



เอกสารสำหรับผู้พัฒนาระบบ Yolov5

จัดทำโดย

กนกพร พงศศิริก 640510604

ธนโชติ วัฒนฐสกุล 640510657

ธนธรณ์ บุญเชิด 640510658

สุภคม ผิวอ่อน 640510687

เสนอ

ผู้ช่วยศาสตราจารย์ ดร.วัฒนา จินดาหลวง

รายงานฉบับนี้เป็นส่วนหนึ่งของรายวิชา

วิชา อัลกอริทึมและการเขียนโปรแกรมในการหาค่าที่เหมาะสมที่สุดเชิงการจัด(204454)

ภาคเรียนที่ 2 ปีการศึกษา 2566

สารบัญ

เรื่อง	หน้า
บทที่ 1 บทนำ	1
บทที่ 2 การติดตั้งและการใช้งาน	2
-ส่วนที่ 1. การเตรียมข้อมูล ข้อมูลจะถูกเตรียมผ่าน Robo flow	2
-ส่วนที่ 2. การเตรียมสร้างโมเดลเพื่อนำมาใช้งาน	10
บทที่ 3 คำสั่งในการปรับปรุงคุณภาพโมเดล	14
บทที่ 4 การใช้โมเดลที่สร้างขึ้นจากสมาชิกในกลุ่ม	17

บทที่ 1 บทนำ

ความเป็นมาของ YOLOv5

YOLOv5 เป็นโมเดลการตรวจจับวัตถุที่ใช้เทคโนโลยี Deep Learning โดยใช้ Convolutional Neural Networks (CNNs) ซึ่งเป็นโครงข่ายประสาทเทียมที่ถูกออกแบบมาเพื่อการประมวลผลข้อมูลที่มีลักษณะเป็นรูปภาพ (Computer vision) ซึ่งถูกพัฒนาขึ้นโดยนักวิจัยชาวอเมริกันชื่อ Andrew S. Yang ผ่านทาง GitHub ภายใต้ชื่อบัญชี "ultralytics" เมื่อปี 2020 โดยเป็นการพัฒนาต่อยอดจาก YOLO (You Only Look Once) ซึ่งเป็นโมเดลการตรวจจับวัตถุที่มีชื่อเสียงทางด้าน Computer Vision

โมเดลที่มีการใช้ใน YOLOv5

1. Convolutional Neural Network (CNN) : หลักการของ CNNs มีการนำเสนอแนวคิดของคอนโวลูชัน (convolution) ที่ใช้ในการสกัดลักษณะ (features) ออกจากภาพ โดยการใช้ตัวกรอง (filter) หรือเคอร์เนล (kernel) ที่มีขนาดเล็กและเลื่อนที่ละหนึ่งพิกเซล (pixel) ที่มีขนาดเล็ก (stride) ไปทั่วภาพ เพื่อสร้าง feature maps ซึ่งจะมีการระบุลักษณะเฉพาะของภาพที่สนใจ เช่น ขอบของวัตถุ รูปร่าง และลักษณะอื่น ๆ ที่สำคัญ[Chat gpt] และ CNNs ยังมีขั้นตอนของการทำ Pooling เพื่อเก็บข้อมูลที่มีลักษณะพิเศษ

2. CSPDarknet53: เป็นโมเดล Convolutional Neural Network (CNN) ที่ใช้ใน YOLOv5 สำหรับการสกัดลักษณะของภาพ โดยมีความลึกและความซับซ้อนในการทำงาน

3. PANet: เป็นโมเดลที่ใช้ในชั้นหลังๆ ของโมเดล YOLOv5 เพื่อช่วยในการปรับปรุงการจำแนกประเภทและตำแหน่งของวัตถุ

4. SAM (Spatial Attention Module): เป็นโมเดลที่ใช้ในชั้นหลังๆ ของโมเดล YOLOv5 เพื่อช่วยในการเน้นความสำคัญของพื้นที่บนภาพ

โดยโมเดลทั้งหมดมีการทำงานร่วมกันเพิ่มประสิทธิภาพและความแม่นยำในการตรวจจับวัตถุ

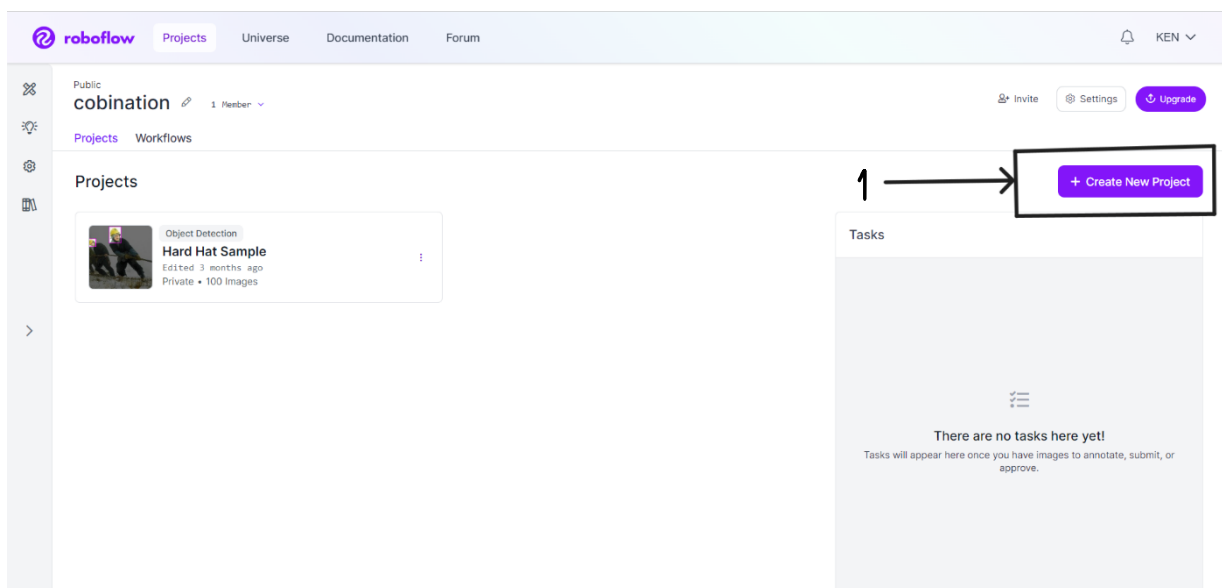
ประสิทธิภาพของ model ขึ้นกับข้อมูลที่มีการใส่เข้าไป

บทที่ 2 การติดตั้งและการใช้งาน

การติดตั้งและการใช้งาน ถูกแบ่งออกเป็น 2 ส่วน คือ ส่วนที่ 1.การเตรียมข้อมูล 2. การใช้งานใช้งาน YOLOv5

ส่วนที่ 1.การเตรียมข้อมูล ข้อมูลจะถูกเตรียมผ่าน Robo flow
ขั้นตอนการเตรียมข้อมูล

1. เมื่อผู้ใช้ทำการ login roboflow จะพบหน้าต่างดังนี้ หลังจากนั้นผู้ใช้กดปุ่ม Create New Project เพื่อทำการสร้าง dataset



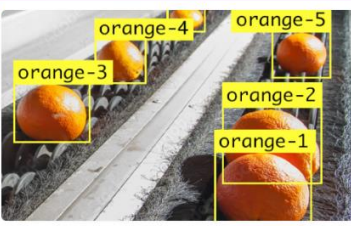
2. ทำการกรอกชื่อโปรเจกต์ใน ช่อง Project Name และทำการเลือก Project Type เป็น Object Detection หลังจากนั้นทำการระบุกลุ่มของวัตถุที่ต้องการจะตรวจจับในรูปแบบ เช่น คน ส้ม
หมายเหตุ: รูปภาพตรง Project Type แสดงถึง ประเภทโปรเจกต์ที่ต้องการสร้าง

Let's create your project.

cobination > [New Public Project](#)


Project Name Persons License [CC BY 4.0](#)

Project Type



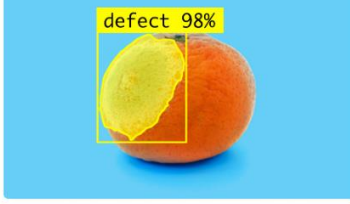
Object Detection
Identify objects and their positions with bounding boxes.

