

# 多媒體技術與應用

## Spring 2022

Instructor : Yen-Lin Chen(陳彥霖), Ph.D.

Professor

Dept. Computer Science and Information Engineering

National Taipei University of Technology

# Project 2

YOLOv4 影像訓練實作

# 簡介

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

# 使用 GoogleColab 雲端運算平台

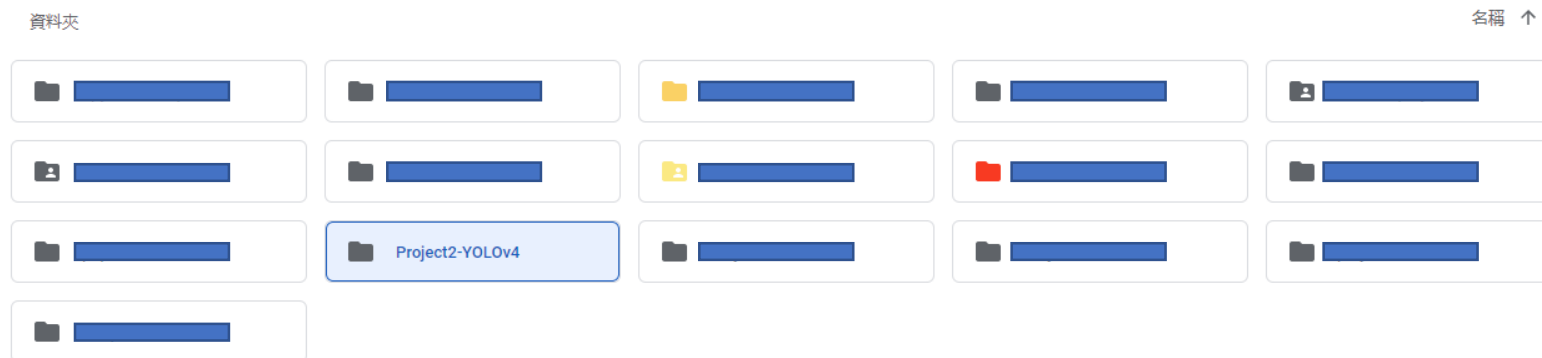
# 使用 Google Colab 雲端運算平台

- 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。

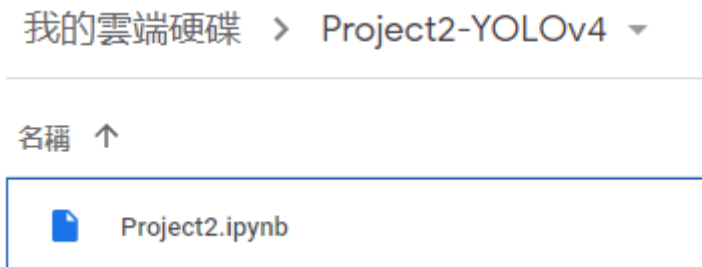


# 使用 Google Colab 雲端運算平台

- 首先在雲端硬碟新增”Project2-YOLOv4”資料夾來存放本次專案的所有檔案

