Chapter 4：YOLOv5

本章重點

介紹如何使用 YOLOv5 預訓練模型來辨識物件。並自行訓練物件偵測模型，本章以訓練和辨識交通號誌為範例。

準備材料

可上網的電腦

學習目標

1. 安裝虛擬環境。
2. 使用 YOLOv5 預模型來偵測物件。
3. 自行訓練 YOLOv5 模型來偵測物件。

5-1. 安裝虛擬環境

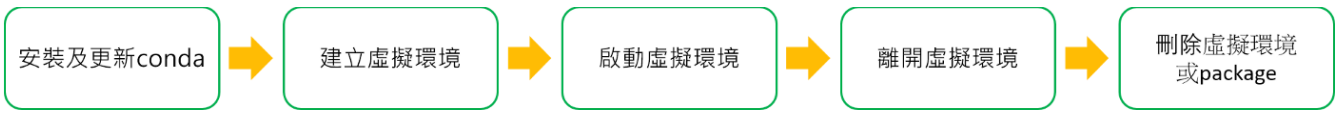
利用 conda 為每一個不同需求的專案建立一個獨立適合的虛擬環境。

用 conda 建立及管理 Python 虛擬環境

開發 Python 專案時，最常遇見的問題就是不同專案可能會有不同的 Python 版本，以及不同的 Package (套件) 需要安裝，例如：Python 2 或 Python 3 的版本。那麼管理不同版本的專案會是一個問題。如果你只需要使用特定的套件，或是想要嘗試各種不同的環境應用，但又不想彼此的開發環境受到影響，那 Anaconda 的套件管理系統 conda 將會是一個不錯的解決方案。

如果慣用 Python 的軟體包管理系統 pip 的人，對於 conda 的指令一定也可以馬上上手，因為它和 pip 指令非常的相似，conda 是套件管理系統，也可用來建立虛擬環境。

conda 命令是管理在安裝不同 package 時的主要介面，使用 conda 時，你可以進行建立 (create)、輸出 (export)、列表 (list)、移除 (remove) 和更新 (update) 環境於不同 Python 版本及 Packages，同時也可以分享你的虛擬環境。接下來筆者將針對 conda 如何建立及管理虛擬環境用下列 5 步驟來說明。



Step 1 安裝及更新 conda

安裝好 Anaconda 後，在 Windows 開始選單 (Start menu) 中選擇 Anaconda Prompt。

檢查目前 Python 版本：

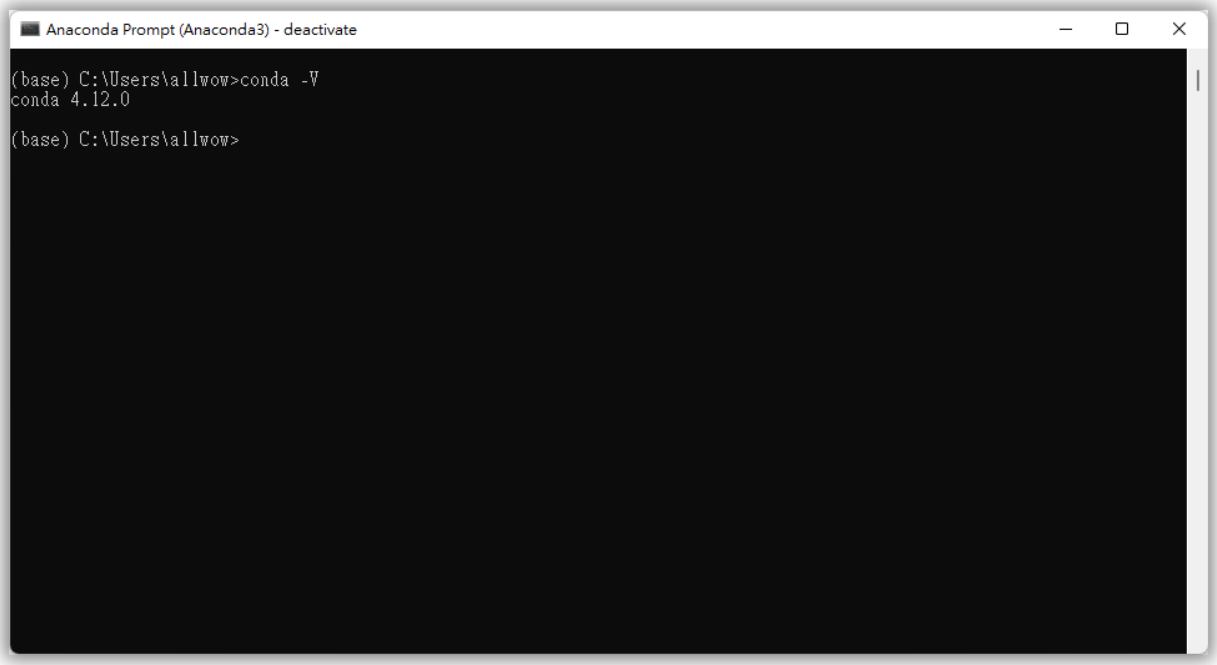
```
python –V
```

檢查目前 Python 環境安裝了那些 Package：

```
pip list
```

檢查目前 conda 版本：

```
conda –V
```



檢查目前 conda 環境安裝了那些 Package：

```
conda list
```

若想要進行更新，可以輸入下列命令：(更新必須以系統管理員執行 Anaconda Prompt)

