



BÁO CÁO ĐỒ ÁN CUỐI KỲ

PHÁT HIỆN MỘT SỐ BỆNH TRÊN LÁ CÂY CÀ PHÊ

- Giáo viên hướng Dẫn :

STT	Họ tên	Email
1	PGS.TS. Lê Đình Duy	duyld@uit.edu.vn
2	Ths. Phạm Nguyễn Trường An	truonganpn@uit.edu.vn

- Giới thiệu thành viên nhóm

STT	MSSV	Họ tên	Gmail
1	19521322	Huỳnh Ngọc Công Danh	19521322@gm.uit.edu.vn
2	19522524	Nguyễn Phú Vinh	19522524@gm.uit.edu.vn
3	19521858	Võ Tuấn Minh	19521858@gm.uit.edu.vn

Chương 0. Giải trình chỉnh sửa sau vấn đáp

Cách để đánh giá mô hình:

- Nhóm đã tiến hành cập nhật về đánh giá mô hình và cách xác định True Positive, False Positive sau những góp ý của thầy. Lý do chọn mean average precision là metric để đánh giá. [link](#)

Số lượng số lượng sai sót của mỗi class trong mỗi mô hình:

- Nhóm tiến hành thống kê True Positive ,False Positive của mỗi class trong mỗi mô hình để tiến hành xác định số lượng. [link](#)

Định dạng của dataset dùng để training model:

- Nhóm đã tiến hành tìm hiểu và xác định nội dung dataset dùng để training model. [link](#)

Chương 1. TỔNG QUAN

1.1. Mô tả bài toán

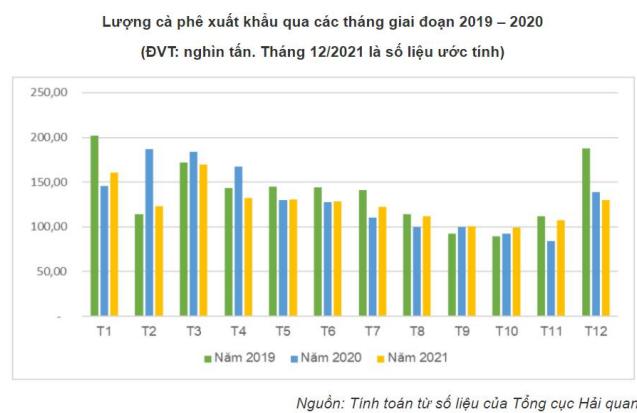
- Ngữ cảnh ứng dụng :

- Hiện nay, cà phê là một trong những loại thức uống được sử dụng phổ biến lẩn trong và ngoài nước. Cà phê được sản xuất từ những hạt cà phê rang, lấy trên cây cà phê.



Hình 1. Ảnh minh họa (Nguồn: Internet).

- Theo báo điện tử VTV, trên 90% tổng sản lượng cà phê của Việt Nam dành cho việc xuất khẩu, khoảng 10% còn lại là dành cho việc chế biến và tiêu thụ trong nước. Cà phê nằm trong nhóm hàng nông sản xuất khẩu chủ lực của Việt Nam, hiện đứng thứ 2 thế giới (chỉ sau Brazil). Tính chung cả năm 2021, xuất khẩu cà phê của Việt Nam đạt 1,52 triệu tấn.



Hình 2. Lượng cà phê xuất khẩu qua các tháng giai đoạn từ 2019-2021.

- Sản lượng tuy lớn nhưng cà phê vẫn chưa đáp ứng được các tiêu chuẩn về chất lượng. Nhằm nâng cao chất lượng của hạt cà phê, việc phát hiện và xử lý những căn bệnh trên lá của cây rất quan trọng. Nhận thấy được vấn đề đó nên nhóm đã quyết định áp dụng những kiến thức của mình và những công nghệ trong lĩnh vực Machine Learning để giải quyết bài toán phát hiện một số loại bệnh trên lá cây cà phê.
- Mô hình hướng tới người sử dụng là người trồng cây cà phê, xây dựng một ứng dụng có thể giúp người trồng có thể phát hiện chính xác hơn các loại bệnh đang gặp trên lá của cây và đưa ra được giải pháp phù hợp nhằm loại bỏ bệnh và tác nhân gây bệnh.

- Input và Output:

- Input:

- Một tấm ảnh chụp hình lá của cây cà phê đang bị bệnh.



Hình 3. Ví dụ về ảnh input thích hợp.

- Các điều kiện ràng buộc :
 - Ảnh chụp tập trung vào lá đang bị bệnh
 - Chụp được từ cuống lá đến chóp lá
 - Chụp trong điều kiện ánh sáng ban ngày
- Output:
 - Bounding box bao quanh lá cây bị bệnh
 - Tên loại bệnh
 - ➔ Trong ứng dụng thực tế hoàn chỉnh, dựa vào tên loại bệnh xác định được ứng dụng sẽ đưa ra các giải pháp phù hợp cho người trồng

1.2. Mô tả dữ liệu

- Dữ liệu của bài toán được nhóm tự thu thập từ một số vườn chuyên trồng cà phê trên địa bàn huyện Lạc Dương và địa bàn thành phố Đà Lạt thuộc tỉnh Lâm Đồng. Trong quá trình thu thập dữ liệu, nhóm gặp nhiều khó khăn như việc di chuyển đến các vườn cà phê khá xa so với nhà riêng (khoảng hơn 20 km), dịch bệnh COVID-19 khiến cho việc đi qua các chốt phong tỏa khó khăn.



Hình 4. Vườn cà phê thuộc xã Trại Hành, thành phố Đà Lạt, tỉnh Lâm Đồng.

- Bộ dữ liệu về lá cây cà phê hiện nay chưa có ai thu thập nên số lượng dữ liệu mà nhóm có vẫn còn hạn chế do dữ liệu tự thu thập và xử lý. Mục đích của việc tự thu thập dữ liệu là để phù hợp với ngữ cảnh ứng dụng của bài toán.

Chương 2. CÁC NGHIÊN CỨU TRƯỚC

- Bài toán của nhóm đặt ra là muốn hướng đến bài toán thuộc loại Object Detection (Phát hiện đối tượng), định vị đối tượng trong ảnh và xác định đối tượng thuộc loại nào. Trong lĩnh vực thị giác máy tính thì bài toán phát hiện đối tượng đạt được nhiều kết quả khi áp dụng hướng tiếp cận Deep learning. Có thể kể đến một số hướng tiếp cận tiên tiến hiện nay bao gồm RCNN, Fast RCNN, Faster RCNN, Mask RCNN, RetinaNet, YOLO, v.v
- Faster RCNN
 - Phương pháp Faster RCNN là một trong các phương pháp phát hiện đối tượng sử dụng mạng Deep learning đạt độ chính xác cao trên các tập dữ liệu chuẩn như COCO . Faster RCNN được cải tiến dựa trên 2 phương pháp trước đó là RCNN và Fast RCNN.
- Mask RCNN
 - Phương pháp Mask RCNN là phương pháp thực hiện song song 2 bài toán là phân vùng đối tượng (Instance Segmentation) và phát hiện đối tượng. Mask RCNN là phương pháp được cải tiến từ Faster RCNN.
- Restinanet
 - RetinaNet là một phương pháp tiếp cận one-stage tức là ngay trong bản thân cấu trúc mạng của phương pháp đã bao gồm thao tác đưa ra vùng đề xuất
- YOLO
 - YOLO được xem là phương pháp đầu tiên xử lý dữ liệu theo thời gian thực và vẫn đạt được độ chính xác cao.
- Một số kết quả nghiên cứu được đánh giá trên tập COCO. COCO (Common Objects in Context) là một tập datasets phục vụ cho các bài toán Object Detection, Segmentation, Image Captioning. Tập dữ liệu tổng cộng có khoảng 1.5 triệu object thuộc về 80 class khác nhau.