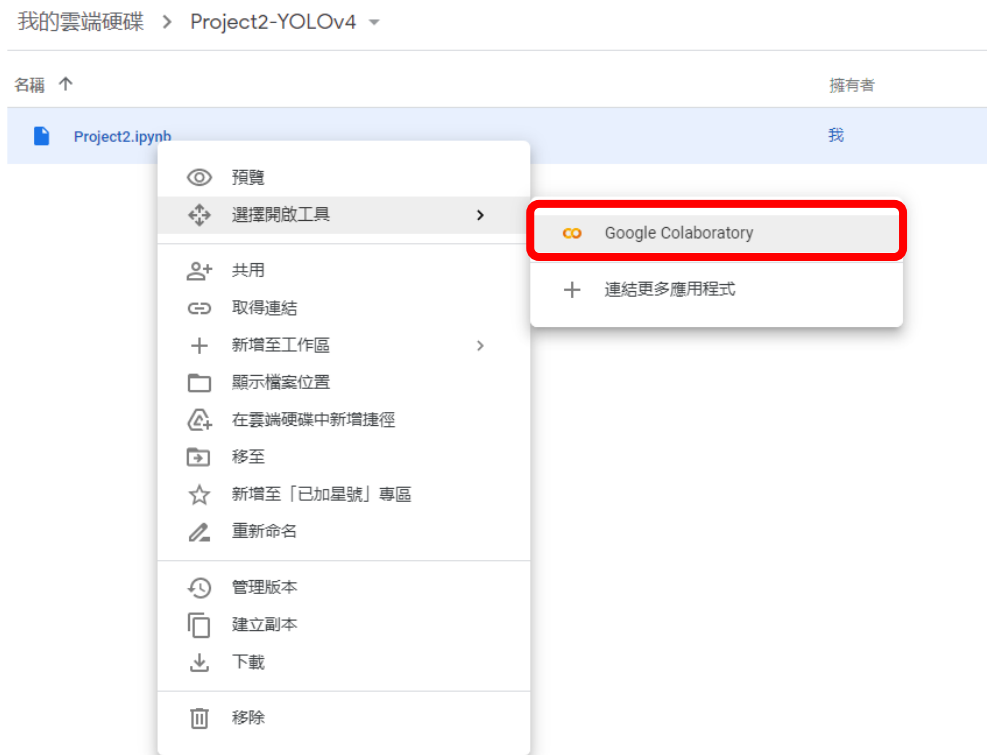


- 在資料夾中上傳助教提供的Project2.ipynb檔



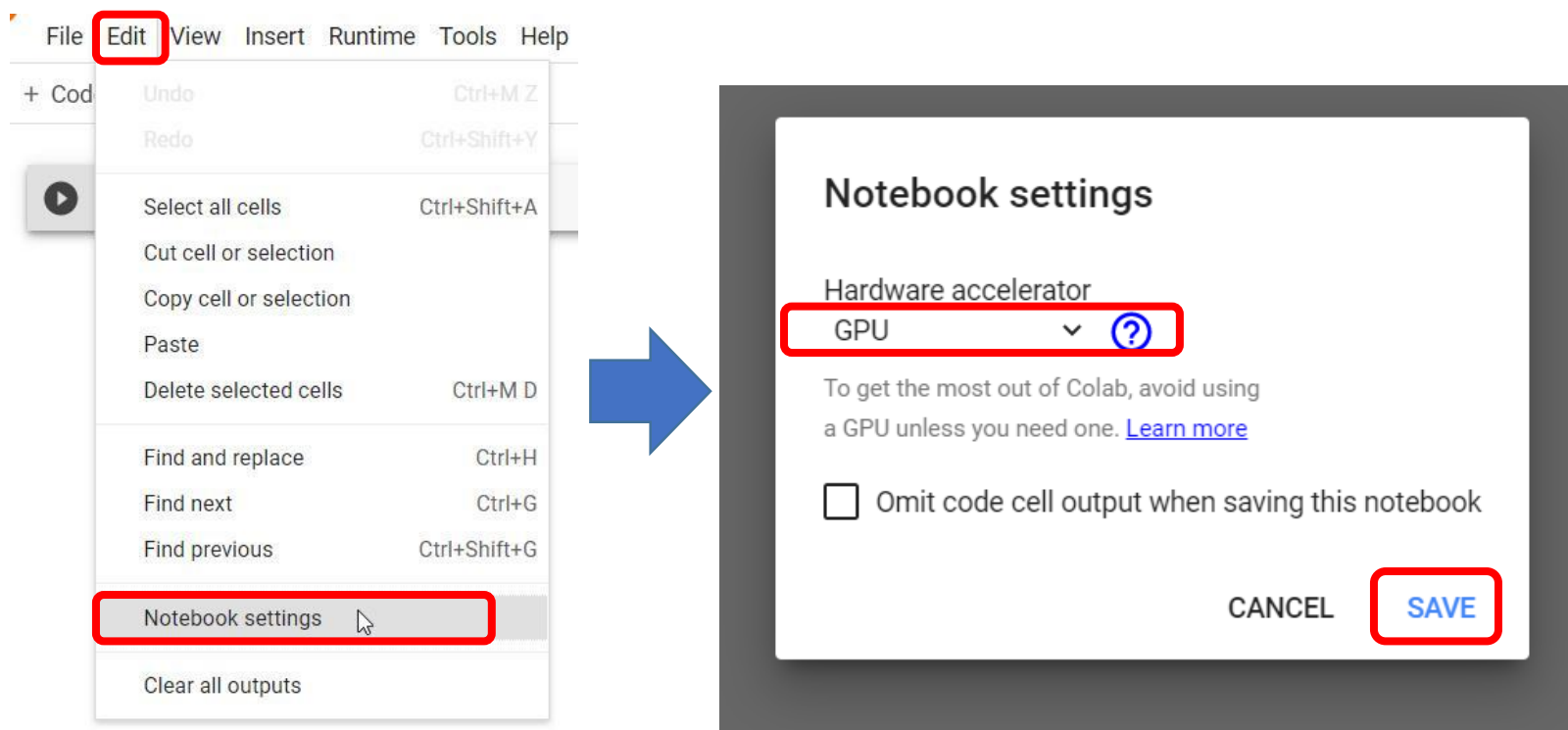
# 使用 Google Colab 雲端運算平台

- 使用 Colab 開啟 Project2.ipynb



# 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/>





# 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')

!ln -s /content/drive/My\ Drive/ /mydrive
%cd /content/drive/MyDrive/Project2-YOLOv4
```

Permit this notebook to access your Google Drive files?

