Embedded Vision Intelligent Laboratory

嵌入式智慧影像分析與實境界面
Fall 2021

Instructor：Yen-Lin Chen(陳彥霖), Ph.D.
Professor
Dept. Computer Science and Information Engineering
National Taipei University of Technology

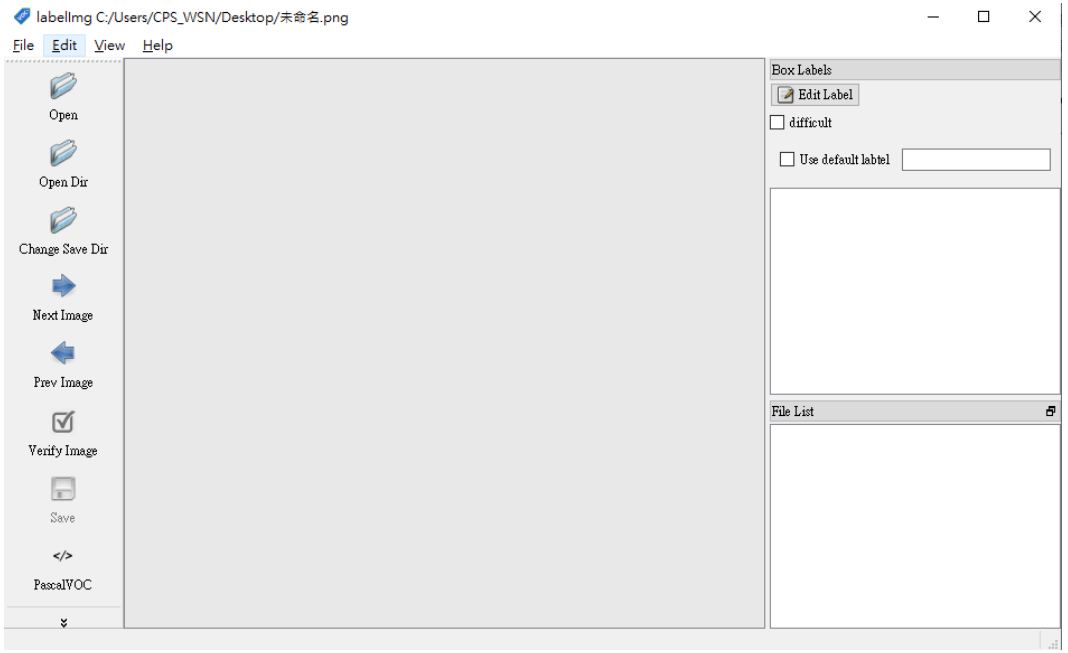
簡介

- 本投影片將介紹：
  1. 使用LabelImg標記工具
  2. Google Colab使用說明
  3. 使用YOLOv4 (訓練模型、執行模型)

使用LabelImge工具

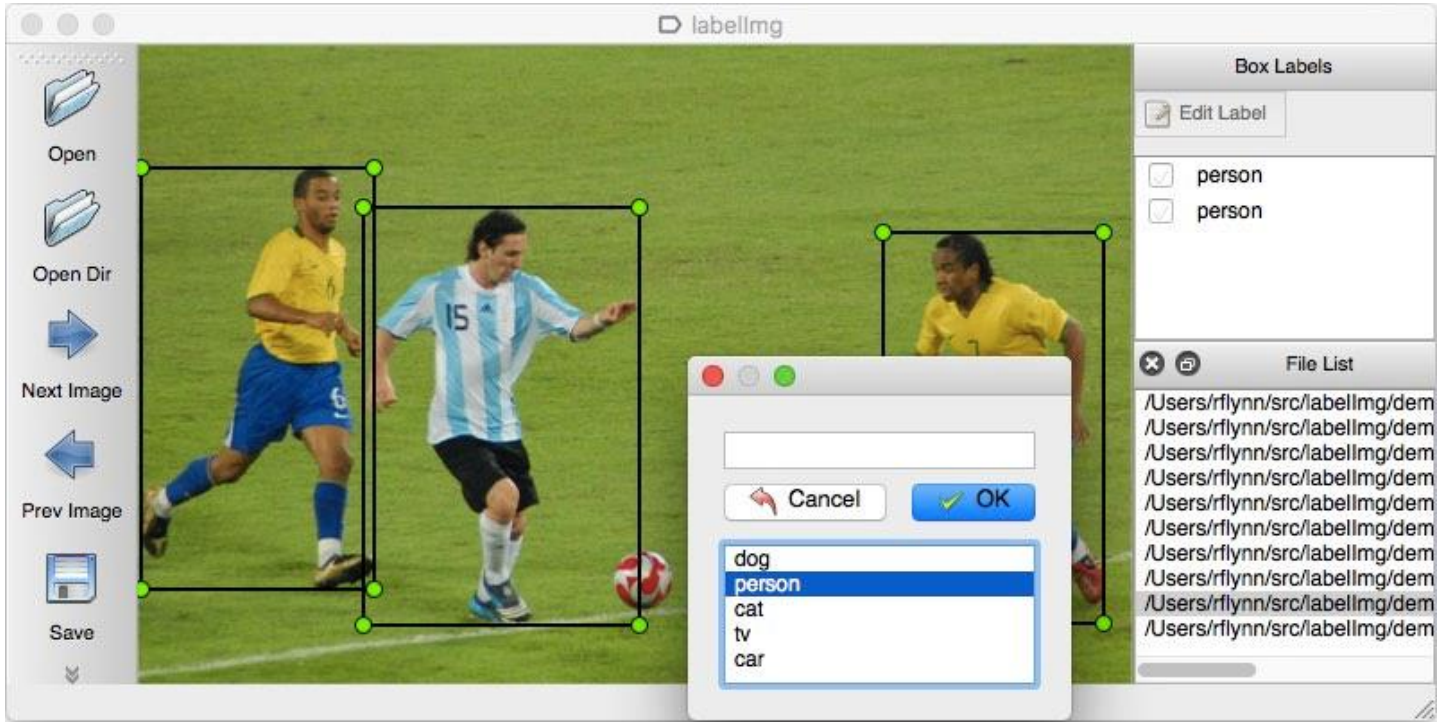
LabelImg

- LabelImg是一款Open Source軟體，專用於影像標記：https://github.com/tzutalin/labelImg
- 到Release下載Windows版，若要編譯需下載Source Code。