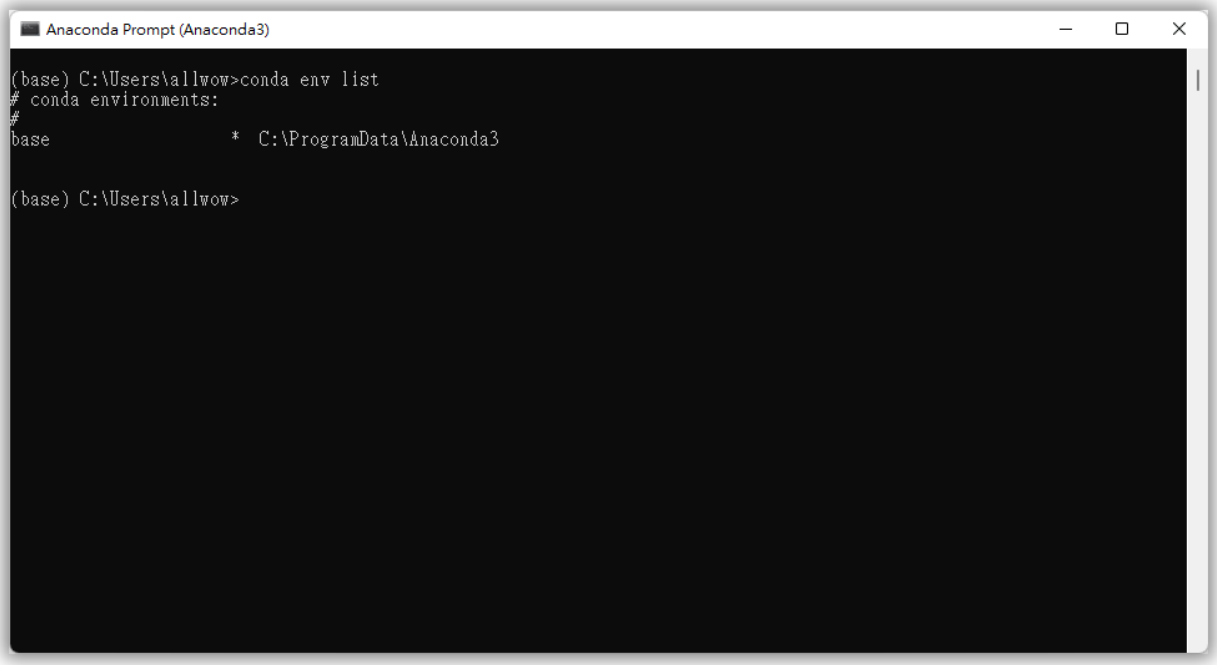
```
conda update conda
```

Step 2　建立虛擬環境

看目前系統已經安裝幾個虛擬環境：

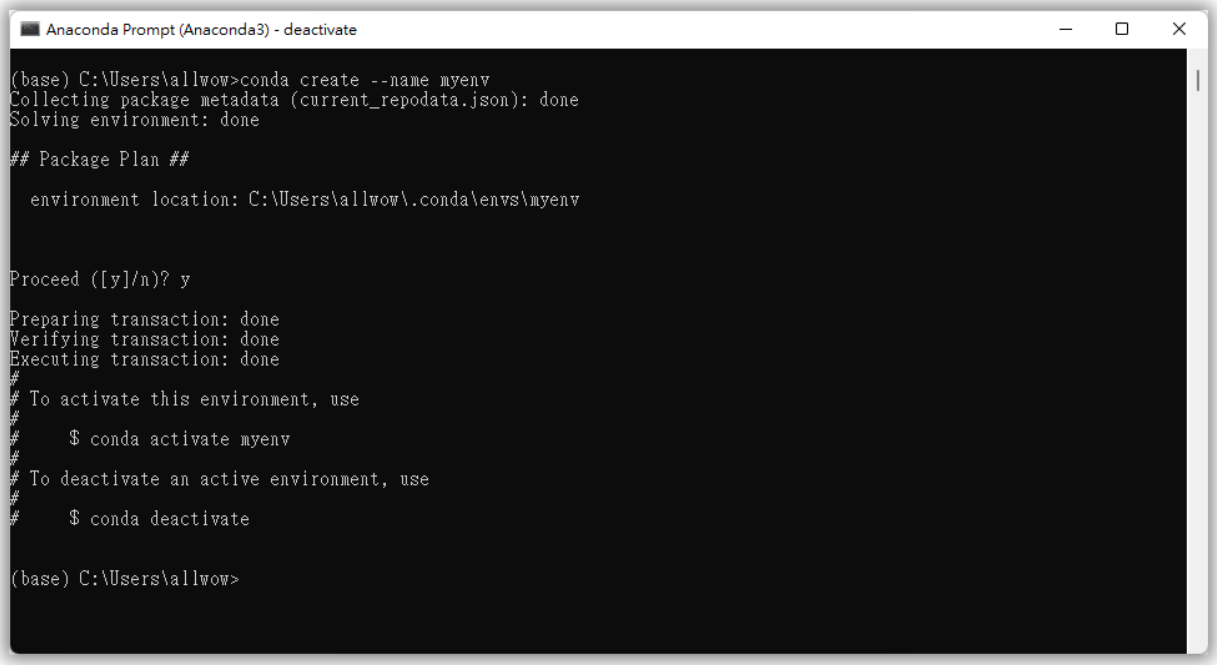
```
conda env list
```

以下預設只有一個環境，為系統預設的 base：



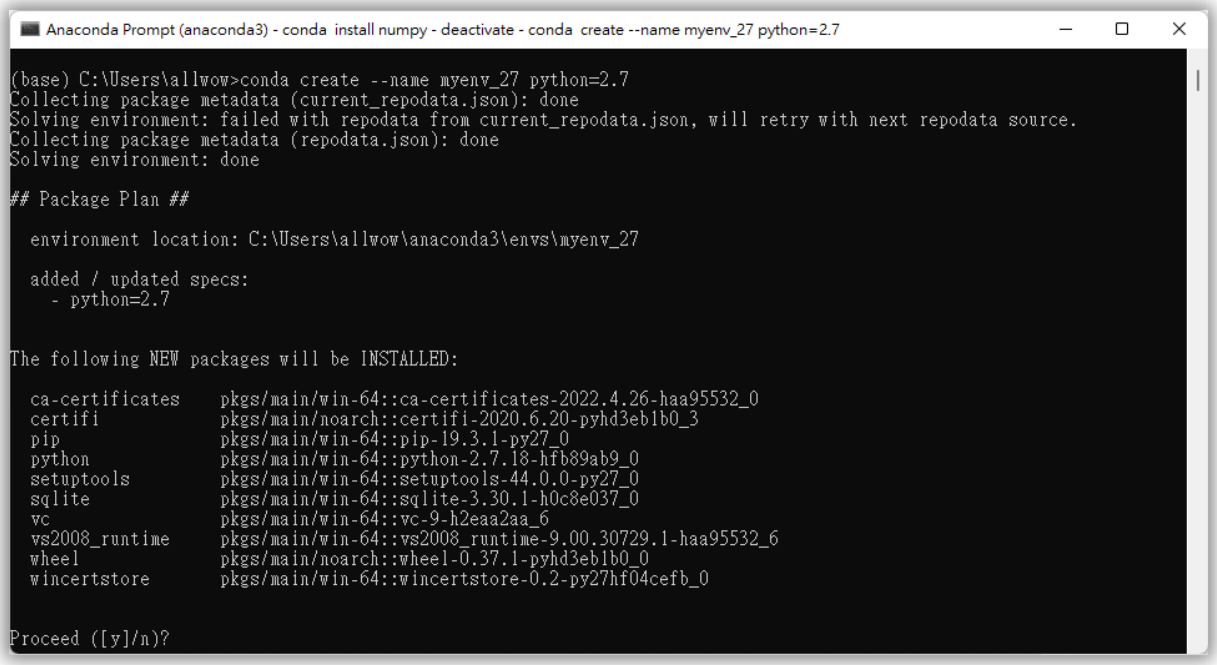
若要建立一個叫做 myenv 的虛擬環境，並且是安裝最新的 Python 的版本：

```
conda create --name myenv
```

