BỘ GIÁO DỤC VÀ ĐÀO TẠO
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN
KHOA KHOA HỌC MÁY TÍNH

ĐỒ ÁN MÔN HỌC

CS338 - NHẬN DẠNG

DETECTING, TRACKING AND COUNTING VIETNAM TRAFFIC VEHICLE

Giảng viên hướng dẫn	**:**	**ThS. Đỗ Văn Tiến**
Sinh viên thực hiện 1	**:**	**Phạm Thanh Lâm**
Mã sinh viên 1	**:**	**21520055**
Sinh viên thực hiện 2	**:**	**Đoàn Lê Tuấn Thành**
Mã sinh viên 2	**:**	**21521438**
Sinh viên thực hiện 3	**:**	**Trương Quang Nghĩa**
Mã sinh viên 3	**:**	**21522376**
Sinh viên thực hiện 4	**:**	**Hoàng Minh Hiếu**
Mã sinh viên 4	**:**	**21520232**
Lớp	**:**	**CS338.O21**

Tp HCM, tháng 6 năm 2024

BẢNG PHÂN CÔNG THỰC HIỆN ĐỒ ÁN MÔN HỌC

Họ tên SV1: **Phạm Thanh Lâm** MSSV: **21520055**	Họ tên SV2: **Đoàn Lê Tuấn Thành** MSSV: **21521438**
1. Thu thập dữ liệu	1. Thu thập dữ liệu
2. Huấn luyện mô hình	2. Huấn luyện mô hình
3. Viết báo cáo đồ án	3. Viết báo cáo đồ án
4. Làm slide thuyết trình	4. Làm slide thuyết trình
Họ tên SV3: **Trương Quang Nghĩa** MSSV: **21522376**	Họ tên SV4: **Hoàng Minh Hiếu** MSSV: **21520232**
1. Thu thập dữ liệu	1. Thu thập dữ liệu
2. Huấn luyện mô hình	2. Deploy mô hình
3. Viết báo cáo đồ án	3. Viết báo cáo đồ án
4. Làm slide thuyết trình	4. Làm slide thuyết trình

SV thực hiện 1 *(Ký tên)* **Phạm Thanh Lâm**	**SV thực hiện 2** *(Ký tên)* **Đoàn Lê Tuấn Thành**
SV thực hiện 3 *(Ký tên)* **Trương Quang Nghĩa**	**SV thực hiện 4** *(Ký tên)* **Hoàng Minh Hiếu**

LỜI CẢM ƠN

Chúng em xin bày tỏ lòng biết ơn chân thành đến Th. s Đỗ Văn Tiến, giảng viên giảng dạy chúng em môn học Nhận dạng. Sự hướng dẫn và kiến thức sâu rộng của thầy đã vô cùng quý giá trong suốt quá trình thực hiện làm đồ án môn học và viết báo cáo về chủ đề "Detecting, tracking and counting vietnam traffic vehicle".

Chúng em cũng xin gửi lời cảm ơn đến các thành viên trong nhóm vì sự tận tâm và tinh thần đồng đội trong việc hoàn thành báo cáo này. Những đóng góp và ý kiến của từng thành viên đã rất cần thiết trong việc thực hiện nghiên cứu phương pháp, phân tích dữ liệu và đưa ra kết luận.

Ngoài ra, chúng em xin cảm ơn những cá nhân đã hỗ trợ, góp ý và giúp đỡ trong quá trình thực hiện công việc. Những ý kiến quý báu và góp ý xây dựng của họ đã góp phần quan trọng vào chất lượng của báo cáo này.

Cuối cùng, chúng em xin ghi nhận các nguồn tài liệu, tham khảo và công cụ đã đóng vai trò quan trọng trong dự án cuối cùng của chúng em. Sự sẵn có của các tài liệu này đã làm phong phú thêm hiểu biết và kiến thức của chúng em về chủ đề và hỗ trợ thực hiện đồ án.

Chúng em xin chân thành cảm ơn.

Sinh viên thực hiện

Hoàng Minh Hiếu - Phạm Thanh Lâm

Đoàn Lê Tuấn Thành – Trương Quang Nghĩa

NHẬN XÉT CỦA GIẢNG VIÊN HƯỚNG DẪN

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

Tp.HCM, ngày … tháng 06 năm 2024

GVHD

ThS. Đỗ Văn Tiến

MỤC LỤC

Phần 1:
GIỚI THIỆU CHUNG

1.1. Tổng quan về đề tài

- Giới thiệu: Với tình trạng giao thông phức tạp và đa dạng ở Việt Nam, việc quản lý và giám sát hiệu quả các phương tiện giao thông là một thách thức lớn. Hàng ngày, các thành phố lớn như Hà Nội, TP. Hồ Chí Minh phải đối mặt với lưu lượng phương tiện dày đặc, gồm xe máy, ô tô, xe tải và xe buýt. Điều này không chỉ gây ra ùn tắc mà còn tạo ra nhiều khó khăn trong việc kiểm soát giao thông và đảm bảo an toàn. Đề tài "Detecting, tracking and counting Vietnam traffic vehicle" tập trung vào việc phát triển một hệ thống thông minh để nhận diện, theo dõi và đếm số lượng phương tiện giao thông, từ đó hỗ trợ quản lý giao thông hiệu quả hơn.
- Mục tiêu và mong đợi:
    - Xây dựng một hệ thống phát hiện và theo dõi phương tiện giao thông**:** Hệ thống này có khả năng nhận diện và phân loại các phương tiện giao thông phổ biến tại Việt Nam như xe máy, ô tô, xe tải và xe buýt trong thời gian thực.
    - Tối ưu hóa hiệu suất để hệ thống có thể hoạt động mượt mà trên các thiết bị giám sát với cấu hình khác nhau, từ các máy tính cá nhân đến các máy chủ phân tích dữ liệu lớn.
    - Cung cấp thông tin chính xác về lưu lượng giao thông**:** Hệ thống sẽ không chỉ đếm số lượng phương tiện mà còn theo dõi sự di chuyển của chúng, giúp đưa ra những thông tin chi tiết và giá trị cho việc quản lý giao thông.

1.2. Mô tả đề tài

Với bài toán "Detecting, tracking and counting Vietnam traffic vehicle" được thể hiện như sau:

- Đầu vào của bài toán: là một video chứa các phương tiện giao thông.
- Đầu ra của bài toán: một video đã qua xử lý phát hiện, theo dõi và đếm số lượng các đối tượng được phũ các bounding box.
- Với hệ thông hỗ trợ này sẽ giúp quản lý và điều phối tình trạng giao thông một cách kịp thời giúp các cơ quan chức năng điều chỉnh đèn tín hiệu giao thông và đưa ra các biện pháp điều phối phù hợp để giảm ùn tắc. Qua đó giúp cải thiện luồng giao thông tại các khu vực đông đúc, giúp giảm nguy cơ tai nạn, đặc biệt là ở các nút giao thông phức tạp đồng thời hệ thống này có thể đóng vai trò quan trọng trong việc phát triển và quản lý các hệ thống giao thông thông minh trong tương lai.

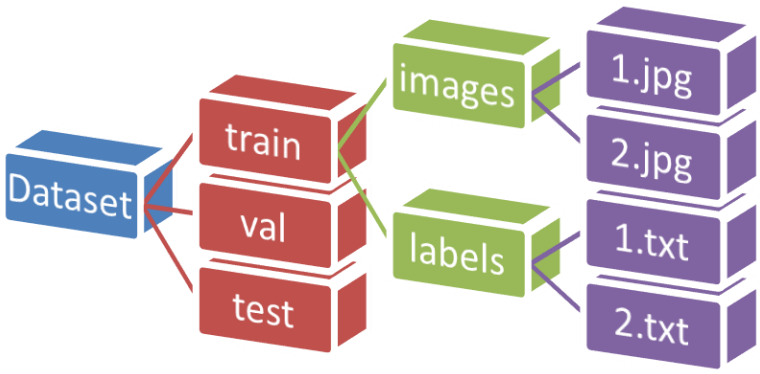
1.3. Mô tả dữ liệu

- Bộ dữ liệu gồm 2361 ảnh của 4 loại phương tiện giao thông sau: Car, Motorbike, Truck và Bus.

