➔ Gây ảnh hưởng đến việc nhận dạng và xác định đối tượng.

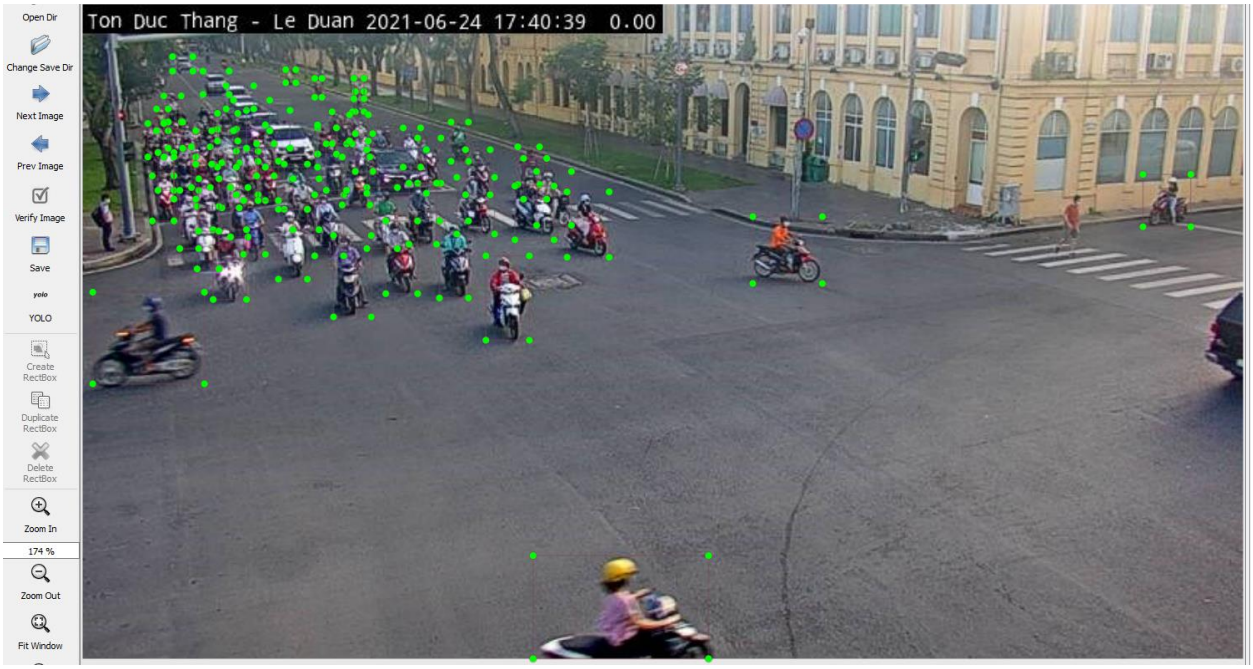
- Tỷ lệ sử dụng phương tiện giao thông ở Việt Nam không đồng điều, do đó dẫn đến sự chênh lệch tỷ lệ của các đối tượng trong bộ dữ liệu. Chúng ta có thể thấy được được phần mô tả chi tiết về bộ dữ liệu trong chương 3.

- Sau khi chạy thử model và quyết định tăng cường dữ liệu, cải thiện số lượng của một vài đối tượng. Tuy nhiên, do tình hình dịch bệnh COVID 19 và thành phố đang dãn cách xã hội, nên việc tăng cường dữ liệu cũng gặp nhiều khó khăn, mặc dù đã quan sát rất nhiều khung giờ. (nội dung sẽ được trình bày rõ trong những phần sau)

- Tốc độ quay của camera thành phố còn kém, có một vài khung hình bị mờ bị nhòe, không hiển thị rõ đối tượng.

- **Lý do tự thu thập dữ liệu**: Mặc dù các đối tượng mà nhóm định sẽ label bao gồm: truck, van, car, bicycle, motorbike đều có trong COCO Dataset, tuy nhiên để phù hợp hơn với tình hình giao thông Việt Nam, có nhiều đối tượng khác như xe ba bánh, ngoài ra thì các góc quay ở các camera giao thông đều tương đối rộng, các đối tượng như xe máy, xe đạp sẽ rất nhỏ và mờ, bị mất nét, do vậy nhóm quyết định tự xây dựng bộ dữ liệu từ camera

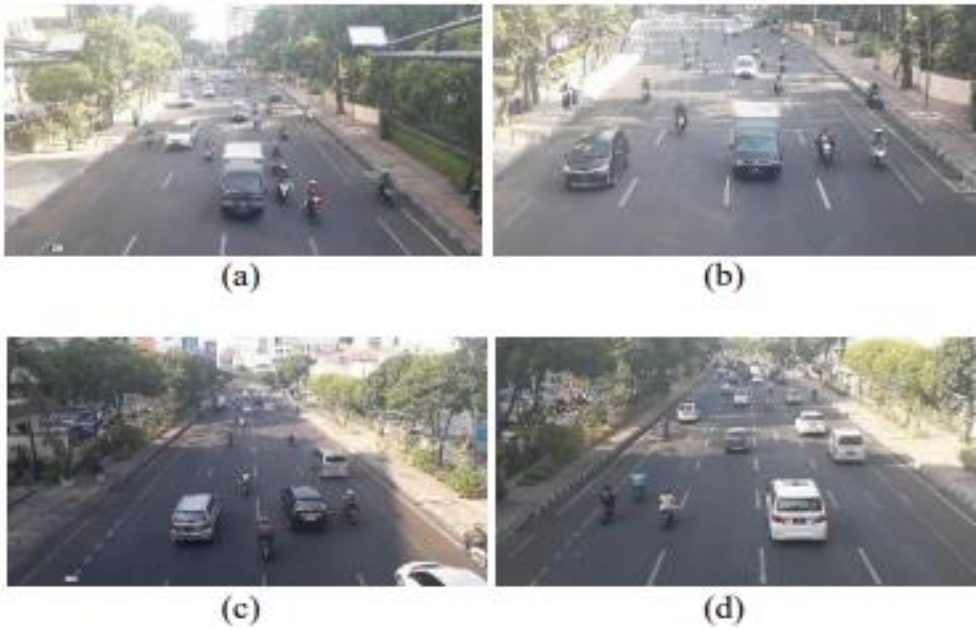
giao thông ở Việt Nam. Với góc quay trên xuống trực tiếp từ camera giao thống sẽ phù hợp với ngữ cảnh bài toán và những ứng dụng sau này. Nhóm chọn camera ở thành phố HCM vì mật độ giao thông trong thành phố khá cao so với các nơi khác, hơn nữa thành phố Hồ Chí Minh là một trong những số ít tỉnh, thành phố trực thuộc Trung ương công khai các hình ảnh giao thông thu được. (Một số tỉnh thành khác cũng có camera được công khai là Bình Định, Đà Nẵng).



Chương II: Các nghiên cứu trước

1. Bài nghiên cứu: A Simple Vehicle Counting System Using Deep Learning with YOLOv3 Model của Muhammad Fachrie[2]

- Model: YOLOv3
- Thu thập data:
    - Video ở độ phân giải full HD (1080p) được quay bằng camera điện thoại thông minh 8 mega-pixel trong 30 khung hình/giây.
    - Video được quay thủ công từ đỉnh cầu đi bộ ở một trong những thành phố lớn ở Indonesia.
    - Khoảng cách giữa mặt đường đến đỉnh cầu cao xấp xỉ 5 mét.
    - Điều kiện: đường một chiều, tình trạng giao thông bình thường (không xảy ra va chạm)