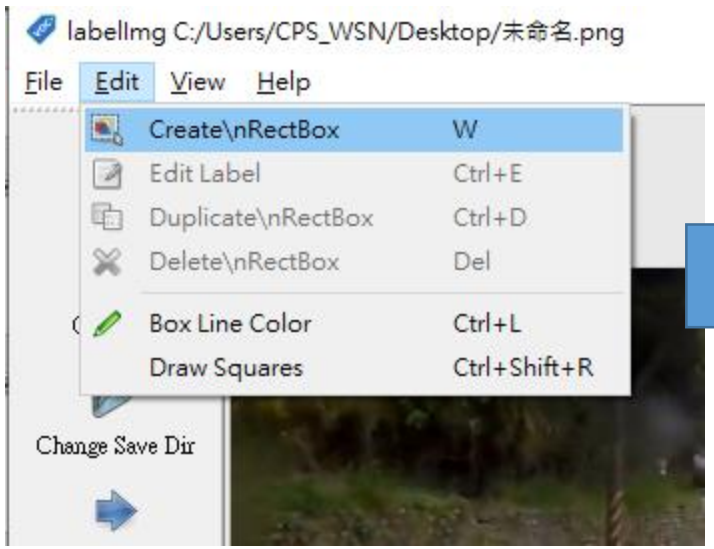
LabelImg

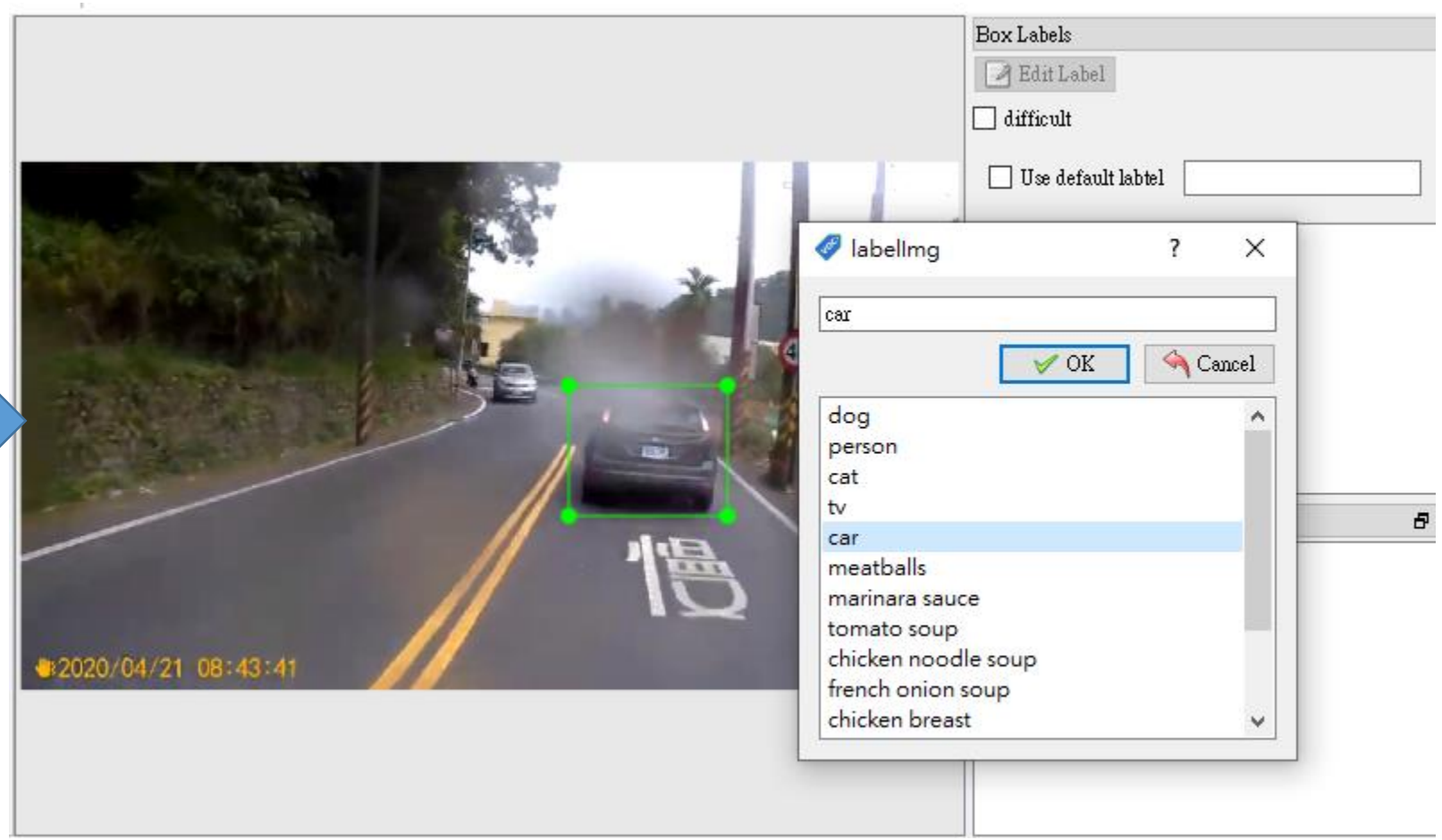
- 產生給機器學習和深度學習使用的樣本
- 支援普遍使用的PASCAL VOC或是YOLO的資料型態