Loại phương tiện	Nhãn
Car	0
Motor	1
Truck	2
Bus	3

Bảng 1.1. Bảng mapping 4 loại phương tiện

- Kích thước của mỗi mẫu dữ liệu là 320x320 pixel, được lưu với định dạng jpg.



1

Hình 1.1. Cấu trúc trong thư mục dataset

- Tất cả các file label được gom vào thư mục labels. Mỗi file label được lưu ở định dạng txt. Mỗi dòng trong file này chứa thông tin về 1 bounding box của một đối tượng trong ảnh với định dạng:

  [label 1] [xmean 1] [ymean 1] [width 1] [height 1]

  [label 2] [xmean 2] [ymean 2] [width 2] [height 2]

- Giá trị [label] tương ứng với giá trị mapping của tên các loại phương tiện ở trên [xmean] [ymean] là toạ độ điểm trung tâm và [wight] [height] là kích thước của bbox. Lưu ý tất cả đều được normalize bằng cách chia cho kích thước của ảnh.

- Bộ dataset sẽ được chia ngẫu nhiên thành các tập train, validation, test theo tỉ lệ 70:20:10

- Số lượng ảnh trong mỗi tập:
    - Tập Train: 1652 ảnh.
    - Tập Validation: 472 ảnh.
    - Tập Test: 237 ảnh.

Hình 1.2: Mẫu dữ liệu trong điều kiện trời tối

Hình 1.3: Mẫu dữ liệu trong môi trường ban ngày

Phần 2:
CÁC ỨNG DỤNG VÀ NGHIÊN CỨU LIÊN QUAN

2.1. You Only Look Once (YOLO)

YOLO (You Only Look Once) là một framework phổ biến cho nhận diện đối tượng thời gian thực, đã được cải tiến và tinh chỉnh liên tục kể từ khi được công bố lần đầu tiên vào năm 2016 bởi Redmon và các cộng sự. Trong phần này, chúng em sẽ xem xét các phiên bản mới nhất của YOLO, cụ thể là YOLOv5, YOLOv7, và YOLOv8 để lựa chọn ra mô hình tối ưu nhất phù hợp với bài toán của nhóm.

- YOLOv5 là phiên bản YOLO framework được khai độc lập bởi Glenn Jocher vào năm 2020, nhằm đơn giản hóa và tối ưu hóa mã nguồn và kiến trúc mạng. YOLOv5 sử dụng mã nguồn dựa trên PyTorch, dễ tiếp cận và tùy biến hơn so với mã nguồn gốc dựa trên C. YOLOv5 cũng áp dụng một mạng xương sống mới gọi là YOLOv5s, dựa trên MobileNetV3, có ít tham số hơn và tốc độ cao hơn so với Darknet-53. YOLOv5 cho thấy kết quả tương đương hoặc tốt hơn so với YOLOv4 trên tập dữ liệu COCO, đạt mAP 50.4% và FPS 140, nhưng không được đánh giá trên tập dữ liệu PASCAL VOC 2012. YOLOv5 cũng được áp dụng vào nhận diện đối tượng dưới nước bởi Zhang và các cộng sự, với mAP 81.2% và FPS 28.6 trên tập dữ liệu URPC2018.
- YOLOv7 là một cập nhật lớn của YOLO bởi Terven và các cộng sự vào năm 2021, giới thiệu nhiều thành phần và cơ chế mới, như Attention Modules, Dynamic Convolution, Label Smoothing, và Online Hard Example Mining. YOLOv7 cũng sử dụng một mạng xương sống mới gọi là YOLOv7s, dựa trên ResNeXt, có sự đa dạng và phức tạp hơn so với YOLOv6s. YOLOv7 đạt kết quả tiên tiến nhất trên nhiều tập dữ liệu và benchmark, vượt qua các phiên bản YOLO trước và các mô hình nhận diện đối tượng khác. Trên tập dữ liệu PASCAL VOC 2012, YOLOv7 đạt mAP 88.9% và FPS 50, và trên tập dữ liệu COCO, đạt mAP 57.9% và FPS 48. YOLOv7 cũng được áp dụng vào nhận diện thuốc lá trong ảnh chụp từ trên cao bởi Kumar và các cộng sự, với mAP 94.6% và FPS 45.8 trên tập dữ liệu Tobacco-800.
- YOLOv8 là phiên bản mới nhất và tiên tiến nhất của khung YOLO bởi Wang và các cộng sự vào năm 2021, tích hợp các phát triển và sáng kiến mới nhất trong lĩnh vực nhận diện đối tượng, như Transformers, Neural Architecture Search, và Knowledge Distillation. YOLOv8 sử dụng một mạng xương sống mới gọi là YOLOv8s, dựa trên ViT, có khả năng chú ý và tự học cao hơn so với YOLOv7s. YOLOv8 cũng sử dụng một hàm mất mới gọi là Generalized Intersection over Union, cải thiện độ chính xác

và căn chỉnh của các hộp giới hạn. YOLOv8 đạt kết quả tiên tiến nhất trên nhiều benchmark và tập dữ liệu, vượt qua các phiên bản YOLO trước và các mô hình nhận diện đối tượng khác. Trên tập dữ liệu PASCAL VOC 2012, YOLOv8 đạt mAP 90.2% và FPS 30, và trên tập dữ liệu COCO, đạt mAP 59.7% và FPS 28. YOLOv8 cũng được áp dụng vào nhận diện khuôn mặt trong giám sát video bởi Li và các cộng sự, đạt mAP 96.3% và FPS 25.4 trên tập dữ liệu WIDER FACE.

Phương pháp mà chúng em đề xuất dựa trên framework YOLO, và tận dụng các ưu điểm của YOLOv5, YOLOv7, và YOLOv8, đồng thời khắc phục một số hạn chế và thách thức của chúng.

2.2. RCNN, Fast R-CNN, Faster R-CNN

Các Mạng Nơ-ron Tích Chập region-based (R-CNN) là một họ mô hình dùng cho phát hiện đối tượng, sử dụng các đề xuất vùng để trích xuất đặc trưng và phân loại đối tượng trong ảnh. Trong phần này, chúng tôi sẽ xem xét sự phát triển của R-CNN từ mô hình R-CNN ban đầu đến Faster R-CNN, và so sánh chúng với phương pháp mà chúng tôi đề xuất.

- R-CNN là mô hình đầu tiên áp dụng CNN vào phát hiện đối tượng, được đề xuất bởi Girshick và các cộng sự vào năm 2014. R-CNN sử dụng selective search để tạo ra khoảng 2000 đề xuất vùng mỗi ảnh, sau đó áp dụng một CNN đã được huấn luyện trước (AlexNet) để trích xuất một vector đặc trưng có kích thước 4096 chiều cho mỗi vùng. Các vector đặc trưng sau đó được đưa vào một tập các SVM tuyến tính cụ thể cho từng lớp để tạo ra kết quả nhận diện cuối cùng. R-CNN đạt mAP 58.5% trên tập dữ liệu PASCAL VOC 2012, nhưng rất chậm (47 giây mỗi ảnh) và yêu cầu một lượng lớn dung lượng bộ nhớ để lưu trữ các đặc trưng.
- Fast R-CNN là một cải tiến của R-CNN, được đề xuất bởi Girshick vào năm 2015. Fast R-CNN cải thiện tốc độ và hiệu quả bộ nhớ của R-CNN bằng cách chia sẻ tính toán của CNN trên toàn bộ ảnh, và sử dụng một lớp RoI pooling để trích xuất các vector đặc trưng có độ dài cố định cho mỗi đề xuất vùng. Fast R-CNN cũng thay thế các SVM bằng một lớp softmax và một bộ điều chỉnh hộp giới hạn, và huấn luyện toàn bộ mạng theo cách end-to-end sử dụng một hàm mất đa nhiệm. Fast R-CNN đạt mAP 66.9% trên tập dữ liệu PASCAL VOC 2012, và nhanh hơn R-CNN 213 lần (0.22 giây mỗi ảnh).
- Faster R-CNN là một cải tiến tiếp theo của Fast R-CNN, được đề xuất bởi Ren và các cộng sự vào năm 2015. Faster R-CNN cải thiện tốc độ và độ chính xác của Fast R-CNN bằng cách thay thế tìm kiếm chọn lọc bằng một Mạng Đề Xuất Vùng (Region