- Đánh giá
    - Model đạt độ chính xác cao

1080p and 30fps

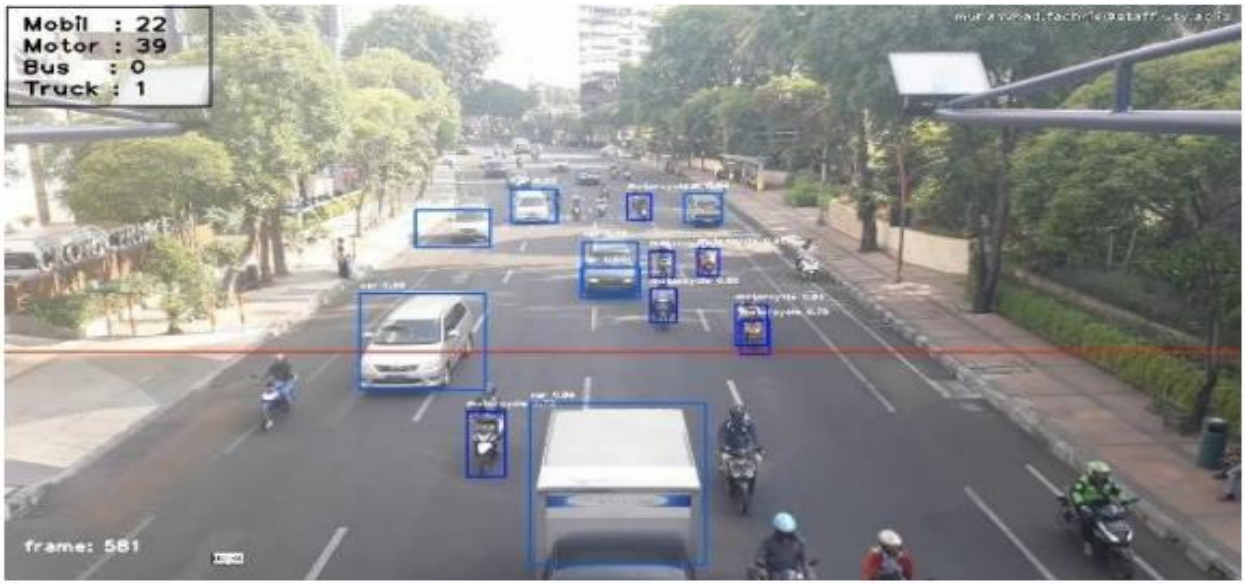
Video	Vehicle	Counting		Counting Error	Overall Accuracy
		Real	System		
Video 1 (frontside 1x zoom)	Car	149	150	+1	96.96%
	Motor	244	246	+2	
	Bus	1	2	+1	
	Truck	1	9	+8	
Video 2 (frontside 2x zoom)	Car	128	114	-14	87.09%
	Motor	232	219	-13	
	Bus	3	4	+1	
	Truck	1	20	+19	
Video 3 (backside 1x zoom)	Car	122	114	-8	90.24%
	Motor	173	170	-3	
	Bus	1	3	+2	
	Truck	1	17	+16	
Video 4 (backside 2x zoom)	Car	99	96	-3	94.52%
	Motor	247	246	-1	
	Bus	0	2	+2	
	Truck	1	14	+13	
				Average Accuracy	92.20%

1080p and 15fps

Video	Vehicle	Counting		Counting Error	Overall Accuracy
		Real	System		
Video 1 (frontside 1x zoom)	Car	149	148	-1	91.14%
	Motor	244	218	-6	
	Bus	1	2	+1	
	Truck	1	8	+7	
Video 2 (frontside 2x zoom)	Car	128	116	-12	85.71%
	Motor	232	212	-20	
	Bus	3	5	+2	
	Truck	1	19	+18	
Video 3 (backside 1x zoom)	Car	122	113	-9	76.09%
	Motor	173	128	-45	
	Bus	1	3	+2	
	Truck	1	16	+15	
Video 4 (backside 2x zoom)	Car	99	94	-5	84.15%
	Motor	247	209	-38	
	Bus	0	2	+2	
	Truck	1	11	+10	
				Average Accuracy	84.27%

Video	Vehicle	Real Count.	Counting (30 fps)		Accuracy (30 fps)		Counting (15 fps)		Accuracy (15 fps)	
			Prev.	New	Prev.	New	Prev.	New	Prev.	New
Video 1 (frontside 1x zoom)	Car	149	150	150	96.96%	**97.72%**	148	148	91.14%	**91.90%**
	Motor	244	246	246			218	218		
	Bus	1	2	2			2	2		
	Truck	1	9	6			8	5		
Video 2 (frontside 2x zoom)	Car	128	114	114	87.09%	**89.83%**	116	116	85.71%	**88.46%**
	Motor	232	219	219			212	212		
	Bus	3	4	4			5	5		
	Truck	1	20	10			19	9		
Video 3 (backside 1x zoom)	Car	122	114	113	90.24%	**91.25%**	113	112	76.09%	**77.44%**
	Motor	173	170	170			128	128		
	Bus	1	3	2			3	2		
	Truck	1	17	14			16	12		
Video 4 (backside 2x zoom)	Car	99	96	96	94.52%	**97.41%**	94	94	84.15%	**86.17%**
	Motor	247	246	246			209	209		
	Bus	0	2	2			2	2		
	Truck	1	14	4			11	4		

- Dự đoán:

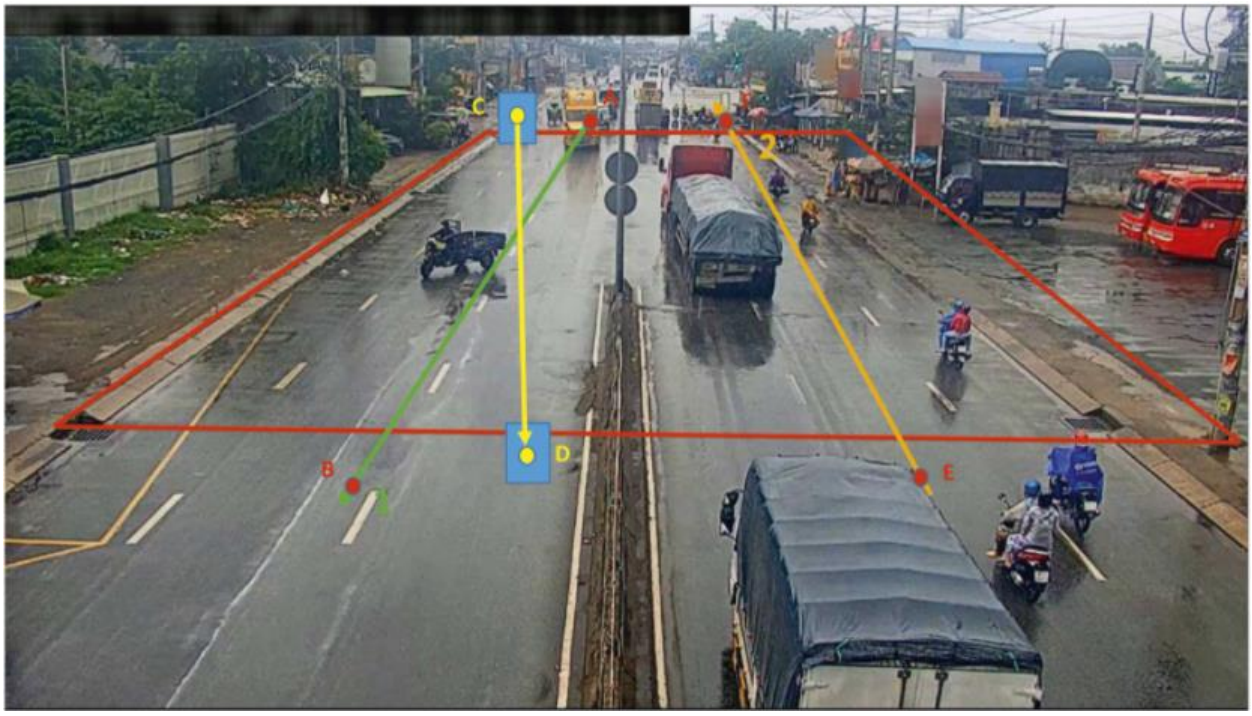


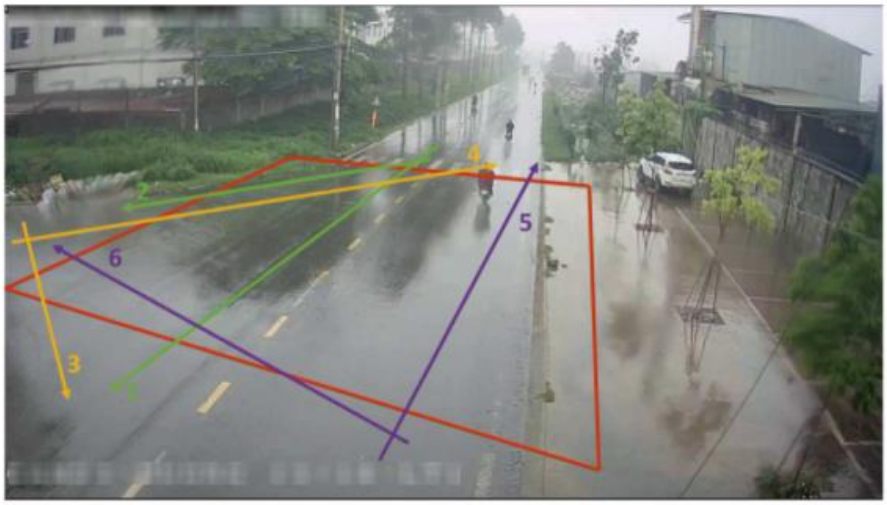
- Nhận định
    - Đối tượng "car" có độ chính xác đếm cao nhất, tiếp theo là "motorbike" và "bus", trong khi "truck" là kém nhất.
    - Nguyên nhân: Tốc độ khung hình của video và khả năng nhận diện phương tiện của YOLOv3

