BỘ GIÁO DỤC VÀ ĐÀO TẠO
ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN

BÁO CÁO ĐỒ ÁN

MÔN CS406 – XỬ LÝ ẢNH VÀ ỨNG DỤNG

ĐỀ TÀI: FACE MASK DETECTION

Giảng viên hướng dẫn : Đỗ Văn Tiến

Nhóm thực hiện :

Trần Nhật Đức - 21521968
Hồ Trung Tín - 21521536
Nguyễn Đức Tuệ - 21521644

Lớp: **CS406.O12.KHCL**

Tp HCM, tháng 12 năm 2023

NHẬN XÉT CỦA GIẢNG VIÊN HƯỚNG DẪN

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

Tp.HCM, ngày … tháng … năm …

GVHD

(Ký tên)

MỤC LỤC

Link đến github: https://github.com/jayjay2k3/CS406_FaceMaskDetection

Link đến dataset: https://www.kaggle.com/datasets/andrewmvd/face-mask-detection

1. Tổng quan về đồ án:

a) Tóm tắt đồ án:

- Mô hình computer vision này được phát triển nhằm mục đích phát hiện và phân loại người trong hình ảnh hoặc video liệu họ có đang đeo khẩu trang hay không.
- Đồ án này có thể được ứng dụng trong nhiều lĩnh vực thực tế:

- Đại dịch COVID-19 trong quá khứ đã làm nổi bật tầm quan trọng của việc đeo khẩu trang như một biện pháp phòng ngừa quan trọng trong bối cảnh dịch bệnh.
- Các mô hình phát hiện khẩu trang có thể được sử dụng trong các trung tâm y tế để đảm bảo rằng bệnh nhân, người thăm và nhân viên tuân thủ các giao thức đeo khẩu trang. Điều này rất quan trọng trong việc ngăn chặn sự lây lan của các bệnh truyền nhiễm trong bệnh viện và phòng khám.
- Trong các ngành công nghiệp nơi việc đeo khẩu trang đóng vai trò rất quan trọng, như các nhà máy sản xuất, các công trường xây dựng, thì việc triển khai mô hình phát hiện người đeo khẩu trang có thể đóng góp vào việc duy trì một môi trường làm việc an toàn.

b) Mô tả Input và Output của bài toán:

- INPUT:

1 hình ảnh hoặc 1 video từ camera an ninh chứa các khuôn mặt người cần được phân tích để xác định xem người đó có đeo khẩu trang hay không, nếu có thì họ có đeo đúng cách không.

- OUTPUT:

- Mỗi khuôn mặt người được phát hiện trong ảnh hoặc video sẽ được 1 bounding box bao quanh.
- Nếu người đó đang đeo khẩu trang, một nhãn "Đeo khẩu trang" sẽ được gán cho bounding box của họ.
- Nếu người đó không đeo khẩu trang, một nhãn "Không đeo khẩu trang" sẽ được gán cho bounding box của họ.

- Nếu người đó đeo khẩu trang không đúng cách (khẩu trang không che mũi, không che miệng), một nhãn "Đeo không đúng cách" sẽ được gán cho bounding box của họ.

2. Dataset:

- Nhóm sử dụng bộ dataset Face Mask Detection từ kaggle, chứa 853 hình ảnh thuộc về 3 lớp, cùng với các bounding boxes của chúng theo định dạng PASCAL VOC.
- Các lớp là:
  - With mask;
  - Without mask;
  - Mask worn incorrectly.

a) Tiền xử lý dữ liệu:

- Sau khi tải về bộ dataset từ kaggle, chúng em chuyển các ảnh và tệp annotation tương ứng có định dạng PASCAL VOC lên Roboflow. Từ đây, bộ dữ liệu sẽ được chuyển đổi định dạng đầu ra thành YOLO Keras TXT.
- Ngoài ra, bộ dataset còn được tiền xử lý để chuẩn hóa và làm sạch. Chúng em sẽ thực hiện các biện pháp xử lý ảnh như cắt tỉa hoặc chỉnh sửa kích thước để làm giảm nhiễu và tối ưu hóa chất lượng dữ liệu. Cuối cùng, bộ dữ liệu sẽ được chia làm 2 tập train và test theo tỉ lệ 80:20.

b) Về Dataset:

- Bộ dữ liệu bao gồm tổng cộng 853 tấm ảnh, mỗi tấm ảnh chứa 1 hoặc nhiều khuôn mặt người, mỗi khuôn mặt người sẽ thuộc một trong ba class: With mask, Without mask, Mask worn incorrectly.

- Mỗi tấm ảnh sẽ có tương ứng một file text riêng, file này sẽ được đặt tên giống với tấm ảnh, nó chứa thông tin về các đối tượng được gắn nhãn trong ảnh, bao gồm vị trí của bounding box và nhãn của chúng.

<label_1> <x_min_1> <y_min_1> <width_1> <height_1>

<label_2> <x_min_2> <y_min_2> <width_2> <height_2>

...

- <label_i>: Nhãn của bounding box thứ i (0 là có đeo khẩu trang, 1 là không đeo khẩu trang, 2 là đeo không đúng cách).
- <x_min_i>, <y_min_i>: Tọa độ của góc trái phía trên của bounding box thứ i, được biểu diễn bằng tỉ lệ so với chiều rộng và chiều cao của hình ảnh.
- <width_i>, <height_i>: Chiều rộng và chiều cao của bounding box thứ i, được biểu diễn bằng tỉ lệ so với chiều rộng và chiều cao của hình ảnh.