Proposal Network), là một mạng hoàn toàn tích chập dự đoán các giới hạn và điểm số của đối tượng tại mỗi vị trí. RPN chia sẻ các đặc trưng tích chập với bộ phát hiện Fast R-CNN, cho phép đề xuất vùng gần như không tốn chi phí. Faster R-CNN đạt mAP 69.9% trên tập dữ liệu PASCAL VOC 2012, và nhanh hơn Fast R-CNN 10 lần (0.02 giây mỗi ảnh).

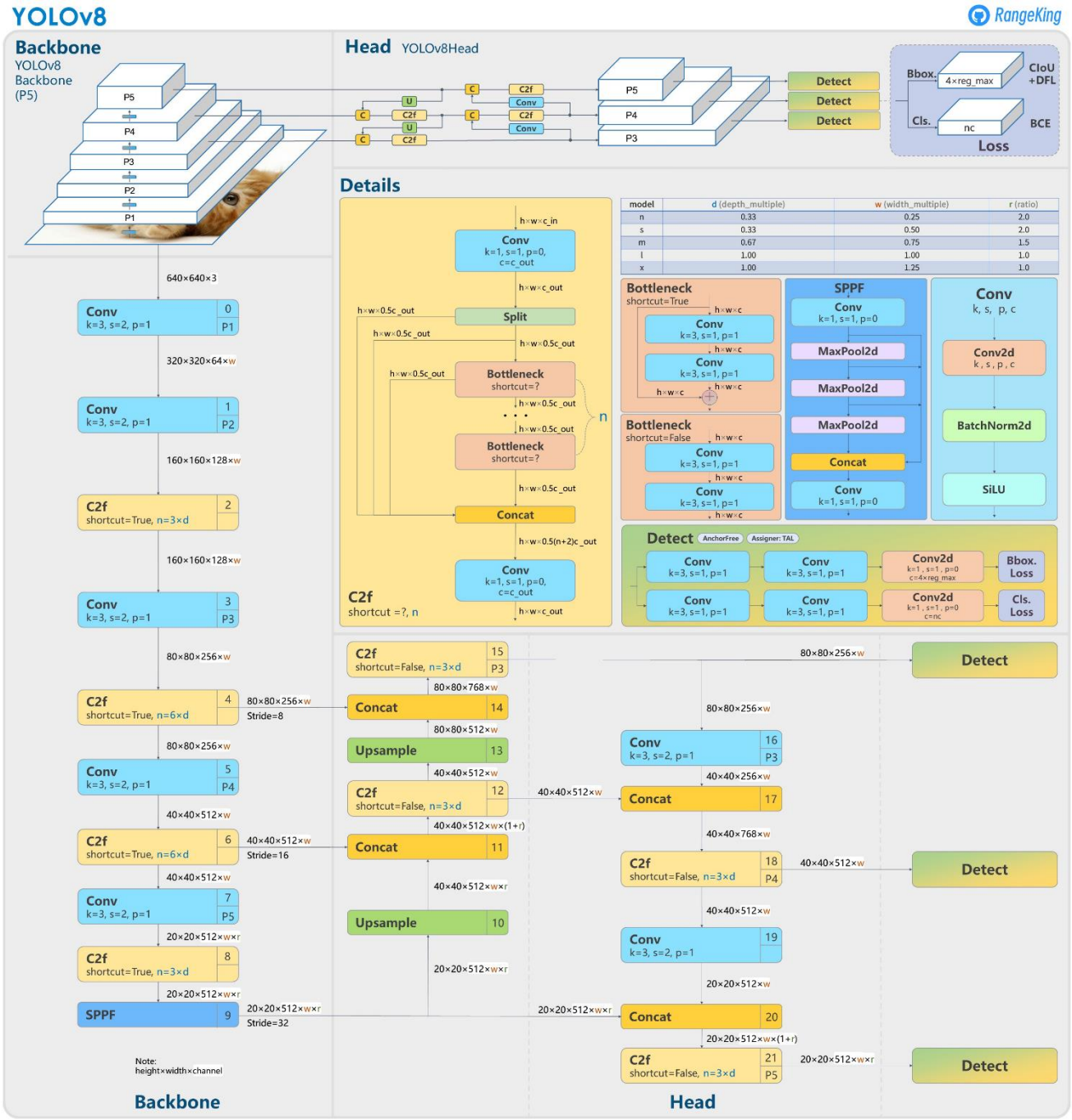
2.3. DeepSORT

- DeepSORT (Simple Online and Realtime Tracking with a Deep Association Metric) là một công cụ theo dõi đối tượng phổ biến, được sử dụng rộng rãi trong các ứng dụng theo dõi đa đối tượng.
- DeepSORT được giới thiệu lần đầu bởi Wojke et al. vào năm 2017, như một sự mở rộng của thuật toán SORT (Simple Online and Realtime Tracking). DeepSORT bổ sung một mạng học sâu để trích xuất đặc trưng nhằm cải thiện độ chính xác trong việc liên kết các đối tượng giữa các khung hình. Điều này giúp giảm thiểu các trường hợp nhầm lẫn đối tượng, đặc biệt khi các đối tượng di chuyển gần nhau hoặc có hình dáng tương tự. DeepSORT sử dụng một mạng Convolutional Neural Network (CNN) để trích xuất các vector đặc trưng từ hình ảnh của đối tượng, và sau đó sử dụng các đặc trưng này để tính toán độ tương đồng giữa các đối tượng ở các khung hình khác nhau. DeepSORT đạt được hiệu suất cao trên nhiều tập dữ liệu khác nhau, bao gồm cả MOT16 và MOT17, với độ chính xác và độ tin cậy vượt trội so với các phương pháp trước đó.

Phần 3:
NỘI DUNG THỰC HIỆN

3.1 Mô hình YOLOv8

- YOLO (You Only Look Once) là một thuật toán phát hiện đối tượng phổ biến đã cách mạng hóa lĩnh vực thị giác máy tính. Nó nhanh và hiệu quả, là một lựa chọn tuyệt vời cho các nhiệm vụ phát hiện đối tượng thời gian thực. YOLO đã đạt được hiệu suất tiên tiến trên nhiều tiêu chuẩn khác nhau và đã được áp dụng rộng rãi trong nhiều ứng dụng thực tế. Trong đồ án này, nhóm chúng em quyết định sử dụng mô hình YOLO để giải quyết cho bài toán Vehicle Detection.
- YOLOv8 được viết và duy trì bởi nhóm Ultralytics. Các mô hình YOLO ban đầu do Joseph Redmon, một nhà khoa học máy tính, tạo ra. Ông đã phát triển ba phiên bản YOLO, với phiên bản thứ ba là YOLOv3, được viết bằng kiến trúc Darknet. Glenn Jocher đã sử dụng phiên bản YOLOv3 trong PyTorch với một số thay đổi nhỏ và đặt tên nó là YOLOv5. Kiến trúc YOLOv5 sau đó đã được điều chỉnh để phát triển YOLOv8.
- Kiến trúc mô hình YOLOv8:
    - Backbone: Đây là phần thân chính của network. YOLOv8 backbone được thiết kế sử dụng kiến trúc CSP-Darknet53 mới, một phiên bản chỉnh sửa của kiến trúc Darknet được sử dụng trong các phiên bản trước.
    - Neck: Đây là phần kết nối backbone và head.
    - Head: Đây là phần phụ trách dự đoán output cuối cùng.