2. Bài nghiên cứu: **Towards AI-Based Traffic Counting System with Edge Computing[3] (**Duc-Liem Dinh, Hong-Nam Nguyen, Huy-Tan Thai, and Kim-Hung Le)

- Model: Khung đếm lưu lượng dựa trên AI chạy trong các thiết bị cạnh
- Thu thập thông tin luồng giao thông qua ba bước: phát hiện, theo dõi và đếm phương tiện
- Bộ dữ liệu bao gồm 22 video về các con đường đô thị và cảnh giao cắt được ghi lại bằng một số camera giao thông (23p mỗi video). Các video được quay từ nhiều góc độ, thời gian và điều kiện thời tiết khác nhau. Đối với mỗi video, chia thành video dài 18 phút để đào tạo và video dài 5 phút để thử

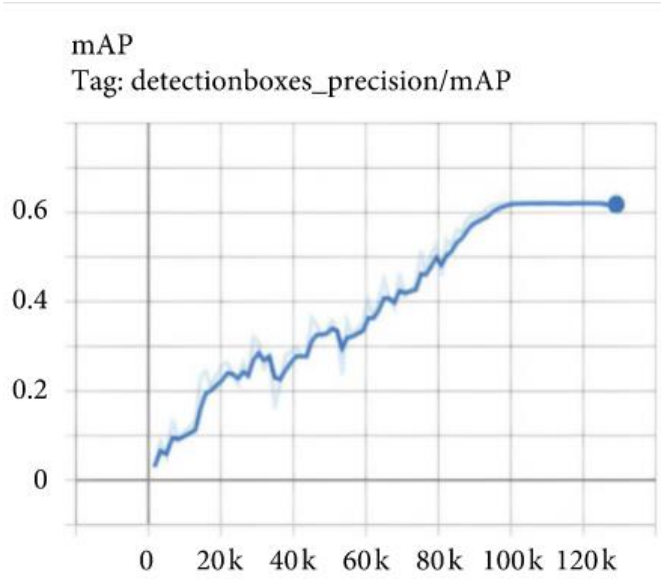
nghiệm. Sau đó, cắt video đào tạo thành các khung hình ảnh và gắn nhãn chúng sau mỗi 3 giây.

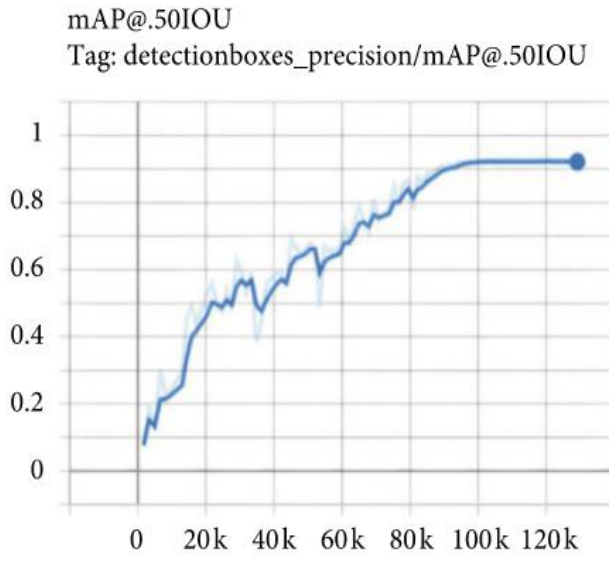




- - - Distance to edges of polygon
- - - Min distance to edges of polygon
● Center point of bounding box of the first frame appearing in ROO
○ Center point of bounding box of the last frame appearing in ROO

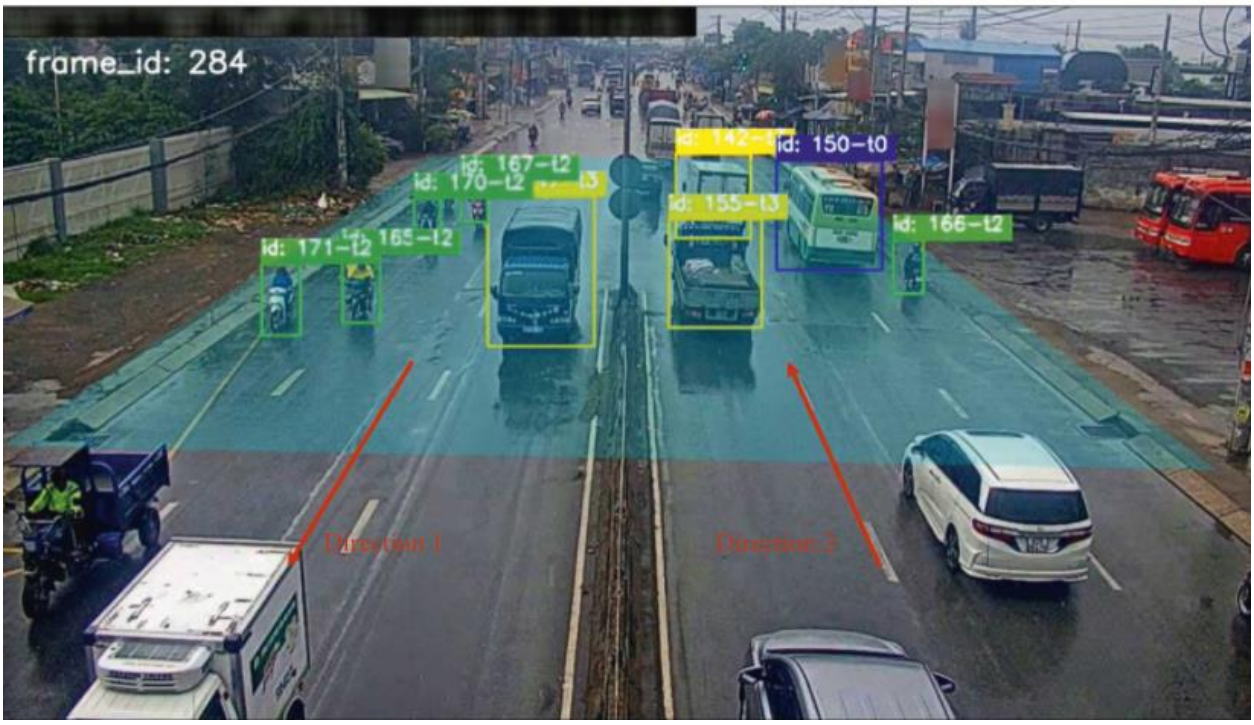
Kết quả: Hiệu suất cao nhất với độ chính xác 92,1% và thời gian suy luận trung bình là 37,313 ms (tương đương FPS trung bình là 26,8)





Đánh giá kết quả đếm lưu lượng thực nghiệm:

Type of vehicle	Direction 1			Direction 2		
	Experimental	Actual	Error (%)	Experimental	Actual	Error (%)
Type 1 (car, taxi,….)	16	20	−20	25	31	−19.4
Type 2 (motorcycle, bicycle,…)	165	155	6.4	109	121	−11.6
Type 3 (bus, coach bus,…)	3	4	−25	4	5	−20
Type 4 (truck, container,…)	35	41	−14.6	31	38	−18.4



Kết quả theo dõi của video thử nghiệm

Nhận xét:

Bài nghiên cứu sử dụng nhiều modun phức tạp để xây dựng một hệ thống đếm lưu lượng giao thông chính xác. Hệ thống đạt độ chính xác cao trên mỗi frame ảnh trong bộ dataset.