- Bộ dữ liệu được chia train và test theo tỉ lệ 80:20. Ngoài ra, model còn được test qua một số video dài từ 10 giây đến 1 phút của nhóm.

3. Cơ sở lý thuyết, training và đánh giá model:

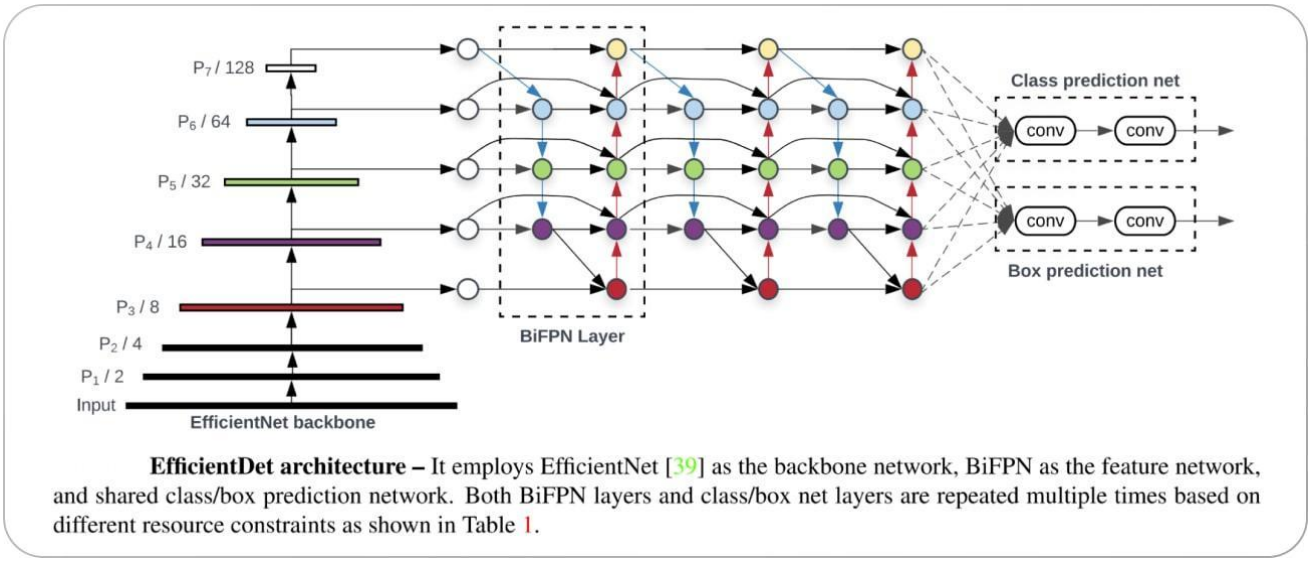
a) YOLO:

- "YOLO", viết tắt của "You Only Look Once", là một đại gia đình của các mô hình phát hiện đối tượng được giới thiệu bởi Joseph Redmon trong một bài báo năm 2016 có tiêu đề "You Only Look Once: Unified, Real-Time Object Detection".
- Kể từ đó, các biến thể của mô hình phát hiện đối tượng YOLO đã nhanh chóng phát triển, với việc phát hành phiên bản YOLO-v8 mới nhất vào tháng 1 năm 2023. Các biến thể của YOLO dựa trên nguyên tắc của hiệu suất phân loại thời gian thực và cao, dựa trên các tham số tính toán hạn chế nhưng hiệu quả.
- YOLO là một phương pháp phát hiện đối tượng thời gian thực trong ảnh và video bằng cách sử dụng một mạng neural network end-to-end để dự đoán các hộp giới hạn (bounding boxes) và xác suất lớp (class probabilities) cùng một lúc.

b) Model YOLOv5:

- YOLOv5 là một sự cải tiến lớn so với các phiên bản trước đó, với nhiều cải tiến về hiệu suất và tính linh hoạt.

- Mô hình YOLOv5 được thiết kế để đơn giản hóa cả quá trình huấn luyện và triển khai, với việc tối ưu hóa mô hình cho tốc độ và hiệu suất cao.
- YOLO v5 được giới thiệu vào năm 2020 bởi cùng một nhóm đã phát triển thuật toán YOLO dưới dạng một dự án mã nguồn mở và được duy trì bởi Ultralytics. YOLO v5 xây dựng dựa trên sự thành công của các phiên bản trước đó và thêm vào đó một số tính năng và cải tiến mới.
- Khác với YOLO, YOLO v5 sử dụng một kiến trúc phức tạp hơn được gọi là EfficientDet, dựa trên kiến trúc mạng EfficientNet. Việc sử dụng một kiến trúc phức tạp hơn trong YOLO v5 cho phép nó đạt được độ chính xác cao hơn và khả năng tổng quát hóa tốt hơn đối với một loạt các loại đối tượng.