Hình 3.1: Kiến trúc mô hình YOLOv8

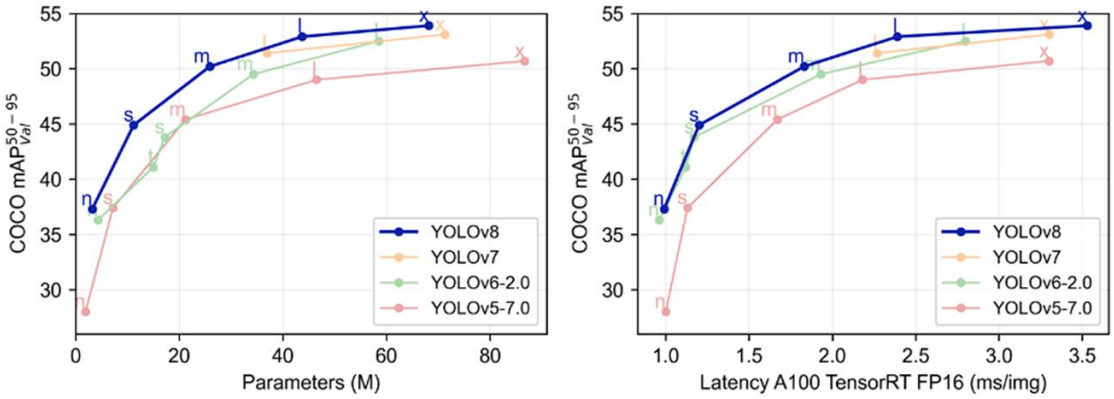
- Những thay đổi so với mô hình YOLOv5:
  - Thay thế mô-đun **C3** bằng mô-đun **C2f**
  - Thay thế **Conv 6x6** đầu tiên bằng **Conv 3x3** trong Backbone
  - Loại bỏ 2 lớp **Conv** (Số 10 và số 14 trong YOLOv5 config)
  - Thay thế **Conv 1x1** đầu tiên bằng **Conv 3x3** trong Bottleneck
  - Sử dụng **decoupled head** và loại bỏ nhánh **objectness**

- So sánh hiệu suất của YOLOv8 với YOLOv5:

Model Size	Detection	Segmentation	Classification
Nano	*+33.21%*	*+32.97%*	*+3.10%*
Small	*+20.05%*	*+18.62%*	*+1.12%*
Medium	*+10.57%*	*+10.89%*	*+0.66%*
Large	*+7.96%*	*+6.73%*	*0.00%*
Xtra Large	*+6.31%*	*+5.33%*	*-0.76%*

Bảng 3.1. Bảng so sánh hiệu suất của YOLOv8 với YOLOv5

- So sánh với các phiên bản YOLO trước đó:



Hình 3.2: Biểu đồ so sánh hiệu năng của YOLOv8 với các phiên bản trước

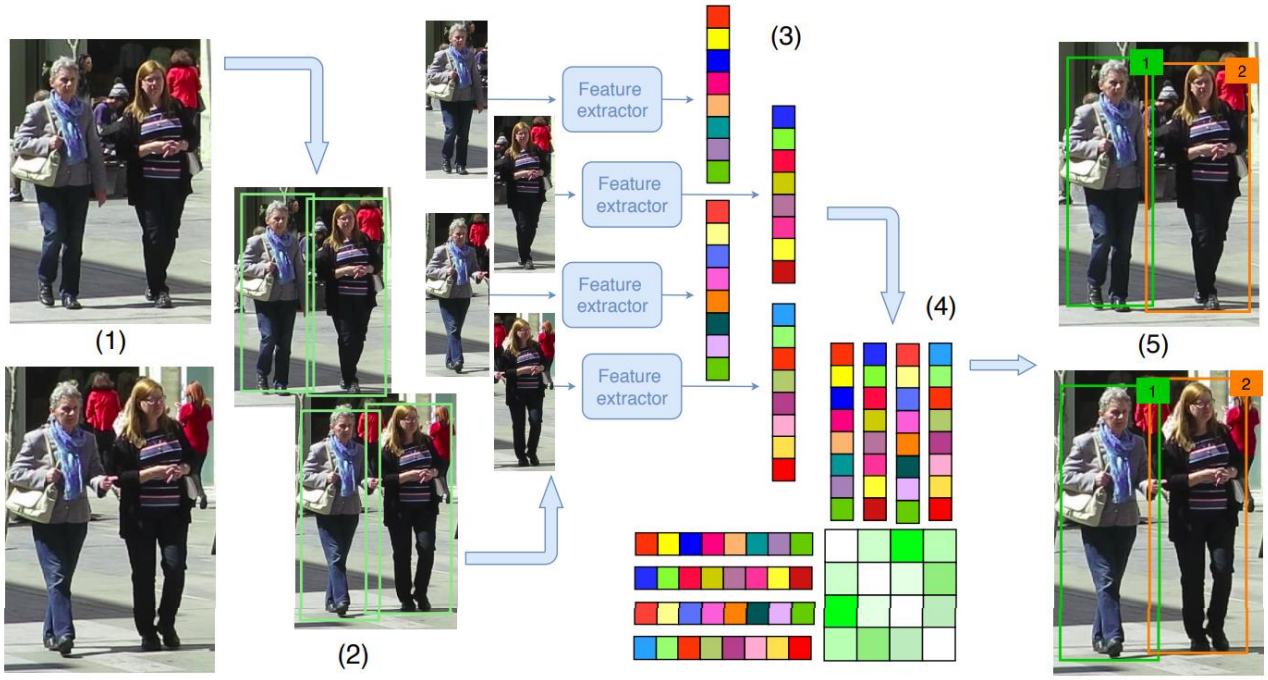
- Để huấn luyện mô hình, nhóm chúng em sử dụng pretrained YOLOv8n.pt (3M) và YOLOv8m.pt vì model này đáp ứng được yêu cầu của bọn em đó là thời gian chạy nhanh và mô hình có độ chính xác tương đối cao cũng như tính linh hoạt của nó trong quá trình sử dụng.
- Việc huấn luyện được thực hiện trên Google Colab và các dữ liệu liên quan được lưu lại trong Google Drive. Quá trình setup môi trường được thực hiện theo các bước bao gồm:
    - Tạo folder lưu trữ model và các dữ liệu liên quan trên Dirve
    - Tạo file Notebook trên Colab, liên kết dữ liệu từ Drive
    - Tạo file data. yaml bao gồm thông tin tập train, tập val và các class sử dụng trong việc train model.
    - Train mô hình và tiến hành thực hiện chạy thử, ghi nhận kết quả

3.2. DeepSORT

3.2.1. Ý tưởng

- 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 để xét và đánh giá nhằm liên kết một detection mới với các track đã được lưu trữ sẵn.
- Track Life Cycle Management: 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



Hình 3.3. Kiến trúc DeepSORT

- Trong deep SORT, nhóm tác giả giải quyết vấn đề data association dựa trên thuật toán Hungary
- Thuật toán dựa trên IOU và 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, hai vector đặc trưng của cùng 1 đối tượng sẽ giống nhau hơn là đặc trưng của 2 đối tượng khác nhau.