Best For
Counting [Tracking](#)



Classification
Assign labels to the entire image.

Best For
[Filtering](#) [Content Moderation](#)



Instance Segmentation
Detect multiple objects and their actual shape.

Best For
[Measurements](#) [Odd Shapes](#)

[Show More](#)

Annotation Group Person

3. ผู้ใช้ทำเลือกปุ่ม Upload data หลังจากนั้นทำการกดปุ่มเลือก Select Folder เพื่อทำการอัปโหลด Folder ของ dataset ที่เตรียมไว้

roboflow Projects Universe Documentation Forum KEN

COBINATION

Persons
Object Detection


Data

- Classes
- Upload Data**
- Assign Images
- Annotate
- Dataset
- Health Check
- Generate
- Versions
- Models

[Upgrade](#)

Upload [Want to change the classes on your annotated images?](#)

Batch Name: Uploaded on 02/24/24 at 12:52 am Tags:


Drag and drop images and annotations to upload them.

OR

[Select Files](#) [Select Folder](#)

Need images to get started? We've got you covered.

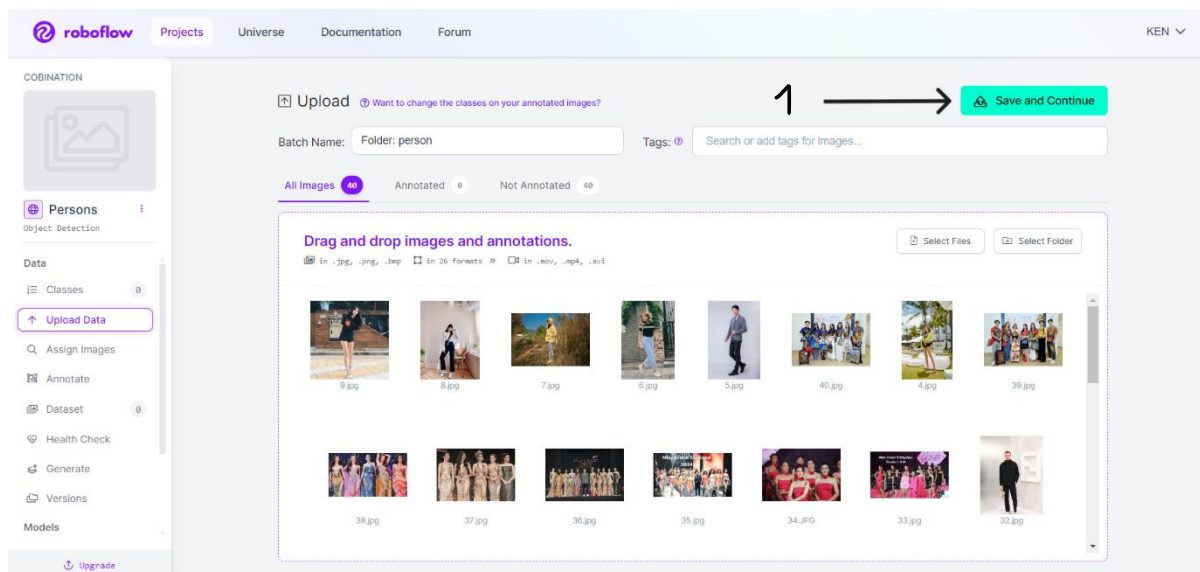
[Import YouTube Video:](#)

[Find a Universe Dataset](#)
Browse over 100k free datasets for images and build a model in minutes.

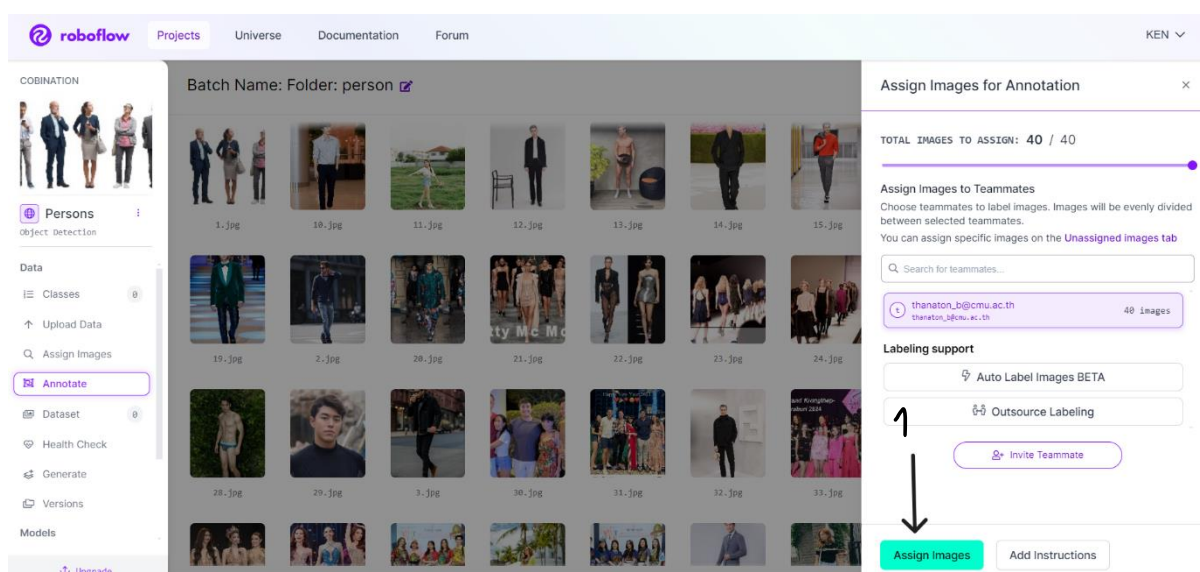
[Integrate Our API](#)
Collect real world images directly from your existing application.

[Upload images directly from cloud storage](#)

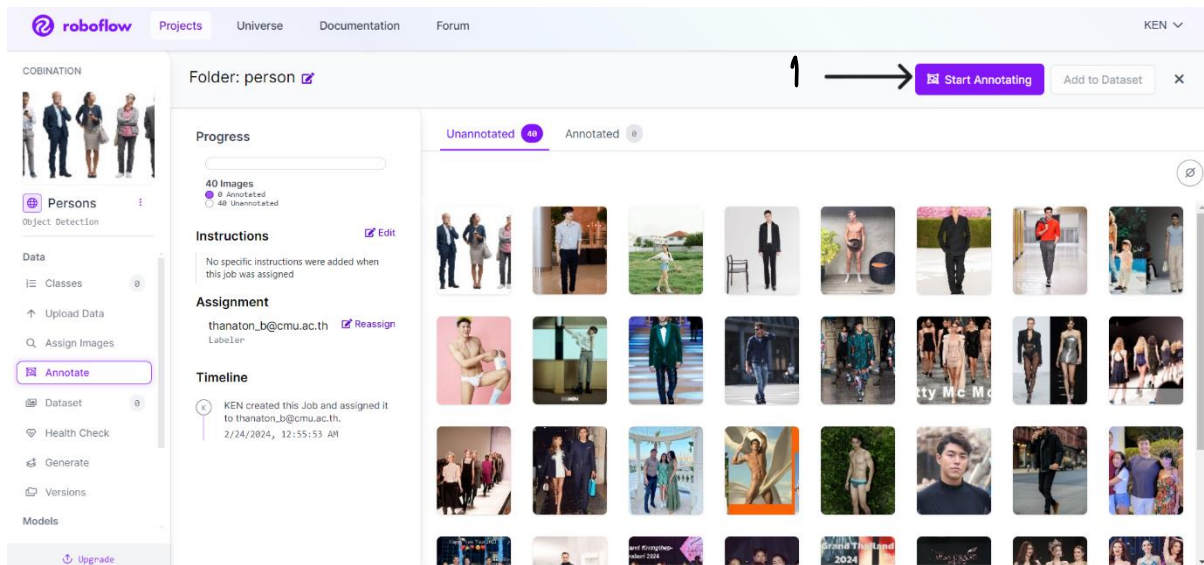
4. เมื่อผู้ใช้ทำการอัปโหลด dataset สำเร็จให้กดปุ่ม Save and Continue เพื่อไปยังขั้นตอนถัดไป