Chương III: Xây dựng bộ dữ liệu

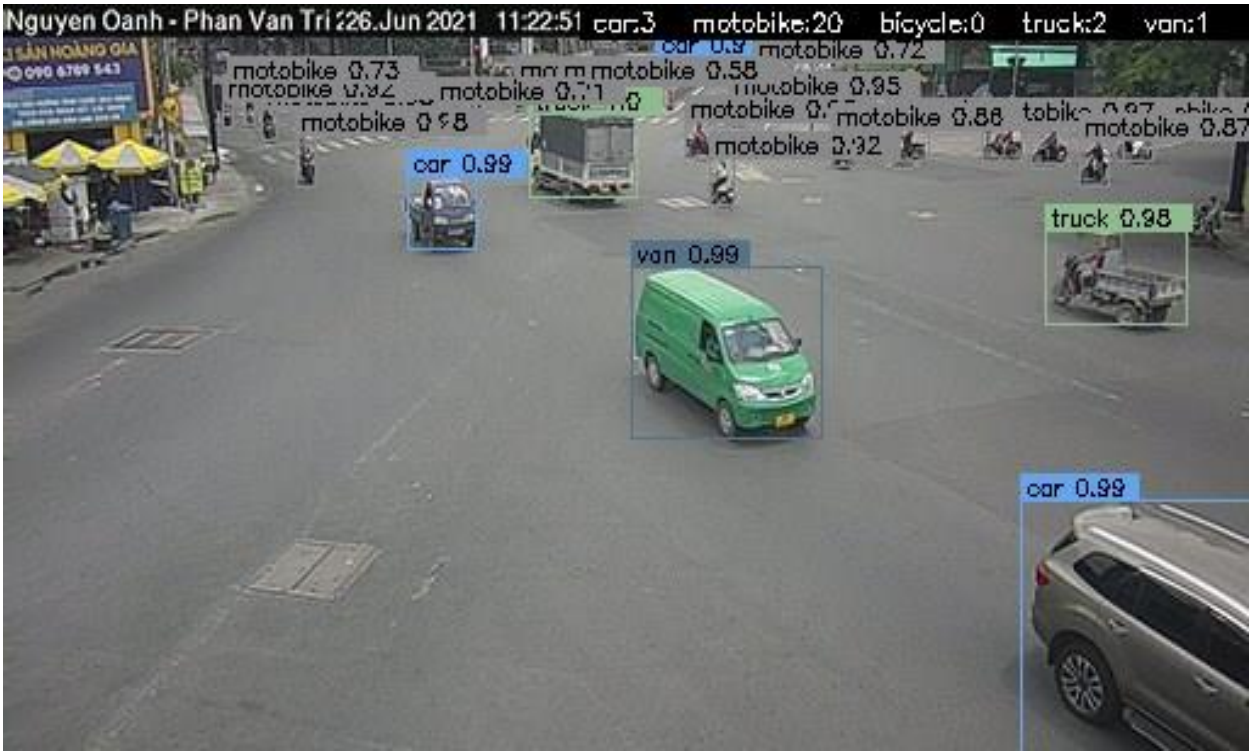
1. Tiêu chí dữ liệu:

- Ưu tiên chọn những camera có góc quay hẹp, hạn chế chọn camera có tầm nhìn quá bao quát toàn bộ đường, tránh việc khó nhận biết các loại phương tiện ở xa.

- Chọn kích thước tối đa của ảnh khi thu thập dữ liệu.Tập trung lấy ảnh ở những nút giao thông (ngã ba, ngã tư) có nhiều phương tiện giao thông di chuyển.
- Label tất cả các phương tiện có trên mặt đường.

2. Mô tả chi tiết dữ liệu:

- Số lượng class: 5 (motorbike, bicycle, van, truck và car)
  Sử dụng công cụ **LabelImg** để label các đối tượng.
- Mô tả chi tiết:
  - Motorbike: là những chiếc xe gắn máy, xe ba bánh nhưng không có thùng chở hàng phía sau.
  - Car: là tất cả các xe ô tô bốn bánh, không dùng để chở hàng, bao gồm xe 4 chỗ xe 7 chỗ.
  - Bicycle: là xe đạp thô sơ và xe đạp điện.
  - Van: là những xe khách và những xe tải đa dụng có thể vận tải người hoặc có thể chuyển đổi công dụng thành vận chuyển hàng hóa.
  - Truck: là những xe có thùng hàng phía sau gắn liền với xe gồm có xe cỡ nhỏ, xe chở container


Nguyen Oanh - Phan Van Tri 26.Jun 2021 11:22:51 car:3 motobike:20 bicycle:0 truck:2 van:1
motobike 0.73
motobike 0.58
motobike 0.95
motobike 0.88
motobike 0.87
motobike 0.98
motobike 0.92
car 0.99
truck 0.98
van 0.99
car 0.99


Nut giao Phu Nhuan 1 (Phan Dinh Phung) 2021-06-25 08:10:01
car:5 motobike:28 bicycle:3 truck:0 van:0
motobike 0.84
motobike 0.85
motobike 1.0
motobike 0.64
car 0.73
bicycle 0.81
motobike 0.99
motobike 1.0
motobike 0.89
bicycle 0.98
car 0.9
car 1.0
motobike 1.0
motobike 0.97
motobike 0.99

3. Độ đa dạng của bộ dữ liệu:

- Bộ dữ liệu trước khi tăng cường data

folder	car	motobike	bicycle	truck	van	number_of_photos
TrainData	2039	17896	208	1545	208	1194
TestData	461	3504	42	320	52	248
ValidateData	261	2627	35	227	33	226

- Bộ dữ liệu sau khi tăng cường dữ liệu

folder	car	motobike	bicycle	truck	van	number_of_photos
TrainData	2039	17896	208	1545	208	1194
TestData	461	3504	42	320	52	248
ValidateData	261	2627	35	227	33	226
ArgumentData	837	3569	45	863	254	454

Tất cả các ảnh sau khi tăng cường đều thêm cho tập train

4. Các trường hợp khó xử lý:

- Các loại phương tiện sát nhau, độ phân giải kém dẫn đến khó label.

- Các góc tối khó nhận diện loại phương tiện

- Phần dữ liệu (455 hình) dùng để tăng cường dữ liệu, tuy nhiên có chất lượng ảnh kém, ngoài ra còn có logo nằm trên đường, gây khó khăn cho việc gán nhãn cũng như là train và predict đối tượng.