3.2.2. Bộ trích xuất đặc trưng

- Nhằm phát triển một bộ trích xuất đặc trưng cho mỗi detection (bounding box) thu được, Nicolai Wojke, Alex Bewley đã phát triển một kiến trúc mạng phần dư mở rộng (Wide Residual Network). Tác vụ này còn được gọi là Cosine Metric Learing
- Sử dụng Wide Residual Network (WRN) các mạng neural thông thường, nhóm tác giả lựa chọn các mạng nông (Shallow Neural Network), cụ thể ở đây là Wide Residual Network (WRN). Kiến trúc này được giới thiệu chỉ với số lớp rất nhỏ (16 lớp), vẫn có

thể đạt performance vượt trội hơn các kiến trúc hàng nghìn lớp khác. Đặc biệt là thời gian training và inference cũng nhanh hơn rất nhiều.

Name	Patch Size/Stride	Output Size
Conv 1	$3 \times 3/1$	$32 \times 128 \times 64$
Conv 2	$3 \times 3/1$	$32 \times 128 \times 64$
Max Pool 3	$3 \times 3/2$	$32 \times 64 \times 32$
Residual 4	$3 \times 3/1$	$32 \times 64 \times 32$
Residual 5	$3 \times 3/1$	$32 \times 64 \times 32$
Residual 6	$3 \times 3/2$	$64 \times 32 \times 16$
Residual 7	$3 \times 3/1$	$64 \times 32 \times 16$
Residual 8	$3 \times 3/2$	$128 \times 16 \times 8$
Residual 9	$3 \times 3/1$	$128 \times 16 \times 8$
Dense 10		128
ℓ_2 normalization		128

Hình 3.4: Các layer của mạng WRN

- Sử dụng cosine softmax classifer: Nicolai Wojke và Alex Bewley sử dụng một classifer mới bằng việc tham số hóa lại softmax classifer tiêu chuẩn

3.2.3. Các độ đo mới

- Deep SORT đề xuất 2 độ đo hoàn toàn mới, được thêm vào để improve cho quá trình liên kết dữ liệu.
- Khoảng cách Mahalanobis:

$$d^{(1)}(i,j) = (\boldsymbol{d}_j - \boldsymbol{y}_i)^{\mathrm{T}} \boldsymbol{S}_i^{-1} (\boldsymbol{d}_j - \boldsymbol{y}_i)$$

Hình 3.5: Công thức khoảng cách Mahalanobis

- Trong đó (y_i, S_i) là giá trị kì vọng và ma trận hiệp phương sai của biến ngẫu nhiên track thứ i, và d_j là giá trị của detection thứ j.
- Khoảng cách Mahalanobis là thước đo khoảng cách giữa điểm P và phân phối D, do P. C. Mahalanobis đưa ra vào năm 1936. Nó là sự tổng quát hóa đa chiều về ý tưởng đo lường độ lệch chuẩn giữa P so với giá trị kì vọng của D. Khoảng cách này bằng 0 nếu P ở giá trị kì vọng của D và tăng lên khi P di chuyển ra xa giá trị kì vọng này.

- Ngoài việc đo lường khoảng cách giữa track và detection, khoảng cách Mahalanobis còn được dùng để loại trừ các liên kết không chắc chắn bằng cách lập ngưỡng khoảng cách Mahalanobis ở khoảng tin cậy 95% được tính từ phân phối $\chi 2$.

$$b_{i,j}^{(1)} = \mathbf{1}\left[d^{(1)}(i,j) \le t^{(1)}\right]$$

Với $t^{(1)} = 9.4877$.

Hình 3.6: Công thức phân phối $\chi 2$.

- Deep SORT đồng thời sử dụng một độ đo khác về đặc trưng của đối tượng, nhằm đảm bảo việc liên kết chuẩn xác dù đối tượng đã biến mất và sau đó xuất hiện trở lại trong khung hình. Đó chính là các đặc trưng học được từ cosine metric learning
- Với mỗi detection, đặc trưng r_j được trích xuất với $||r_j||=1$. Với mỗi track, một danh sách với độ dài khoảng 100 được sử dụng để lưu trữ đặc trưng của 100 track gần nhất: $R_k = \{\boldsymbol{r}_{\boldsymbol{k}}^{(\boldsymbol{i})}\}_{\boldsymbol{k=1}}^{\boldsymbol{Lk}}$, $Lk=100$.
- Khi đó, độ đo mới giữa track và detection được tính bằng khoảng cách cosine:

$$d^{(2)}(i,j) = \min\left\{1 - \boldsymbol{r}_j{}^{\mathrm{T}}\boldsymbol{r}_k^{(i)} \mid \boldsymbol{r}_k^{(i)} \in \mathcal{R}_i\right\}$$

Và

$$b_{i,j}^{(2)} = \mathbf{1}\left[d^{(2)}(i,j) \le t^{(2)}\right]$$

Hình 3.7: Công thức khoảng cách cosine

với $\boldsymbol{t}^{(\mathbf{2})}$) được lựa chọn tùy vào tập dữ liệu sử dụng.

- Khoảng cách Mahalanobis cung cấp các thông tin về vị trí đối tượng dựa trên chuyển động tức thời, đặc biệt hữu ích cho các dự đoán ngắn hạn. Mặt khác, khoảng cách

cosine xem xét thông tin về đặc trưng của đối tượng, đặc biệt hữu ích cho các dự đoán dài hạn hoặc các đối tượng khó phân biệt. Bằng việc kết hợp 2 độ đo với trọng số phù hợp, deep SORT tạo ra một độ đo mới:

$$c_{i,j} = \lambda d^{(1)}(i,j) + (1-\lambda) d^{(2)}(i,j)$$

Với:

$$b_{i,j} = \prod_{m=1}^{2} b_{i,j}^{(m)}$$

Hình 3.8: Công thức khoảng cách kết hợp cả 2 độ đo

- Giá trị sau đó được sử dụng trong chiến lược liên kết matching cascade

3.2.4. Chiến lược đối sánh theo tầng hay Matching Cascade

- Deep SORT sử dụng nhằm cải thiện độ chính xác của liên kết, chủ yếu là vì khi đối tượng biến mất trong thời gian dài, độ không chắc chắn của bộ lọc Kalman sẽ tăng lên rất nhiều và sẽ dẫn đến phân tán xác suất dự đoán liên tục.
- Vì vậy, nếu dự đoán liên tục không được cập nhật, phương sai của phân phối chuẩn sẽ ngày càng lớn.
- Chiến lược đối sánh theo tầng tiến hành lấy lần lượt từng track ở các frame trước đó, để tiến hành xây dựng ma trận chi phí và giải bài toán phân công theo từng tầng. Chi tiết thuật toán được trình bày cụ thể hơn ở mã giả dưới đây:

Listing 1 Matching Cascade

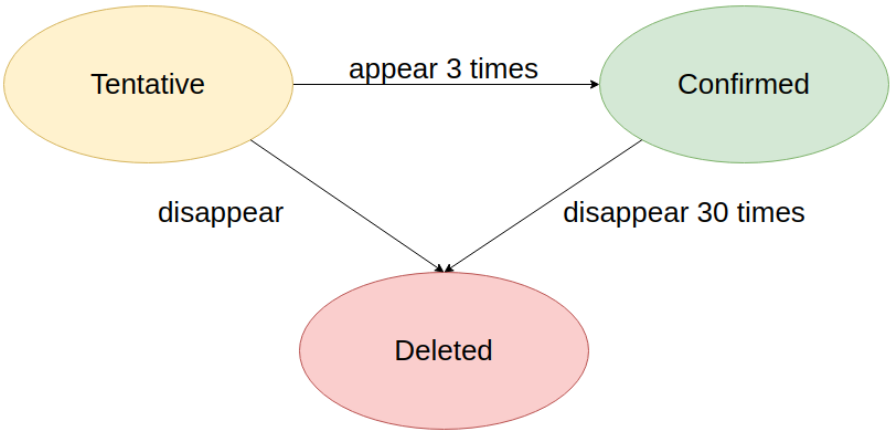
Input: Track indices $\mathcal{T} = \{1, \ldots, N\}$, Detection indices $\mathcal{D} = \{1, \ldots, M\}$, Maximum age $A_{\max}$