5. ทำการกดปุ่ม Assign images เพื่อไปยังขั้นตอนถัดไป

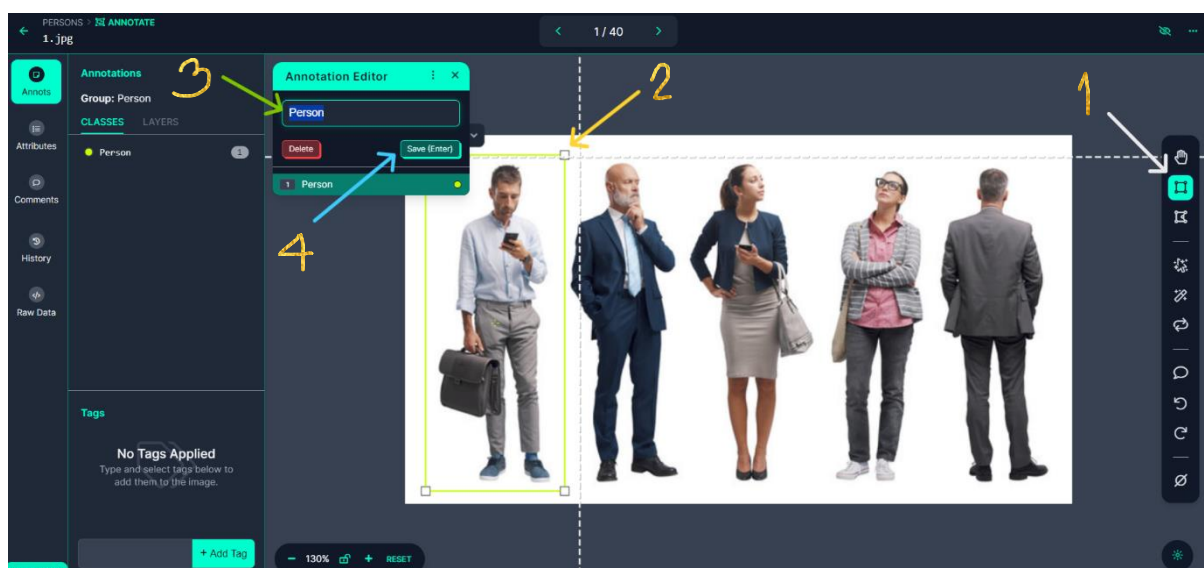


6. ทำ การกดปุ่ม Start Annotating เพื่อไปยังขั้นตอนถัดไป

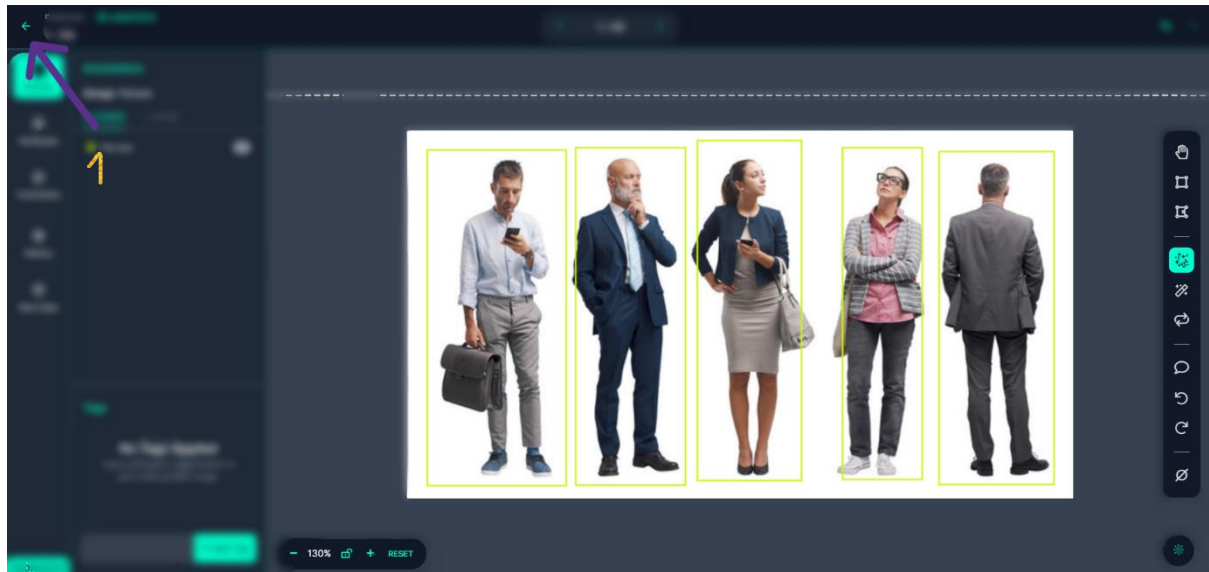


7. ผู้ใช้ทำการใช้เครื่องมือทางด้านขวาที่ลูกศรสีขาวชี้ (Bounding Boxes) เพื่อทำการแบ่งคลาส โดยทำการลากแนวทแยงของวัตถุนั้น(ลูกศรสีเหลือง) หลังจากนั้นจะมีกล่อง Annotation Editor ขึ้นมาเพื่อให้ผู้ใช้ทำการแก้ไขชื่อคลาส และ บันทึกคลาส (ลูกศรสีแดงและสีเหลือง)

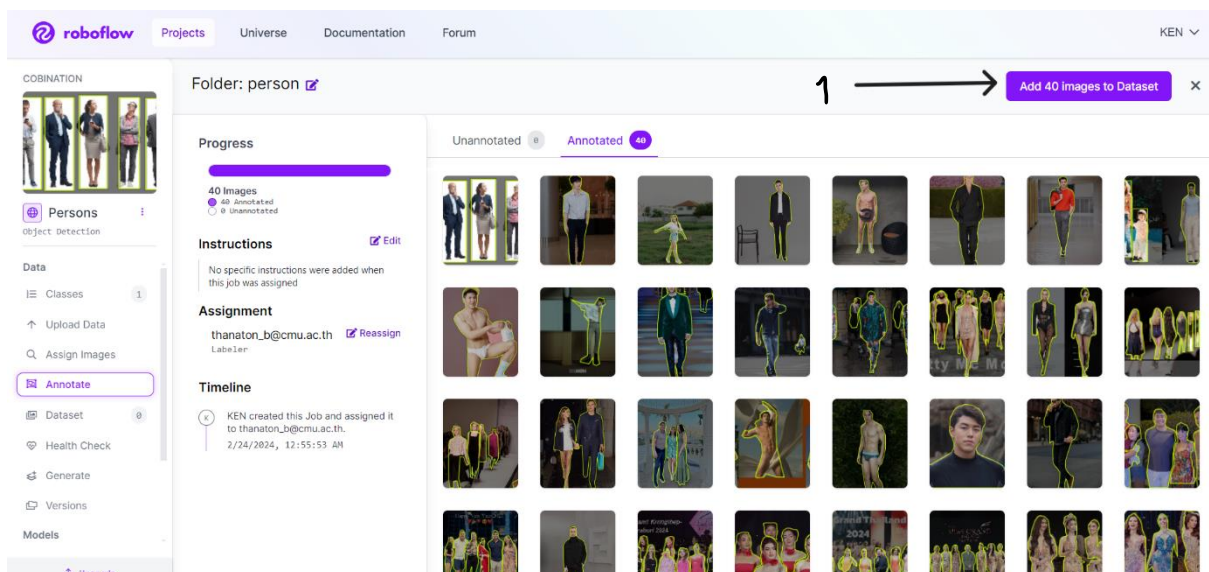
หมายเหตุ: เพื่อเพิ่มประสิทธิภาพที่ดีของโมเดลผู้ใช้ควรเลือกใช้ Bounding Boxes ลากทีละคน เพื่อกำหนดคลาส และ ผู้ใช้ต้องเลือกคลาสให้ถูกต้อง