	backbone	AP	AP ₅₀	AP ₇₅	AP _S	AP _M	AP _L
<i>Two-stage methods</i>							
Faster R-CNN+++ [3]	ResNet-101-C4	34.9	55.7	37.4	15.6	38.7	50.9
Faster R-CNN w FPN [6]	ResNet-101-FPN	36.2	59.1	39.0	18.2	39.0	48.2
Faster R-CNN by G-RMI [4]	Inception-ResNet-v2 [19]	34.7	55.5	36.7	13.5	38.1	52.0
Faster R-CNN w TDM [18]	Inception-ResNet-v2-TDM	36.8	57.7	39.2	16.2	39.8	52.1
<i>One-stage methods</i>							
YOLOv2 [13]	DarkNet-19 [13]	21.6	44.0	19.2	5.0	22.4	35.5
SSD513 [9, 2]	ResNet-101-SSD	31.2	50.4	33.3	10.2	34.5	49.8
DSSD513 [2]	ResNet-101-DSSD	33.2	53.3	35.2	13.0	35.4	51.1
RetinaNet [7]	ResNet-101-FPN	39.1	59.1	42.3	21.8	42.7	50.2
RetinaNet [7]	ResNeXt-101-FPN	40.8	61.1	44.1	24.1	44.2	51.2
YOLOv3 608 × 608	Darknet-53	33.0	57.9	34.4	18.3	35.4	41.9

Hình 5. Performance trên tập COCO.

Chương 3. XÂY DỰNG BỘ DỮ LIỆU

3.1. Quá trình thu thập:

- Dữ liệu được nhóm thu thập thủ công bằng camera của điện thoại.
- Điện thoại sử dụng: Iphone 7 Plus, 32GB.
- Mỗi tấm ảnh gốc có kích thước 3024 x 4032 (camera nằm ngang), 4032 x 3024 (camera nằm dọc)

Property	Value	Property	Value
Date taken	12/20/2021 8:53 AM	Date taken	1/2/2022 9:36 AM
Program name	14.4	Program name	14.4
Date acquired		Date acquired	
Copyright		Copyright	
Image		Image	
Image ID		Image ID	
Dimensions	3024 x 4032	Dimensions	4032 x 3024
Width	3024 pixels	Width	4032 pixels
Height	4032 pixels	Height	3024 pixels
Horizontal resolution	72 dpi	Horizontal resolution	72 dpi
Vertical resolution	72 dpi	Vertical resolution	72 dpi
Bit depth	24	Bit depth	24
Compression		Compression	
Resolution unit	2	Resolution unit	2
Color representation	Uncalibrated	Color representation	Uncalibrated
Compressed bits/pixel		Compressed bits/pixel	
Camera		Camera	
Camera maker	Apple	Camera maker	Apple
Camera model	iPhone 7 Plus	Camera model	iPhone 7 Plus

Hình 6. Độ phân giải và camera sử dụng.

- File ảnh được lưu trữ trong cùng 1 folder trên máy tính dưới dạng tệp .JPG
- Thời gian thu thập dữ liệu:

STT	Thời gian thu thập	Địa điểm thu thập
1	20/12/2021	Huyện Lạc Dương
2	25/12/2021	Xã Trạm Hành
3	1/2/2022	Xã Trạm Hành

3.2. Tiêu chí khi thu thập dữ liệu :

- Chụp rõ nét tập trung vào lá cây bị bệnh.
- Chụp toàn bộ chiếc lá từ phần cuống lá đến chóp lá.

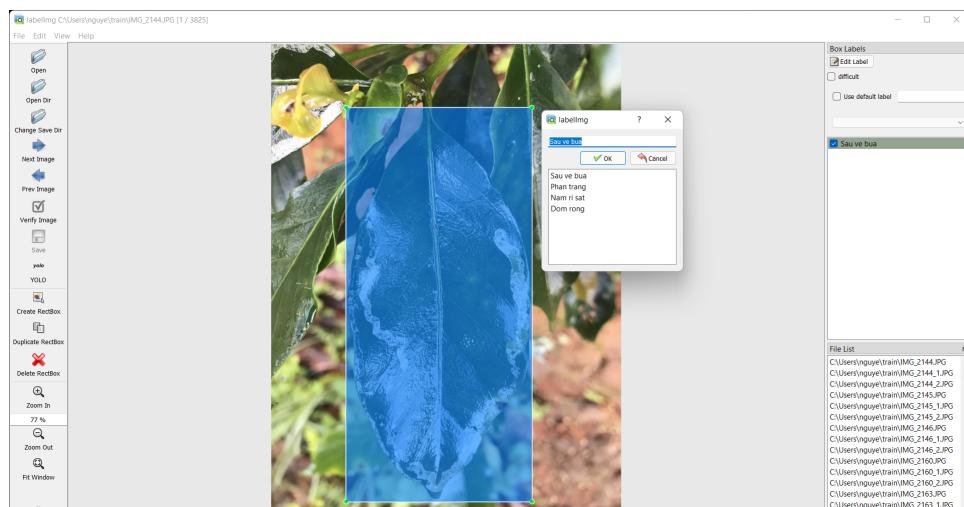
- Đảm bảo ánh sáng ban ngày.

3.3. Giảm độ phân giải của ảnh :

- Do mỗi ảnh có kích thước khá lớn nên dung lượng lưu trữ khá nặng. Đối với folder chứa toàn bộ dữ liệu gốc nặng khoảng 11.2Gb gây khó khăn trong việc lưu trữ nếu nhóm đã giảm độ phân giải xuống ~ 3.33 lần. Độ phân giải sau khi giảm 907 x 1209 và 1209 x 907. Dung lượng lưu trữ sau khi xử lý nặng khoảng 1.6Gb.

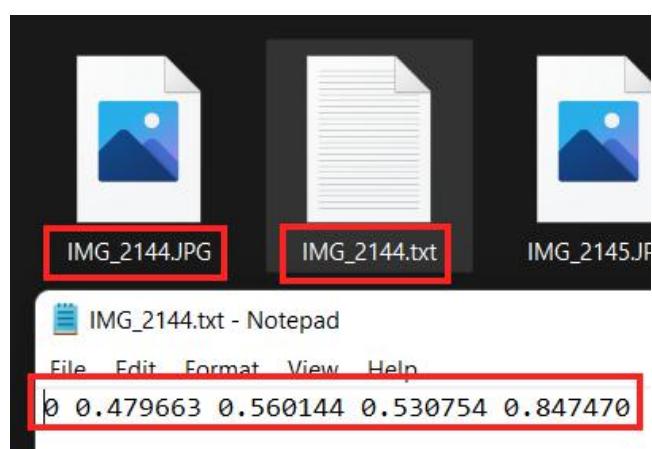
3.4. Gán nhãn dữ liệu :

- Sử dụng công cụ labelImg để tiến hành gán nhãn toàn bộ dữ liệu



Hình 7. Công cụ labelImg.

- Sử dụng thao tác kéo thả chuột để tạo bounding box cho đối tượng. Label được lưu thành file text có cùng tên với ảnh dưới dạng YOLO format.



Hình 8. Ảnh, label và label format của YOLO.

- Trong một ảnh có thể có nhiều lá nhưng chỉ label những lá bị bệnh và thấy rõ từ cuống lá đến chóp lá.
- Số loại label là 4. Được ký hiệu bằng 1 trong các chữ số 0, 1, 2, 3

Label 0: Bệnh sâu vẽ bùa

- Những lá bị sâu vẽ bùa gây hại sẽ bị co lại, biến dạng. Sâu non chui qua lớp biểu bì của lá để ăn phần nhu mô của lá tạo thành đường hầm ngoằn ngoèo màu trắng, trắng đục dưới lớp biểu bì.



Hình 9. Một số ví dụ về bệnh sâu vẽ bùa trên lá cà phê.