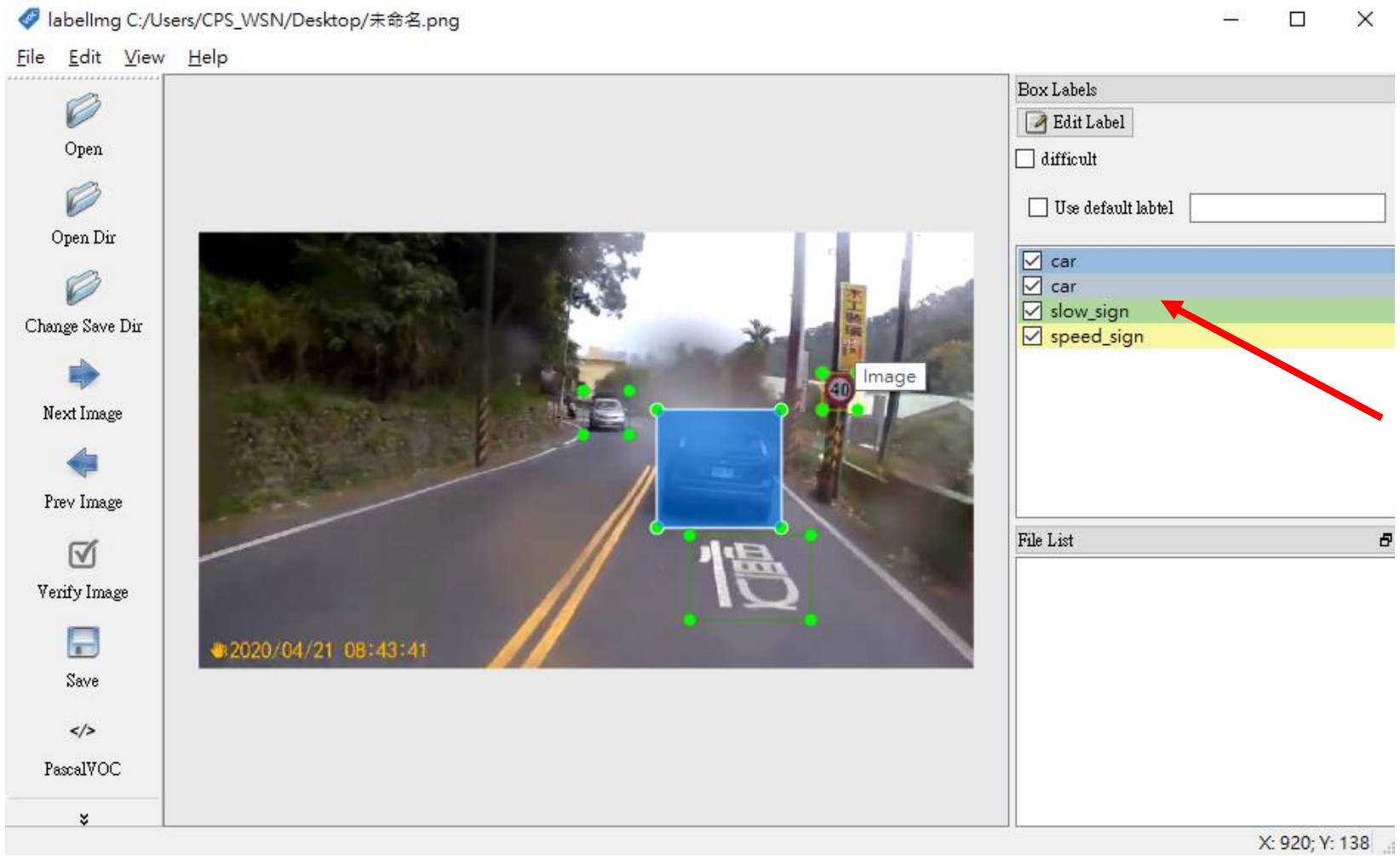
- LabelImg程式示意圖

LabelImg操作


labelImg C:/Users/CPS_WSN/Desktop/未命名.png
File
Edit
View
Help
Create\nRectBox W
Edit Label Ctrl+E
Duplicate\nRectBox Ctrl+D
Delete\nRectBox Del
Box Line Color Ctrl+L
Draw Squares Ctrl+Shift+R
Change Save Dir


Box Labels
Edit Label
difficult
Use default labtel
labelImg
car
OK
Cancel
dog
person
cat
tv
car
meatballs
marinara sauce
tomato soup
chicken noodle soup
french onion soup
chicken breast
2020/04/21 08:43:41

LabelImg操作



已經標過的物件會放在這

LabelImg操作

資料格式可以換成YOLO，pascal_voc儲存的副檔名為xml，YOLO則為txt

LabelImg操作

- 若存成YOLO用的.txt檔，格為：
  1. 物件的類別編號
  2. bndBox的中心座標與圖片寬高的比值，是bndBox正規化後x的中心座標(除以圖片寬度)
  3. bndBox的中心座標與圖片高度的比值，是bndBox正規化後y的中心座標(除以圖片高度)
  4. bndBox的寬度與輸入圖像寬度的比值，是bndBox歸一化後的寬度座標(除以圖片寬度)
  5. bndBox的高度與輸入圖像高度的比值，是bndBox歸一化後的高度座標(除以圖片高度)

```
4 0.671089 0.541463 0.160110 0.269919
4 0.525618 0.413008 0.059469 0.100813
15 0.711345 0.791057 0.156450 0.193496
16 0.827081 0.364228 0.045746 0.084553
```

- 存成PascalVOC的.xml檔，格式為：

```
<?xml version="1.0"?>
- <annotation>
    <folder>Desktop</folder>
    <filename>未命名.png</filename>
    <path>C:/Users/CPS_WSN/Desktop/未命名.png</path>
  + <source>
  - <size>
      <width>1093</width>
      <height>615</height>
      <depth>3</depth>
    </size>
    <segmented>0</segmented>
  + <object>
  + <object>
  + <object>
  - <object>
      <name>speed_sign</name>
      <pose>Unspecified</pose>
      <truncated>0</truncated>
      <difficult>0</difficult>
    - <bndbox>
        <xmin>879</xmin>
        <ymin>198</ymin>
        <xmax>929</xmax>
        <ymax>250</ymax>
      </bndbox>
    </object>
</annotation>
```