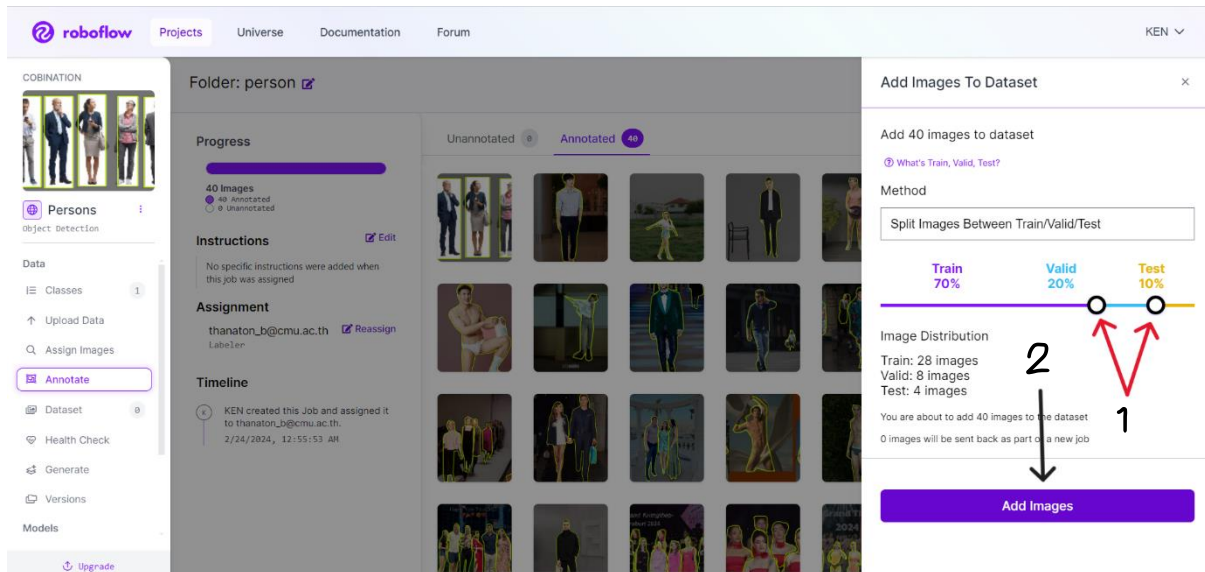
8. เมื่อทำครบทุกรูปให้กดปุ่มย้อนกลับ (ลูกศรสีม่วง)



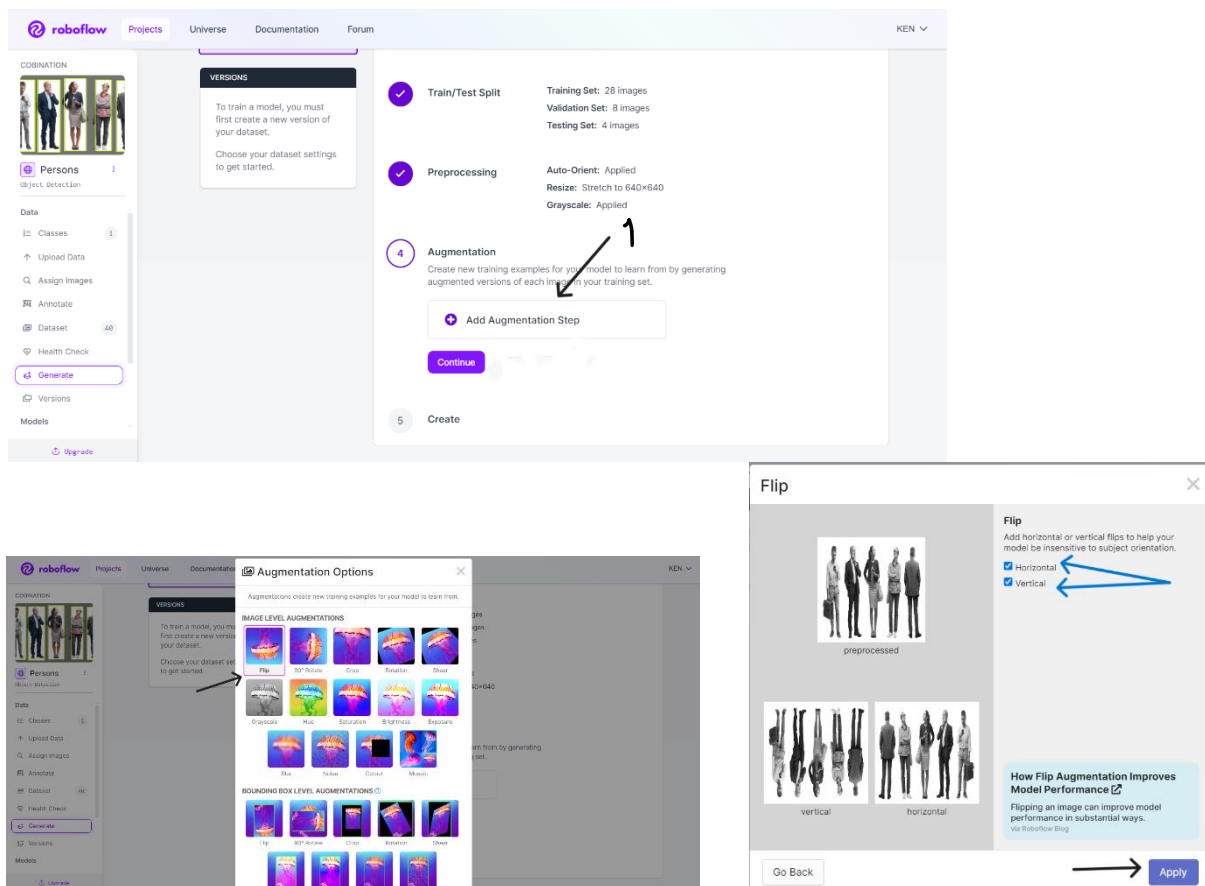
9. ผู้ใช้ทำการกดปุ่ม Add ... images to Dataset เพื่อไปยังขั้นตอนถัดไป



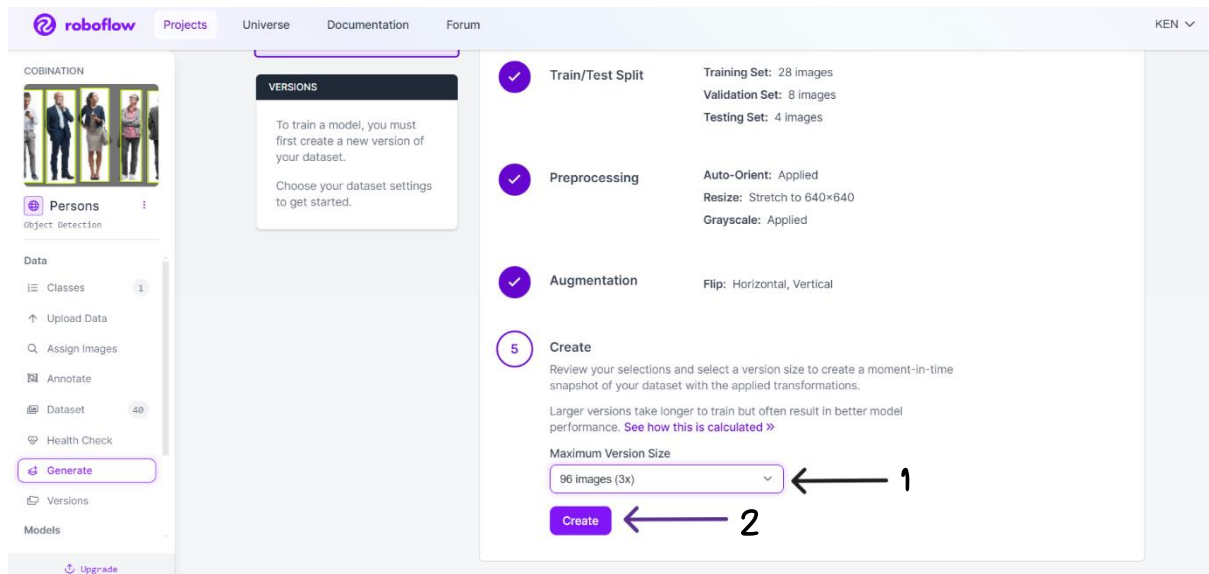
10. ผู้ใช้ทำการแบ่งข้อมูลเป็น Train, Valid, Test ตามลำดับ(ลูกศรสีแดง) หลังจากนั้นกดปุ่ม Add images เพื่อไปยังขั้นตอนถัดไป