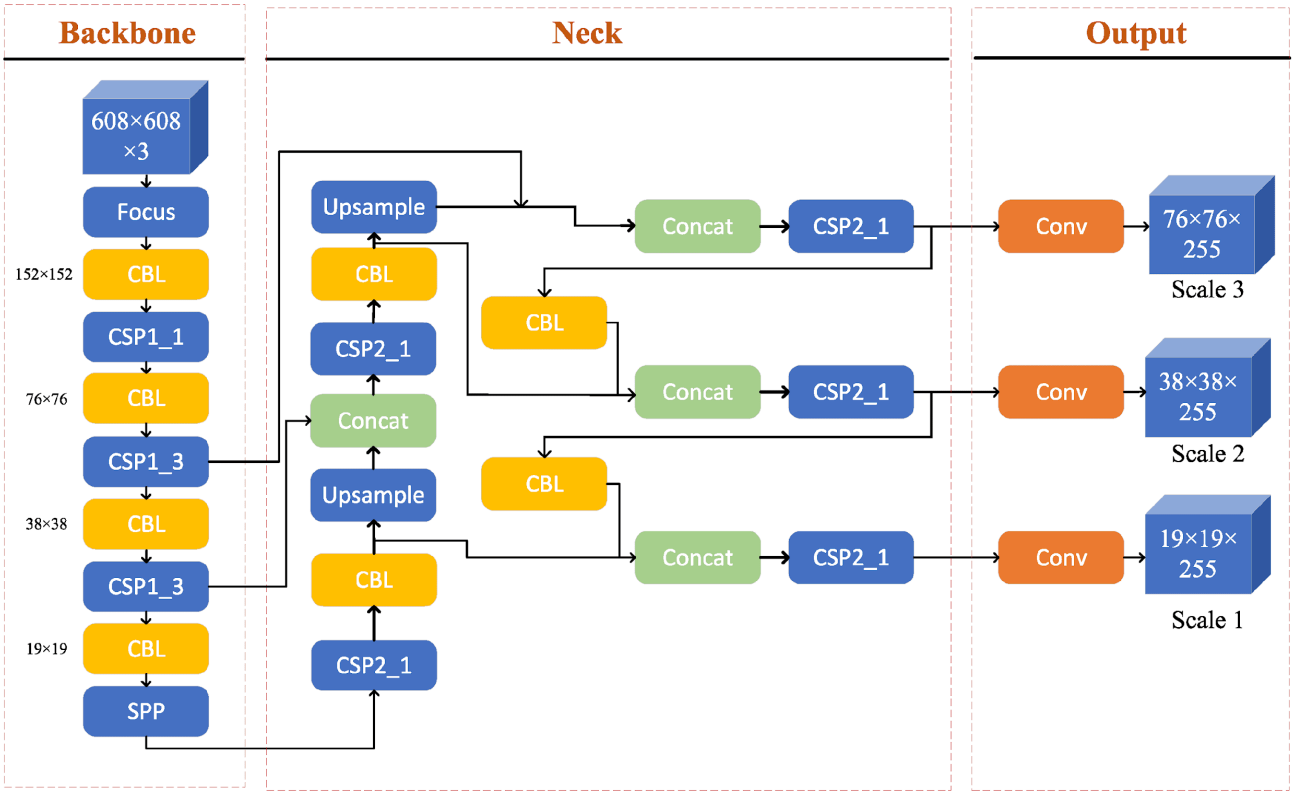
EfficientDet architecture – It employs EfficientNet [39] as the backbone network, BiFPN as the feature network, and shared class/box prediction network. Both BiFPN layers and class/box net layers are repeated multiple times based on different resource constraints as shown in Table 1.

- Dưới đây là một phân tích chi tiết về sự khác biệt giữa kiến trúc YOLOv4 và YOLOv5:
  - Backbone (Lõi):
    - YOLOv4: Sử dụng CSPResidualBlock, một kiến trúc residual block có sự kết hợp của CSP (Cross-Stage Partial connections) và các residual connections.

- YOLOv5: Sử dụng C3 module, một kiến trúc convolutional layer mới được thiết kế, có thể tối ưu hóa hiệu quả hơn so với CSPResidualBlock.