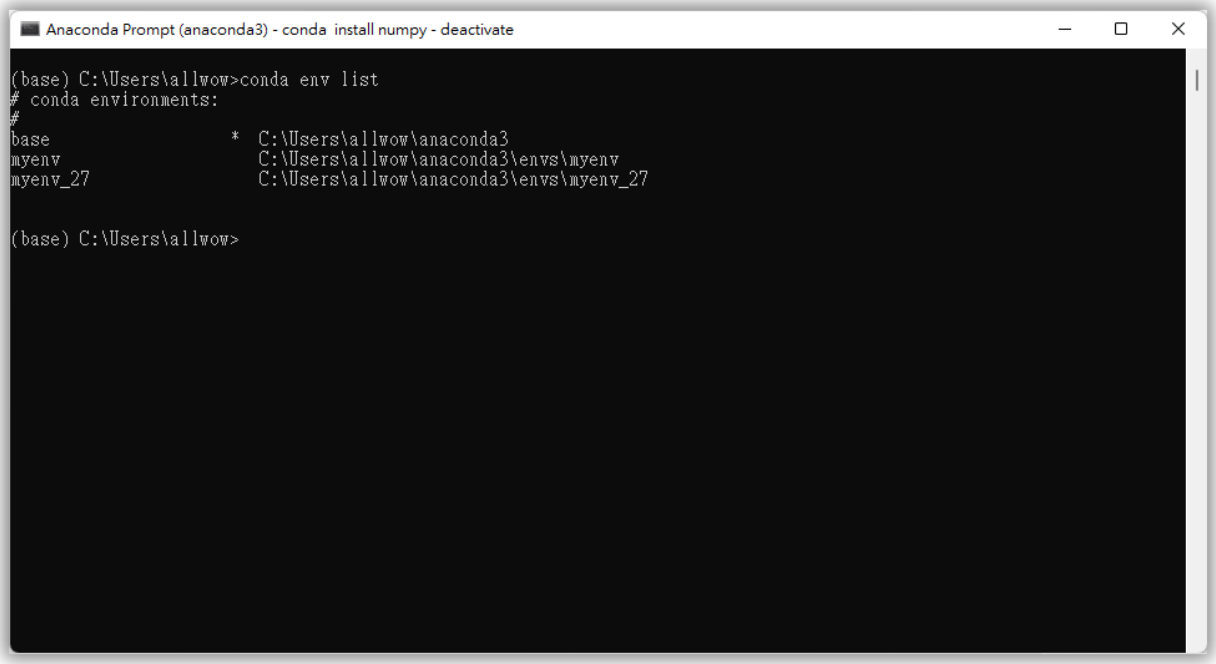


若要建立一個叫做 myenv_27 的虛擬環境，並且指定 Python 2.7 的版本：

```
conda create --name myenv_27 python=2.7
```



再次執行 conda env list，列出目前虛擬環境狀況，將會看到多了兩個剛建立的虛擬環境 myenv 與 myenv_27。



Step 3 啟動虛擬環境

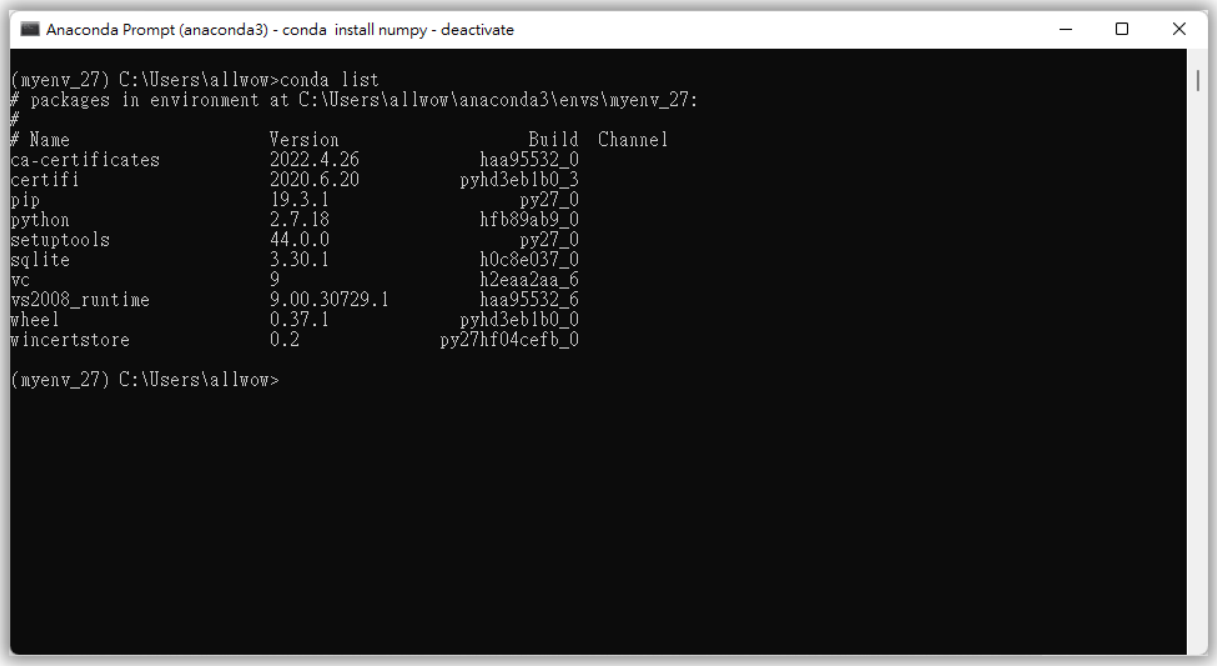
啟動一個新的虛擬環境：

```
activate myenv_27
```

這時候 cmd 模式下前面會有一個 (myenv)，表示你目前是處於此虛擬環境中，這時候你就可以在此虛擬環境中，開始安裝你所需要的各種 Package。

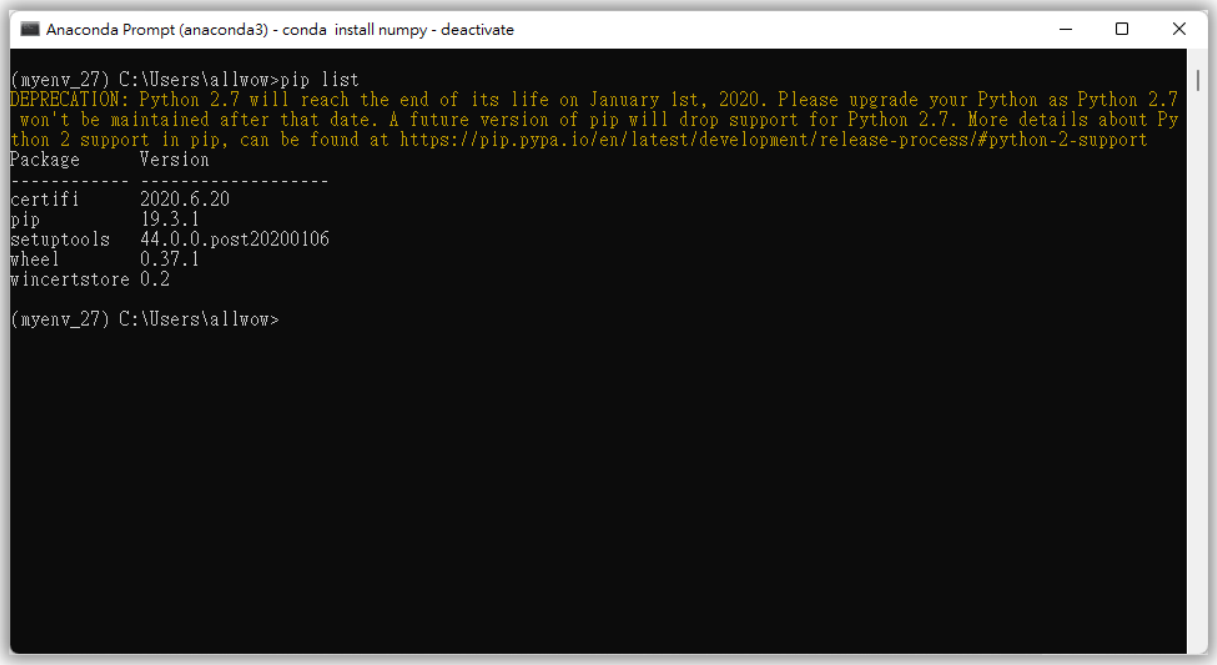
檢查目前此虛擬環境中的 Conda 安裝了那些東西：

```
(myenv_27) conda list
```



檢查目前虛擬環境的 Python 環境，安裝了那些 Package：

```
(myenv_27) pip list
```



要在此虛擬環境下安裝所需套件，例如：numpy。

```
(myenv_27) conda install numpy
```

Step 4 離開虛擬環境

關閉目前的虛擬環境：

```
(myenv_27) deactivate
```

Step 5 刪除虛擬環境的 Package 或虛擬環境本身

要刪除虛擬環境中某個 Package：(例如：虛擬環境 myenv_27 中的 numpy)

```
(myenv_27) conda remove --name myenv_27 numpy
```