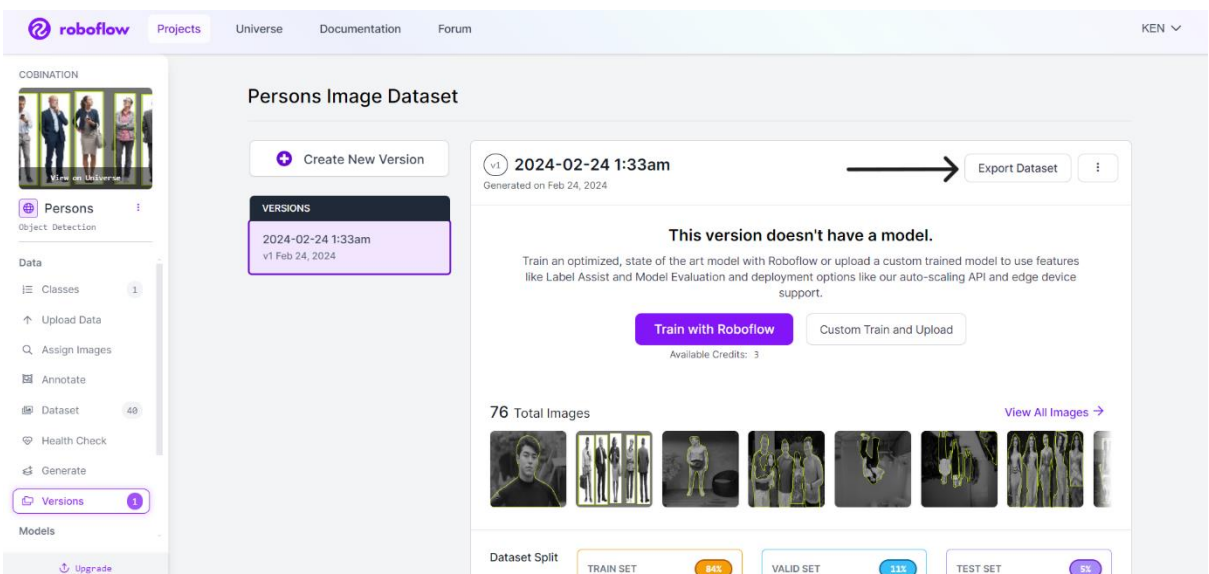
11. ผู้ใช้ทำการกดปุ่ม Add Augmentation เพื่อเพิ่มความหลากหลายและประสิทธิภาพของตัวโมเดล ไม่ว่าจะเป็นการหมุนภาพกลับหัว เป็นต้น จากนั้นกดปุ่ม Continue



12. ทำการเลือก Maximum Version Size เพื่อเพิ่ม dataset จาก Augmentation หลังจากนั้นกดปุ่ม Create



13. ทำการเลือกปุ่ม Export Datasets



14. ทำการเลือก Format เป็น YOLO v5 PyTorch และ ทำตามลูกศรสีม่วงหลังจากนั้นกดปุ่ม Continue Dataset จะถูก Download อัตโนมัติ

Export

Format

YOLO v5 PyTorch

TXT annotations and YAML config used with [YOLOv5](#).

☒ download zip to computer ☐ show download code

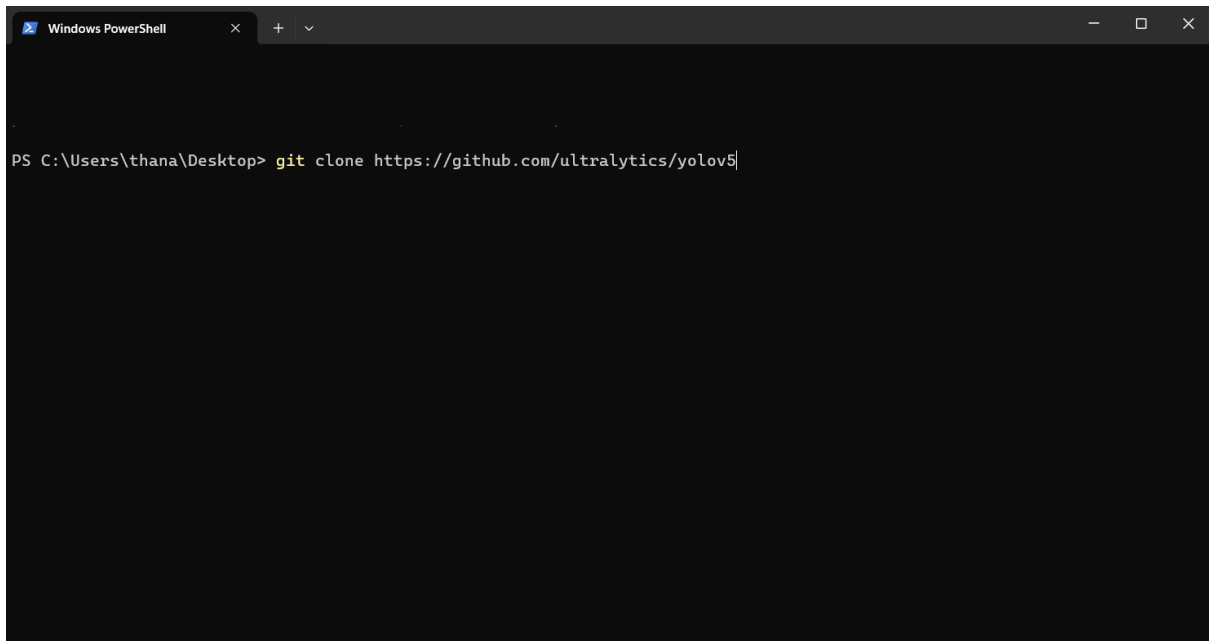
☒ Also train a model for [Label Assist](#) with [Roboflow Train](#).