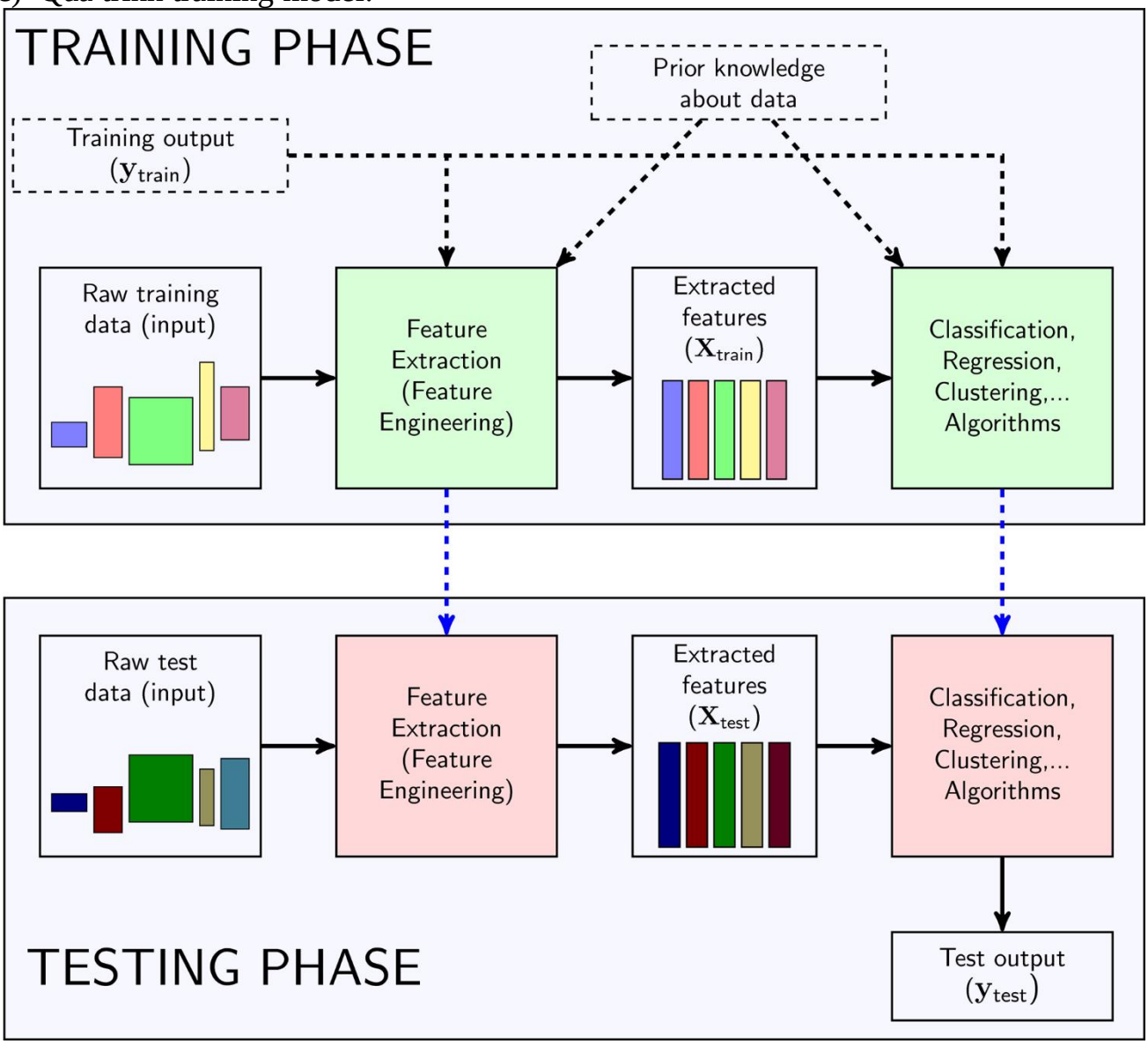
- Neck (Cổ):
  - YOLOv4: Sử dụng SPP (Spatial Pyramid Pooling) và PAN (Path Aggregation Network) để kết hợp thông tin đa phân cấp và cải thiện mạng.
  - YOLOv5: Sử dụng SPPF (Fused Spatial Pyramid Pooling) và PAN, một biến thể của SPP kết hợp thông tin cụ thể từ các đối tượng nhỏ hơn và PAN để tối ưu hóa sự kết hợp thông tin.
- Head (Đầu):
  - YOLOv4: Giữ nguyên từ YOLOv3, sử dụng một loạt các convolutional layers để dự đoán bounding box và xác suất lớp.
  - YOLOv5: Tiếp tục sử dụng đầu từ YOLOv3 mà không có thay đổi đáng kể.
- Các Thay Đổi Khác:
  - Data Augmentation (Tăng cường dữ liệu): YOLOv5 thêm vào các phương pháp tăng cường dữ liệu mới như Mosaic Augmentation, Copy-paste Augmentation, và MixUp Augmentation để cải thiện sự đa dạng và chất lượng của dữ liệu huấn luyện.
  - Loss Function (Hàm mất mát): YOLOv5 thêm hệ số scale cho Objectness Loss để cân bằng giữa các loại lỗi và cải thiện độ chính xác của mô hình.
  - Anchor Box (Hộp Neo): YOLOv5 sử dụng Auto Anchor sử dụng Genetic Algorithm để tự động tối ưu hóa các anchor box cho dữ liệu huấn luyện cụ thể.
  - Loại bỏ Grid Sensitivity: YOLOv5 loại bỏ Grid Sensitivity và thay đổi công thức để cải thiện độ nhạy của mô hình đối với các đối tượng nhỏ.
  - EMA Weight: Cân nhắc sử dụng Exponential Moving Average (EMA) Weight để cải thiện ổn định và hiệu suất của mô hình trong quá trình huấn luyện.

- Những thay đổi này giúp YOLOv5 đạt được hiệu suất và linh hoạt tốt hơn so với các phiên bản tiền nhiệm, với khả năng huấn luyện và triển khai dễ dàng hơn.



YOLOv5 architecture

c) Quá trình training model:



- Từ tập dataset đã có từ trước, dữ liệu đó sẽ được tiền xử lý trước khi đưa vào training.
- Sau đó, chúng sẽ được trích xuất các đặc trưng tạo ra các feature map tương đương.
- Những feature map này cuối cùng được đưa vào những thuật toán máy học để phân loại.

d) Về quá trình trích xuất đặc trưng (feature engineering):

Trong YOLO, quá trình trích xuất đặc trưng diễn ra thông qua một deep neural network được huấn luyện trên dữ liệu ảnh.

- Tiền Xử Lý Ảnh: Trước khi được đưa vào mạng neural network, ảnh đầu vào được tiền xử lý để chuẩn hóa kích thước và giá trị pixel.