Label 1: Bệnh phấn trắng

- Bệnh phấn trắng do một số loại nấm có họ hàng gần gây ra. Triệu chứng chung là chúng tạo ra lớp bột có màu trắng xám trên bề mặt của lá.



Hình 10. Một số ví dụ về bệnh phấn trắng trên lá cà phê.

Label 2: Bệnh nấm rỉ sắt

- Trên lá xuất hiện các vết đốm hình tròn màu nâu cam hơi đỏ (giống rỉ sắt), xung quanh có vầng màu vàng úa.



Hình 11. Một số ví dụ về bệnh nấm rỉ sắt trên lá cà phê.

Label 3: Bệnh đốm rong

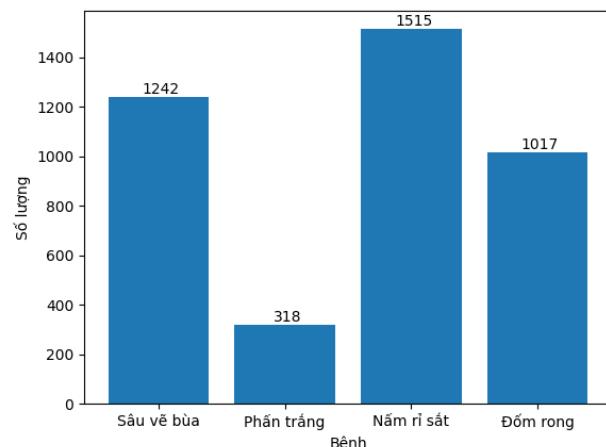
- Đốm bệnh có hình tròn lúc đầu nhỏ khoảng 3 - 5 mm, hơi nhô lên trên mặt lá do rong phát triển thành ung mịn, màu hơi vàng.



Hình 12. Một số ví dụ về bệnh đốm rong trên lá cà phê.

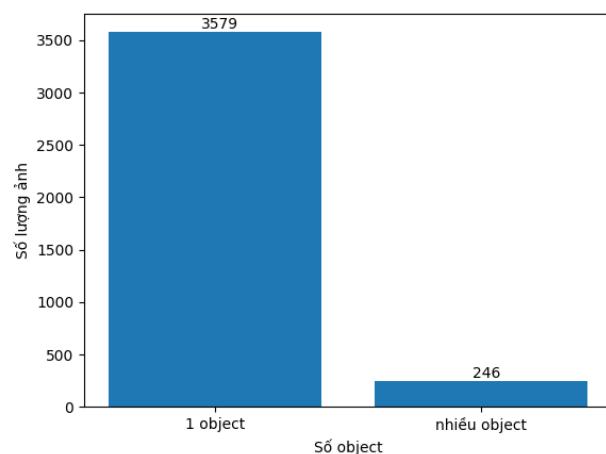
3.5. Thông số bộ dữ liệu :

- Tổng số lượng ảnh trong bộ dữ liệu là: 3825 ảnh
- Tổng số object là: 4092



Hình 13. Số lượng object thuộc từng loại label.

- Số ảnh có 1 object là: 3579
- Số ảnh có nhiều hơn 1 object là: 246



Hình 14. Số lượng ảnh có 1 object và nhiều object.

Nhận xét: Số lượng object thuộc bệnh phản trắng khá ít so với các bệnh khác, nguyên nhân là bệnh này xuất hiện khá ít tại các vườn cà phê thu thập dữ liệu.

- Tập dữ liệu được chia thành hai tập train và test với tỉ lệ là 80% cho tập train và 20% cho tập test



Hình 15. Số lượng object thuộc từng loại label trong tập train.



Hình 16. Số lượng object thuộc từng loại label trong tập test.

Trong đó:

- 0 : Sâu vẽ bùa
- 1 : Phản trăng
- 2 : Nấm rỉ sắt
- 3 : Đỗm rong

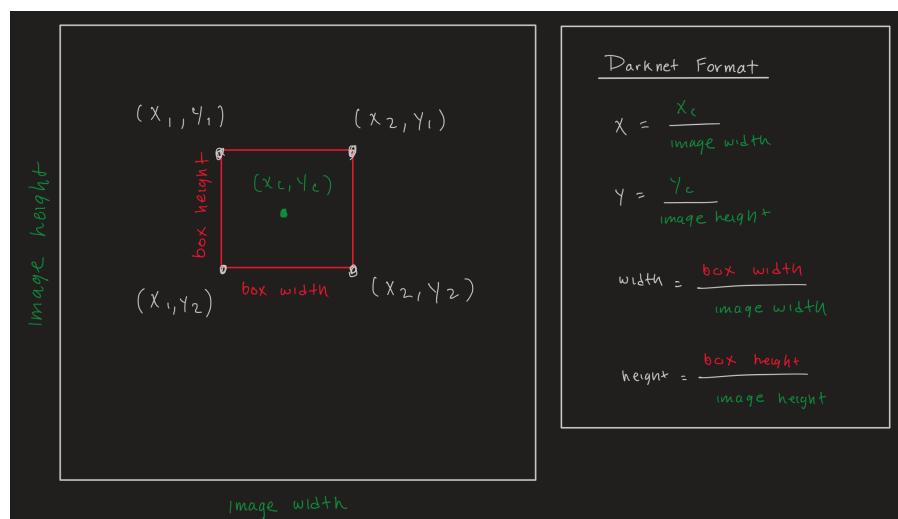
- Dataset được tổ chức lưu trữ trên roboflow. Sử dụng code để tải dataset về trong quá trình train và test.

Chương 4. TRAINING VÀ ĐÁNH GIÁ MODEL

4.1. Nội dung dataset:

4.1.1. YOLO:

- Đối với các model YOLO thì trong tập dataset sẽ gồm các file ảnh và các file *.txt ứng với mỗi tấm ảnh.
- Nội dung của file txt: mỗi object được biểu diễn bằng 1 dòng <object-class> <x> <y> <width> <height>
 - Trong đó <object-class> là số nguyên trong đoạn [0, 3]
 - <x> <y> <width> <height> là các số thực được chuẩn hóa có giá trị nằm trong đoạn [0, 1], biểu diễn bounding box của đối tượng.



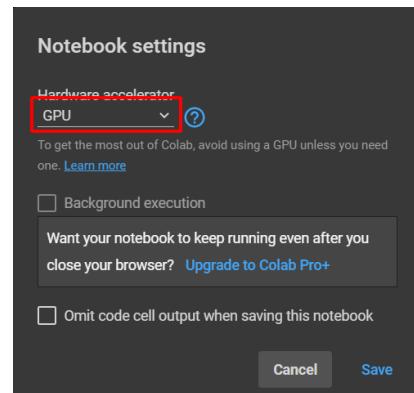
Hình 17. Cách tính các giá trị x, y, width, height.

4.1.2. Faster-RCNN:

- Đối với Faster-RCNN nhóm sử dụng roboflow để tự chuyển đổi từ định dạng YOLO darknet format sang COCO json format.