Compute cost matrix $C = \left[c_{i,j}\right]$
Compute gate matrix $B = \left[b_{i,j}\right]$
Initialize set of matches $\mathcal{M} \leftarrow \emptyset$
Initialize set of unmatched detections $\mathcal{U} \leftarrow \mathcal{D}$
for $n \in \{1, \ldots, A_{\max}\}$ **do**
  Select tracks by age $\mathcal{T}_n \leftarrow \left\{i \in \mathcal{T} \mid a_i = n\right\}$
  $\left[x_{i,j}\right] \leftarrow$ min cost matching $(C, \mathcal{T}_n, \mathcal{U})$
  $\mathcal{M} \leftarrow \mathcal{M} \cup \left\{(i,j) \mid b_{i,j} \cdot x_{i,j} > 0\right\}$
  $\mathcal{U} \leftarrow \mathcal{U} \setminus \left\{j \mid \sum_i b_{i,j} \cdot x_{i,j} > 0\right\}$
end for
return $\mathcal{M}, \mathcal{U}$

Hình 3.9: Mã giả matching cascade

3.2.5. Quản lý vòng đời 1 track

- Deep SORT quản lí vòng đợi của 1 track dựa trên 1 biến trạng thái với 3 giá trị (tentative, confirmed, deleted)
- Các trạng thái này lúc mới khởi tạo sẽ được gán 1 giá trị mang tính thăm dò (tentative).
- Giá trị này nếu vẫn đảm bảo duy trì được trong 3 frame tiếp theo, trạng thái sẽ chuyển từ thăm dò sang xác nhận (confirmed),
- Các track có trạng thái confirmed sẽ cố gắng được duy trì theo dõi, dù bị biến mất thì Deep SORT vẫn sẽ duy trì theo dõi trong 30 frame tiếp theo.
- Ngược lại, nếu mất dấu khi chưa đủ 3 frame, trạng thái sẽ bị xóa khỏi trình theo dõi (deleted)



Hình 3.10: Quá trình quản lí vòng đợi 1 track

3.3. Đếm số lượng phương tiện

- Hệ thống đếm số lượng đối tượng đi vào và ra khỏi khu vực xác định bằng cách phân tích hướng di chuyển của các đối tượng được theo dõi là Bắc hay Nam.
- Khi một đối tượng đang được theo dõi di chuyển và nó có ID không nằm trong danh sách các đối tượng trước đó hệ thống sẽ tăng bộ đếm tương ứng với loại phương tiện lên 1.
- Điều này cho phép hệ thống không chỉ theo dõi mà còn đếm được số lượng đối tượng đi vào và ra khỏi khu vực một cách chính xác.

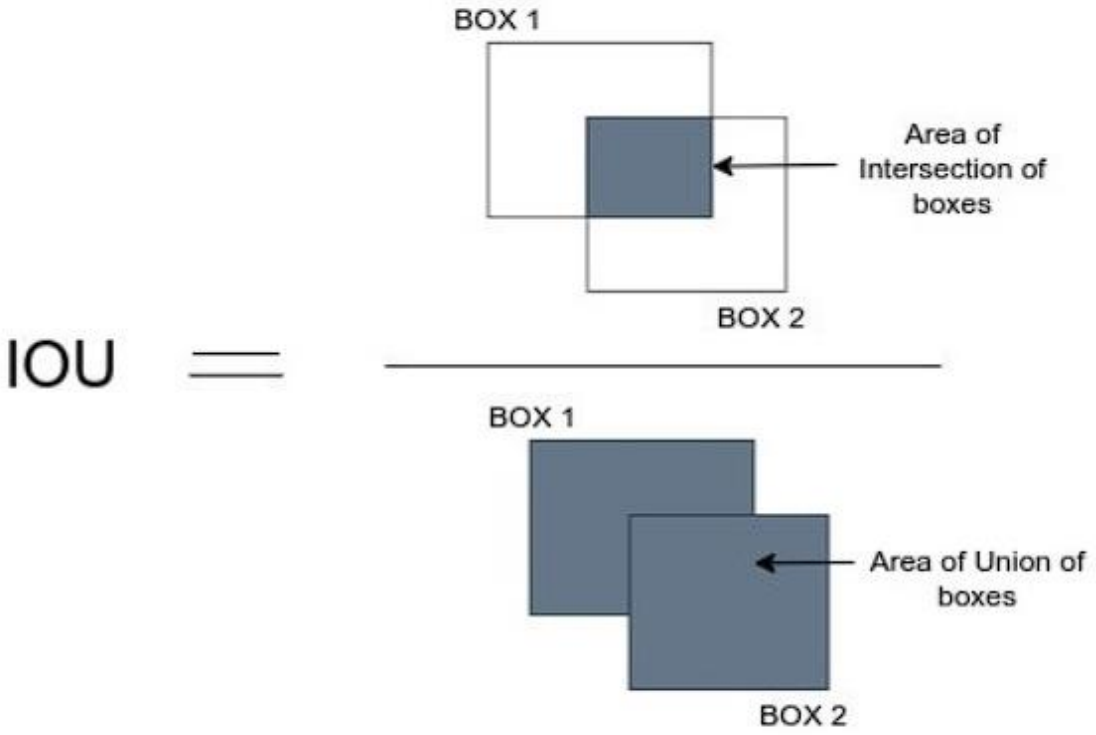
Phần 4:
ĐÁNH GIÁ VÀ PHÂN TÍCH

4.1. Các phương pháp đánh giá:

4.1.1. Giới thiệu về IoU

- IoU là viết tắt của Intersection over Union, là Giao trên Hợp.

$$IOU = \frac{Area\ of\ Intersection\ of\ two\ boxes}{Area\ of\ Union\ of\ two\ boxes}$$



Hình 4.1: Công thức tính IoU

- IoU là tỉ lệ giữa giao và hợp của bouding box model dự đoán ra và ground truth.
- Như vậy: Tỉ IoU càng lớn thì càng tốt, đồng nghĩa với việc phần giao là lớn và phần hợp là nhỏ (nhãn dự đoán ra giống với nhãn thực).