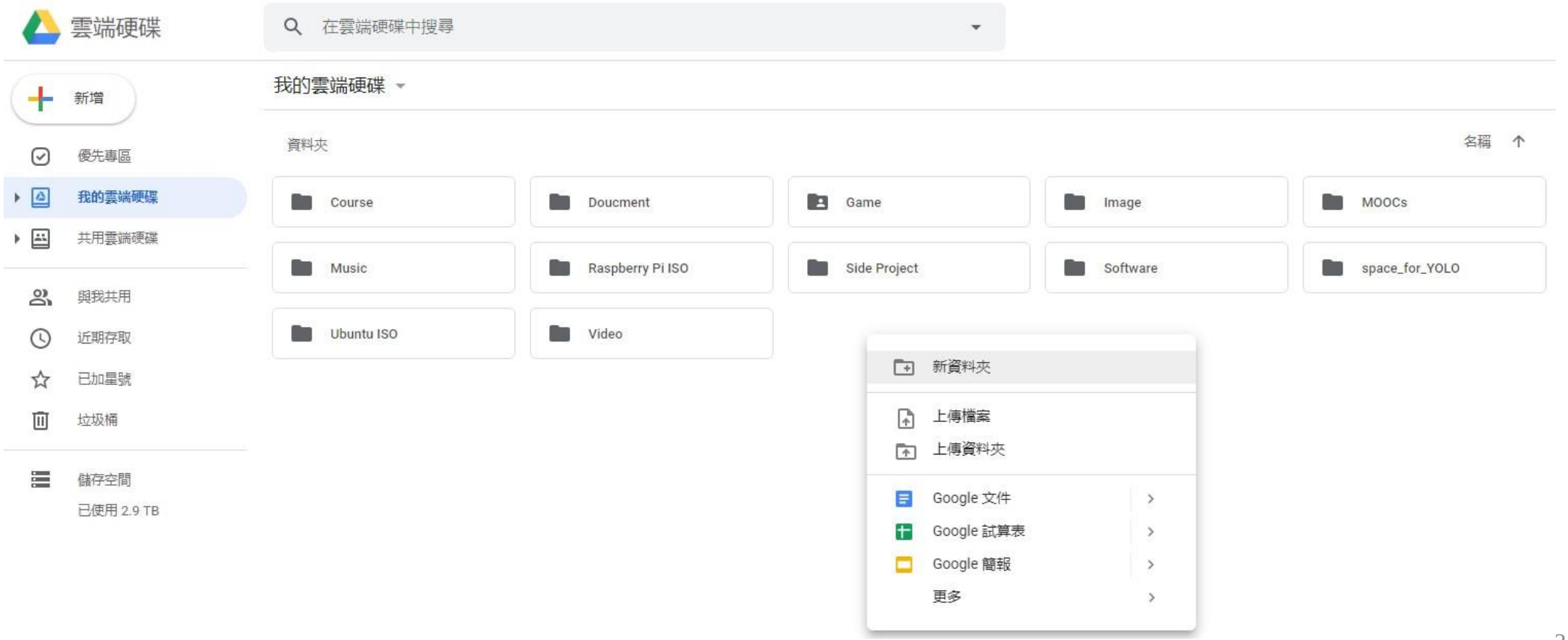
使用GoogleColab雲端運算平台

使用Google Colab雲端運算平台

- Google Colaboratory (簡稱為Google Colab) 可讓你在瀏覽器上撰寫及執行 Python，且具備下列優點：
  - 不必進行任何設定
  - 免費使用 GPU
  - 輕鬆共用
- Colab 筆記本可讓你在單一文件中結合可執行的程式碼和 RTF 格式，並附帶圖片、HTML、LaTeX 等其他格式的內容。你建立的 Colab 筆記本會儲存到你的 Google 雲端硬碟帳戶中。你可以輕鬆將 Colab 筆記本與同事或朋友共用，讓他們在筆記本上加上註解，或甚至進行編輯。