4.2. Cấu hình train và test:

Để train model nhóm sử dụng tài nguyên của Google Colab với thiết lập runtime type là GPU da



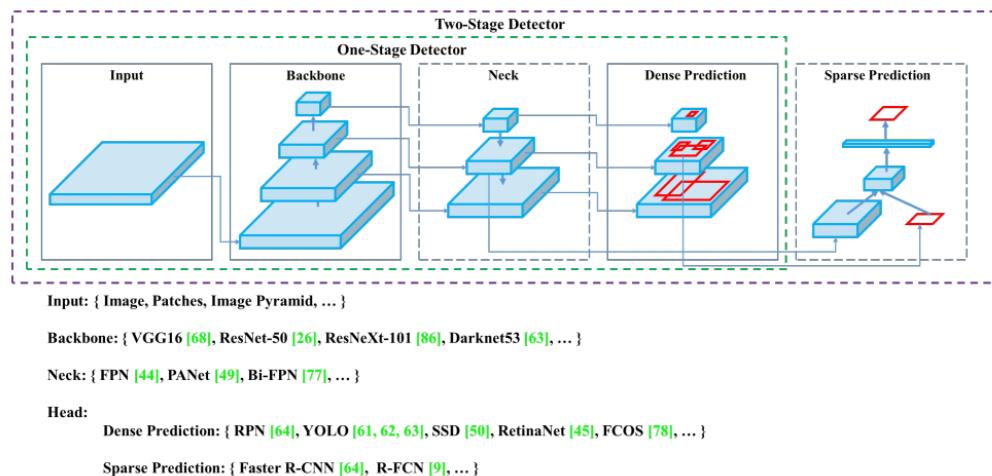
Hình 18. Bật GPU trên Google Colab.

```
Thu Jan 6 11:02:11 2022
+-----+
| NVIDIA-SMI 495.44      Driver Version: 460.32.03    CUDA Version: 11.2 |
| GPU  Name Persistence-M  Bus-Id Disp.A  Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap | Memory-Usage | GPU-Util  Compute M. |
|          |          |          |          |          |          |          | MIG M. |
+-----+
|   0  Tesla K80      Off  00000000:00:04.0 Off   0MiB / 11441MiB |  0%      Default |
| N/A   39C   P8    27W / 149W |           |          |          |          |          | N/A      |
+-----+
+
| Processes:
| GPU  GI CI      PID  Type  Process name          GPU Memory Usage
|   ID  ID
+-----+
| No running processes found
+-----+
```

Hình 19. Cấu hình dùng để train và test.

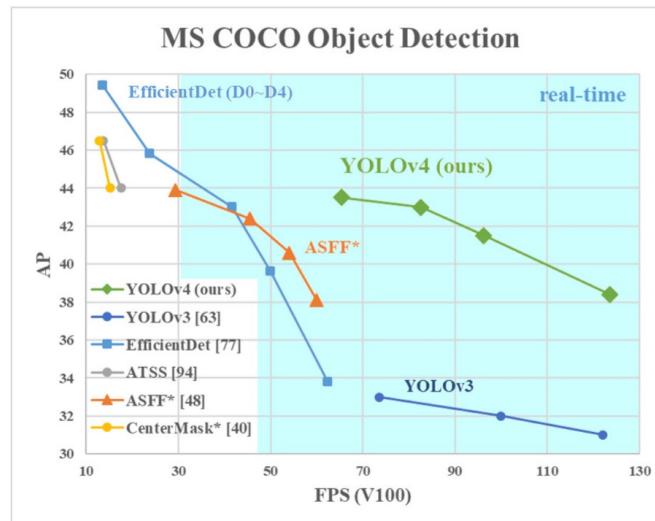
4.3. YOLOv4:

4.3.1. Sơ lược về YOLOv4



Hình 20. Cấu trúc mô hình YOLOv4.

- YOLOv4 được giới thiệu bởi Alexey Bochkovskiy, Chien-Yao Wang, and Hong-Yuan Mark Liao trong bài báo [YOLOv4: Optimal Speed and Accuracy of Object Detection](#) xuất bản ngày 23/4/2020.
- YOLOv4 là một loạt các cải tiến về tốc độ so với YOLOv3 và được cài đặt từ một bản fork của Darknet. Kiến trúc của YOLOv4 đã đưa bài toán object detection dễ tiếp cận hơn với những người không có tài nguyên tính toán mạnh.



Hình 21. So sánh YOLOv4 với các mô hình khác.

- Kết quả so sánh YOLOv4 với các mô hình khác ở thời điểm hiện tại. YOLOv4 chạy nhanh gấp đôi EfficientDet và tăng AP và FPS so với YOLOv3 lần lượt là 10% và 12%. Hình ảnh từ paper YOLOv4. Nhìn vào biểu đồ, ta dễ dàng thấy được sự hiệu quả của YOLOv4 so với các mạng tốt nhất hiện nay. Cụ thể hơn YOLOv4 đạt 43.5% AP trên tập dữ liệu MS COCO ở tốc độ 65 FPS, trên GPU Tesla V100.

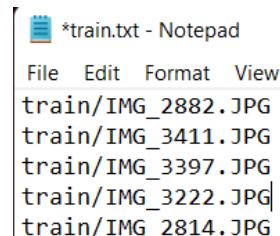
4.3.2 Thiết lập training

- Nhóm sử dụng darknet repository của tác giả để huấn luyện cho model và thiết lập các thông số trong file Makefile như sau

```
Makefile - Note
File Edit Format
GPU=1
CUDNN=1
CUDNN_HALF=1
OPENCV=1
```

Hình 22. Thiết lập các thông số Makefile để sử dụng GPU.

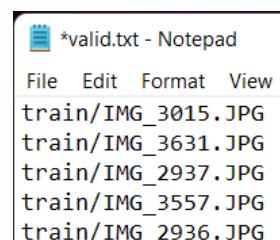
- Chỉnh sửa các thông số của model YOLOv4 trong file yolov4-custom.cfg theo hướng dẫn của tác giả:
 - batch: 32
 - subdivisions = 32
 - max_batches = 8000 (Bằng số class * 2000)
 - steps = 6400, 7200 (Bằng 0.8 * max_batches, 0.9 * max_batches)
 - width = 416, height = 416 (Kích thước của ảnh)
 - classes = 4 (Số class)
 - filters = 27 (Tính theo công thức filters = (classes + 5) * 3)
- Tạo file train.txt chứa đường dẫn tới các ảnh dùng để train (3138 ảnh)



```
*train.txt - Notepad
File Edit Format View
train/IMG_2882.JPG
train/IMG_3411.JPG
train/IMG_3397.JPG
train/IMG_3222.JPG
train/IMG_2814.JPG
```

Hình 23. File train.txt

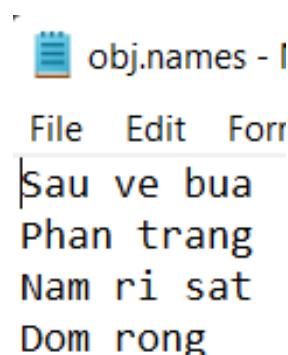
- Tạo file valid.txt chứa đường dẫn tới các ảnh dùng để đánh giá trong quá trình train (687 ảnh)



```
*valid.txt - Notepad
File Edit Format View
train/IMG_3015.JPG
train/IMG_3631.JPG
train/IMG_2937.JPG
train/IMG_3557.JPG
train/IMG_2936.JPG
```

Hình 24. File valid.txt

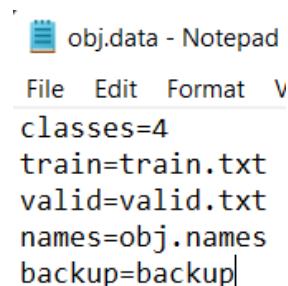
- Tạo file obj.names chứa tên của các class



```
obj.names - I
File Edit Forr
Sau ve bua
Phan trang
Nam ri sat
Dom rong
```

Hình 25. File obj.names

- Tạo file obj.data có nội dung như sau



```
obj.data - Notepad
File Edit Format V
classes=4
train=train.txt
valid=valid.txt
names=obj.names
backup=backup|
```

Hình 26. File obj.data

Trong đó:

classes: là số lượng class

train: đường dẫn tới file train.txt

valid: đường dẫn tới file valid.txt

names: đường dẫn tới file obj.names

backup: đường dẫn tới folder backup chứa các trọng số được lưu lại trong quá trình train