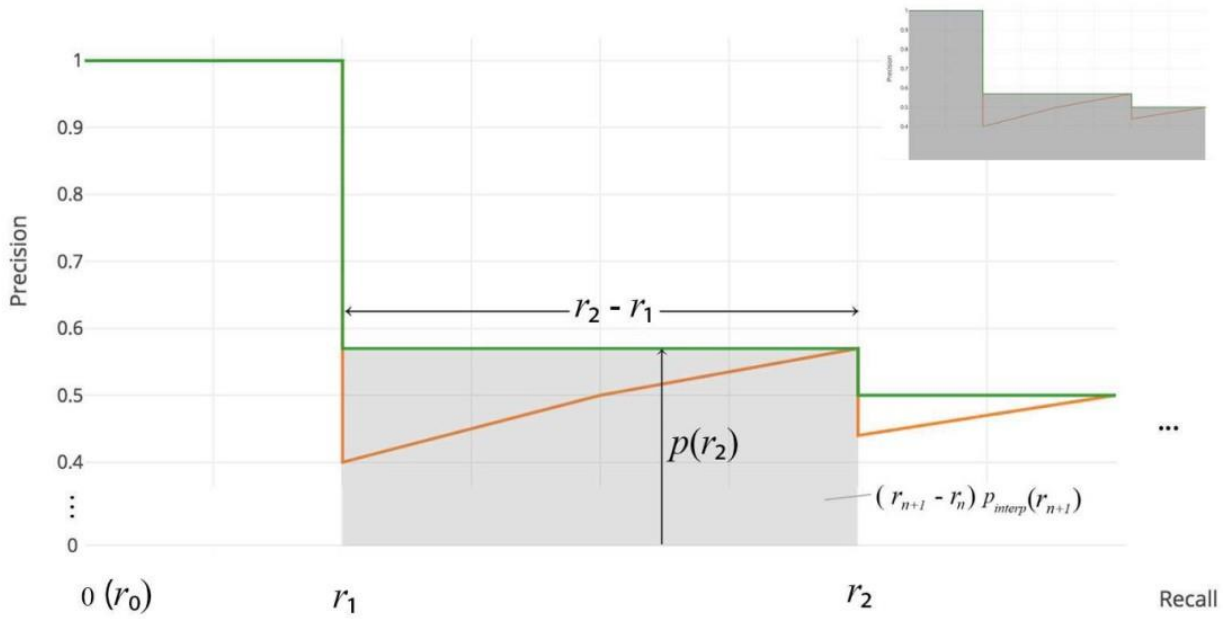
Chương IV. Training và đánh giá model

1. Chuẩn bị các file training

- File labelled_data.data: lưu trữ đường dẫn các file cần thiết để train YOLO
    - File classes.name: chứa thông tin 5 class
    - File validate_data.txt: chứa đường dẫn của các ảnh trong tập validate
    - File train_data.txt: chứa đường dẫn của các ảnh trong tập train
- File cfg.txt: tinh chỉnh các thông số cần thiết để train với bộ data tùy chỉnh của chúng ta, theo gợi ý của tác giả[4].
- File pre-trained sử dụng lại từ file pre-trained của tác giả

2. Tiêu chí đánh giá

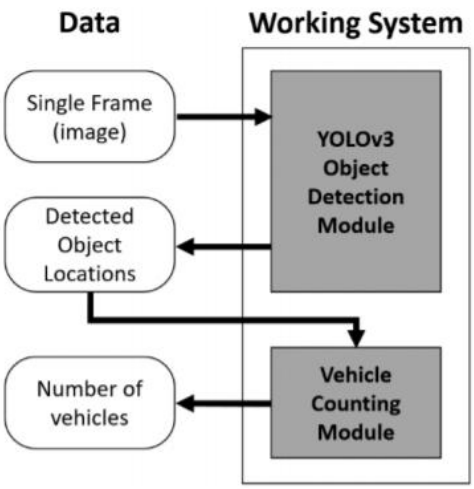
- mAP là 1 metric đánh giá được sử dụng phổ biến trong các bài toán về Object Detection[5].
- Trước khi tìm hiểu mAP được tính như thế nào thì chúng ta cần quan tâm một vài định nghĩa:
    - IoU (tỉ lệ giữa đo lường mức độ giao nhau giữa hai đường bao) để xác định hai khung hình có bị đè chồng lên nhau không.
      Tính bằng công thức:
      $$IoU = \frac{Diện\ tích\ giao\ nhau}{Diện\ tích\ hợp\ nhau}$$
    - TP: đối tượng nhận diện đúng (khi IoU > = 0.5)
      FP: đối tượng nhận dạng sai (khi IoU < 0.5)
      FN: đối tượng không được nhận dạng
    - Lúc này:
        - $Precision = \frac{Số\ lần\ dự\ đoán\ chính\ xác}{Tổng\ số\ lần\ dự\ đoán} = \frac{TP}{TP+FP}$
        - $Recall = \frac{Số\ lần\ dự\ đoán\ chính\ xác}{Số\ lần\ nhận\ dạng\ đúng\ có\ thể\ có} = \frac{TP}{TP+FN}$
    - Giả sử có n ngưỡng để tính giá trị của precision và recall, với mỗi ngưỡng cho một cặp giá trị (precision, recall), sau khi vẽ trên trục tọa độ và nối chúng lại với nhau. Ta xác định được:
      $$AP = \sum_{i=1}^{n} (R_i - R_{i-1}) * P_i$$



- mAP là giá trị trung bình của các AP được tính cho tất cả các lớp

3. Cấu tạo model

- Model được xây dựng với 2 modun chính:
  - Modun phát hiện đối tượng (loại phương tiện): đọc bức ảnh đầu vào và thực hiện phát hiện xe bằng thuật toán YOLO. Modun này cho kết quả vị trí của mọi phương tiện được phát hiện (giới hạn tọa độ)
  - Modun đếm số lượng phương tiện được phát hiện trong bức ảnh dựa trên tọa độ hoặc vị trí của phương tiện (sử dụng thuật toán OpenCV)



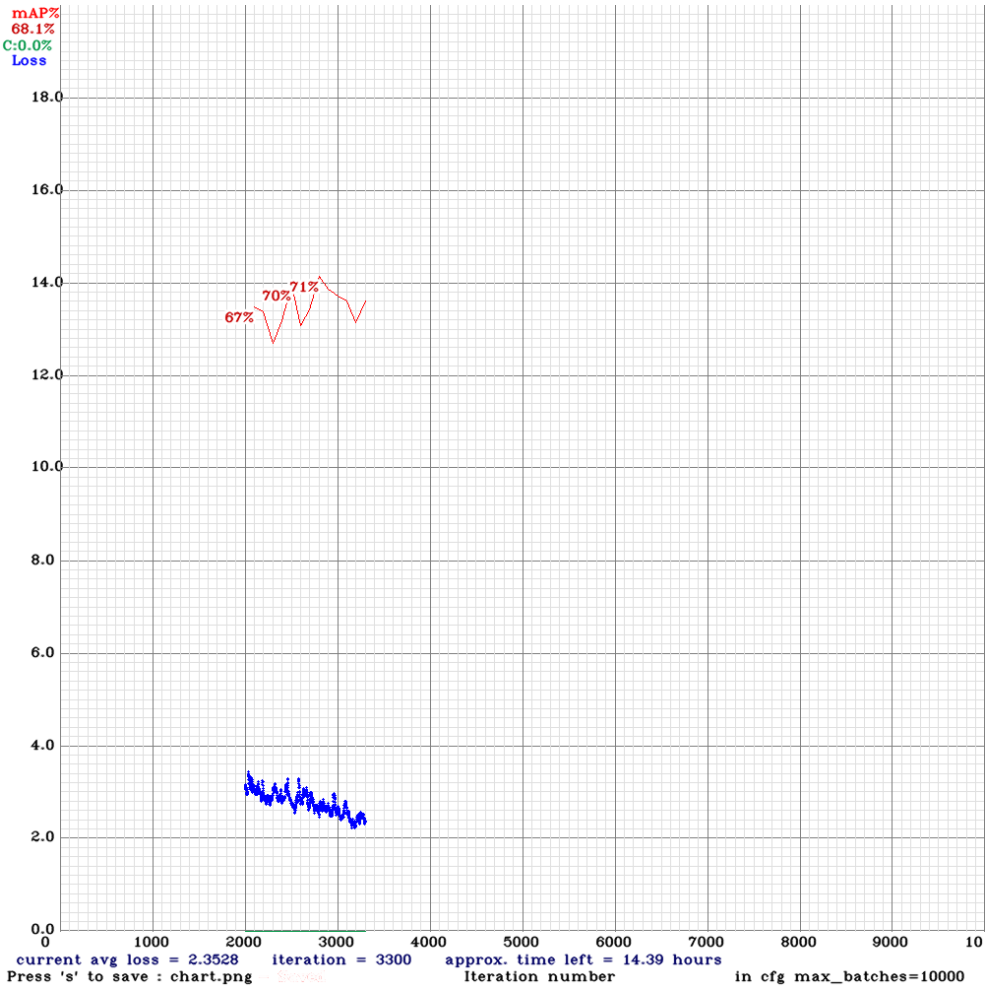
- Kết quả của modun phát hiện đối tượng đóng vai trong quan trong trong hệ thống này, nếu phương tiện không được nhận diện thì nó sẽ không được tính.