Cancel Continue

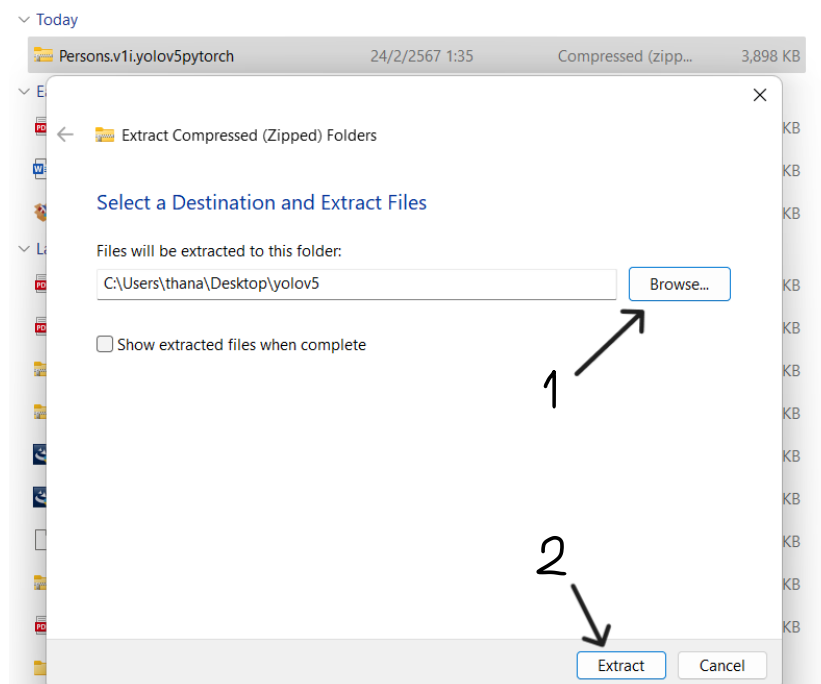
15. หลังจากทำทุกขั้นตอนก่อนหน้านี้หมดแล้ว Dataset จะถูกโหลดมาเก็บไว้ในเครื่อง

ส่วนที่ 2. การเตรียมสร้างโมเดลเพื่อนำมาใช้งาน

1. ใช้คำสั่ง `git clone https://github.com/ultralytics/yolov5` ใน CMD , PowerShell หรือ download file zip ได้จาก link



2. ผู้ใช้ทำการแตกไฟล์ dataset เข้าไปในโฟลเดอร์ yolov5 ที่โคลนจาก git hup



3. ทำการติดตั้ง Library ที่จำเป็นโดยใช้คำสั่ง `pip install -r requirements.txt`

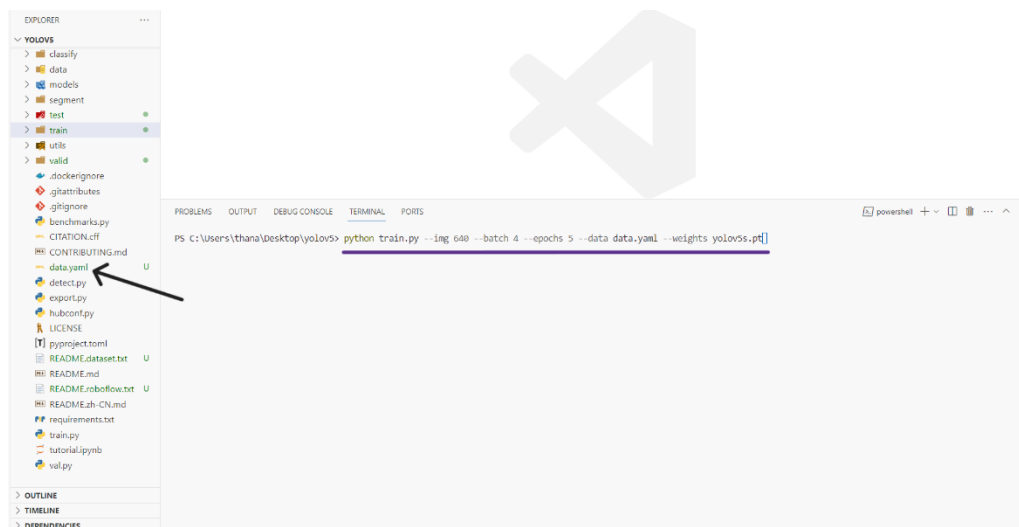
```

PS C:\Users\thana\Desktop\yolov5> pip install -r .\requirements.txt
Requirement already satisfied: gitpython>=3.1.30 in c:\users\thana\appdata\local\programs\python\python311\lib\site-pack
ages (from -r .\requirements.txt (line 5)) (3.1.42)
Requirement already satisfied: matplotlib>=3.3 in c:\users\thana\appdata\local\programs\python\python311\lib\site-packag
es (from -r .\requirements.txt (line 6)) (3.8.2)
Requirement already satisfied: numpy>=1.23.5 in c:\users\thana\appdata\local\programs\python\python311\lib\site-packages
 (from -r .\requirements.txt (line 7)) (1.26.3)
Requirement already satisfied: opencv-python>=4.1.1 in c:\users\thana\appdata\local\programs\python\python311\lib\site-p
ackages (from -r .\requirements.txt (line 8)) (4.9.0.80)
Requirement already satisfied: Pillow>=9.4.0 in c:\users\thana\appdata\local\programs\python\python311\lib\site-packages
 (from -r .\requirements.txt (line 9)) (10.2.0)
Requirement already satisfied: psutil in c:\users\thana\appdata\roaming\python\python311\site-packages (from -r .\requir
ements.txt (line 10)) (5.9.8)
Requirement already satisfied: PyYAML>=5.3.1 in c:\users\thana\appdata\local\programs\python\python311\lib\site-packages
 (from -r .\requirements.txt (line 11)) (6.0.1)
Requirement already satisfied: requests>=2.23.0 in c:\users\thana\appdata\local\programs\python\python311\lib\site-packa
ges (from -r .\requirements.txt (line 12)) (2.31.0)
Requirement already satisfied: scipy>=1.4.1 in c:\users\thana\appdata\local\programs\python\python311\lib\site-packages
 (from -r .\requirements.txt (line 13)) (1.12.0)
Requirement already satisfied: thop>=0.1.1 in c:\users\thana\appdata\local\programs\python\python311\lib\site-packages (
from -r .\requirements.txt (line 14)) (0.1.1.post2209072238)
Requirement already satisfied: torch>=1.8.0 in c:\users\thana\appdata\local\programs\python\python311\lib\site-packages
 (from -r .\requirements.txt (line 15)) (2.2.1)
Requirement already satisfied: torchvision>=0.9.0 in c:\users\thana\appdata\local\programs\python\python311\lib\site-pac
kages (from -r .\requirements.txt (line 16)) (0.17.1)
Requirement already satisfied: tqdm>=4.64.0 in c:\users\thana\appdata\local\programs\python\python311\lib\site-packages
 (from -r .\requirements.txt (line 17)) (4.66.1)
Requirement already satisfied: ultralytics>=8.0.232 in c:\users\thana\appdata\local\programs\python\python311\lib\site-p
ackages (from -r .\requirements.txt (line 18)) (8.1.18)
Requirement already satisfied: pandas>=1.1.4 in c:\users\thana\appdata\local\programs\python\python311\lib\site-packages

```

4. ทำการเทรนโมเดลโดยใช้คำสั่ง `python train.py --img 640 --batch 16 --epochs 2 --data data.yaml --weights yolov5s.pt --cache --patience`

หมายเหตุ: หากมี error แจ้งว่าไม่พบข้อมูลให้เข้าไปตรวจสอบ file data.yaml และในส่วนของ path file: train, test, valid ว่า path ตรงกับที่ของ file จริง ๆ



5. เมื่อทำการเทรนโมเดลเสร็จเรียบร้อยแล้วจะแสดงตามภาพ และ บอกว่า model จะถูกบันทึกไปยัง path ไหน ซึ่ง model จะถูกบันทึกไปยัง /run/train/exp ซึ่งใน folder exp จะมี folder weights สำหรับเก็บ model ที่มีการ train โดยมีสอง file คือ best.pt(model ตัวที่ดีที่สุด) กับ last.pt(model ตัวสุดท้ายที่มีการเทรน)

AutoAnchor: 2.74 anchors/target, 0.996 Best Possible Recall (BPR). Current anchors are a good fit to dataset
 Plotting labels to runs/train/exp/labels.jpg...
 Image sizes 640 train, 640 val
 Using 2 dataloader workers
 Logging results to runs/train/exp
 Starting training for 3 epochs...