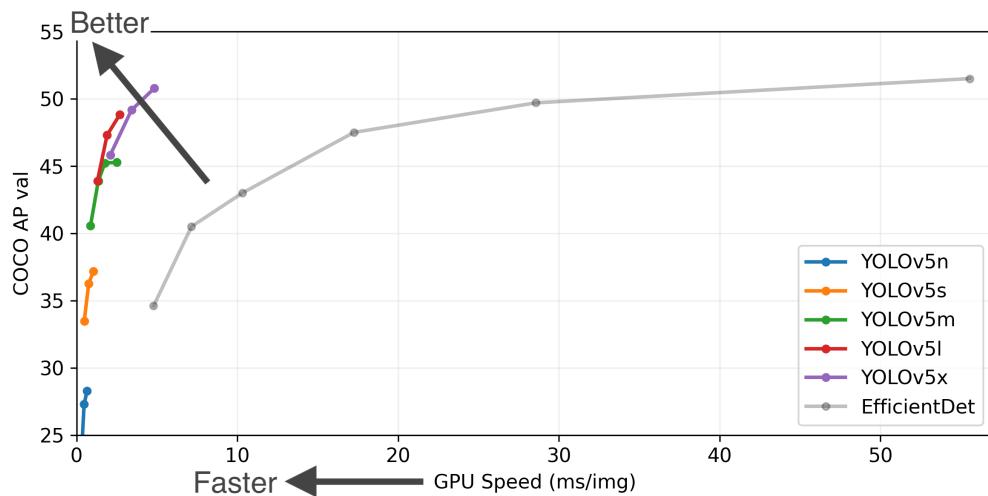
4.3.3. Train model

- Tải file trọng số yolov4.conv.137.weights và tiến hành train trên file trọng số này
- Trong quá trình train model các file trọng số được lưu lại trong đó có 2 file quan trọng là:
 - yolov4-custom_last.weights (Trọng số của iteration mới nhất)
 - yolov4-custom_best.weights (Trọng số tốt nhất)
- Quá trình training khá lâu vượt qua thời gian cho phép của Google Colab nên ở những lần train tiếp theo nhóm tiến hành train tiếp trên file trọng số mới nhất
- Thời gian train model: khoảng 28 tiếng
- Thời gian test trên 687 ảnh: 53 giây

4.4. YOLOv5:

4.4.1. Sơ lược về YOLOv5

- Không lâu sau khi YOLOv4 được phát hành chính thức thì 1 phiên bản khác của YOLO xuất hiện là YOLOv5 sử dụng frame work Pytorch. YOLOv5 được giới thiệu bởi Glenn Jocher vào ngày 18/5/2020, YOLOv5 có mã nguồn mở ở [Github](#). Với những số liệu của tác giả cung cấp thì mô hình này khá triển vọng. Tuy nhiên YOLOv5 hiện vẫn chưa có paper chính thức.



Hình 27. Perfomance của các phiên bản YOLOv5 trên tập COCO

Model	size (pixels)	mAP _{0.5:0.95}	mAP _{0.5}	Speed CPU b1 (ms)	Speed V100 b1 (ms)	Speed V100 b32 (ms)	params (M)	FLOPs @640 (B)
YOLOv5n	640	28.4	46.0	45	6.3	0.6	1.9	4.5
YOLOv5s	640	37.2	56.0	98	6.4	0.9	7.2	16.5
YOLOv5m	640	45.2	63.9	224	8.2	1.7	21.2	49.0
YOLOv5l	640	48.8	67.2	430	10.1	2.7	46.5	109.1
YOLOv5x	640	50.7	68.9	766	12.1	4.8	86.7	205.7

Hình 28. Kết quả đánh giá trên tập COCO

- Từ những kết quả mà tác giả đưa ra có thể thấy YOLOv5 có thể đạt tới 68.9% mAP@0.5 trên tập COCO.

4.4.2. Thiết lập training

- Tạo file data.yaml có nội dung như sau

```
data.yaml - Notepad
File Edit Format View Help
names:
- '0'
- '1'
- '2'
- '3'
nc: 4
train: yolov5-train-1/train/images
val: yolov5-test-1/valid/images
```

Hình 29. File data.yaml

Trong đó:

names: lần lượt là tên của các label được đặt trong dấu ngoặc kép

nc: số lượng class

train: đường dẫn tới các file ảnh train

valid: đường dẫn tới các file ảnh dùng để valid trong quá trình train

- Thiết lập training

- batch: 32
- img size: 416
- epoch: 500

4.4.3. Train model

- Tải file trọng số của model YOLOv5s và tiến hành train trên file trọng số này.
- Trong quá trình train model các file trọng số được lưu lại trong đó có 2 file quan trọng là:
 - last.pt (Trọng số của epoch mới nhất)
 - best.pt (Trọng số tốt nhất)
- Quá trình training khá lâu vượt qua thời gian cho phép của Google Colab nên ở những lần train tiếp theo nhóm tiến hành train tiếp trên file trọng số mới nhất
- Vào epoch 450 thì xuất hiện thông báo dừng train model như sau:

```
Epoch 450/499  gpu_mem      box      obj      cls      labels    img_size
 3.49G  0.009875  0.005941  0.0006422      7      416: 100% 99/99 [01:26<00:00,  1.15it/s]
          Class   Images   Labels      P      R    mAP@.5  mAP@.5:.95: 100% 11/11 [00:07<00:00,  1.50it/s]
            all     687     728     0.995     0.998     0.995     0.956
Stopping training early as no improvement observed in last 100 epochs. Best results observed at epoch 350, best model saved as best.pt.
```

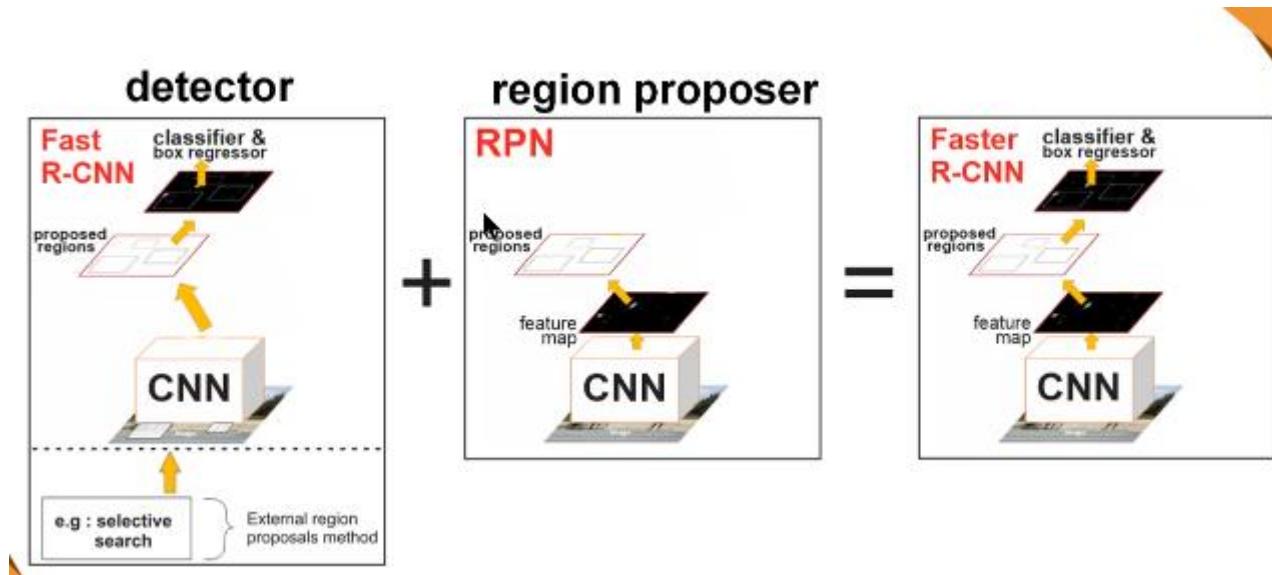
Hình 30. Early stopping YOLOv5

- Do trong 100 epoch gần nhất thì model không còn tốt lên được nữa (model hội tụ) nên tự động dừng train.
- Thời gian train model: khoảng 8 tiếng
- Thời gian test trên 687 ảnh: 32 giây

4.5. Faster-RCNN:

4.5.1. Sơ lược về Faster RCNN

- Faster-RCNN được giới thiệu bởi Shaoqing Ren, Kaiming He, Ross Girshick, Jian Sun.Trong bài báo [Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks](#)
- Faster-RCNN là một phương pháp phát hiện đối tượng sử dụng deep learning.Faster RCNN là một sự cải tiến dựa trên hai phương pháp trước đó là RCNN và Fast RCNN. Faster RCNN là sự kết hợp giữa Fast-RCNN với một mạng mới có tên gọi là region proposal network(rpn).
- Region Proposal Netwok là một mạng Convolutional Neural Network(CNN) có chức năng để tìm ra các vùng có khả năng chứa đối tượng thường được gọi là region proposal
- Fast-RCNN là một mạng CNN dùng để trích xuất các features từ các region proposal và trả ra các bounding box cùng với label cho từng cái bounding box đó.