- Mạng Neural Network Convolutional (CNN): YOLO sử dụng một mạng neural network tích chập (CNN) để trích xuất các đặc trưng từ ảnh đầu vào. Mạng CNN này có thể bao gồm nhiều lớp convolutional, pooling, và các lớp kích hoạt phi tuyến tính như ReLU.
- Lớp Global Average Pooling (GAP): Sau khi đi qua các lớp convolutional, đầu ra của mạng CNN thường là một bản đồ đặc trưng (feature map) có kích thước lớn. Để giảm số lượng tham số và tính toán, YOLO sử dụng lớp GAP để chuyển đổi mỗi feature map thành một giá trị đặc trưng duy nhất bằng cách tính trung bình của tất cả các giá trị pixel trên feature map.
- Lớp Fully Connected Layer (FC): Sau lớp GAP, các giá trị đặc trưng được đưa vào một hoặc nhiều lớp fully connected layer để ánh xạ từ không gian đặc trưng sang không gian dự đoán. Các lớp này có thể liên kết các đặc trưng với các lớp phân loại và dự đoán các bounding box.
- Lớp Output: Cuối cùng, các lớp output của mạng neural network sẽ dự đoán các bounding box và xác suất của các lớp đối tượng trong ảnh. Các bounding box thường được dự đoán bằng cách sử dụng các hàm kích hoạt như sigmoid để dự đoán tọa độ (vị trí và kích thước) của bounding box, cùng với các xác suất của các lớp đối tượng thông qua softmax hoặc sigmoid.

e) Lý do chọn model:

Lý do chúng em chọn mô hình YOLOv5 cho đồ án này là dựa trên các ưu điểm sau:

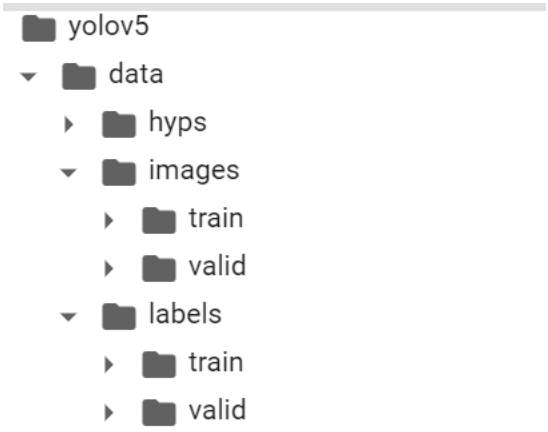
- Hiệu Suất Cao: YOLOv5 là một trong những mô hình object detection hiện đại nhất, với hiệu suất cao và tốc độ xử lý nhanh, phù hợp với yêu cầu của dự án.
- Dễ Triển Khai: Mô hình YOLOv5 có cấu trúc đơn giản và dễ dàng triển khai, giúp giảm bớt thời gian và công sức trong việc triển khai hệ thống.
- Tính Linh Hoạt: YOLOv5 hỗ trợ các ứng dụng trên nhiều nền tảng và môi trường khác nhau, từ các ứng dụng di động đến các hệ thống nhúng, giúp dễ dàng tích hợp vào các ứng dụng thực tế.

f) Quá trình train model:

- Upload bộ dữ liệu lên google drive.
- Tải các thư viện và framework cần thiết để train model.

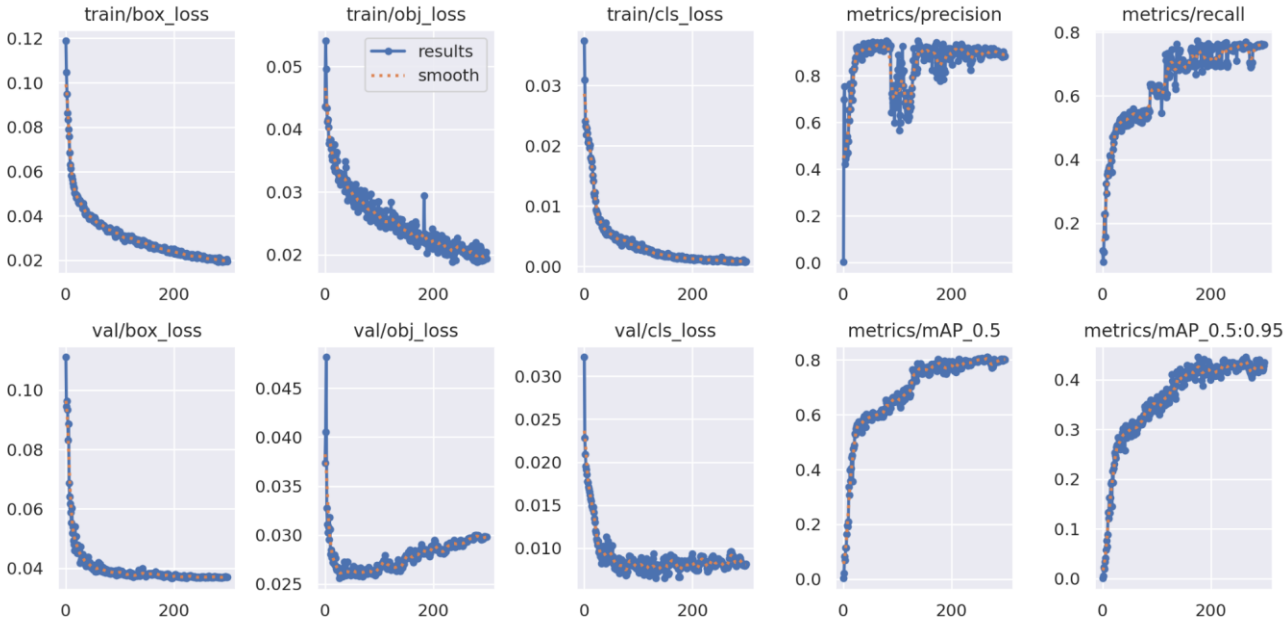
```
git clone https://github.com/ultralytics/yolov5
cd yolov5
pip install -r requirements.txt
```

- Giải nén bộ dữ liệu, chia bộ dữ liệu thành 2 tập train và test. Tỉ lệ là 80:20. Kiểm tra lại sau khi giải nén thì folder images/ và labels/ phải nằm trong yolov5/data như sau:

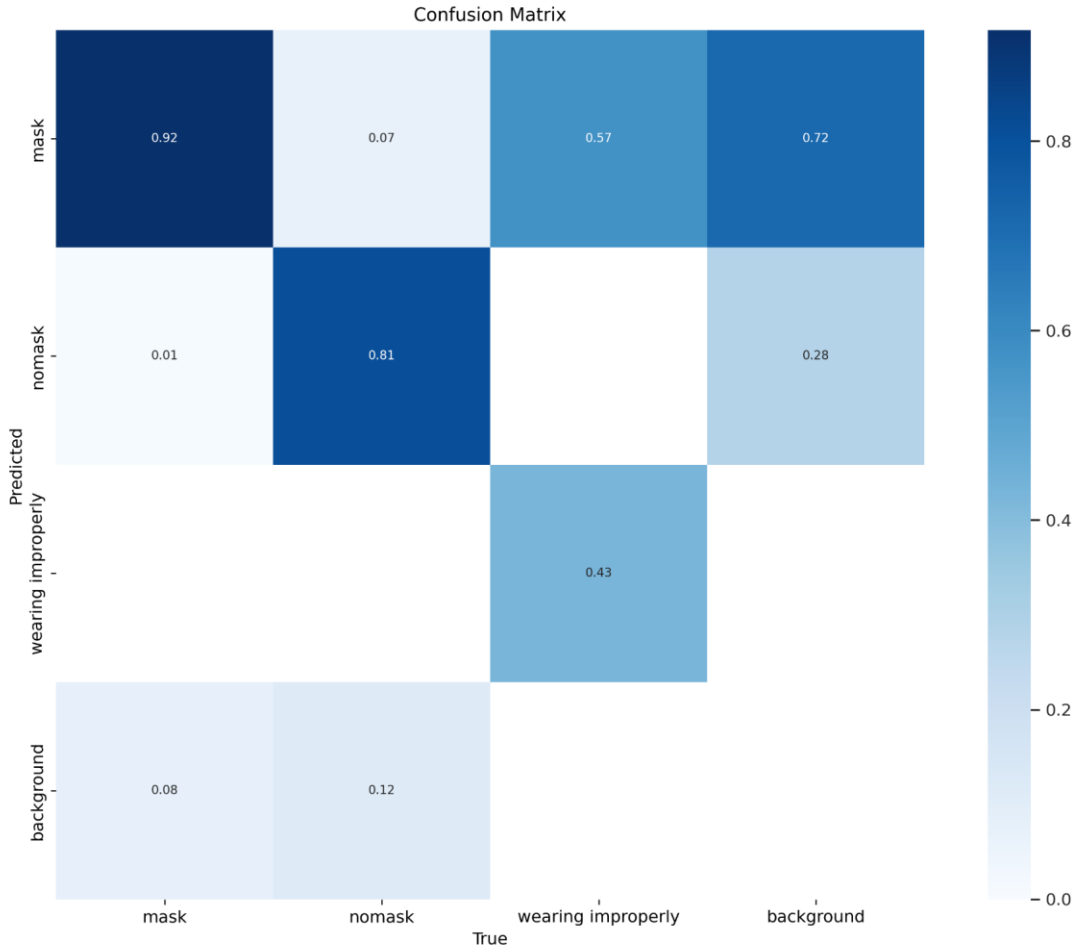


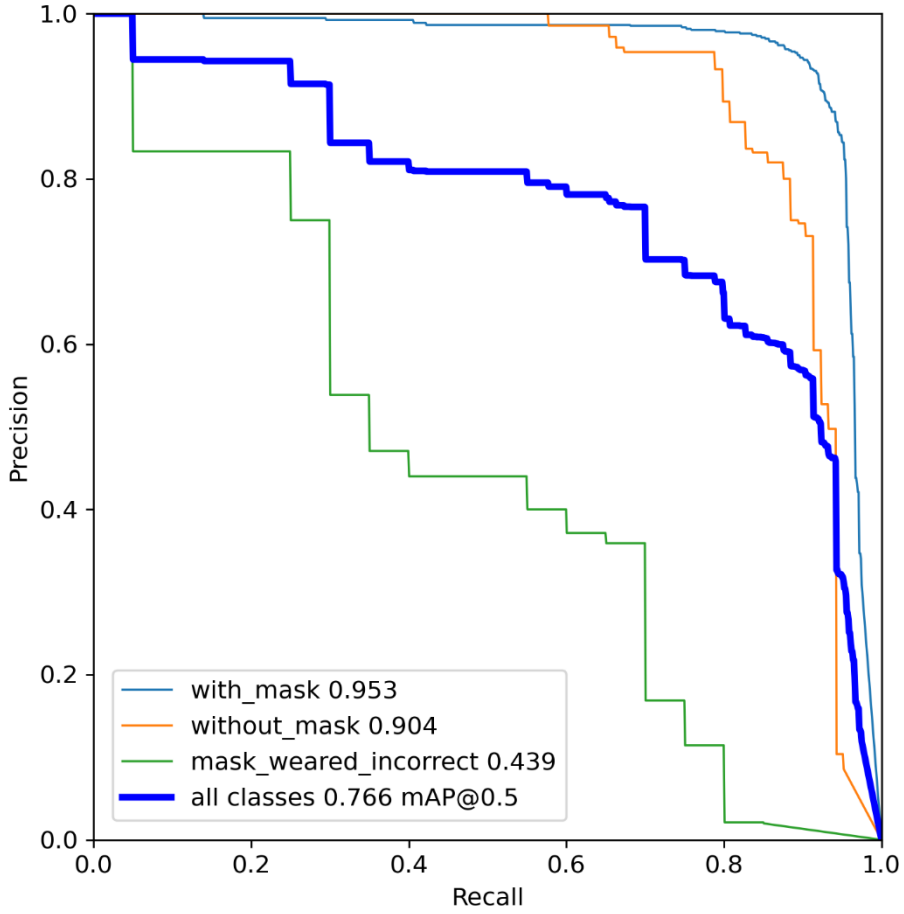
- Để huấn luyện mô hình, chúng em chạy train.py, với các đối số như sau:
  - img: kích thước ảnh đầu vào – 416.
  - batch: kích thước batch – 16.
  - epochs: số lượng epochs – 300.
  - data: đường dẫn đến tệp dataset.yaml
  - weights: đường dẫn trọng số ban đầu, mặc định là yolov5s.pt