Epoch	GPU_mem	box_loss	obj_loss	cls_loss	Instances	Size
0/2	0G	0.1167	0.04297	0	11	640: 100%
Class	Images	Instances	P	R	MAP50	MAP50-95: 0%
Class	Images	Instances	P	R	MAP50	MAP50-95: 50%
Class	Images	Instances	P	R	MAP50	MAP50-95: 100%
all	8	43	0.00167	0.0233	0.000868	0.00026
1/2	0G	0.1137	0.04528	0	18	640: 100%
Class	Images	Instances	P	R	MAP50	MAP50-95: 0%
Class	Images	Instances	P	R	MAP50	MAP50-95: 50%
Class	Images	Instances	P	R	MAP50	MAP50-95: 100%
all	8	43	0.00167	0.0233	0.000964	0.00028
2/2	0G	0.1065	0.04822	0	32	640: 100%
Class	Images	Instances	P	R	MAP50	MAP50-95: 0%
Class	Images	Instances	P	R	MAP50	MAP50-95: 50%
Class	Images	Instances	P	R	MAP50	MAP50-95: 100%
all	8	43	0.0155	0.209	0.0214	0.00647

3 epochs completed in 0.041 hours.
 Optimizer stripped from runs/train/exp/weights/last.pt, 14.4MB
 Optimizer stripped from runs/train/exp/weights/best.pt, 14.4MB
 Validating runs/train/exp/weights/best.pt...
 Fusing layers...
 Model summary: 157 layers, 7012822 parameters, 0 gradients, 15.8 GFLOPs

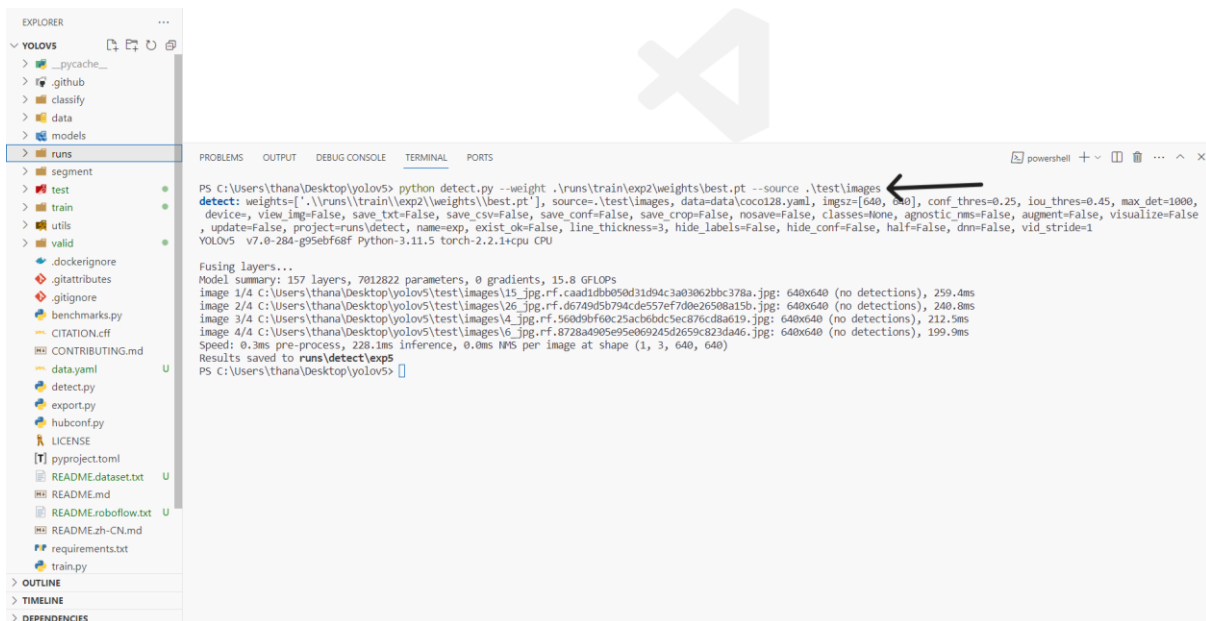
Class	Images	Instances	P	R	MAP50	MAP50-95
Class <td>Images</td> <td>Instances</td> <td>P</td> <td>R</td> <td>MAP50</td> <td>MAP50-95: 0%</td>	Images	Instances	P	R	MAP50	MAP50-95: 0%
Class <td>Images</td> <td>Instances</td> <td>P</td> <td>R</td> <td>MAP50</td> <td>MAP50-95: 50%</td>	Images	Instances	P	R	MAP50	MAP50-95: 50%
Class <td>Images</td> <td>Instances</td> <td>P</td> <td>R</td> <td>MAP50</td> <td>MAP50-95: 100%</td>	Images	Instances	P	R	MAP50	MAP50-95: 100%
all	8	43	0.0153	0.209	0.0214	0.00647

Results saved to runs/train/exp

6. ทำการทดสอบ model ที่สร้าง ด้วยคำสั่ง

`python detect.py --weight .\runs\train\exp2\weights\best.pt` (โมเดลที่สร้างถูกจัดเก็บตรงนี้)

`--source .\test\image` (หากเปลี่ยน เป็น 0 หมายถึงมีการใช้กล้องของอุปกรณ์) และผลจะถูกบันทึกไปยัง `runs\detect\exp5`



บทที่ 3 คำสั่งในการปรับปรุงคุณภาพโมเดล

--weights: เป็นการเลือกตัว model

--source: 0 #web cam, img.jpg #picture, vid.mp4 #video,

--epochs: จำนวนรอบในการ train

--data: data ที่เราใช้ train ต้องเป็น .yaml

--patience: จำนวนครั้งที่เทรนแล้วไม่ฉลาดแล้วให้หยุด train

--cfg: เป็นตัวเลือกที่ใช้ระบุไฟล์ .yaml ที่ใช้กำหนดโครงสร้างของโมเดล เช่น YOLOv5s, YOLOv5m, YOLOv5l, YOLOv5x หรือการปรับแต่งโมเดลเอง

--data ใช้สำหรับระบุไฟล์ .yaml ที่ใช้กำหนดข้อมูลเกี่ยวกับการฝึกโมเดล เช่นตำแหน่งของชุดข้อมูลฝึก, จำนวนคลาส, และอื่น ๆ โดยทั่วไปแล้วไฟล์ .yaml นี้จะมีโครงสร้างเช่นนี้:

```
yaml
train: path/to/train.txt # ไฟล์ที่ระบุเส้นทางไปยังภาพที่ใช้สำหรับฝึกโมเดล
val: path/to/val.txt # ไฟล์ที่ระบุเส้นทางไปยังภาพที่ใช้สำหรับการตรวจสอบโมเดล
test: path/to/test.txt # ไฟล์ที่ระบุเส้นทางไปยังภาพที่ใช้สำหรับการทดสอบโมเดล (ถ้ามี)
nc: 80 # จำนวนคลาส
names: ['class1', 'class2', ..., 'classN'] # ชื่อคลาส
```