This notebook is requesting access to your Google Drive files. Granting access to Google Drive will permit code executed in the notebook to modify files in your Google Drive. Make sure to review notebook code prior to allowing this access.

No thanks

Connect to Google Drive

%cd 指令使目前工作目錄位置為 Project2-YOLOv4，  
之後將在這個目錄下執行程式

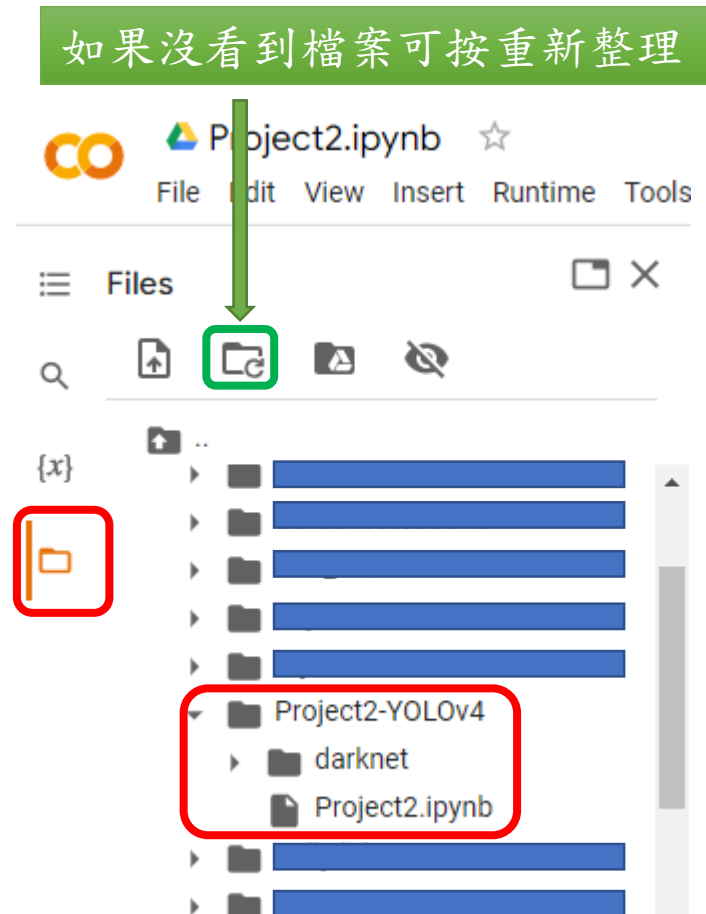
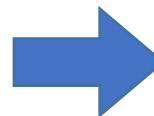
# 使用 Google Colab 雲端運算平台

- Copy 一份 darknet 到你的雲端硬碟

## Step 1 : Download the darknet repository