g) Kết quả và đánh giá:
(training loss)



(confusion matrix)





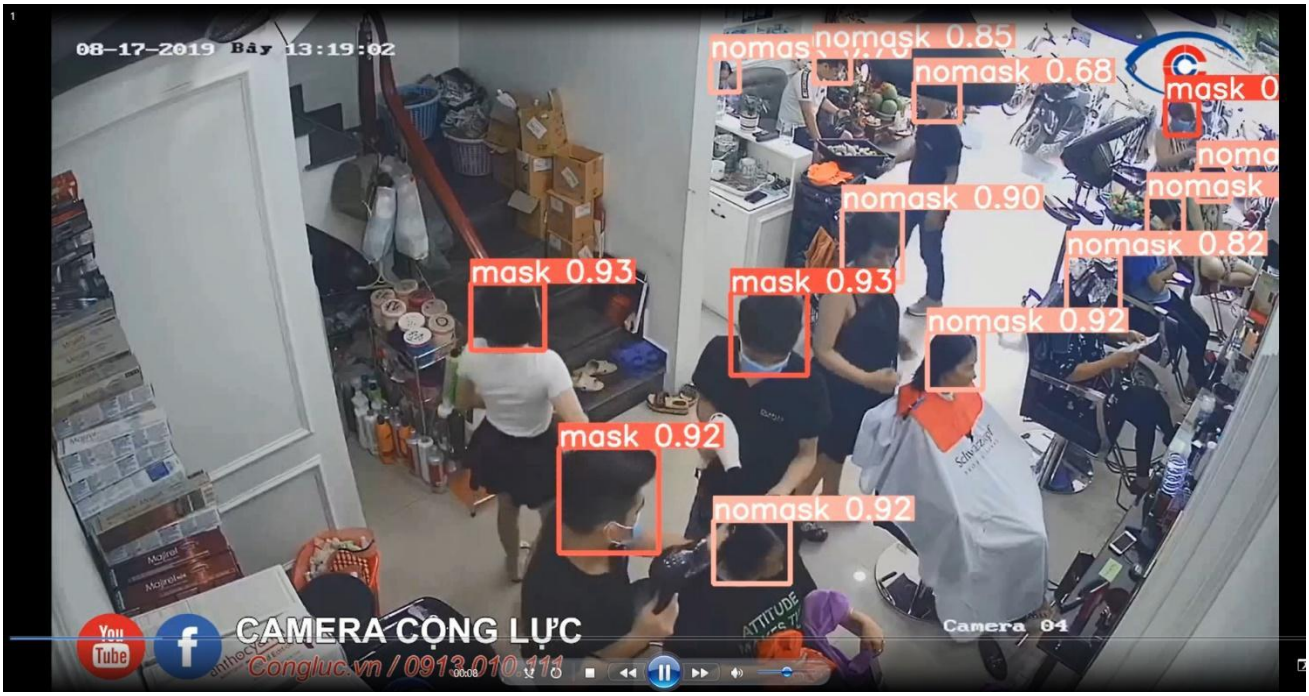
(đánh giá IoU và mAP)

Class	#Labels	Precision	Recall	mAP^{val} 0.5	mAP^{val} 0.5:0.95
with mask	630	0.94	0.9	0.95	0.64
without mask	104	0.86	0.82	0.9	0.6
mask weared incorrectly	20	0.72	0.3	0.43	0.24
total	754	0.84	0.67	0.76	0.49

h) Thực hiện dự đoán:

Để thực hiện dự đoán trên ảnh và luồng video, chúng em gọi detect.py và điều chỉnh các tham số sau:

- weights: trọng số của mô hình đã huấn luyện
- source: tệp/thư mục đầu vào để thực hiện dự đoán, 0 để sử dụng webcam
- output: thư mục để lưu kết quả
- conf-thres: ngưỡng độ tin cậy của đối tượng



Nhận xét:

- Mô hình nhìn chung hoạt động khá tốt trong việc phát hiện người đeo hoặc không đeo khẩu trang trên ảnh và video.
- Mô hình chưa phát hiện tốt các trường hợp đeo khẩu trang không đúng cách, điều này là do bộ dữ liệu chưa được cân bằng số lượng giữa các nhãn, nhãn "Mask" và "No mask" có số lượng lớn hơn nhiều so với nhãn "Wear

Improperly". Sự đa dạng trong dữ liệu huấn luyện có thể cần được tăng cường thêm để cải thiện hiệu suất của mô hình.