使用Google Colab雲端運算平台

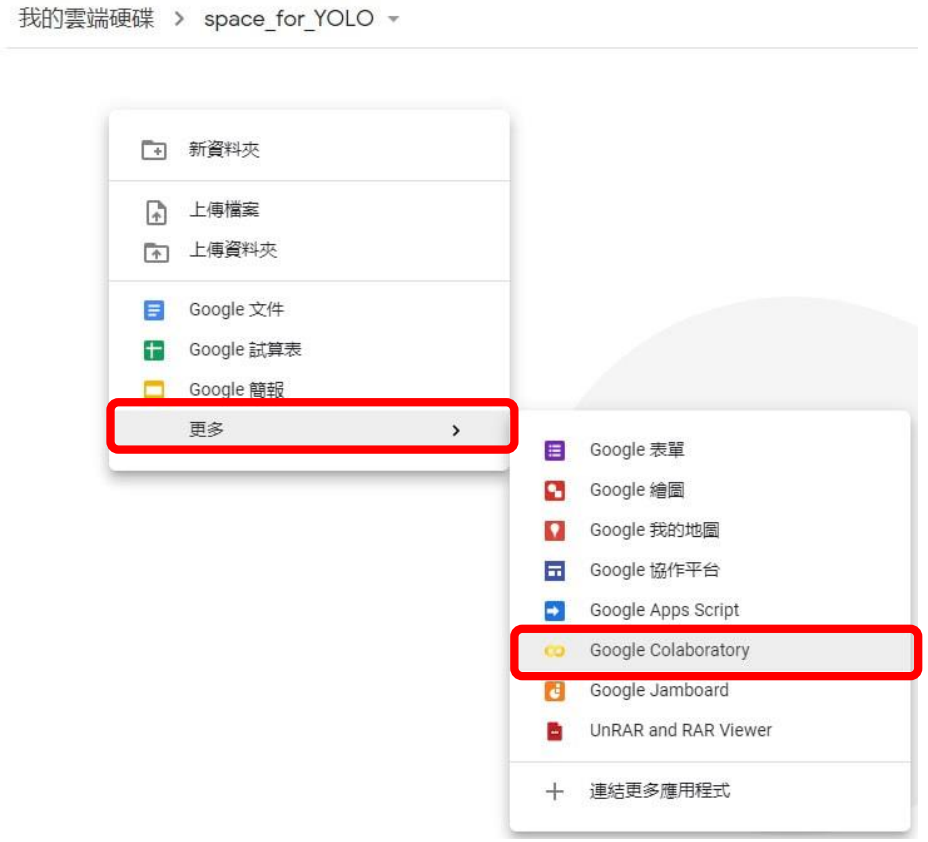
- 先在個人的 Google Drive 上新增一個資料夾。



ref:https://jason-chen-1992.weebly.com/home/-google-colab-yolov4

使用Google Colab雲端運算平台

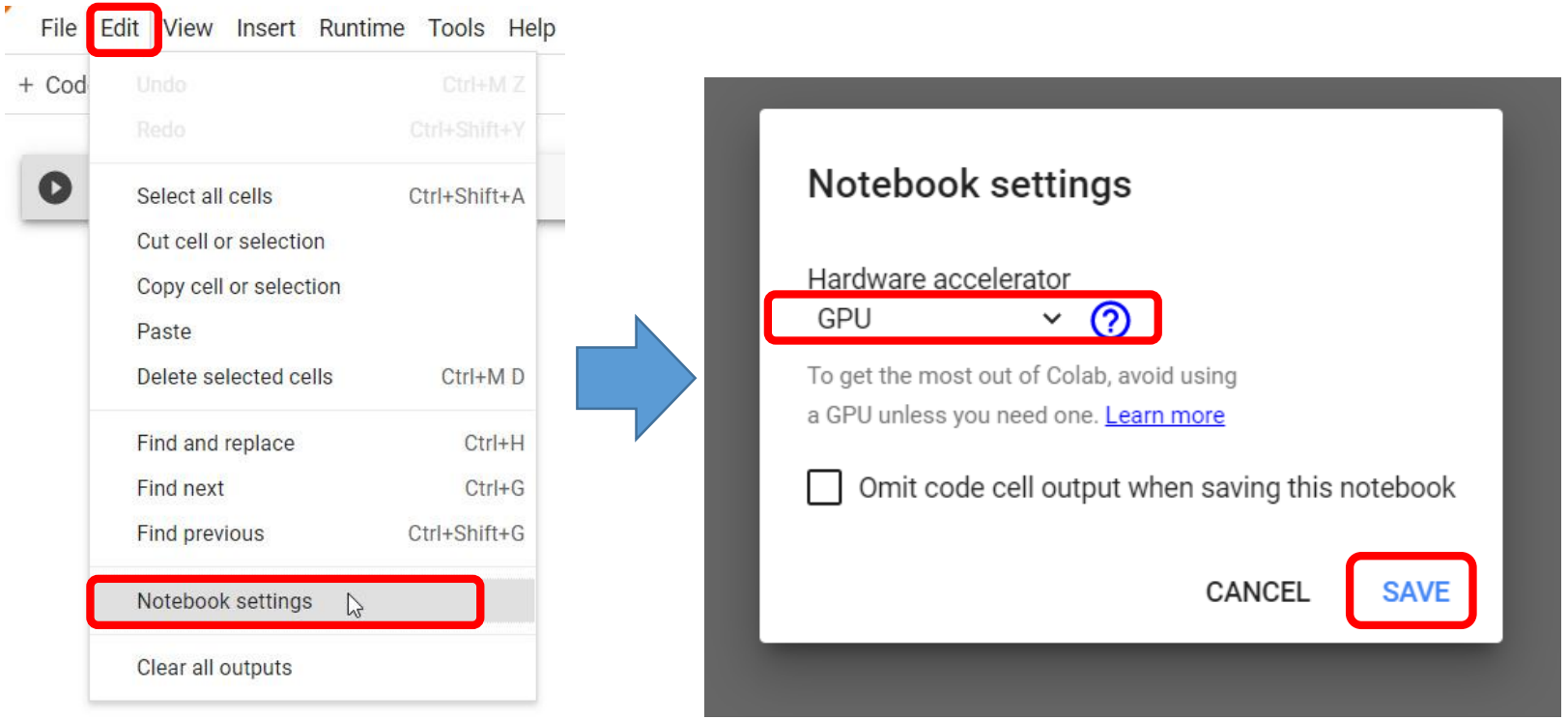
- 進入該資料夾後新增一個 Colab 文件



ref:https://jason-chen-1992.weebly.com/home/-google-colab-yolov4

使用Google Colab雲端運算平台

- 開啟 Google Colab 然後啟用免費的GPU
- 網址：https://colab.research.google.com/



ref:https://jason-chen-1992.weebly.com/home/-google-colab-yolov4

使用Google Colab雲端運算平台

- 首先我們先 Copy 一份 darknet 到你的Colab空間

ref:https://jason-chen-1992.weebly.com/home/-google-colab-yolov4

使用Google Colab雲端運算平台

- 下載完成後，我們需要修改 Makefile 裡面的四個參數，分別是：
  - GPU=0 要改成 GPU=1 (開啟GPU加速)
  - OPENCV=0 要改成 OPENCV=1 (用來讀取影像、影片、畫框等功能)
  - CUDNN=0 要改成 CUDNN=1 (用於加速tensorflow、pytorch等深度學習框架)
  - CUDNN_HALF=0 要改成 CUDNN_HALF=1 (建構tensor核心，加速偵測物件)

備註：若以上指令無法更改可自colab左邊目錄直接打開 /content/darknet/Makefile檔案，進行修改。

ref:https://jason-chen-1992.weebly.com/home/-google-colab-yolov4

使用Google Colab雲端運算平台

- Makefile 修改完成後，我們就 make 指令來生成 darknet 這個深度學習引擎了。

- 在 make 完成後，其實 darknet 這套深度學習引擎就已經安裝完畢了，接下來我們會想要測試他是否能正常work，所以這時候我們就先去下載一些已經預先 train 好的 weights 檔做為測試用。

ref:https://jason-chen-1992.weebly.com/home/-google-colab-yolov4

使用Google Colab雲端運算平台

Step 4: Download pretrained YOLOv3 and YOLOv4 weights

YOLOv3 and YOLOv4 has been trained already on the coco dataset which has 80 classes that it can predict. We will grab these pretrained weights so that we can run YOLOv3 and YOLOv4 on these pretrained classes and get detections.

```
[ ] download_from_official = False

    if download_from_official:
      # download weights form official
      !wget https://pjreddie.com/media/files/yolov3.weights
      !wget https://pjreddie.com/media/files/darknet53.conv.74
      !wget https://github.com/AlexeyAB/darknet/releases/download/darknet_yolo_v3_optimal/yolov4.weights
      !wget https://github.com/AlexeyAB/darknet/releases/download/darknet_yolo_v3_optimal/yolov4.conv.137
    else:
      # download weights form Jason Chen's google drive (should be faster)
      !gdown https://drive.google.com/uc?id=1hSaT4Yc19atZZulW3Q3BUDFIohfGteXN
      !gdown https://drive.google.com/uc?id=1XpMMC_eUfHKaIpfmxZa71IyocMW6ssCb
      !gdown https://drive.google.com/uc?id=1vOlvou7Pgv36l-ahej5IIfYdb_9nmy0A
      !gdown https://drive.google.com/uc?id=1Zl3rh0ZOPVj4DPZdae8WqqZU0NLX7Ncp
```

```
Downloading...
From: https://drive.google.com/uc?id=1hSaT4Yc19atZZulW3Q3BUDFIohfGteXN
To: /content/darknet/yolov3.weights
248MB [00:03, 62.3MB/s]
Downloading...
From: https://drive.google.com/uc?id=1XpMMC_eUfHKaIpfmxZa71IyocMW6ssCb
To: /content/darknet/darknet53.conv.74
162MB [00:01, 100MB/s]
Downloading...
From: https://drive.google.com/uc?id=1vOlvou7Pgv36l-ahej5IIfYdb_9nmy0A
To: /content/darknet/yolov4.weights
258MB [00:02, 101MB/s]
Downloading...
From: https://drive.google.com/uc?id=1Zl3rh0ZOPVj4DPZdae8WqqZU0NLX7Ncp
To: /content/darknet/yolov4.conv.137
170MB [00:00, 205MB/s]
```

ref:https://jason-chen-1992.weebly.com/home/-google-colab-yolov4

使用Google Colab雲端運算平台

- 在 weights 下載好之後，我們就可以使用：
  !./darknet detect <path of .cfg file> <path of .weights file> <path of picture>
  這個指令進行辨識，辨識的結果會以 "predictions.jpg" 存在跟 darknet 同一個目錄底下，為了可以直接 show 在 notebook 上面，我們可以多寫一個 imshow 的 function 來實現。

ref:https://jason-chen-1992.weebly.com/home/-google-colab-yolov4

使用Google Colab雲端運算平台

Step 5 : Run Object Detection with Darknet and YOLOv3/v4

Define the show image function

```
[ ]  def imShow(path):
       import cv2
       import matplotlib.pyplot as plt
       %matplotlib inline

       image = cv2.imread(path)
       height, width = image.shape[:2]
       resized_image = cv2.resize(image,(3*width, 3*height), interpolation = cv2.INTER_CUBIC)

       fig = plt.gcf()
       fig.set_size_inches(18, 10)
       plt.axis("off")
       plt.imshow(cv2.cvtColor(resized_image, cv2.COLOR_BGR2RGB))
       plt.show()
```

ref:https://jason-chen-1992.weebly.com/home/-google-colab-yolov4

使用Google Colab雲端運算平台

▾ >>> 5-1. Run detection with YOLOv3

```
[ ]  # run darknet detection
     !./darknet detect cfg/yolov3.cfg yolov3.weights data/person.jpg

     # show result
     imShow('predictions.jpg')
```