要刪除整個虛擬環境：(不能刪除當前的虛擬環境，必須先關閉目前的虛擬環境)

```
conda env remove --name myenv_27
```

Step 6 複製虛擬環境

複製虛擬環境：(myEnvNameDes：目的檔案，myEnvNameSou：來源檔案)

```
conda create -n myEnvNameDes --clone myEnvNameSou
```

Step 7 為虛擬環境安裝 Anaconda

Anaconda 的 Spyder 有除錯模式，可單步執行追蹤程式碼。目前 Anaconda 的最新版本是支援 Python 3.9，所以先創建一個 Python 3.9 的虛擬環境，然後安裝 Anaconda 整合開發環境後，再安裝各種 Python 的 Package，這樣的虛擬環境會比較方便專案管理。

建一個 Python 3.9 的虛擬環境 myenv_39，並且切換至 myenv_39：

```
conda create --name myenv_39 python=3.9 & activate myenv_39
```

安裝 Anaconda：

```
(myenv_39) conda install anaconda
```

建立好基本的虛擬開發環境後，以後就可以使用複製虛擬環境的方式，快速管理。

5-2. YOLOv5 物件偵測

安裝 YOLOv5 環境

首先，建立一個 YOLOv5 專用的 Python 3.9 虛擬環境，可由上一節完成的 myenv_39 來複製，名稱為 yolov5env：

```
conda create -n yolov5env --clone myenv_39
```

載入 Python 虛擬環境 yolov5env：

```
activate yolov5env
```

下載 YOLOv5 檔案

安裝 git for Windows 的下載指令：

下載 YOLOv5 原始碼：

```
# 下載 YOLOv5 原始碼
(yolov5env) git clone https://github.com/ultralytics/yolov5
```

在 Python 虛擬環境之下，安裝 YOLOv5 所需要的各種 Python 的 Package：

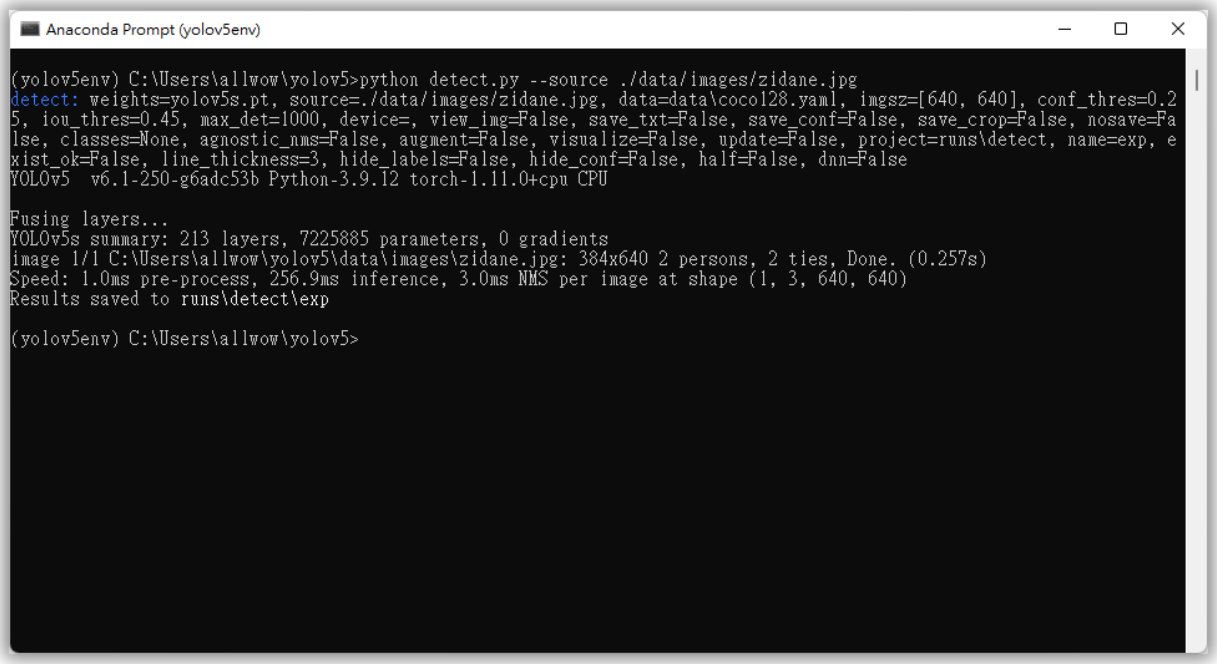
```
# 安裝 YOLOv5 所需環境
(yolov5env) cd yolov5
(yolov5env) pip install -r requirements.txt
```

使用 YOLOv5 模型偵測物件

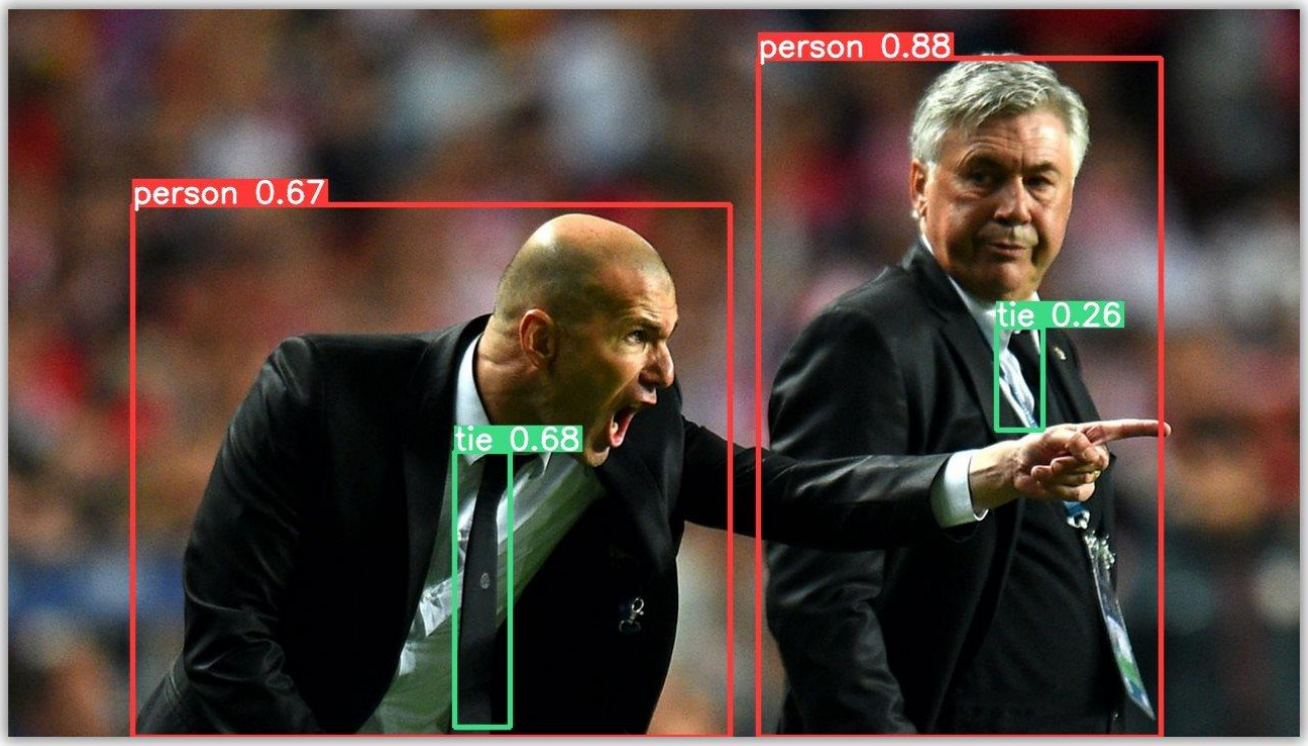
detect.py 指令稿

在 YOLOv5 的原始碼中，有附帶一個偵測物件用的 Python 指令稿 detect.py，可以使用指定的 YOLOv5 模型來進行物件偵測。例如：拿官方的 zidane.jpg 圖檔 (位於 data/images 資料夾) 進行物件偵測：

```
# 以 YOLOv5 模型偵測圖片物件
(yolov5env) python detect.py --source data/images/zidane.jpg
```