```
[ ] !git clone https://github.com/AlexeyAB/darknet
```

```
Cloning into 'darknet'...  
remote: Enumerating objects: 3, done.  
remote: Counting objects: 100% (3/3), done.  
remote: Compressing objects: 100% (3/3), done.  
remote: Total 13221 (delta 0), reused 1 (delta 0), pack-reused 13218  
Receiving objects: 100% (13221/13221), 11.92 MiB | 11.19 MiB/s, done.  
Resolving deltas: 100% (9042/9042), done.
```



# 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)

▼ Step 2 : Modify the Makefile to have GPU and OpenCV enabled

```
[ ] %cd darknet
    !sed -i 's/GPU=0/GPU=1/' Makefile
    !sed -i 's/OPENCV=0/OPENCV=1/' Makefile
    !sed -i 's/CUDNN=0/CUDNN=1/' Makefile
    !sed -i 's/CUDNN_HALF=0/CUDNN_HALF=1/' Makefile
```

📁 /content/darknet

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。

# 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。

▼ Step 3 : Make darknet

```
[ ] !make
```

- 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。

# 使用 Google Colab 雲端運算平台

- 下載YOLOv4預訓練權重和測試權重檔

```
[▶] !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
```

# 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。

# 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()
```

# 使用 Google Colab 雲端運算平台

- 使用 YOLOv4 測試權重檔來對 person.jpg 進行 detect



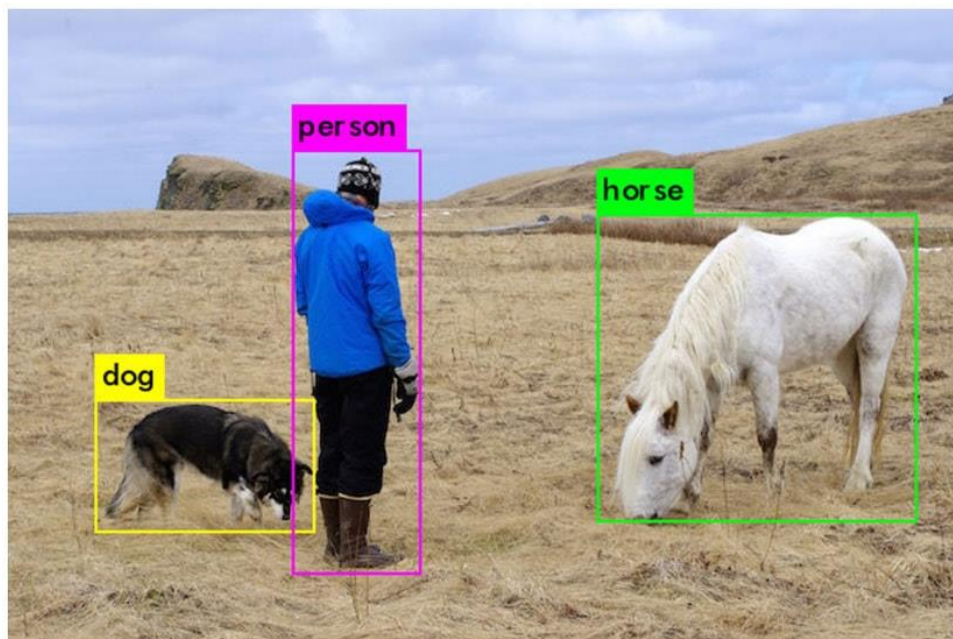
```
#run darknet detection
!./darknet detect cfg/yolov4.cfg yolov4.weights data/person.jpg

imshow("predictions.jpg")
```