- Mô hình gặp khó khăn trong việc phát hiện khẩu trang dưới một số điều kiện cụ thể, ví dụ, nó có xu hướng nhầm lẫn giữa râu dài và một chiếc khẩu trang. Điều này có thể được giảm thiểu bằng cách thêm nhiều sự phong phú trong bộ dữ liệu huấn luyện và có thể thực hiện thêm các kỹ thuật data augmentation.
- Có thể cần thực hiện các bước tinh chỉnh mô hình và thử nghiệm với các siêu tham số khác nhau để cải thiện hiệu suất và đảm bảo rằng mô hình hoạt động hiệu quả trong các điều kiện đa dạng.

4. API:

a) FastAPI:

FastAPI là framework hiện đại, hiệu suất cao để xây dựng web API với Python 3.7+ dựa trên các tiêu chuẩn của Python. Nó giúp các nhà phát triển code ít hơn, triển khai nhanh hơn và hỗ trợ docs API đầy đủ, cũng cải thiện tốc độ khi tích hợp API vì đã support docs API đầy đủ. Sẽ là một sự lựa chọn hoàn hảo cho một dự án cần triển khai trong thời gian ngắn nhưng vẫn đảm bảo đầy đủ các yếu tố đi kèm.

b) Các tính năng của FastAPI:

- Triển khai nhanh
- Trực quan và tiện dụng (Intuitive): Được nhiều trình chỉnh sửa hỗ trợ, có thể chạy trên nhiều nền tảng khác nhau. Tốn ít thời gian gỡ lỗi hơn (hưởng lợi từ Python là một ngôn ngữ thông dịch)
- Dễ dàng tiếp cận (Easy): Thiết kế dễ sử dụng và hỏi tập nhanh thông qua docs.
- Code ngắn gọn (Short): Code ngắn gọn, bao gồm nhiều tính năng đã được triển khai sẵn và giúp giảm thiểu bugs.
- Mạnh mẽ (Robust): Code sẵn sàng cho môi trường production và API docs được sinh tự động.

c) Lí do sử dụng FastAPI

- Nhóm cần triển khai một api nhanh chóng với 2 phương thức post và get
- Sử dụng ngôn ngữ python để xây dựng api

d) Các phương thức của API

- Phương thức post:

Input: ảnh (định dạng jpg) hoặc video (định dạng mp4)

Output: kết quả sau khi xử lí:

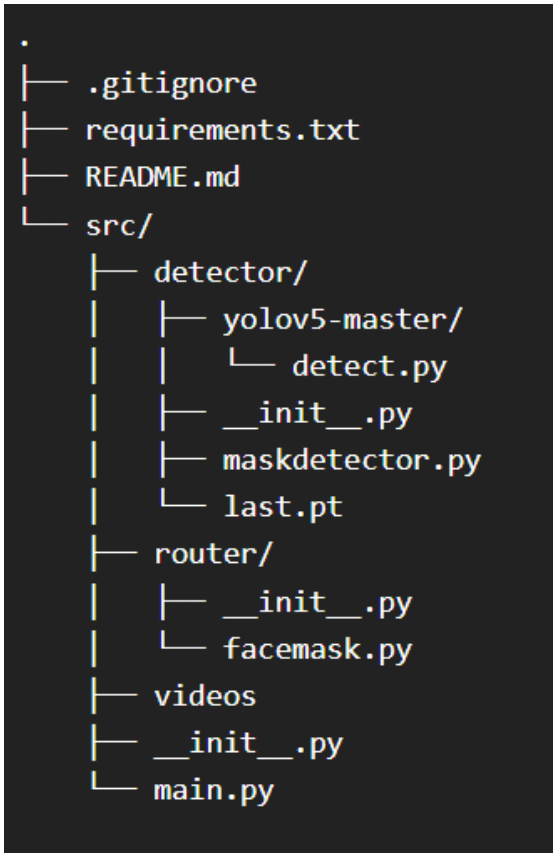
+ Thành công: tên video / ảnh sau khi đã xử lí
+ Thất bại: lí do không thể xử lí được video / ảnh

- Phương thức get:

Input: Tên của ảnh / video đã qua xử lí của mô hình

Output: Trả về ảnh / video nếu có tồn tại trên server

e) Cấu trúc thư mục API



- src/detector/maskdetector.py: wrapper cho mô hình phát hiện khẩu trang sử dụng yolov5

-src/detector/yolov5-master/detect.py: triển khai mô hình phát hiện khẩu trang

-src/detector/yolov5-master/last.pt: bộ trọng số sử dụng cho mô hình phát hiện khẩu trang

src/router/facemask.py: triển khai 2 phương thức get và post của api

src/main.py: khởi tạo API với fastAPI

TÀI LIỆU THAM KHẢO:

1. https://towardsdatascience.com/face-mask-detection-using-yolov5-3734ca0d60d8
2. https://github.com/iAmEthanMai/mask-detection-dataset
3. https://colab.research.google.com/github/ultralytics/yolov5/blob/master/tutorial.ipynb#scrollTo=N3qM6T0W53gh
4. You Only Look Once: Unified, Real-Time Object Detection
5. https://medium.com/analytics-vidhya/covid-19-face-mask-detection-using-yolov5-8687e5942c81
6. https://github.com/spacewalk01/yolov5-face-mask-detection
7. https://www.v7labs.com/blog/yolo-object-detection#:~:text=YOLO%20v5%20uses%20a%20new,clusters%20as%20the%20anchor%20boxes.
8. https://blog.roboflow.com/yolov5-improvements-and-evaluation/
9. https://iq.opengenus.org/yolov5/
10. https://viblo.asia/p/tong-hop-kien-thuc-tu-yolov1-den-yolov5-phan-1-naQZRRj0Zvx