其中 --source 參數可以用來指定影像來源，支援的格式非常多，包含：電腦攝影機、圖片檔案（jpg、png)、影片檔案（mp4)、包含圖片的目錄、YouTube 網址、RTSP 串流位址等，而偵測結果預設會擺放在 runs/detect/exp 目錄之下，以下是偵測的結果。



進行攝影機偵測，第一台攝影機參數為 --source 0，第二台為 --source 1，依此類推：

```
# 以 YOLOv5 模型偵測攝影機物件
(yolov5env) python detect.py --source 0
```

YOLOv5 實際上又依據模型複雜度分為 yolov5n、yolov5s (預設)、yolov5m、yolov5l、yolov5x，我們可以使用 --weight 參數來指定要採用的模型，例如：使用 yolov5n。

```
# 以 yolov5n 的模型偵測攝影機物件
(yolov5env) python detect.py --source 0 --weight yolov5n.pt
```

detect.py 指令稿還有支援其他相當多的參數，可以使用 --help 查詢：

```
# 查詢參數用法
(yolov5env) python detect.py --help
```

自行撰寫 Python 偵測物件

除了使用既有的指令稿之外，我們也可以自行撰寫 Python 來使用 YOLOv5 模型偵測物件：

5-1.py 匯入 torch 來使用 YOLOv5 模型偵測物件。

```
import torch

# 從 PyTorch Hub 下載 YOLOv5s 預訓練模型，可選用的模型有 yolov5s, yolov5m, yolov5x 等
model = torch.hub.load('ultralytics/yolov5', 'yolov5s')
# 影像來源，支援檔案、路徑、PIL、OpenCV, NumPy, list 等
img = 'https://ultralytics.com/images/zidane.jpg'
# 進行物件偵測
results = model(img)
# 顯示結果摘要
results.print()
```

輸出結果：

```
image 1/1: 720x1280 2 persons, 2 ties
Speed: 1071.2ms pre-process, 186.9ms inference, 2.0ms NMS per image at shape (1, 3, 384, 640)
```

偵測出物件之後，可以顯示或儲存結果圖片：

```
# 顯示結果圖片
results.show()
# 儲存結果圖片
results.save()
```

亦可顯示所有偵測到的物件列表，包含位置、信心（confidence）、類別編號與名稱：

```
# 顯示物件列表
print(results.pandas().xyxy[0])
```

輸出結果：

```
     xmin        ymin         xmax        ymax      confidence  class  name
0  742.974792   48.395599  1141.844604  720.000000    0.881052      0  person
1  442.007660  437.522461   496.653961  709.973511    0.675214     27     tie
2  123.024261  193.287292   715.662109  719.723877    0.665813      0  person
3  982.803162  308.417358  1027.365845  419.987000    0.260076     27     tie
```

5-3. 自行訓練 YOLO 模型

自行訓練模型

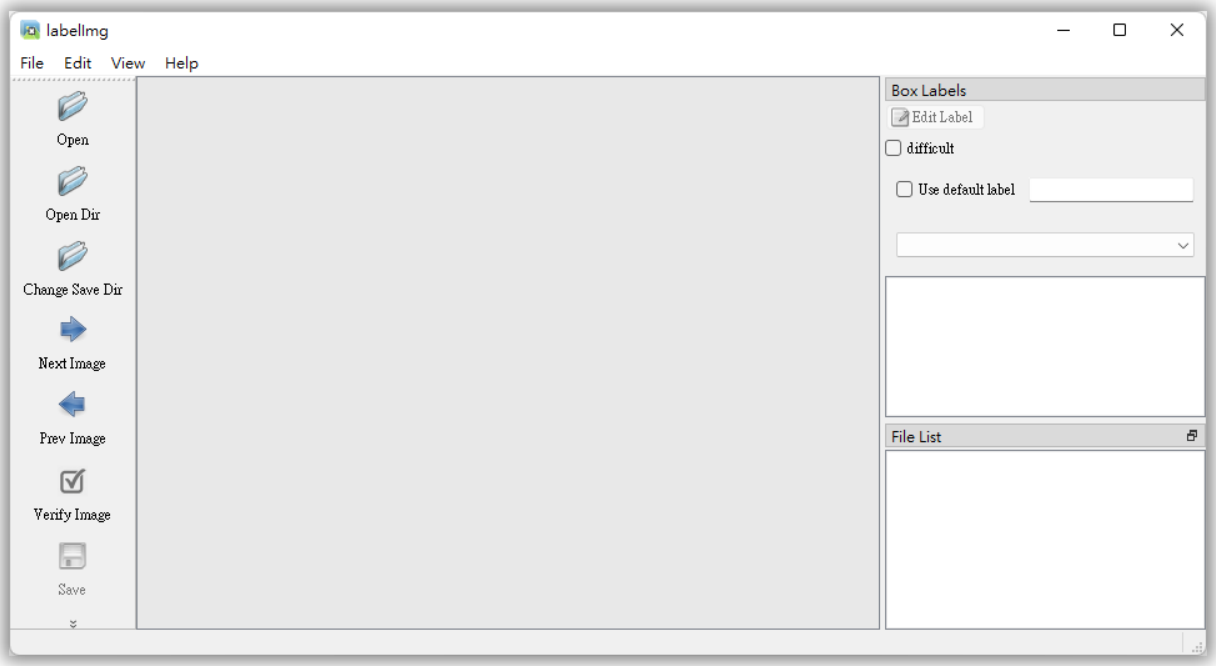
若想要針對自己的資料，訓練出自己的 YOLO 模型，可以先準備好標註的圖片之後，再用預訓練的 YOLO 模型，進行遷移式學習（transfer learning）。遷移式學習使用已有問題的解決模型，並將其利用在其他不同但相關問題上。例如：用來辨識汽車的知識（或者是模型）也可以被用來提升識別卡車的能力。

安裝 labelImg 標註工具

實作影像的物件偵測時，需要有大量的已知資料集，也就是照片加上物件的所在位置（特徵）以及物件的名稱（標籤），初期都會使用人工的方式來手動標註，而 labelImg 就是用來標註照片中物體位置與名稱的小工具。此工具是以 Python 所開發的，可以用 pip 安裝：

```
# 從 PyPI 下載與安裝 labelImg
(yolov5env) pip install labelImg
# 執行 labelImg，start /min 為背景執行
(yolov5env) start /min labelImg
```

開啟畫面如下：



標註圖片

Step 1 自行拍攝或下載圖片數據集。

你需要自行拍攝或下載圖片數據集 (dataset)。

例如：到 MakeML (https://makeml.app/datasets/road-signs) 下載訓練照片檔 (檔案名稱為 RoadSignsPascalVOC.zip)。

以下是照片的規格：

- 數據集包括以下的類別：Trafic Light (紅綠燈)、Stop (停車)、Speedlimit (限速)、Crosswalk (人行橫道)。
- 圖片數量：877 個 .png。