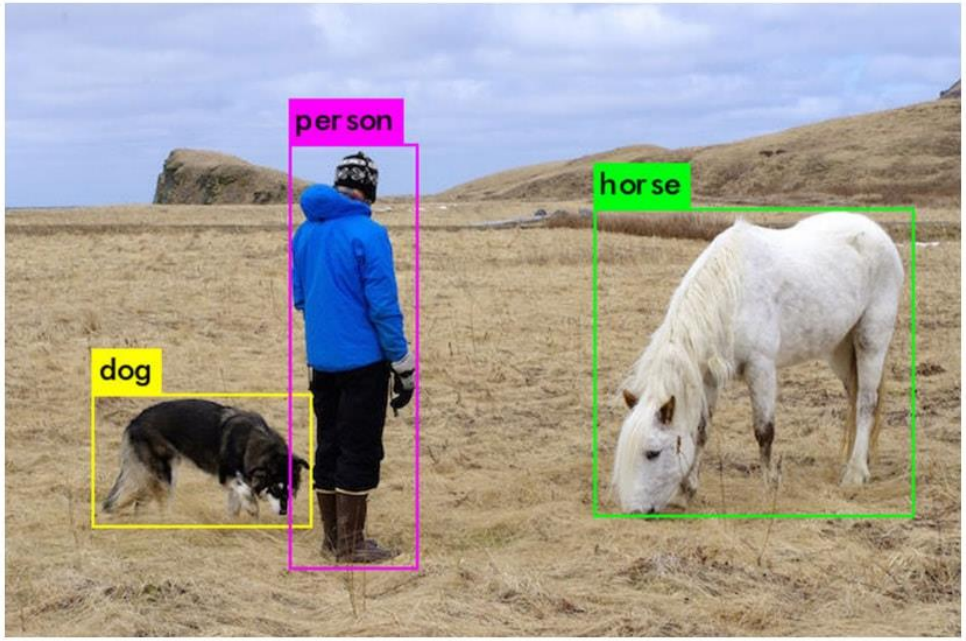
▾ >>> 5-2. Run detection with YOLOv4

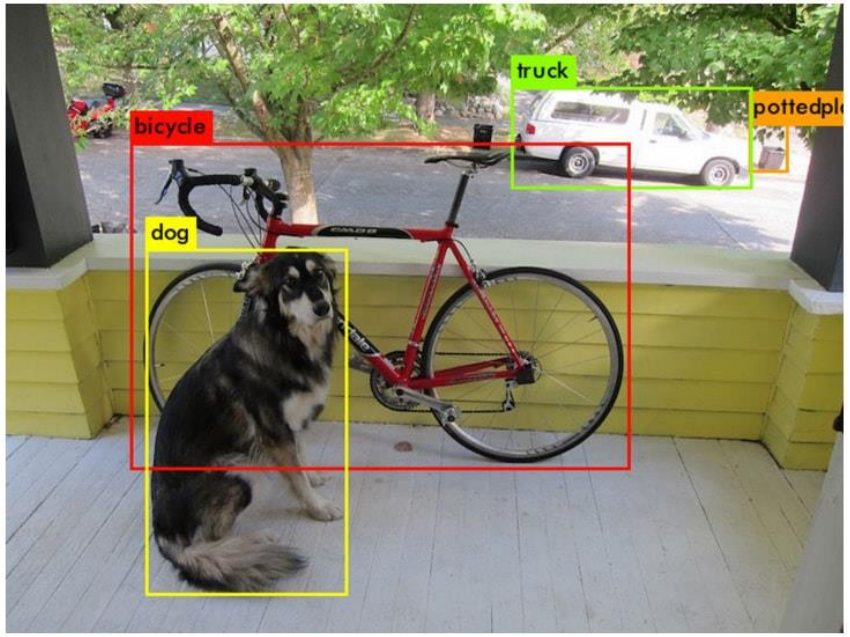
```
[ ]  # run darknet detection
     !./darknet detect cfg/yolov4.cfg yolov4.weights data/dog.jpg

     # show result
     imShow('predictions.jpg')
```

ref:https://jason-chen-1992.weebly.com/home/-google-colab-yolov4

使用Google Colab雲端運算平台

- 如果你的 darknet 能夠正常 work 的話，你應該能夠看到下方這兩張圖的辨識結果。





ref:https://jason-chen-1992.weebly.com/home/-google-colab-yolov4

使用Google Colab雲端運算平台

- Google Colab連結自己的Google雲端硬碟空間
  - 由於Colab的空間是一個臨時的工作空間，在建立當初是沒有任何檔案的，此時可將Colab連結自己的Google雲端硬碟空間，以利將事先準備好的obj.names、obj.data等檔案複製到Colab的工作空間，或是將訓練好的.weight權重檔案複製回雲端。

```
from google.colab import drive
drive.mount('/content/drive')

#將google雲端硬碟路徑簡寫為mydrive
!ln -s /content/drive/My\ Drive/ /mydrive
!ls /mydrive
```

ref:https://blog.csdn.net/longlong068/article/details/105791941

使用Google Colab雲端運算平台

- 準備好訓練YOLOv4模型所需檔案，並上傳到雲端：
  1. obj.zip (包含資料集影像與同名的.txt座標文件(使用LabelImg工具取得))
  2. obj.names (自己資料集的類別名稱)
  3. obj.data (相關路徑及各種參數設定)
  4. yolov4_custom.cfg (自己根據資料及類別數量修改)
  5. generate_train.py (用來產生train.txt)

ref:https://blog.csdn.net/longlong068/article/details/105791941

使用Google Colab雲端運算平台

- 上傳obj.zip到雲端硬碟：
  - 通過LabelImg或Yolo_mark等標記工具生成專案所需的與圖片同名的.txt檔，將原圖片與同名.txt文件放在同一文件obj下，將資料夾壓縮為obj.zip並上傳到Google雲端硬碟的yolov4資料夾下。obj.zip包含的檔案如下所示。

名稱	修改日期	類型	大小
1.jpg	2021/5/5 下午 07:53	JPG 檔案	232 KB
1.txt	2021/5/10 下午 03:03	文字文件	1 KB
2.jpg	2021/5/5 下午 07:53	JPG 檔案	233 KB
2.txt	2021/5/5 下午 08:53	文字文件	1 KB
10.jpg	2021/5/5 下午 07:53	JPG 檔案	240 KB
10.txt	2021/5/10 下午 01:54	文字文件	1 KB
11.jpg	2021/5/5 下午 07:53	JPG 檔案	241 KB
11.txt	2021/5/5 下午 09:00	文字文件	1 KB
12.jpg	2021/5/5 下午 07:53	JPG 檔案	241 KB
12.txt	2021/5/5 下午 09:04	文字文件	1 KB
13.jpg	2021/5/5 下午 07:53	JPG 檔案	240 KB
13.txt	2021/5/10 下午 01:56	文字文件	1 KB
14.jpg	2021/5/5 下午 07:53	JPG 檔案	241 KB
14.txt	2021/5/10 下午 02:05	文字文件	1 KB
15.jpg	2021/5/5 下午 07:53	JPG 檔案	240 KB
15.txt	2021/5/10 下午 02:20	文字文件	1 KB
16.jpg	2021/5/5 下午 07:53	JPG 檔案	240 KB
16.txt	2021/5/10 下午 03:03	文字文件	1 KB
17.jpg	2021/5/5 下午 07:53	JPG 檔案	240 KB
17.txt	2021/5/10 下午 03:03	文字文件	1 KB