# 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張圖的辨識結果。



# 使用 Google Colab 雲端運算平台

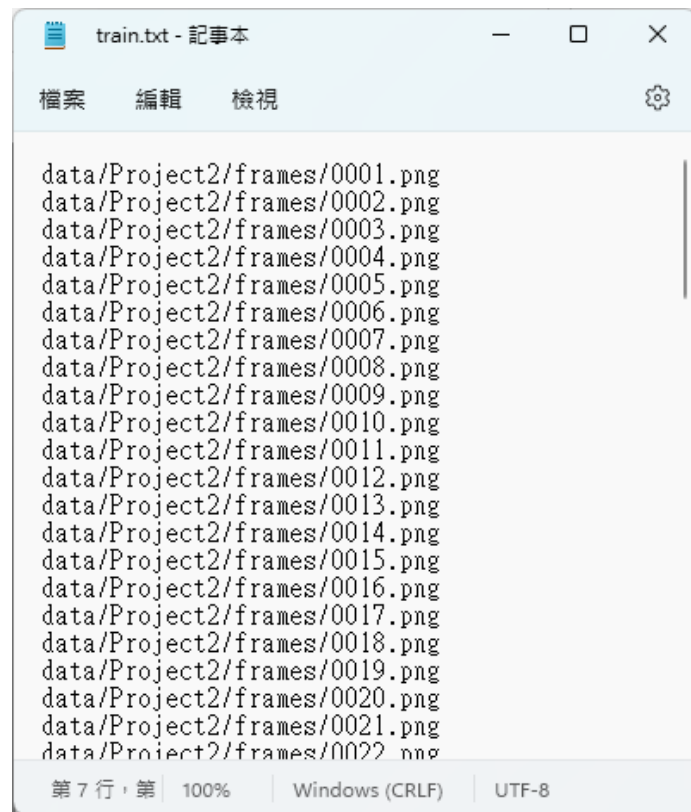
- 準備好訓練YOLOv4模型所需檔案，並上傳到雲端硬碟的Project2-YOLOv4資料夾中：
  1. frames(包含資料集影像與同名的.txt座標文件(使用LabelImg工具取得))
  2. obj.names (自己資料集的類別名稱)
  3. obj.data (相關路徑及各種參數設定)
  4. train.txt(訓練資料路徑，未生成)

# 使用 Google Colab 雲端運算平台

- 由於本次實作僅有一個類別，若在使用 labelImg 標記時未修改 classes.txt，可能會導致物件類別 ID 不為 0，因此需重新命名 Label ID
- 執行助教提供的 relabel.py，可將所有物件類別 ID 重設為 0

# 使用 Google Colab 雲端運算平台

- 生成train.txt：
  1. 執行generate\_train.py
  2. 生成訓練資料路徑檔



```
data/Project2/frames/0001.png
data/Project2/frames/0002.png
data/Project2/frames/0003.png
data/Project2/frames/0004.png
data/Project2/frames/0005.png
data/Project2/frames/0006.png
data/Project2/frames/0007.png
data/Project2/frames/0008.png
data/Project2/frames/0009.png
data/Project2/frames/0010.png
data/Project2/frames/0011.png
data/Project2/frames/0012.png
data/Project2/frames/0013.png
data/Project2/frames/0014.png
data/Project2/frames/0015.png
data/Project2/frames/0016.png
data/Project2/frames/0017.png
data/Project2/frames/0018.png
data/Project2/frames/0019.png
data/Project2/frames/0020.png
data/Project2/frames/0021.png
data/Project2/frames/0022.png
```

# 使用 Google Colab 雲端運算平台

- 修改obj.data和obj.names：
  - 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/Project2/train.txt
3 valid = data/Project2/train.txt
4 names = data/Project2/obj.names
5 backup = backup/
```

obj.data