Trafic Light

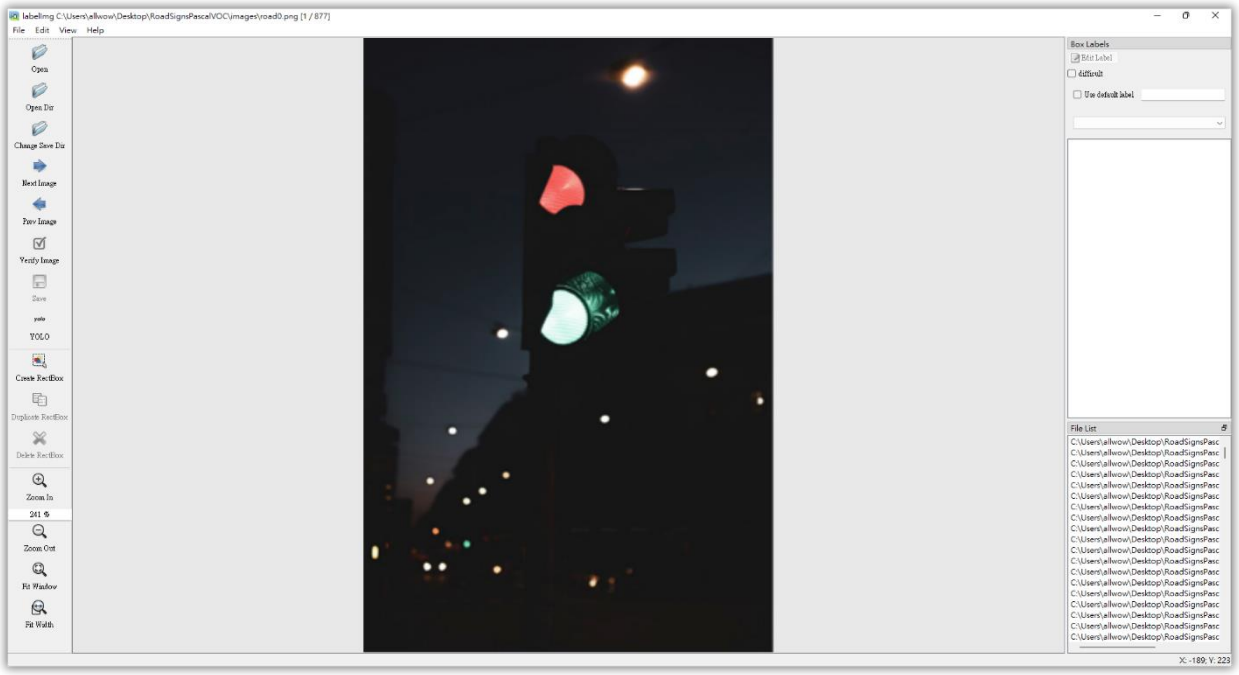
Stop

Speedlimit

Crosswalk

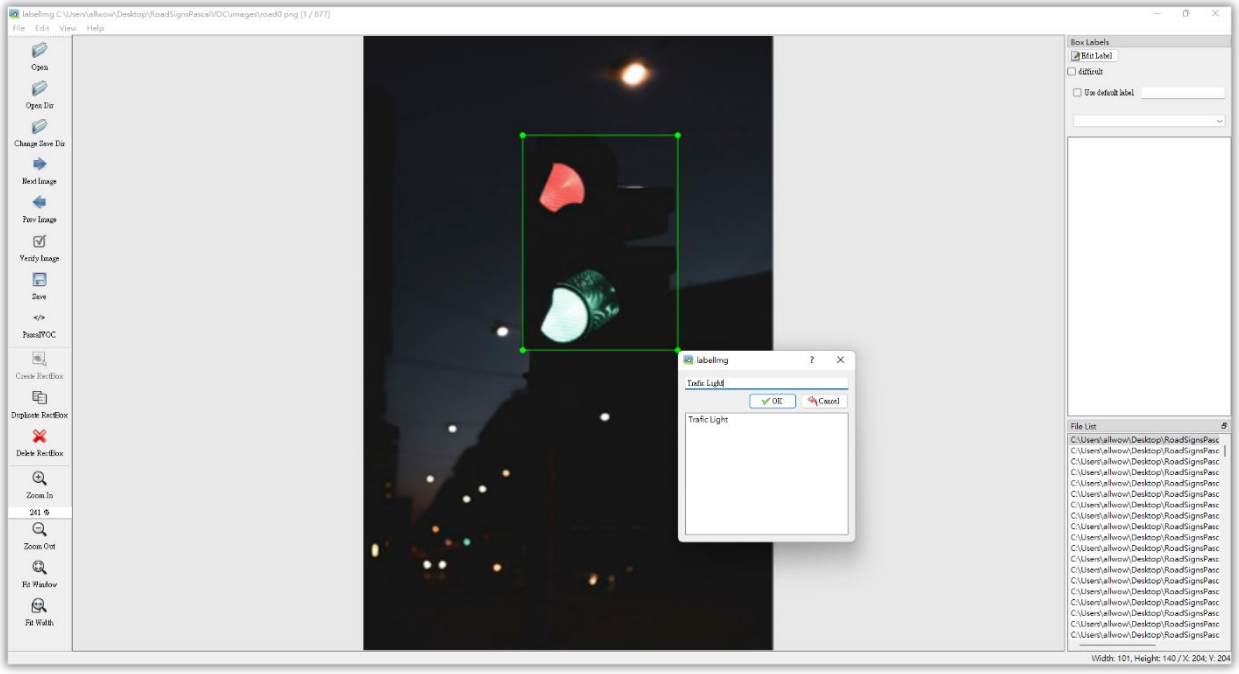
Step 2　開啟圖片

開啟 labelImg 之後，首先開啟想要進行標註的圖片檔，開啟時可以選擇『 Open 』開啟單張圖檔，或是以『 Open Dir 』開啟整個目錄中所有的圖檔。例如：解壓縮 RoadSignsPascalVOC.zip 後，以『 Open Dir 』開啟 images 資料夾：