--hyp ใช้สำหรับระบุไฟล์ .yaml ที่ใช้กำหนดค่า hyperparameters ต่าง ๆ ของการฝึกโมเดล เช่น learning rate, momentum, weight decay, เป็นต้น โดยไฟล์ .yaml นี้จะมีโครงสร้างดังนี้

```
lr0: 0.01 # อัตราการเรียนรู้เริ่มต้น
lrf: 0.2 # อัตราการเรียนรู้สุดท้าย
momentum: 0.937 # โมเมนตัม
weight_decay: 0.0005 # ค่า weight decay
warmup_epochs: 3.0 # จำนวน epochs ที่ใช้ในการเตรียมการฝึกโมเดล
warmup_momentum: 0.8 # ค่าโมเมนตัมในช่วงการเตรียมการ
warmup_bias_lr: 0.1 # อัตราการเรียนรู้ของ bias ในช่วงการเตรียมการ
box: 0.05 # ค่าตัวเลขสำหรับปรับค่าพารามิเตอร์ของกล่องกลาง
cls: 0.5 # ค่าตัวเลขสำหรับปรับค่าพารามิเตอร์ของคลาส
cls_pw: 1.0 # สำหรับปรับค่าพารามิเตอร์ของคลาสในหลายหมวดหมู่
obj: 1.0 # ค่าตัวเลขสำหรับปรับค่าพารามิเตอร์ของการตรวจจับวัตถุ
obj_pw: 1.0 # สำหรับปรับค่าพารามิเตอร์ของการตรวจจับวัตถุในหลายหมวดหมู่
iou_t: 0.2 # ค่าตัวเลขสำหรับปรับค่าพารามิเตอร์ของ iou threshold
anchor_t: 4.0 # ค่าตัวเลขสำหรับปรับค่าพารามิเตอร์ของ anchor_t threshold
fl_gamma: 0.0 # ค่าตัวเลขสำหรับปรับค่าพารามิเตอร์ของ focal loss
hsv_h: 0.015 # ค่าตัวเลขสำหรับปรับค่าพารามิเตอร์ของการปรับสี hsv_h
hsv_s: 0.7 # ค่าตัวเลขสำหรับปรับค่าพารามิเตอร์ของการปรับสี hsv_s
hsv_v: 0.4 # ค่าตัวเลขสำหรับปรับค่าพารามิเตอร์ของการปรับสี hsv_v
degrees: 0.0 # ค่าตัวเลขสำหรับปรับค่าพารามิเตอร์ของการหมุนภาพ
translate: 0.1 # ค่าตัวเลขสำหรับปรับค่าพารามิเตอร์ของการย้ายภาพ
scale: 0.5 # ค่าตัวเลขสำหรับปรับค่าพารามิเตอร์ของการปรับขนาดภาพ
shear: 0.0 # ค่าตัวเลขสำหรับปรับค่าพารามิเตอร์ของการเอียงภาพ
perspective: 0.0 # ค่าตัวเลขสำหรับปรับค่าพารามิเตอร์ของการเส้นทางมุม
flipud: 0.0 # ค่าตัวเลขสำหรับปรับค่าพารามิเตอร์ของการสลับภาพตามแนวดิ่ง
fliplr: 0.5 # ค่าตัวเลขสำหรับปรับค่าพารามิเตอร์ของการสลับภาพตามแนวนอน
mosaic: 1.0 # ค่าตัวเลขสำหรับปรับค่าพารามิเตอร์ของการใช้งานฟังก์ชัน mosaic
mixup: 0.0 # ค่าตัวเลขสำหรับปรับค่าพารามิเตอร์ของการใช้งานฟังก์ชัน mixup
copy_paste: 0.0 # ค่าตัวเลขสำหรับปรับค่าพารามิเตอร์ของการใช้งานฟังก์ชัน copy_paste
```

python train.py --img 640 --batch 16 --epochs 100 --data your_data.yaml --hyp your_hyperparameters.yaml (คำสั่งใน cmd)

--batch-size ใช้สำหรับระบุขนาดของ batch ในแต่ละรอบการฝึกโมเดล โดย batch size คือจำนวนภาพที่ใช้ในการฝึกโมเดลในแต่ละรอบ โดยทั่วไปแล้วมีการเลือกขนาด batch ที่เหมาะสมเพื่อให้การฝึกโมเดลเสถียรและมีประสิทธิภาพ โดยใช้ batch size ที่มากๆ อาจทำให้ใช้หน่วยความจำมากขึ้น Ex. --batch-size 16

--imgsz ใช้สำหรับระบุขนาดของภาพนำเข้า (input image size) ที่ใช้ในการฝึกโมเดล โดยทั่วไปแล้วการเลือกขนาดของภาพนำเข้าจะขึ้นอยู่กับความต้องการและข้อจำกัดของโปรเจกเฉพาะ ๆ Ex. --imgsz 640

--rect จะเป็น boolean เพื่อระบุว่าควรใช้ bounding box รูปทรงสี่เหลี่ยมหรือไม่ Ex. --rect

--resume: เป็นตัวเลือกที่ใช้ระบุให้การฝึกโมเดลดำเนินการต่อจากการฝึกที่หยุดไว้ก่อนหน้านี้ โมเดลจะถูกโหลดมาและการฝึกจะดำเนินการต่อจาก epoch ล่าสุดที่หยุด Ex. --resume

--nosave: เป็นตัวเลือกที่ใช้ระบุให้ไม่บันทึกโมเดลหลังจากการฝึกโมเดลเสร็จสิ้น โดยไม่มีการบันทึกโมเดลที่ได้หลังจากการฝึกโมเดลเสร็จสิ้น Ex. --noseve

--resume จะอ่าน weights จากไฟล์ที่ระบุและใช้เป็นโมเดลเริ่มต้นสำหรับการฝึกต่อ หากต้องการฝึกโมเดลใหม่โดยใช้ weights จากการฝึกเก่า เพียงแค่ระบุไฟล์ weights

--device 0 หมายถึงการใช้ GPU หมายเลข 0 ในการฝึก ส่วนการใช้ --device cpu หมายถึงการใช้ CPU ในการฝึกโมเดล

-multi-scale จะเปิดใช้งาน multi-scale training และในแต่ละรอบการฝึกโมเดล จะมีการสุ่มเลือกขนาดของภาพนำเข้าจากช่วงขนาดที่กำหนดไว้ ซึ่งช่วยเพิ่มประสิทธิภาพของโมเดลในการตรวจจับวัตถุที่มีขนาดต่าง ๆ ได้

--single-cls ใช้ในกรณีที่ต้องการฝึกโมเดลในโหมด single-class detection ซึ่งหมายความว่าโมเดลจะถูกฝึกเพื่อตรวจจับวัตถุเพียงหนึ่งประเภทเท่านั้น Ex --single-cls

-sync-bn จะเปิดใช้งาน Synchronized Batch Normalization ในโมเดล ทำให้การคำนวณ Batch Normalization สามารถทำงานได้รวดเร็วและมีประสิทธิภาพมากขึ้น โดยอัตโนมัติ เฉพาะเมื่อใช้ GPU หลายๆ ตัวพร้อมกัน

--workers ใช้ในการระบุจำนวนของเวิร์กเกอร์ (workers) ที่ใช้ในการโหลดข้อมูลขณะทำการฝึก โมเดล เวิร์กเกอร์เหล่านี้จะทำงานพร้อมกันเพื่อเร่งความเร็วของการฝึก Ex. --workers 4

--freeze ใช้ในการแช่แข็ง (freeze) ชั้นของโมเดลในขณะฝึกโดยไม่มีการอัปเดตน้ำหนักของชั้นเหล่านั้น ซึ่งช่วยให้โมเดลสามารถฝึกได้อย่างเร็วและประหยัดทรัพยากร

บทที่ 4 การใช้โมเดลที่สร้างขึ้นจากสมาชิกในกลุ่ม

เป็นการนำโมเดลที่มีการสร้างโดยนักศึกษามาใช้งาน

1. ใช้คำสั่ง `git clone https://github.com/KENTHN658/detect_people_from_video` ใน CMD , PowerShell หรือ download file zip ได้จาก link

```
PS C:\Users\thana\Desktop> git clone https://github.com/KENTHN658/detect_people_from_video
Cloning into 'detect_people_from_video'...
remote: Enumerating objects: 205, done.
remote: Counting objects: 100% (205/205), done.
remote: Compressing objects: 100% (145/145), done.
remote: Total 205 (delta 66), reused 193 (delta 58), pack-reused 0Receiving objects: 97% (199/205), 964.00 KiB | 1.84 M
Receiving objects: 100% (205/205), 2.01 MiB | 2.13 MiB/s, done.
```

2. ใช้คำสั่ง `pip install -r requirements.txt`

```
Windows PowerShell
PS C:\Users\thana\Desktop\detect_people_from_video> pip install -r requirements.txt
```

3. ใช้คำสั่ง `python detect.py --weight humandetect.pt --source 0` เพื่อใช้โมเดลที่ถูกสร้างโดยนักศึกษา หลังจากนั้นจะมีหน้าต่างกล้องแสดงขึ้นมา