4. Train với YOLOv3

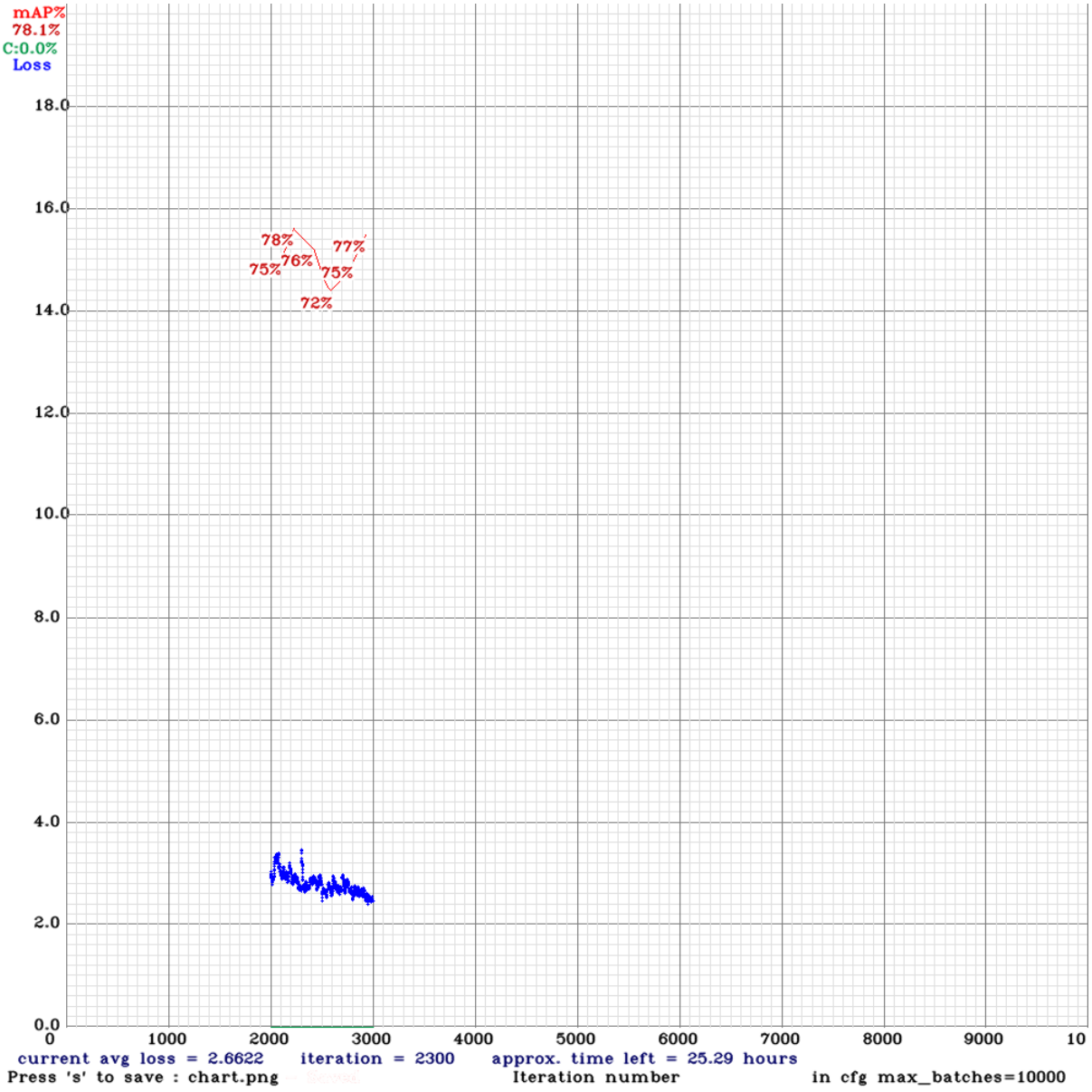
- Sử dụng model YOLOv3 có sẵn của darknet[6]
- Lần train đầu tiên:
    - Bộ dữ liệu: 1668 bức ảnh

folder	car	motobike	bicycle	truck	van	number_of_photos
TrainData	2039	17896	208	1545	208	1194
TestData	461	3504	42	320	52	248
ValidateData	261	2627	35	227	33	226

    - Số vòng train: 3300
    - Thời gian train > 15 tiếng
    - Đánh giá: sau quá trình train một khoảng thời gian, nhóm nhận thấy rằng mAP của model YOLOv3 khá thấp (cao nhất được 71%)



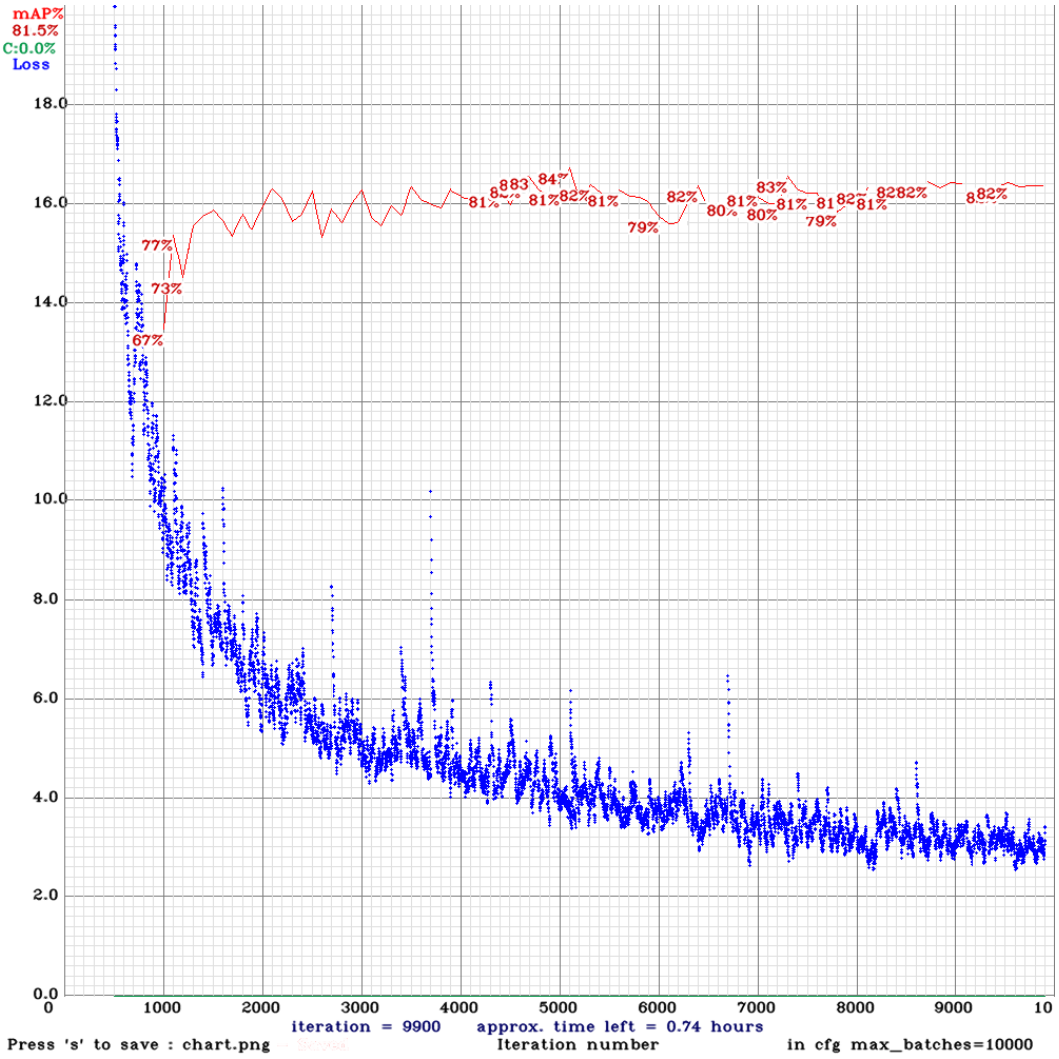
- Cải thiện model và train lại từ đầu với YOLO v3
    - Tăng cường dữ liệu: bổ sung thêm 454 bức ảnh
    - Tinh chỉnh file yolov3_cfg.txt (tham khảo từ model darknet)
        - Thêm cờ random=1 tăng độ chính xác bằng cách huấn luyện Yolo cho các độ phân giải khác nhau
        - Tăng độ phân giải mạng (height=608, width=608)
- Đánh giá:
    - Thời gian train > 10 tiếng
    - Số vòng train 3000 vòng
    - Chỉ số mAP cao nhất đạt được là 78.1%



- Nhận xét: Mặc dù chỉ số mAP của model đã tăng lên, tuy nhiên vẫn chưa đạt mục tiêu nhóm đặt ra (khoảng 85%), do đó nhóm đã chuyển sang sử dụng model YOLOv4
- Nhận định của nhóm:
    - Chất lượng ảnh và các object không hoàn toàn phù hợp với tiêu chuẩn dữ liệu của YOLOv3
    - Các thông số tinh chỉnh model chưa hợp lý.
    - Sự chênh lệch giữa các loại phương tiện dẫn đến tỉ lệ dự đoán của mỗi loại có sự khác biệt đáng kể.