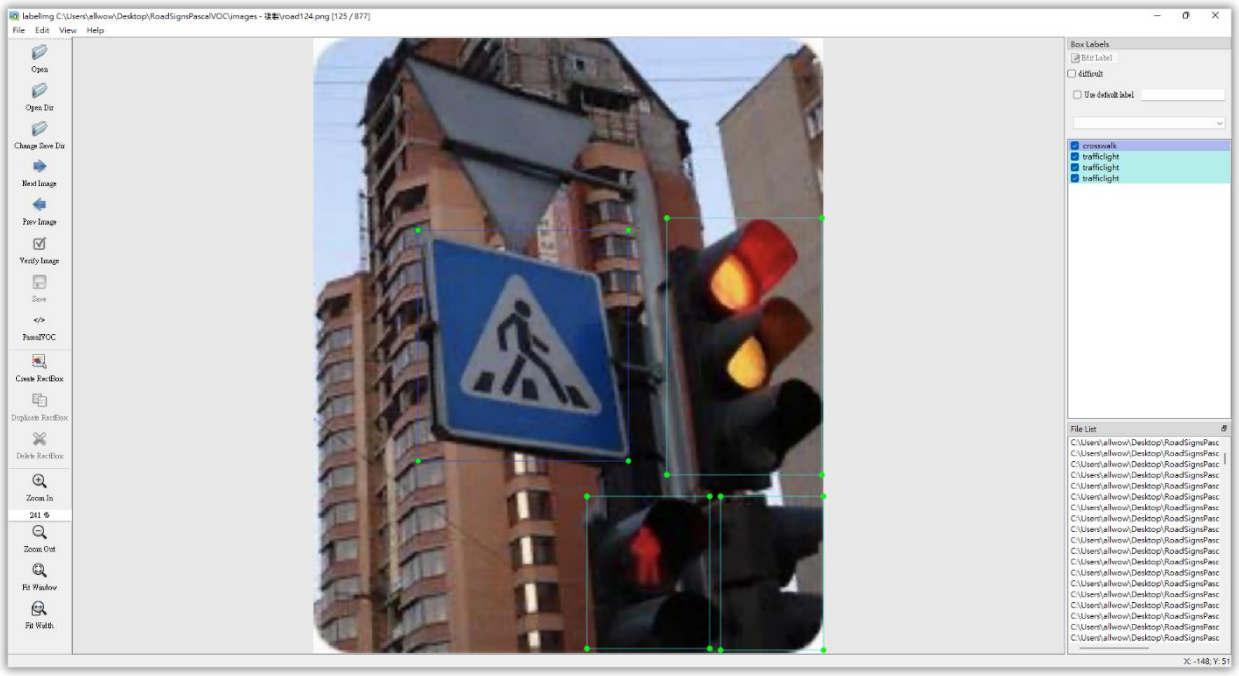


Step 3　標示出物件的位置與標籤。

選擇『 Create RectBox 』，使用滑鼠將物件框起來，並輸入這個物件的標籤。

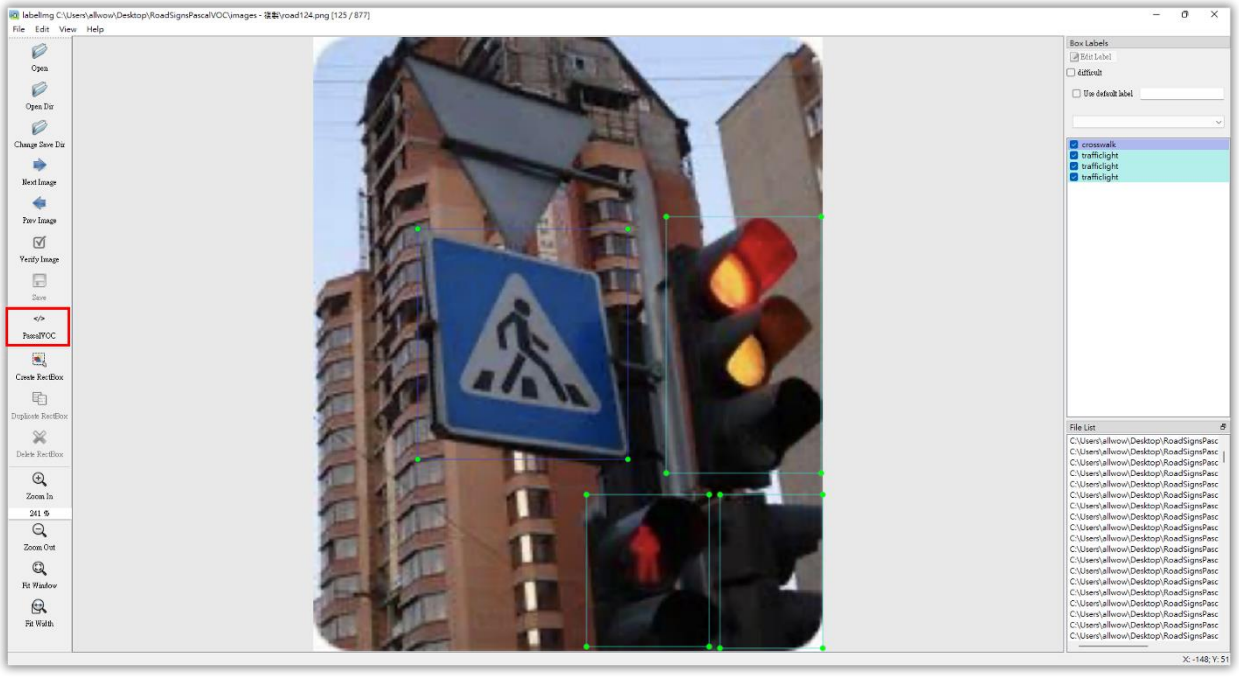


一張圖有多少想要分類的類別，都要標註。標示完成後，在視窗的右方會列出所有標示好的物件，我們可以對任何的物件進行修改或刪除等動作。



Step 4 轉檔與儲存。

當一張圖片上的物件都標註完成後，記得按下左側的『 Save 』鍵儲存，labelImg 會使用 XML 格式（PascalVOC）來儲存標註資訊。



介紹兩種常用的深度學習引擎的圖片標註格式：

- PascalVOC：xml 檔。使用 LabelImg 開啟照片時，會自動檢查是否有對應的標註檔，如果它發現同目錄下有對應的 xml 標註檔，就會自動載入它並顯示出來。例如：

```
<annotation>
    <folder>images</folder>
    <filename>road1.png</filename>
    <size>
        <width>400</width>
        <height>283</height>
        <depth>3</depth>
    </size>
    <segmented>0</segmented>
    <object>
        <name>trafficlight</name>
        <pose>Unspecified</pose>
        <truncated>0</truncated>
        <occluded>0</occluded>
        <difficult>0</difficult>
        <bndbox>
            <xmin>154</xmin>
            <ymin>63</ymin>
            <xmax>258</xmax>
            <ymax>281</ymax>
        </bndbox>
    </object>
</annotation>
```

- YOLO：txt 檔。訓練 YOLO 模型時可以直接使用的文字檔，例如：

```
1 0.515000 0.607774 0.260000 0.770318
```

YOLO 的格式是普通文字檔，每一行代表一個標註物件。

格式為：`class x_center y_center width height`

5 個欄位以空白分隔，意義如下：

1. class：類別 ID。
2. x_center：中心點 X 座標 (標準化)。
3. y_center：中心點 Y 座標 (標準化)。
4. width：方框寬度 (標準化)。
5. height：方框高度 (標準化)。

Step 5 按 YOLO 的目錄管理方式存儲圖片與 TXT 檔。

將所有的圖片標註好之後，參考 coco128 資料集 的目錄結構，將自己的資料也依照同樣的目錄結構存放，類似像這樣：

```
road
├─── images
│    └─── train
│         ├─── road0.png
│         ├─── road1.png
│         └─── road2.png
└─── labels
     └─── train
          ├─── road0.txt
          ├─── road1.txt
          └─── road2.txt
```

YOLOv5 訓練程式在讀取圖檔所對應的標註檔案時，會將圖檔路徑中最後一個 images 替換為 labels，然後在這個路徑中尋找對應的標註檔案。

這裡只是舉例來說明圖片與標註檔案的目錄結構，實際上不會只用三筆資料就進行模型訓練，依照 YOLOv5 官方網頁 上的建議，每個類別最好收集到 1,500 張影像，並包含 10,000 個標註物件。

資料集 YAML 設定檔

接著為資料集準備一個給訓練程式用的 road.yaml 設定檔，並儲存至 data 目錄，內容如下：

```yaml
# 設定圖檔路徑
path: ./datasets/road    # 資料根目錄
train: images/train    # 訓練用資料集 (相對於 path)
val: images/train    # 驗證用資料集 (相對於 path)
test:                # 測試用資料集 (相對於 path，可省略)

# 物件類別設定
nc: 4    # 類別數量
names: ['traficlight', 'speedlimit', 'crosswalk', 'stop']    # 類別名稱
```