ref:https://blog.csdn.net/longlong068/article/details/105791941

使用Google Colab雲端運算平台

- 上傳obj.data和obj.names到雲端：
  - obj.data 根據自己資料集的類別個數進行修改而得到。
  - obj.names中每行寫入自己資料集的類別名稱。

```
1 classes= 1
2 train  = data/train.txt
3 valid  = data/train.txt
4 names = data/obj.names
5 backup = backup/
6
7
```

obj.data

```
1 fish
2
```

obj.names

ref:https://blog.csdn.net/longlong068/article/details/105791941

使用Google Colab雲端運算平台

- 上傳yolov4_custom.cfg到雲端：
  - yolov4_custom.cfg在darknet/cfg/yolov4-custom.cfg的基礎上，根據自己資料集的類別個數進行修改得到。
  - batch = 64(每個batch學習採用多少的樣本資料)
  - subdivisions =32(在一個batch學習中，要拆成幾個個mini-batch)
  - max_batches = 2000(模型最大的iteration次數)
  - Steps 配合scales調整learning rate，右圖為在max_batches的80%時將原本learning rate * 0.1，在90%時將learning rate再 * 0.1

```
1  [net]
2  # Testing
3  #batch=1
4  #subdivisions=1
5  # Training
6  batch=64
7  subdivisions=32
8  width=608
9  height=608
10 channels=3
11 momentum=0.949
12 decay=0.0005
13 angle=0
14 saturation = 1.5
15 exposure = 1.5
16 hue=.1
17
18 learning_rate=0.001
19 burn_in=1000
20 max_batches = 2000
21 policy=steps
22 steps=1600,1800
23 scales=.1,.1
24
```

(可依據自己專案情況決定增加or減少訓練迭代次數)

ref:https://blog.csdn.net/longlong068/article/details/105791941

使用Google Colab雲端運算平台

- filters=(5+classes)x3 (如資料集種類個數classes為1 則filters=18)
- classes =6(分類個數)
- 共要修改三個部分，如下圖(預設)所示：

```
959  [convolutional]
960  size=1
961  stride=1
962  pad=1
963  filters=18
964  activation=linear
965
966
967  [yolo]
968  mask = 0,1,2
969  anchors = 12, 16, 19, 36, 40
970  classes=1
971  num=9
972  jitter=.3
973  ignore_thresh = .7
974  truth_thresh = 1
```

```
1047  [convolutional]
1048  size=1
1049  stride=1
1050  pad=1
1051  filters=18
1052  activation=linear
1053
1054
1055  [yolo]
1056  mask = 3,4,5
1057  anchors = 12, 16, 19, 36,
1058  classes=1
1059  num=9
1060  jitter=.3
1061  ignore_thresh = .7
1062  truth_thresh = 1
```

```
1135  [convolutional]
1136  size=1
1137  stride=1
1138  pad=1
1139  filters=18
1140  activation=linear
1141
1142
1143  [yolo]
1144  mask = 6,7,8
1145  anchors = 12, 16, 19, 36, 40
1146  classes=1
1147  num=9
1148  jitter=.3
1149  ignore_thresh = .7
1150  truth_thresh = 1
```

ref:https://blog.csdn.net/longlong068/article/details/105791941

使用Google Colab雲端運算平台

- 上傳generate_train.py文件到雲端：
  - 此yolov4專案訓練自己的資料集需要train.txt
  - train.txt檔可透過generate_train.py產生(需要創建generate_train.py，並上傳到雲端，以便後續複製到Colab中並運行)
  - generate_train.py文件如下：

```python
import os

image_files = []
os.chdir(os.path.join("data", "obj"))
for filename in os.listdir(os.getcwd()):
    if filename.endswith(".jpg"):
        image_files.append("data/obj/" + filename)
os.chdir("..")
with open("train.txt", "w") as outfile:
    for image in image_files:
        outfile.write(image)
        outfile.write("\n")
    outfile.close()
os.chdir("..")
```

執行此python檔會根據obj.zip內檔案內容自動產生所有路徑的train.txt檔案

ref:https://blog.csdn.net/longlong068/article/details/105791941

使用Google Colab雲端運算平台

- 將訓練所需檔案上傳至雲端後，依照右側指令依序copy至Colab工作空間，全部準備完成後就可以開始訓練自己的YOLOv4模型了。

```
[ ]  #確保當前路徑在darknet下
     !ls
     #將資料及圖片及標記的的txt文件的壓縮檔上傳
     !cp  /mydrive/yolo_fish/obj.zip  ../
     #解壓縮到data文件下
     !unzip  ../obj.zip  -d  data/obj
```

```
[ ]  #從google  drive上傳根據自己資料集修改的.cfg文件
     !cp  /mydrive/yolo_fish/yolov4_custom.cfg  ./cfg
```

```
[ ]  #從google  drive上傳自己資料集的obj.data  obj.names
     0
     !cp  /mydrive/yolo_fish/obj.data    ./data
     !cp  /mydrive/yolo_fish/obj.name    ./data
```

```
[ ]  #上傳generate_train.py
     !cp  /mydrive/yolo_fish/generate_train.py  ./
```

```
[ ]  #運行在data文件下生成train.txt
     !python  generate_train.py
```

ref:https://blog.csdn.net/longlong068/article/details/105791941

使用Google Colab雲端運算平台

在train自己的model時，須有一個預訓練權重檔案當基底

```
!./darknet detector train data/obj.data cfg/yolov4_custom.cfg yolov4.conv.137 -dont_show
```

進行訓練　obj.data位置　.cfg位置　預訓練權重檔案　訓練過程不要顯示任何視窗

```
!./darknet detector train data/obj.data cfg/yolov4_custom.cfg backup/yolov4_custom_last.weights -dont_show
```

ref:https://blog.csdn.net/longlong068/article/details/105791941

使用Google Colab雲端運算平台

```
#備份.weight檔案
!cp  ./backup/yolov4_custom_last.weights  /mydrive/yolo_fish
```

```
#copy影片至darknet資料夾
!cp  /mydrive/yolo_fish/fish4.mp4  ./
```

```
#進行影片辨識
!./darknet  detector  demo  data/obj.data  cfg/yolov4_custom.cfg  backup/yolov4_custom_last.weights  -dont_show  fish4.mp4  -out_filename  fish_out.mp4
```