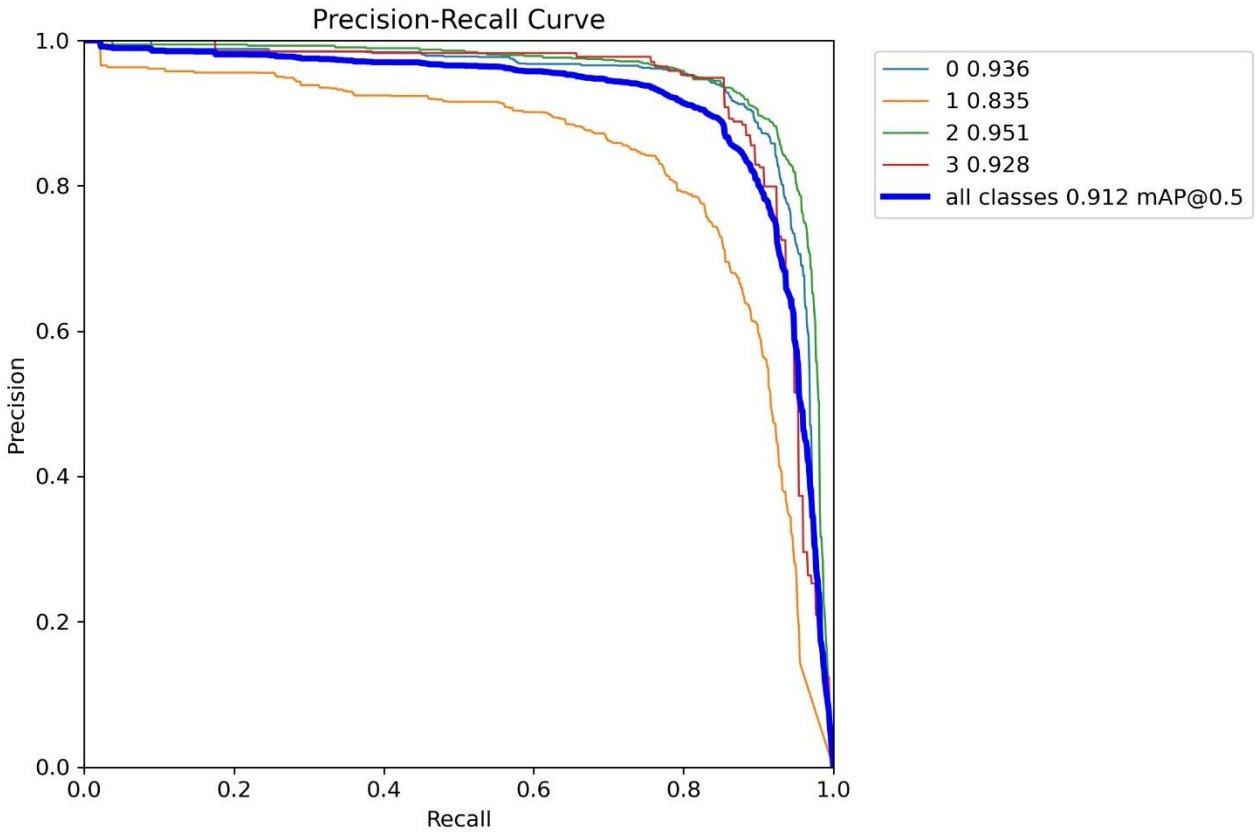
4.1.2. Precision, Recall

- Precision cho biết rằng những cái mà model dự đoán là Positive thì có bao nhiêu % là Positive thật.
- Precision = Dự đoán Positive đúng (True Positive) / Tổng dự đoán

- Positive [Số dự đoán Positive đúng (True Positive) + Số dự đoán Positive sai (False Positive)]
- Ví dụ: Trong 10 người được dự đoán, mô hình đoán 8 người bị ung thư, nhưng thực tế chỉ có 5 người bị ung thư, còn lại 3 người bị dự đoán sai Thì True Positive là 5 người được dự đoán ung thư là dúng False Positive là 3 người được dự đoán ung thư là sai. Precision = 5 / (5+3) = 5/8 = 0.625
- Recall sẽ cho biết model có thể tóm đúng được bao nhiêu Positive trong dữ liệu được cho.
- Recall = Dự đoán Positive đúng (True Positive) / Tổng số Positive thật trong thực tế [Số dự đoán Positive đúng (True Positive) + số dự đoán Negative sai (False Negative)]
- Ví dụ: Trong 10 người vào khám bênh, có 5 người bị ung thư, nhưng model nhận diện được 3 người là ung thư, còn 7 người còn lại là không ung thư. Trong đó True Positive là 3 người ung thư được nhận diện là ung thư. False Negative là 2 người ung thư được nhận diện là không ung thư. Recall = 3/ (3+2) = 3/5 = 0.6

4.1.3. Precision Recall Curve

- Để nói đến Precision Recall Curve thì phải nhắc đến IoU threshold (Ngưỡng của IoU). Với các dự đoán của model object detection, nếu IoU > threshold thì được coi là True Positive (Nhận diện đúng) và ngược lại nếu IoU < threshold thì sẽ được coi là False Positive (Nhận là Positive nhưng lại bị sai).
- Từ đó, chúng ta thay đổi IoU threshold, tính toán ra Precision và Recall tại các mức độ và vẽ ra được Precision Recall Curve như sau:



-

Hình 4.2. Precision Recall Curve của mô hình

- Nhìn vào đồ thị trên, phần diện tích bên dưới đường cong chính là Average Precision (AP).
    - AP lớn nếu vùng này lớn, suy ra đường cong có xu hướng gần góc trên bên phải và có nghĩa là lại các threshold khác nhau thì Precision và Recall đều khá cao. Từ đó suy ra model tốt.
    - AP nhỏ thì cả Precision và Recall đều khá thấp và model không tốt

4.1.4. mAP (mean Average Precision)

- Theo như AP ở trên thì mỗi class sẽ có một giá trị AP. Mà một bài toán detection thì có nhiều class nên để nói đến cả mô hình thì chúng ta phải tính trung bình cộng tất cả các giá trị AP của các class khác nhau. Và trung bình cộng đó chính là mAP, có nghĩa là Mean Average Precision.

4.2. Đánh giá kết quả mô hình

4.2.1. Bảng đánh giá kết quả mô hình

- Lý thuyết về mAP là vậy nhưng khi dùng YOLO thì mô hình đã tính luôn mAP và trả về cho chúng ta mà chúng ta không cần phải tính thủ công lại.

Class	Images	Instances	Precision	Recall	mAP50	mAP50-95
all	472	2568	0.862	0.869	0.911	0.603
0 (Car)	332	673	0.883	0.899	0.936	0.654
1 (Motor)	271	922	0.804	0.782	0.831	0.376
2 (Truck)	365	819	0.891	0.909	0.95	0.699
3 (Bus)	128	172	0.869	0.885	0.926	0.682

Bảng 4.1: Đánh giá kết quả mô hình YOLOv8n

Class	Images	Instances	Precision	Recall	mAP50	mAP50-95
all	472	2568	0.898	0.884	0.933	0.66
0 (Car)	332	673	0.911	0.912	0.948	0.704
1 (Motor)	271	922	0.882	0.768	0.874	0.432
2 (Truck)	365	819	0.895	0.944	0.96	0.751
3 (Bus)	128	172	0.903	0.913	0.951	0.751

Bảng 4.2: Đánh giá kết quả mô hình YOLOv8m

- Nhận xét tổng quan: Precision và Recall tổng quan đều khá cao, cho thấy 2 mô hình hoạt động tốt trong việc nhận diện và định vị các đối tượng trong hình ảnh. Chỉ số mAP50 (mean Average Precision tại ngưỡng IoU 50%) rất cao (0.911 và 0.933), cho thấy mô hình hoạt động tốt khi đánh giá tại ngưỡng này. Tuy nhiên, chỉ số mAP50-95, trung bình độ chính xác qua các ngưỡng IoU từ 50% đến 95%, thấp hơn (0.603 và 0,66), cho thấy hiệu suất giảm dần ở các ngưỡng IoU cao hơn.
- So sánh 2 mô hình:
    - Tổng quát (All Classes): Mô hình YOLOv8m có hiệu suất tổng thể tốt hơn so với YOLOv8n trên tất cả các chỉ số. Điều này cho thấy YOLOv8vm hoạt động tốt hơn trong việc phát hiện và phân loại các đối tượng.

    - Lớp 0 (Car): YOLOv8m vượt trội hơn về tất cả các chỉ số so với YoloV8-n. Sự cải thiện lớn nhất là ở mAP50-95.
    - Lớp 1 (Motor): YOLOv8m có Precision và mAP50-95 tốt hơn đáng kể, mặc dù Recall có chút giảm nhẹ so với YoloV8-n.
    - Lớp 2 (Truck): YOLOv8m thể hiện vượt trội hơn với Recall và mAP50-95 cao hơn đáng kể.
    - Lớp 3 (Bus): YOLOv8m có hiệu suất tốt hơn trên tất cả các chỉ số, đặc biệt là mAP50-95.
- Mô hình YOLO cho thấy hiệu suất tổng quan mạnh mẽ, đặc biệt đối với Xe Hơi, Xe Tải và Xe Buýt. Tuy nhiên, nó gặp nhiều khó khăn hơn với Xe Máy, được thể hiện qua các chỉ số độ chính xác, độ nhạy và mAP thấp hơn cho lớp này. Tổng kết, mô hình khá hiệu quả nhưng còn có thể cải thiện thêm trong việc xử lý xe máy.
- Vì vấn đề hiệu suất khi triển khai mô hình thực tế, nhóm quyết định sử dụng mô hình YOLOv8n để deploy.