Hình 31. Cấu trúc Faster RCNN

4.5.2. Thiếp lập training

- Nhóm sử dụng detectron2 một thư viện của Facebook AI Research để tiến hành huấn luyện cho Faster RCNN. Nhóm đã sử dụng file pretrained weights X-101-32x8d.pkl để tiếp tục train cho model của mình.

Faster R-CNN:

Name	lr sched	train time (s/iter)	inference time (s/itm)	train mem (GB)	box AP	model id	download
R50-C4	1x	0.551	0.102	4.8	35.7	137257644	model metrics
R50-DC5	1x	0.380	0.068	5.0	37.3	137847829	model metrics
R50-FPN	1x	0.210	0.038	3.0	37.9	137257794	model metrics
R50-C4	3x	0.543	0.104	4.8	38.4	137849393	model metrics
R50-DC5	3x	0.378	0.070	5.0	39.0	137849425	model metrics
R50-FPN	3x	0.209	0.038	3.0	40.2	137849458	model metrics
R101-C4	3x	0.619	0.139	5.9	41.1	138204752	model metrics
R101-DC5	3x	0.452	0.086	6.1	40.6	138204841	model metrics
R101-FPN	3x	0.286	0.051	4.1	42.0	137851257	model metrics
X101-FPN	3x	0.638	0.098	6.7	43.0	139173657	model metrics

Hình 32. File pretrained được sử dụng

- Thiết lập thông số:
 - BATCH_SIZE_PER_IMAGE = 64
 - CLASSES : 5 (Tổng số class + 1)
 - MAX_ITER : 15000
 - STEP_SIZE : 6000,10000
- Nhóm chỉ chỉnh sửa một vài thông số để tiến hành việc training.

4.5.3. Train model

- Thời gian train của Faster RCNN: 12 tiếng
- Thời gian test trên 687 ảnh: 175 giây

4.6 Đánh giá model

4.6.1. Metric đánh giá

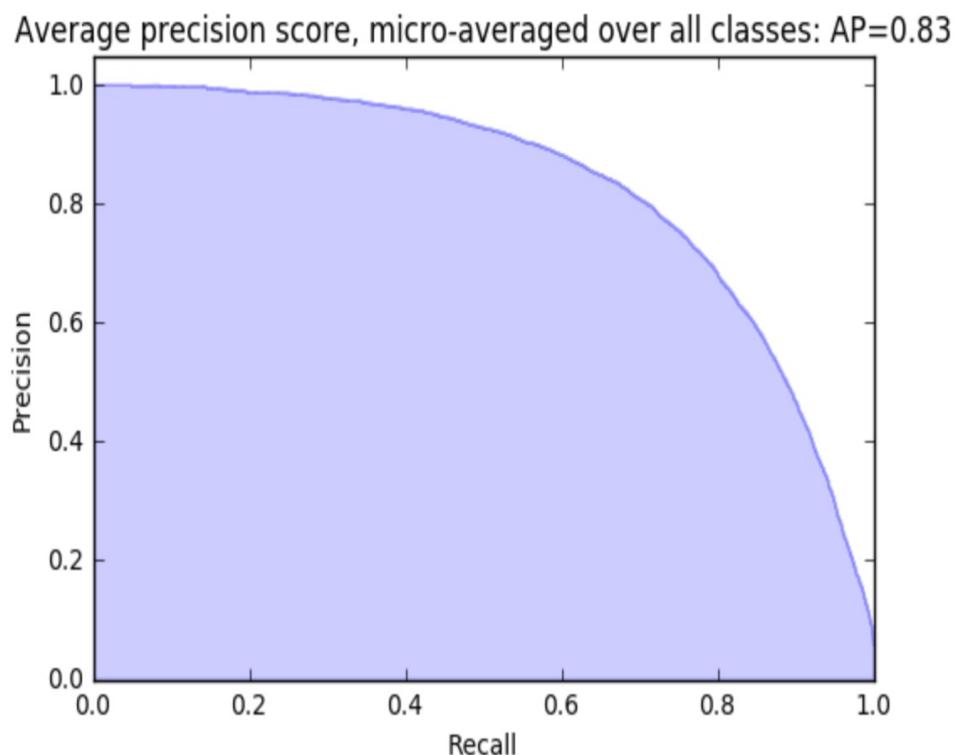
- Để đánh giá model thì nhóm sử dụng mean average precision để đánh giá model. Trước tiên để hiểu được mean average là gì thì trước tiên chúng tôi sẽ giới thiệu một số khái niệm cơ bản.
- IOU là tỷ lệ giữa phần giao của bounding box dự đoán với ground truth(vùng đối tượng thật mà chúng tôi label) và phần hợp của chúng.

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

Hình 33. Độ đo IOU

- Giá trị IOU trong khoảng (0,1). Dựa vào đó có thể xác định được wrong detection hay correct detection. Dựa vào ngưỡng để xác định. Nếu IOU lớn hơn hoặc bằng ngưỡng thì đó là một correct detection còn lại thì là wrong detection.
- Dựa vào những khái niệm trên để định nghĩa True/false positive/negative.
 - True Positive (TP): IoU lớn hơn hoặc bằng ngưỡng, là một correct detection

- False Positive (FP): IoU bé hơn ngưỡng, là một wrong detection
- False Negative (FN): trường hợp mà ground truth không có predicted bounding box
- Nếu có nhiều predicted bounding box xếp chồng lên nhau trong cùng một ground truth thì ta sẽ chọn predicted bounding box nào có IOU lớn hơn ngưỡng cao nhất là True Positive, còn lại là False Positive.



Hình 34. AP

- AP là diện tích màu xanh nằm dưới đường cong.
- Mỗi lớp bài toán sẽ có một giá trị AP, Mean Average Precision là trung bình AP cho tất cả các lớp.
- →**Lý do chọn mean average precision là metric để đánh giá mô hình :**
 - Mỗi quan hệ giữa precision – recall giúp mAP đánh giá được về độ chính xác của mô hình.
 - Precision – Recall thay đổi khi ngưỡng IoU thay đổi. Do đó, tại một giá trị IoU xác định, ta có thể do/đánh giá được mô hình một cách tốt nhất.