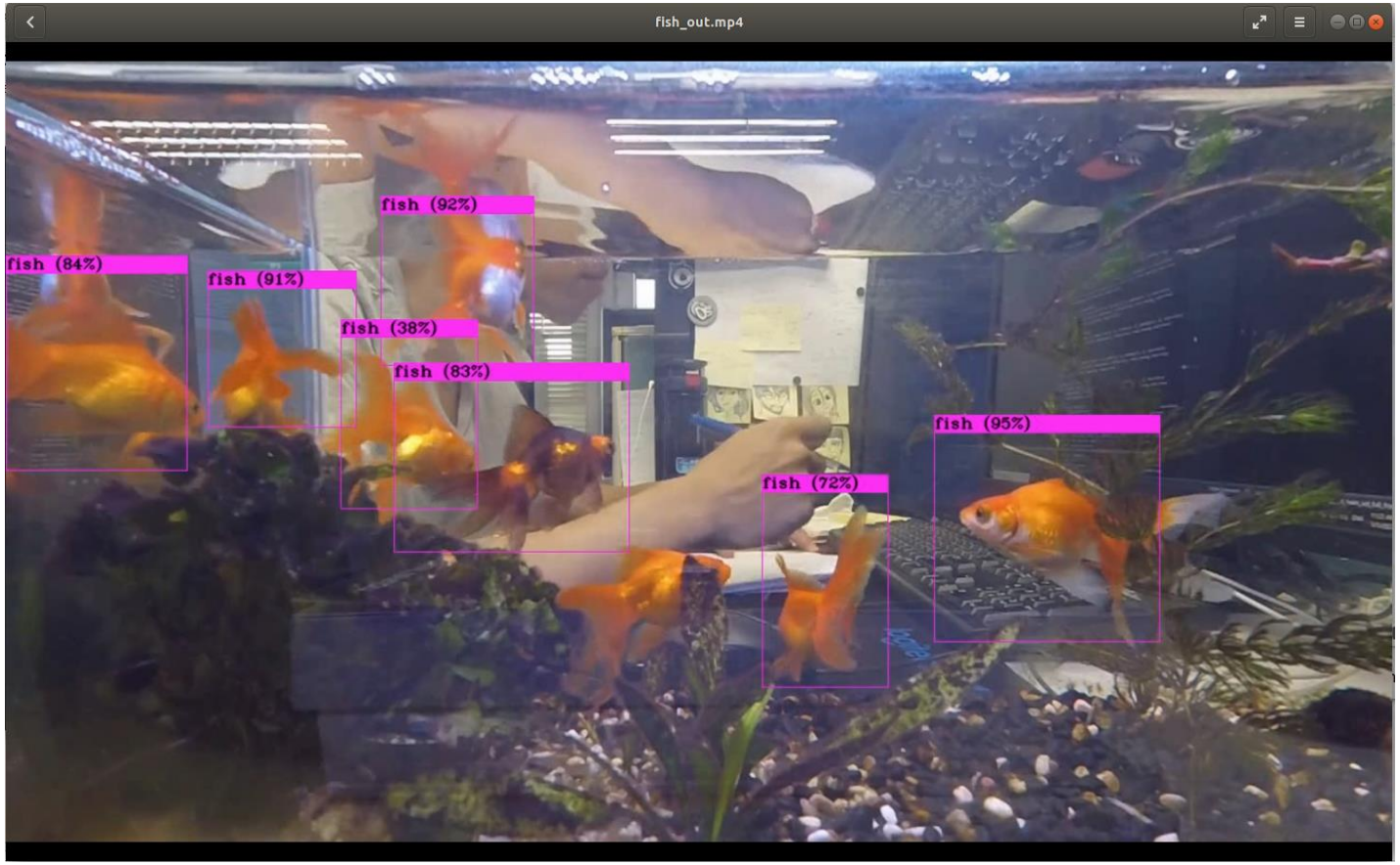
進行辨識　obj.data位置　.cfg位置　訓練好的權重檔案　過程不要顯示任何視窗　src filename　output filename

```
#copy影片至雲端
!cp  ./fish_out.mp4  /mydrive/yolo_fish
```

ref:https://blog.csdn.net/longlong068/article/details/105791941

使用Google Colab雲端運算平台

- YOLOv4深度學習模型辨識影片範例(以魚群辨識為例)：



ref:https://blog.csdn.net/longlong068/article/details/105791941

使用Google Colab雲端運算平台

- 使用Google Colab需要注意的事項：
  - 由於Google Colab提供的是免費使用的運算平台，資源有限，若是訓練時閒置過久會可能會主動中斷連結，這時候已執行的結果都會被清空。
  - Colab 最長的執行時間為 12 小時，但訓練 YOLO 可能長達數天以上，因此可參考下方參考資料建立一個專用的 Colab disk 空間，讓每次重新執行 Colab 時，不會遺失訓練結果，更可以很快設定好訓練環境並從上次中斷的地方繼續訓練。請參考：https://makerpro.cc/2020/02/use-google-colab-to-train-yolo/

ref:https://makerpro.cc/2020/02/use-google-colab-to-train-yolo/

參考資料

- LabelImg影像標記程式： https://github.com/tzutalin/labelImg
- YOLOv4 darknet： https://github.com/AlexeyAB/darknet
- Google Colab上執行Yolo教學1：
- https://jason-chen-1992.weebly.com/home/-google-colab-yolov4
- Google Colab上執行Yolo教學2：
- https://blog.csdn.net/longlong068/article/details/105791941
- Google Colab上執行Yolo教學3：
- https://makerpro.cc/2020/02/use-google-colab-to-train-yolo/
- YOLOv4 darknet作者提供的colab教學：
- https://colab.research.google.com/drive/12QusaaRj_lUwCGDvQNfICpa7kA7_a2dE

Project 2

影像標記與訓練

專案實作

- 專案項目:影像標記與訓練。
- 專案要求:
  - 以組為單位進行影像label標記與訓練。
  - 各組自行上youtube搜尋「行車紀錄器」相關影片
  - 使用opencv將影片切割成frames(可自行決定是否進行down sample)
  - 使用LabelImg標記車子的label
  - 在colab上使用yolov4進行訓練

小組報告格式規定

- 專案情境
  - 小組所討論出來的議題，並簡明扼要描述議題的情境。
- 定義問題
  - 將議題中的問題定義出來，並收斂問題方向。
- 方案構思
  - 簡單描述如何解決定義好的問題，並預計使用的技術。
- 解決方法
  - 說明實際上如何完成此議題的方案構思。
- 成果影片(僅小組報告)
  - 以影片方式呈現，並上傳至youtube後，將連結遷入至小組報告最後一頁。
- 分工
  - 說明小組成員分工內容與比例。

個人報告內容

- 個人報告內容須要有以下內容：
  - 你一開始所提出的議題是?你的議題是否有被選為小組議題候選?
  - 你在小組議題中，提出了那些問題與解決方案?是否有被小組接受?
    - 如個人所提出的方案沒被接受，是因為那些原因?
    - 為了這個議題，你去找了那些資料?你是如何分析找到的資料?
    - 其他小組成員所提出的提議有哪些?而你對於其他人的提議意見如何?
  - 在小組決定小組議題過程中，你對於小組最後提出的議題討論是否能接受?接受理由為何?不接受理由為何?
  - 你是否能接受最後的議題與方案?如接受請說明接受與否的理由?
  - 本次專案個人的心得
  - 本次你認為小組成員的貢獻比例及理由

專案繳交規則

- 小組報告繳交期限:110/10/29 23:59(以I學園上傳時間為基準)
- 個人報告繳交期限:110/10/29 23:59(以I學園上傳時間為基準)
- 補交規則
  - 超過正常繳交期限兩周內成績**打8折**
  - 超過兩周後不再接受補交