```
1 helmet
```

obj.names

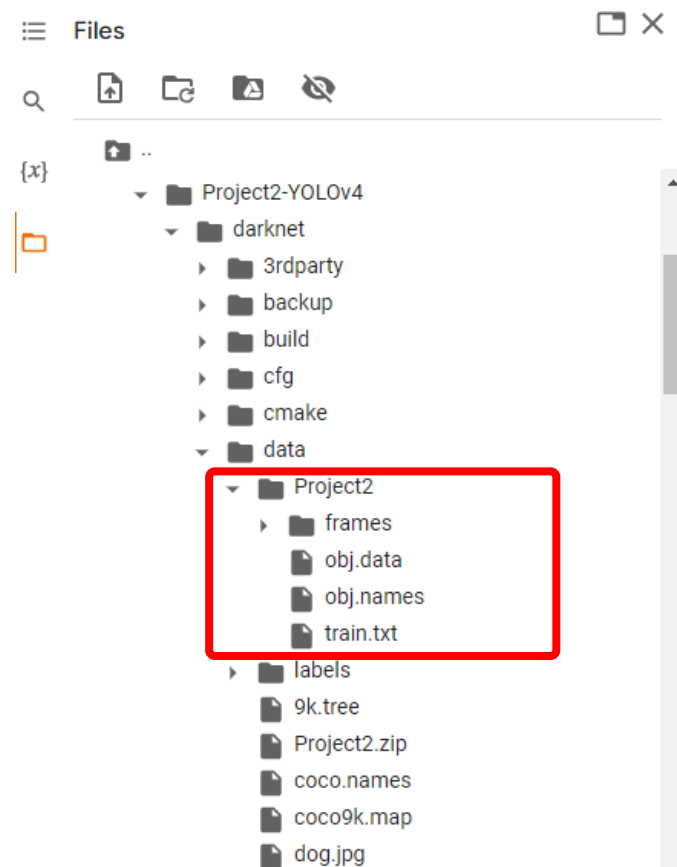
# 使用 Google Colab 雲端運算平台

- 將以上檔案壓縮至Project2.zip，並上傳到雲端後解壓縮：

1. 上傳至雲端硬碟中Project2-YOLOv4/darknet/data
2. 將當前工作目錄設為darknet
3. 解壓縮檔案至data路徑內

```
%cd /content/drive/MyDrive/Project2-YOLOv4/darknet  
!pwd
```

```
!unzip data/Project2.zip -d data
```



# 使用 Google Colab 雲端運算平台

- 複製yolov4的設定檔來作修改

```
%cp cfg/yolov4-custom.cfg cfg/yolov4-project2.cfg
```

- 修改yolov4-project2.cfg：

- yolov4-project2.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。
- 修改一：

- batch = 64, subdivisions = 16不變當訓練運行顯示“out of memory”時，可以嘗試將subdivisions升至32或64。
- max\_batches = ? 為自己資料集種類個數\*2000，如果有一類即為2000
- Steps = ?, ? 分別為max\_batches的80%和90%

- batch: 一次訓練(iteration)要多少訓練樣本，取決於GPU記憶體大小，主要影響訓練速度。
- subdivision: yolo模型特殊訓練參數，為將batch拆分為mini-batch數量。若batch=64, subdivision=16，每次訓練會將4筆資料放入GPU記憶體中，將16個mini-batch運算梯度平均後，再進行權重更新。

yolov4-project2.cfg x

```
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)

# 使用 Google Colab 雲端運算平台

- 修改二：

- $\text{filters} = (5 + \text{classes}) \times 3$ 。如資料集種類個數  $\text{classes}$  為 1 則  $\text{filters} = (5 + 1) \times 3 = 18$
- $\text{classes} = 1$  (分類個數)
- 共要修改三個段落，如下圖所示：

```
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
```



# 使用 Google Colab 雲端運算平台

- 初次訓練

在初次訓練，須有一個預訓練權重檔案當基底

```
!./darknet detector train data/Project2/obj.data cfg/yolov4-project2.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

- 接續訓練

```
!./darknet detector train data/Project2/obj.data cfg/yolov4-project2.cfg backup/yolov4-custom_last.weights -dont_show
```

接續上次權重紀錄訓練

# 使用 Google Colab 雲端運算平台

- 對影片進行物件辨識