5. Train với YOLOv4

- Nhằm cải thiện độ chính xác cho hệ thống, nhóm quyết định sử dụng YOLOv4 để làm modun nhận diện loại phương tiện giao thông
- Nguồn tham khảo: https://github.com/AlexeyAB/darknet
- Thiết lập file:
    - Sử dụng file pre-trained weights-file: yolov4.conv.137
    - Sử dụng file yolov4-custom.cfg và sửa đổi để phù hợp với data của nhóm theo hướng dẫn của tác giả
        - batch = 64
        - classes = 5
        - subdivisions = 16
        - max_batches = 10000
        - width=608
        - height=608
        - filters = 30
- Thời gian train mô hình: > 30 tiếng
- Quá trình train:
    - Sau train được 3200 vòng với bộ data cũ (tương tự với YOLOv3) chỉ số mAP tốt nhất là 82%.
    - Sau khi tăng cường thêm data (454 hình) kết quả đến vòng thứ 5100 thì được mAP tốt nhất là 84%.
    - Sau đó, những iteration tiếp theo thì mAP có xu hướng giảm, đến iteration 8200 thì nhóm quyết định phân bố lại data, sử dụng 200 tập hình của tập test bỏ vào tập train tuy nhiên mAP trên tập validate vẫn giữ trung bình là 82%.



- Nhận xét: chỉ số mAP của YOLOv4 có tăng hơn so với YOLOv3 (cao nhất là 83.56% tại iteration 5100)
- Nhận định:
    - Mặc dù chỉ số mAP có tăng nhưng không tăng nhiều so với YOLOv3, cho dù đã tăng cường dữ liệu
    - Nguyên nhân chủ yếu là do sự không đồng đều trong chất lượng hình ảnh làm ảnh hưởng đến khả năng nhận diện của YOLOv4
    - Các object của các class có sự chênh lệch nhiều dẫn đến độ chính xác của các class khác nhau (tác giả khuyến nghị data cần thiết là 2000 * số class ảnh)
    - Chỉ số mAP không cao do ảnh hưởng một phần từ hoạt động gán nhãn (vì nhiều người gán nhãn khác nhau nên có thể có sự sai lệch giữa các box được label).

6. Đánh giá model YOLOv4

- Kết quả đánh giá trên tập test thu được ở iteration 5100 (có mAP trên tập validate cao nhất là 83.56%)

```
248
 detections_count = 9533, unique_truth_count = 4379
class_id = 0, name = car, ap = 91.03%           (TP = 418, FP = 94)
class_id = 1, name = motobike, ap = 91.15%      (TP = 3241, FP = 871)
class_id = 2, name = bicycle, ap = 57.15%       (TP = 28, FP = 20)
class_id = 3, name = truck, ap = 90.54%         (TP = 287, FP = 81)
class_id = 4, name = van, ap = 69.62%           (TP = 34, FP = 13)

 for conf_thresh = 0.25, precision = 0.79, recall = 0.92, F1-score = 0.85
 for conf_thresh = 0.25, TP = 4008, FP = 1079, FN = 371, average IoU = 62.32 %

 IoU threshold = 50 %, used Area-Under-Curve for each unique Recall
 mean average precision (mAP@0.50) = 0.798984, or 79.90 %
Total Detection Time: 101 Seconds
```

- Kết quả đánh giá trên tập test thu được ở iteration 9900 (có mAP trên tập validate là 82.02%)

```
248
 detections_count = 7895, unique_truth_count = 4379
class_id = 0, name = car, ap = 95.50%           (TP = 436, FP = 63)
class_id = 1, name = motobike, ap = 93.85%      (TP = 3285, FP = 641)
class_id = 2, name = bicycle, ap = 71.14%       (TP = 29, FP = 12)
class_id = 3, name = truck, ap = 93.63%         (TP = 294, FP = 58)
class_id = 4, name = van, ap = 80.41%           (TP = 41, FP = 19)

 for conf_thresh = 0.25, precision = 0.84, recall = 0.93, F1-score = 0.88
 for conf_thresh = 0.25, TP = 4085, FP = 793, FN = 294, average IoU = 67.71 %

 IoU threshold = 50 %, used Area-Under-Curve for each unique Recall
 mean average precision (mAP@0.50) = 0.869060, or 86.91 %
Total Detection Time: 13 Seconds
```

- Nhận xét:
    - Nhìn chung class “bicycle” và “van” có chỉ số AP thấp nhất đó là nguyên nhân chính dẫn đến mAP của model thấp, đây là điều dễ dàng dự đoán được trong quá trình phân tích dữ liệu số lượng đối tượng này trong tập dataset quá ít để máy có thể học được với độ chính xác cao.
- Chỉ số AP của class “car”, “truck”, “motorbike” tương đối cao >90%, nhìn vào dữ liệu có trong tập dataset ta có thể thấy được số lượng các class này đều >2000, do vậy chúng ta có thể kết luận rằng để cải thiện model này thì chúng ta cần tăng cường dữ liệu cho class “bicycle” và “van”.

7. Phân tích

a. Mẫu dữ liệu model dự đoán sai

- Lỗi nhận diện sai class

Nhận diện người đi đường thành class “bicycle”

Nhận diện class “motorbike” thành class “bicycle”

Nhận diện class “car” thành class “truck”

- Lỗi không nhận diện được vật thể

Không nhận diện được object thuộc class “truck”

Không nhận diện được object thuộc class “bicycle”

b. <u>Nhận định của nhóm:</u>

- Chất lượng ảnh không đạt dạng tốt nhất để có thể dự đoán tốt
- Do vài class có số lượng object ít nên tỉ lệ dự đoán thấp, dẫn đến không thể nhận diện đầy đủ đặc trưng của class gây nên dự đoán nhầm class cho các đối tượng không rõ ràng.
- Góc ảnh chụp khác nhau không đa dạng nên một số phương tiện như truck, van sẽ có đặc trưng giống nhau (đầu xe van giống xe trục ở góc chụp chỉ thấy đầu xe).
- Các vật thể như cột đèn đường gây nhiễu ảnh làm model không nhận diện một số phương tiện.
- Nhìn chung, class “car” và “motorbike” có độ chính xác khá tốt (do số lượng object tương đối nhiều, phù hợp với tình hình giao thông Hồ Chí Minh). Bên cạnh đó, class “bicycle” và “van” có độ chính xác thấp nhất (số lượng object thấp nhất).

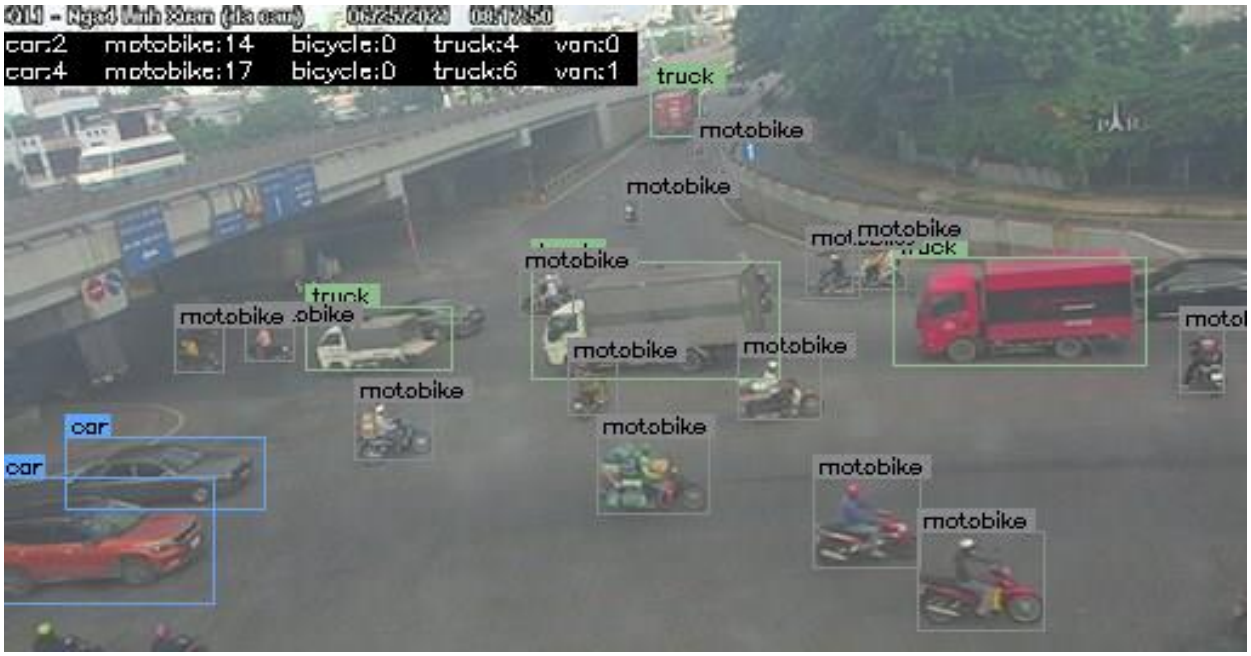
c. So sánh dự đoán YOLOv3 và YOLOv4:

Ghi chú: Ở góc trên cùng bên phải

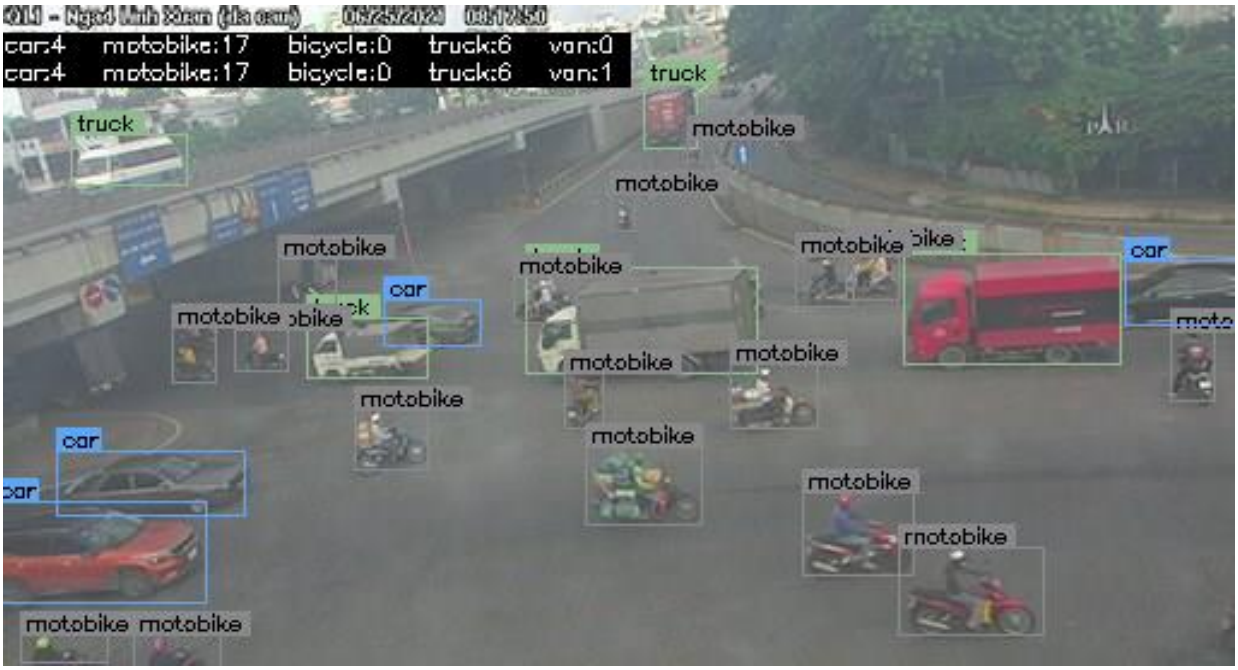
Dòng đầu tiên là số lượng phương tiện model nhận diện được

Dòng thứ hai là số lượng phương tiện thực tế

- YOLOv3



- YOLOv4



- Ảnh ban đầu

-YOLOv3

- YOLOv4

Nhận xét:

YOLOv4 có khả năng nhận diện chính xác hơn so với YOLOv3, do đó số phương tiện được đếm của YOLOv4 sẽ chính xác hơn so YOLOv3.

So với YOLOv3 thì YOLOv4 có những cải tiến để nhận diện vật thể nhanh và chính xác hơn.

Nhìn chung, chất lượng ảnh thu thập ảnh hưởng đến quá trình phân tích và nhận diện vật thể của cả hai model trên.

Chương V. Ứng dụng và hướng phát triển.

- Từ khả năng nhận dạng được các đối tượng phương tiện giao thông, chúng ta có thể đếm được các đối tượng có trên frame ảnh. Tuy nhiên, để vận dụng việc đếm xem frame ảnh có bao nhiêu đối tượng phương tiện giao thông trong thực tế là một điều vô nghĩa. Thực tế, chúng ta có thể vận dụng nó trong việc đếm phương tiện giao thông trên một làn đường hoặc nhiều hơn thế, và từ đó chúng ta có thể biết được số lượng các phương tiện đang lưu thông trong khung thời gian thực nào đó. Việc đó sẽ giúp ích rất nhiều điều tiết được lưu lượng giao thông, các đèn giao thông sẽ hoạt động phụ thuộc vào từng thời điểm trong ngày một cách tự động và thông minh.
- Với ứng dụng thực tế ấy, nhóm đã tìm hiểu và thực hiện demo vận dụng model nhận dạng đối tượng phương tiện giao thông đã train trước đó (sử dụng YOLOv4) để lưu lượng di xe di chuyển trong một thời điểm xác định

Nội dung demo:

- Sử dụng video trích xuất được từ camera giao thông thành phố Đà Nẵng (link youtube)
- Sử dụng code trong hướng dẫn (link youtube) (sử dụng thư viện OpenCV). Tuy nhiên thì nhóm có thêm hàm để lọc lại các box, giúp cải thiện khả năng đếm.
- Mô tả: Sử dụng thư viện OpenCV để vẽ 2 line, đối đượng nằm giữa 2 line này thì sẽ được nhận diện và thêm vào danh sách các đối tượng, sau mỗi frame thì sẽ cập nhập lại danh sách, đối tượng nào đi qua khỏi line “end” thì sẽ tăng giá trị biến đếm của đối tượng đó.

Nhận xét:

Vì sử dụng YOLO nhưng thời gian detect trên mỗi frame ảnh là rất lớn: với phiên bản YOLOv4 Tiny là 0.14348s , với phiên bản YOLOv4 Custom Data là 0.82547s, khi chạy trên máy Intel(R) Core(TM) i7-8565U CPU @ 1.80GHz 1.99 GHz, RAM 8,00 GB. Do vậy để có thể ứng dụng thực tế với thời gian thực lấy video trực tiếp từ youtube thì cần phải cải thiện rất nhiều.


car : 00
motorbike : 12
bicycle : 00
truck : 00
van : 01
www.0511.vn


car : 02
motorbike : 18
bicycle : 00
truck : 00
van : 01
motorbike: 77.67%
motorbike: 98.58%
www.0511.vn

Tài liệu tham khảo

[1] **http://giaothong.hochiminhcity.gov.vn/map.aspx**

[2] (Fachrie, M. (2020). A Simple Vehicle Counting System Using Deep Learning with YOLOv3 Model. *Jurnal RESTI (Rekayasa Sistem Dan Teknologi Informasi)*, *4*(3), 462-468.)

[3] Dinh, D. L., Nguyen, H. N., Thai, H. T., & Le, K. H. (2021). Towards AI-Based Traffic Counting System with Edge Computing. *Journal of Advanced Transportation*, *2021*.

[4] AlexeyAB/darknet: YOLOv4 / Scaled-YOLOv4 / YOLO - Neural Networks for Object Detection (Windows and Linux version of Darknet) (github.com)

[5] https://devai.info/2020/12/17/tim-hieu-mapmean-average-precision-danh-gia-mo-hinh-object-detection-su-dung-yolov4/

[6] https://github.com/AlexeyAB/darknet