4.6.2. Kết quả đánh giá

- Thiết lập các ngưỡng để đánh giá
 - confidence thresh: 0.25
 - iou thresh: 0.5
- YOLOv4

```

detections_count = 1487, unique_truth_count = 728
class_id = 0, name = Sau ve bua, ap = 100.00%
class_id = 1, name = Benh phan trang, ap = 98.44%
class_id = 2, name = Nam ri sat, ap = 98.79%      (TP = 250)
class_id = 3, name = Dom rong, ap = 98.63%        (TP = 187)

for conf_thresh = 0.25, precision = 0.90, recall = 0.98,
for conf_thresh = 0.25, TP = 714, FP = 80, FN = 14, average AP = 0.989642

IoU threshold = 50 %, used Area-Under-Curve for each unique class
mean average precision (mAP@0.50) = 0.989642, or 98.96 %
Total Detection Time: 53 Seconds

```

Hình 35. Kết quả đánh giá model YOLOv4

Class	AP@0.5
0	1.000
1	0.984
2	0.988
3	0.986

- YOLOv5

```

Model Summary: 213 layers, 7020913 parameters, 0 gradients, 15.8 GFLOPs
val: Scanning 'yolov5-test-1/test/labels.cache' images and labels... 687 found, 0 missing, 0 empty, 0 corrupted: 100% 687/687
      Class   Images   Labels      P      R    mAP@.5:1.95: 100% 22/22 [00:21<00:00,  1.01it/s]
      all     687     728   0.988   0.99   0.993   0.87
      0     687     217       1   0.998   0.995   0.877
      1     687      64   0.984   0.984   0.99   0.848
      2     687     258   0.981   0.992   0.992   0.85
      3     687     189   0.989   0.984   0.994   0.903
Speed: 0.4ms pre-process, 6.4ms inference, 1.8ms NMS per image at shape (32, 3, 640, 640)

```

Hình 36. Kết quả đánh giá model YOLOv5s

Class	AP@0.5
0	0.995
1	0.99
2	0.992
3	0.994

- Faster RCNN

```

[01/15 14:50:08 d2.evaluation.coco_evaluation]: Evaluation results for bbox:
|   AP   |   AP50  |   AP75  |   APs  |   APm  |   APl  |
|:-----|:-----|:-----|:-----|:-----|:-----|
| 92.200 | 99.557 | 99.557 |  nan  |  nan  | 92.200 |
[01/15 14:50:08 d2.evaluation.coco_evaluation]: Note that some metrics cannot be computed.
[01/15 14:50:08 d2.evaluation.coco_evaluation]: Per-category bbox AP:
| category | AP      | category | AP      | category | AP      |
|:-----|:-----|:-----|:-----|:-----|:-----|
| uit     | nan    | 0        | 98.683 | 1        | 99.748 |
| 2       | 99.954 | 3        | 99.840 |          |          |

```

Hình 37. Kết quả đánh giá model Faster RCNN (AP per-category là AP0.5)

Class	AP@0.5
0	0.987

Class AP@0.5

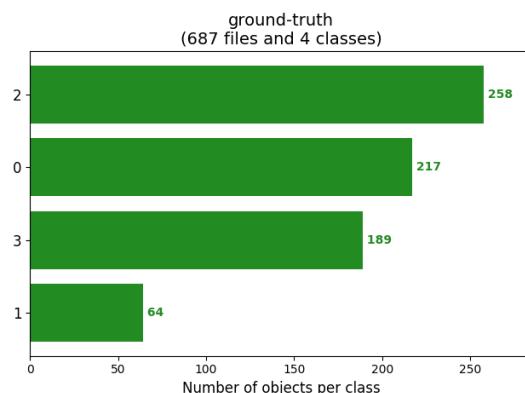
1	0.997
2	0.999
3	0.998

- Tổng kết đánh giá

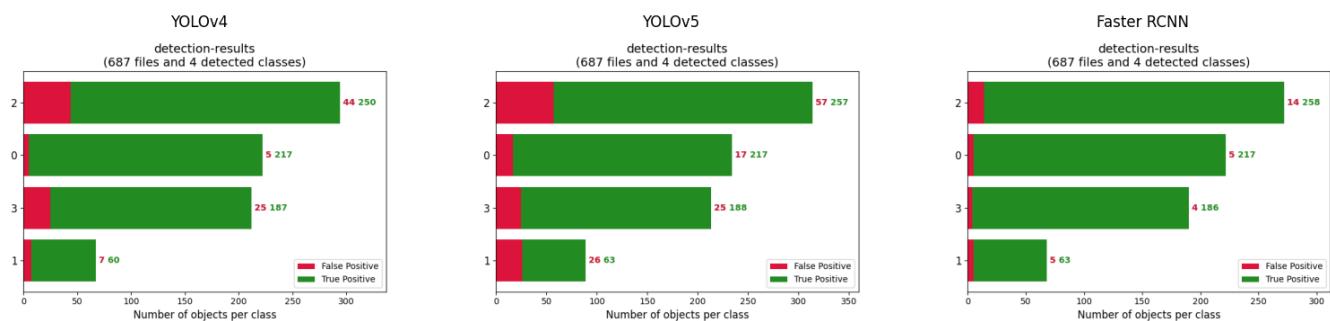
Model\Class	0	1	2	3
YOLOv4	<u>1.000</u>	0.984	0.988	0.986
YOLOv5	0.995	0.99	0.992	0.994
Faster RCNN	0.987	<u>0.997</u>	<u>0.999</u>	<u>0.998</u>

→ Khi đánh giá bằng điểm AP@0.5, đối với class 0 model YOLOv4 cho kết quả cao nhất. Đối với 3 class còn lại, Faster RCNN đều cho kết quả tốt hơn

- Số lượng sai sót của mỗi class:



Hình 38. Ground truth



Hình 39. Detection result

→ Qua các biểu đồ thông kê trên ta có thể thấy rằng số lượng các True Positive được model tìm thấy khá cao và gần bằng so với ground truth. Ở model YOLOv4 và model YOLOv5 có nhiều các False Positive được tìm ra.

Model Precision Recall mAP@0.5

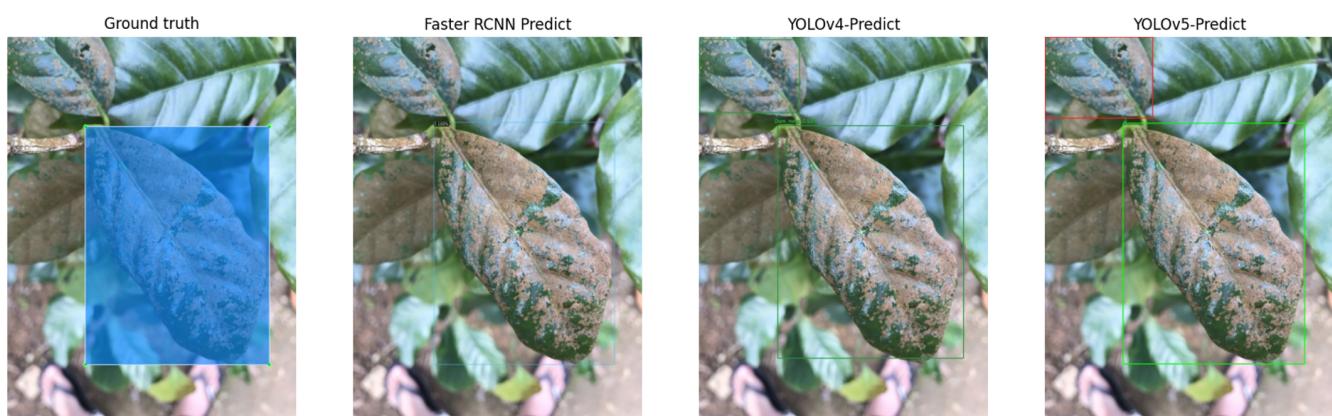
Model	Precision	Recall	mAP@0.5
YOLOv4	0.900	0.98	0.989
YOLOv5s	0.988	0.99	0.993
Faster-RCNN	<u>0.996</u>	<u>0.997</u>	<u>0.996</u>

→ Khi đánh giá bằng mAP@0.5 cả 3 model đều cho kết quả rất tốt. Faster RCNN cho kết quả tốt nhất.