Phần 5:
TRIỂN KHAI MODEL

5.1. Ý tưởng thực hiện

5.1.1. Streamlit

Streamlit là một framework mã nguồn mở bằng Python, giúp việc xây dựng các ứng dụng web tương tác dễ dàng và nhanh chóng, đặc biệt là cho các dự án liên quan đến dữ liệu và machine learning. Dưới đây là một số điểm nổi bật về Streamlit:

- Streamlit được thiết kế để đơn giản hóa quy trình phát triển ứng dụng web. Bạn chỉ cần một vài dòng mã Python để tạo ra một ứng dụng web hoàn chỉnh. Cú pháp của Streamlit rất trực quan và thân thiện với các lập trình viên, đặc biệt là những người đã quen thuộc với Python.
- Streamlit hoạt động tốt với các thư viện phổ biến như NumPy, Pandas, Matplotlib, Plotly, và các framework machine learning như TensorFlow, PyTorch, và Scikit-learn. Điều này giúp bạn dễ dàng trực quan hóa dữ liệu, hiển thị các mô hình machine learning, và nhiều hơn nữa mà không cần phải chuyển đổi giữa các công cụ khác nhau.
- Một trong những tính năng mạnh mẽ của Streamlit là khả năng cập nhật giao diện ứng dụng theo thời gian thực khi dữ liệu thay đổi. Bạn không cần phải lo lắng về việc làm mới trang web hoặc cập nhật thủ công; Streamlit sẽ tự động thực hiện điều đó cho bạn.
- Streamlit có thể được triển khai trên nhiều nền tảng khác nhau, bao gồm cả local server, cloud services (như AWS, GCP), và thậm chí là các dịch vụ triển khai liên tục (CI/CD).
- Streamlit cho phép bạn mở rộng và tùy chỉnh ứng dụng của mình bằng cách thêm các widget tương tác như nút, hộp kiểm, thanh trượt, và nhiều hơn nữa. Bạn cũng có thể tùy chỉnh giao diện người dùng để phù hợp với nhu cầu cụ thể của dự án.

Hình 5.1. Giao diện Website

5.1.2. Triển khai bằng server của thầy đã cung cấp

5.1.2.1. Docker

Docker là một nền tảng mã nguồn mở giúp đơn giản hóa quá trình xây dựng, triển khai và chạy các ứng dụng bằng cách sử dụng các container. Container là các gói phần mềm nhẹ, độc lập và có thể di chuyển được, chứa tất cả những gì cần thiết để chạy ứng dụng, bao gồm mã nguồn, runtime, thư viện hệ thống, và các thiết lập. Dưới đây là một số điểm chính về Docker:

5.1.2.2. Container và Image

- **Container:** Là một instance của một image. Container chạy các ứng dụng một cách cô lập nhưng nhẹ hơn so với các máy ảo (VMs).
- **Image:** Là một mẫu (template) không thay đổi, dùng để tạo các container. Images chứa tất cả các tệp tin cần thiết để chạy ứng dụng.

5.1.2.3. Lợi ích của Docker

- Cô lập môi trường: Mỗi container hoạt động như một hệ thống độc lập với môi trường của nó, giúp tránh xung đột giữa các ứng dụng.
- Khả năng di động: Container có thể chạy nhất quán trên bất kỳ máy nào có Docker, bất kể máy chủ đó là máy chủ vật lý, máy chủ ảo hay trên đám mây.

- Quản lý tài nguyên hiệu quả: Container nhẹ và sử dụng ít tài nguyên hơn so với máy ảo.
- Tốc độ: Triển khai và khởi động container nhanh chóng, giúp tiết kiệm thời gian trong quá trình phát triển và triển khai ứng dụng.

5.1.2.4. Các thành phần chính của Docker

- Docker Engine: Là thành phần cốt lõi của Docker, bao gồm một daemon (dockerd) để quản lý các container, một REST API để tương tác với daemon và một giao diện dòng lệnh (CLI).
- Docker Hub: Là một kho lưu trữ trực tuyến cho các Docker images. Người dùng có thể tải lên và tải xuống các images từ Docker Hub.
- Docker Compose: Là một công cụ để định nghĩa và chạy các ứng dụng Docker multi-container. Sử dụng tệp YAML để cấu hình các dịch vụ của ứng dụng.

5.1.2.5. Quy trình triển khai đồ án

- **Viết Dockerfile:** Tạo một tệp Dockerfile chứa các lệnh để build image.
- **Build Image:** Sử dụng lệnh docker build để tạo image từ Dockerfile.
- **Push Docker Image:** Sử dụng lệnh này để đẩy Docker image lên Dockerhub
- **Pull Docker Image:** Đăng nhập vào Server triển khai ứng dụng và kéo image về
- **Run Container:** Sử dụng lệnh docker run để khởi động container từ image.
- **Docker ps:** Giám sát và kiểm tra các container đang chạy

➔ **Truy cập vào địa chỉ IP streamlit cung cấp trong dòng lệnh**

Phần 6:
KẾT LUẬN

6.1. Ưu điểm của đồ án:

- Xây dựng được mô hình có thể phát hiện được các loại phương tiện giao thông phổ biến tại đường phố Việt Nam.
- Thành công áp dụng thuật toán tracking để theo dõi đối tượng và đếm số lượng phương tiện trong khung hình.
- Triển khai mô hình lên website để trực quan hóa kết quả và người dùng dễ dàng thao tác. Hạn chế của đồ án:

6.2. Nhược điểm của đồ án:

- Bởi vì tự thu thập dữ liệu nên dữ liệu còn chưa được nhiều và số lượng class còn hạn chế, số lượng ảnh ở các lớp còn chênh lệch nhau nhiều.
- Mô hình YOLOv8n cho thời gian phát hiện đối tượng nhanh, tuy nhiên thuật toán DeepSORT tracking còn khá chậm, gây ảnh hưởng đến hiệu suất chạy realtime.
- Mô hình vẫn còn phát hiện sai dẫn đến đưa kết quả sai cho người sử dụng.
- Website triển khai còn đơn giản.

6.3. Hướng phát triển của đồ án:

- Thu thập nhiều dữ liệu hơn, tạo ra nhiều dữ liệu hơn để có thể huấn luyện mô hình mang lại độ chính xác cao hơn.
- Làm giàu dữ liệu bằng cách tăng cường dữ liệu theo nhiều cách để có thể biểu diễn dữ liệu được cả những trường hợp mà chúng ta không tự thu thập dữ liệu được.
- Tìm hiểu những thuật toán mới hơn và hiệu quả hơn để tăng hiệu suất của mô hình.
- Áp dụng vào các bài toán thực tế hơn như thay đổi thời gian đèn giao thông tự động dựa trên số phương tiện, …

TÀI LIỆU THAM KHẢO

[1] Đỗ Văn Tiến (2024), *Slide bài giảng môn Nhận dạng*, Khoa Khoa học Máy tính, Trường Đại học Công Nghệ Thông Tin – Đại học Quốc gia Thành phố Hồ Chí Minh.

[2] Bui Tien Trung: SORT - Deep SORT: Một góc nhìn về Object Tracking https://viblo.asia/p/sort-deep-sort-mot-goc-nhin-ve-object-tracking-phan-2-djeZ1m78ZWz

[3] Vietnam Vehicle Dataset https://universe.roboflow.com/car-classification/vietnamese-vehicle/dataset/1

[4] G. Jocher, A. Chaurasia, and J. Qiu, "YOLO by Ultralytics." https://github.com/ultralytics/ultralytics, 2023.

[5] YOLOv8: Comprehensive Guide to State of The Art Object Detection: https://learnopencv.com/ultralytics-yolov8