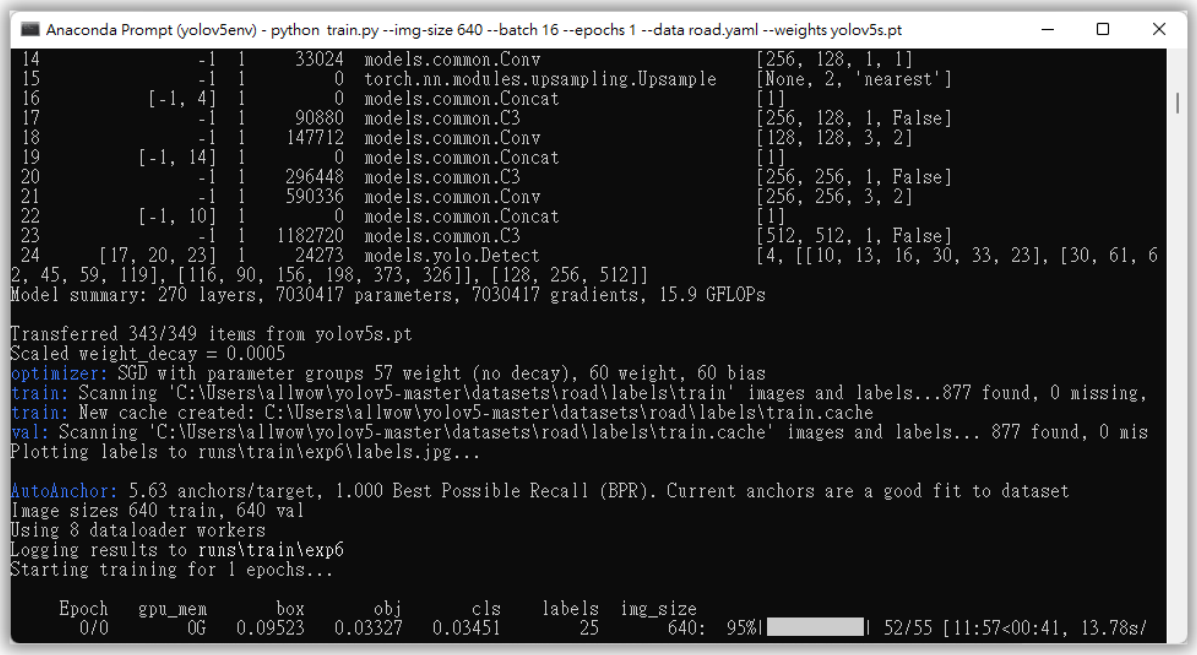
其他資料集的 yaml 設定檔，可以從 YOLOv5 原始碼的 data 目錄中查看。

訓練 YOLOv5 模型

準備好標註資料與設定檔之後，執行 YOLOv5 原始碼中附帶的模型訓練指令稿 (執行時間要靠 GPU，請自行上網搜尋相關技術文，或使用 Google Colab 來訓練)：

```
# 進行 YOLOv5s 模型訓練
(yolov5env) python train.py --img-size 640 --batch 16 --epochs 500 \
--data road.yaml --weights yolov5s.pt
```

- --img-size：可以指定影像的大小，對於偵測比較小的物件時，可以加大影像的大小，預設的影像大小為 640。
- --weights：可用來指定預訓練的模型。



將參數改為 --epochs 1，只訓練一個回合來暫時完成所有的結果，並且檢查以下的輸出結果：

- 訓練完成之後，新的模型與各種訓練過程的資訊都會放在 runs/train/exp 目錄中，其中 result.png 可以查看模型的收斂狀況，如果模型收斂的情況不理想，可能就要考慮調整參數並重新訓練。
- 而在 runs/train/exp/weights 目錄中存放了最佳的模型參數 best.pt 以及訓練到最後的模型參數 last.pt，通常我們會使用 best.pt 這個最佳的模型參數來做後續的預測。
- 若發生 Error，請在 train.py 加入以下程式碼：

```
os.environ["KMP_DUPLICATE_LIB_OK"] = "TRUE"
```

使用 Google Colab 來訓練模型

以下使用 Colab 來訓練模型：

- 請先上傳我們整理好的 yolov5-master.zip 的壓縮檔至自己的 Google Drive，這樣上傳單一個壓縮檔時間才不會花太多時間。
- 登入你的 Colab，使用 Colab 進行解壓縮後，再進行模型訓練。

首先，登入 Goolge Drive，並掛載到 Colab 預設的 content 目錄下：

```
# 登入 Googe Drive
from google.colab import drive
drive.mount('/content/gdrive')    # 此處需要登入 google 帳號
```

進入雲端硬碟目錄：

```
# 進入掛載的雲端硬碟資料夾
cd gdrive/MyDrive/
```

解壓縮 yolov5-master.zip，因為 Colab 的作業系統是 Linux，必須使用 Linux 的指令，而 Colab 在某些 Linux 指令前要加上驚嘆號 ！：

```
# 解壓縮 yolov5-master.zip
!unzip yolov5-master.zip
```

進入 yolov5-master 目錄：

```
# 進入掛載的雲端硬碟目錄
cd yolov5-master
```

使用模型訓練指令稿：

```
# 進行 YOLOv5s 模型訓練
!python train.py --img-size 640 --batch 16 --epochs 500 \
--data road.yaml --weights yolov5s.pt
```

- 以筆者的為例，訓練 500 個 epochs 的時間為 5 個小時。

使用自己的模型進行預測

使用自己的模型進行預測：detect.py 指令稿

訓練出來的 YOLO 模型可在 Anaconda Prompt 中，直接套用 detect.py 進行預測：

```
# 使用自行訓練的 YOLO 模型進行預測
(yolov5env) python detect.py --weight runs/train/exp/weights/best.pt \
--source road0.png --conf-thres 0.5
```

- --conf-thres：設定信心門檻值。偵測的結果同樣會放在 runs/detect/exp 目錄之下。

以下是偵測攝影機物件：

```
# 以自行訓練的 YOLO 模型偵測攝影機物件
(yolov5env) python detect.py --weight runs/train/exp/weights/best.pt \
--source 0
```

以下是拿 Youtube 的影片來測試：

```
# 以自行訓練的 YOLO 模型偵測 Youtube 的影片
(yolov5env) python detect.py --weight runs/train/exp/weights/best.pt \
--source https://www.youtube.com/watch?v=7BwW2dH6ZI0
```

使用自己的模型進行預測：自行撰寫 Python

自行訓練的 YOLOv5 模型，也可以用於自行撰寫的 Python 指令稿中，使用方式與前述標準的 YOLOv5 模型類似：

5-2.py　使用自己的 YOLOv5 模型偵測物件。

```
import torch

# 載入自行訓練的 YOLOv5 模型
model = torch.hub.load('ultralytics/yolov5', 'custom', \
path = 'runs/train/exp/weights/best.pt')

# 影像來源
img = 'eggs-test.jpg'

# 設定 IoU 門檻值
model.iou = 0.3

# 設定信心門檻值
model.conf = 0.5

# 進行物件偵測
results = model(img)

# 顯示結果圖片
results.show()

# 顯示結果摘要
results.print()
```

輸出結果：

```
image 1/1: 400x300 2 trafficlights, 1 crosswalk
Speed: 11.0ms pre-process, 210.9ms inference, 1.0ms NMS per image at
shape (1, 3, 640, 480)
```

附錄

常見的終端機命令

常用的指令	Windows	Linux
列出目前所有的檔案/目錄	dir	ls
變更目錄到指定位置	cd 目錄路徑	cd 目錄路徑
回到到上一層	cd ..	cd ..
中斷程序	Ctrl + C	Ctrl + C