Model	Thời gian test 687 ảnh (giây)
YOLOv4	53
YOLOv5s	32
Faster-RCNN	175

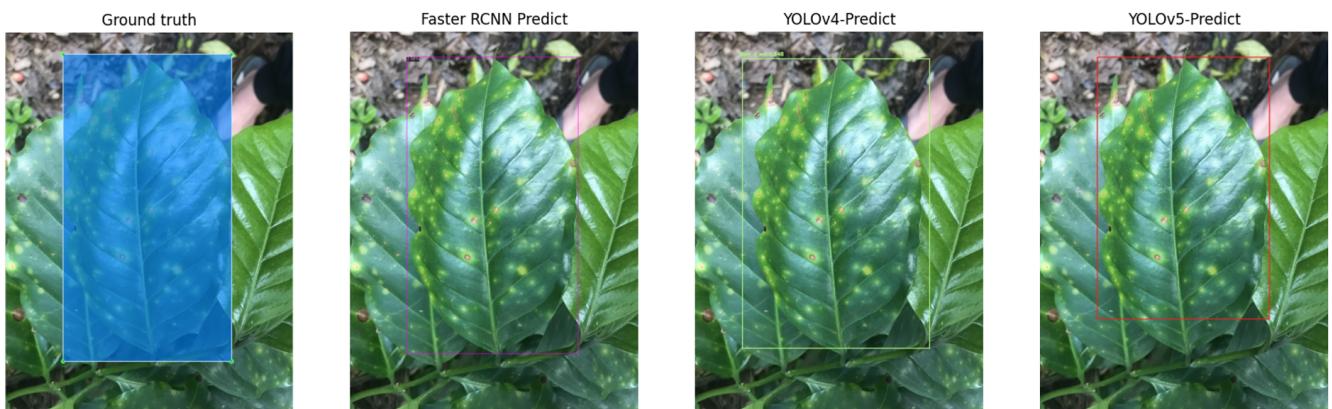
→ Khi thử nghiệm trên cùng một cấu hình, mặc dù Faster RCNN cho kết quả mAP@0.5 tốt nhất nhưng cũng tốn thời gian nhiều nhất so với 2 model còn lại.

- Một số hình ảnh test



Hình 40. Kết quả test

Model YOL0v4 và YOL0v5 detect sai 1 phần lá bị bệnh đốm rong ở góc trên bên trái



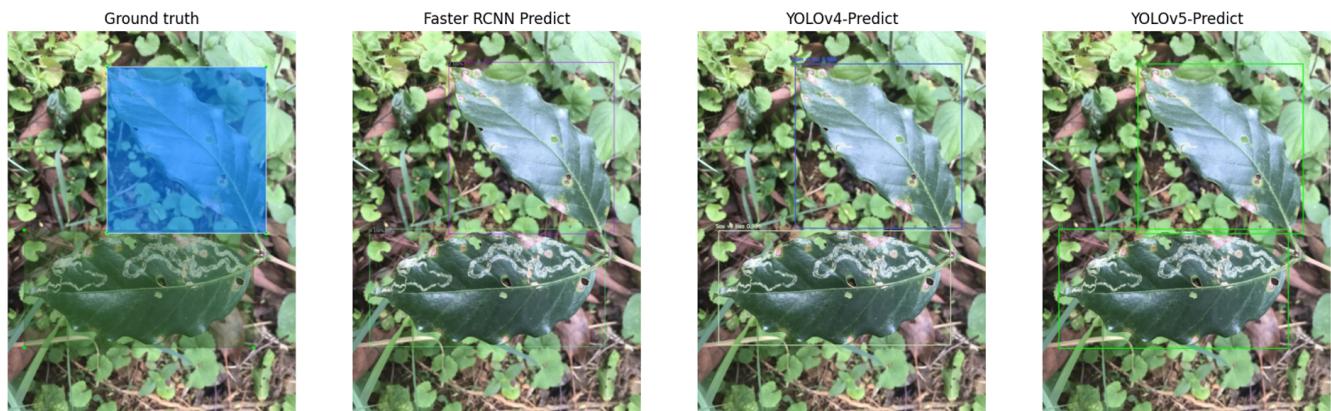
Hình 41. Kết quả test

YOLOv5 detect sai loại bệnh (Ground truth là nấm rỉ sắt - Predict đốm rong)



Hình 42. Kết quả test

YOLOv5 detect 1 lá bình thường ở góc trên thành bệnh sâu vẽ bùa



Hình 43. Kết quả test

Cả ba model đều cho kết quả chính xác khi detect được 2 lá bị bệnh.



Hình 44. Kết quả test

YOLOv4 cho kết quả chính xác, Faster RCNN và YOLOv5 detect sai 1 lá bình thường ở bên trái thành bệnh sâu vẽ bùa

- Nhìn chung kết quả thử nghiệm đều khá tốt.
 - Một số lá bình thường bị detect nhầm thành bệnh sâu vẽ bùa và nấm rỉ sắt do 1 số ảnh trong tập train bệnh còn nhẹ và khá giống với lá bình thường
 - Một số lá bị nấm rỉ sắt nhìn khá giống với bệnh đốm rong làm cho model bị nhầm lẫn.
 - YOLOv4 và YOLOv5 đều có những trường hợp detect ra 1 phần lá bị bệnh (đối tượng không đủ từ cuống đến chóp lá). Trường hợp này xảy ra nhiều hơn đối với model YOLOv5

Chương 5. Ứng dụng và hướng phát triển:

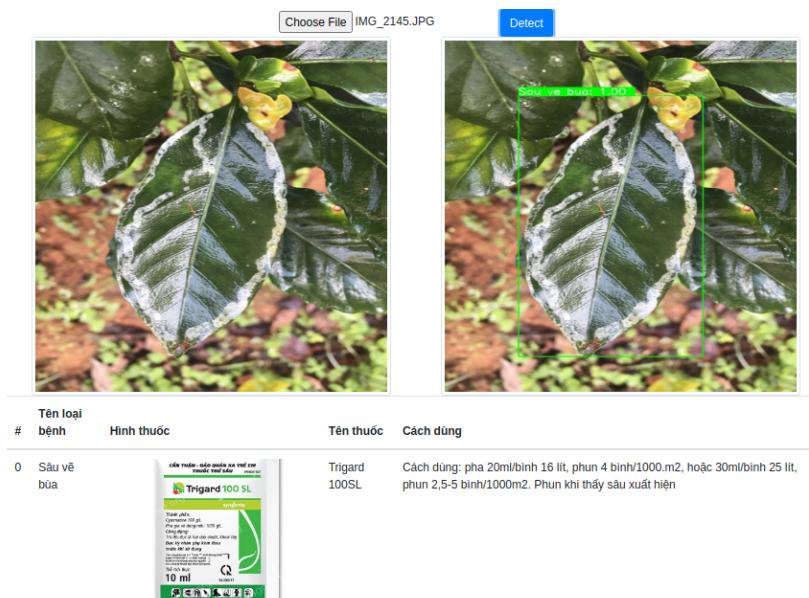
Ứng dụng:

- Ứng dụng hướng tới người sử dụng chính là người trồng cà phê, giúp người trồng có thể phát hiện được các loại bệnh xuất hiện trên lá từ đó có thể xử lý đúng cách và hiệu quả.
- Như đã đề cập trước đó, việc giúp người trồng cà phê phát hiện được bệnh xuất hiện trên lá sẽ góp phần nâng cao được chất lượng sản phẩm cà phê, đáp ứng được các tiêu chuẩn về hàng xuất khẩu từ đó nguồn thu nhập của người dân sẽ được tăng lên. Việc sử các mô hình máy học sẽ thúc đẩy quá trình ứng dụng khoa học kỹ thuật vào trong nông nghiệp.

Hướng phát triển:

- Thu thập thêm nhiều dữ liệu về các loại bệnh nhầm giúp ứng dụng phát hiện được nhiều loại bệnh và chính xác hơn.
- Có thể hướng tới việc phát hiện các loại bệnh trên nhiều loại lá cây nông nghiệp khác nhau dựa trên các đặc điểm giống nhau của các loại bệnh khi xuất hiện trên lá.

Chương 6. Demo:



Hình 45. Demo

- Source code : <https://github.com/danhhuynh25029/CS114.M11/tree/main/finalProject/app>