```
!./darknet detector demo data/Project2/obj.data cfg/yolov4-project2.cfg backup/yolov4-custom_last.weights -dont_show helmet3.mp4 -out_filename helmet3_out.mp4
```

進行辨識

obj.data位置

.cfg位置

訓練好的權重檔案

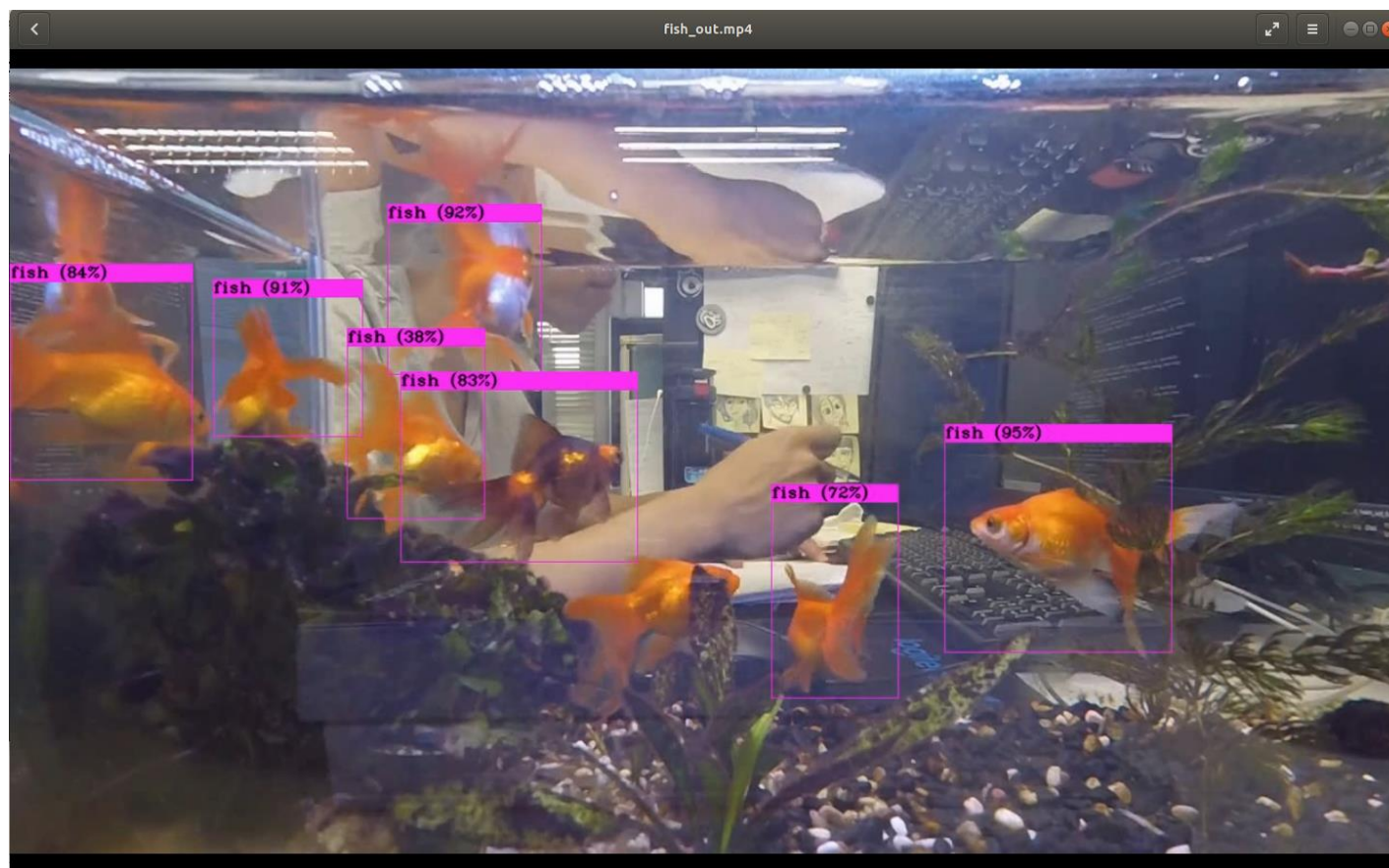
過程不要顯示任何視窗

src filename

output filename

# 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，資源總是有限，因此若是訓練時間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，除了本次介紹的連接 Google 雲端硬碟來儲存資料外，還可以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/>

# 專案要求

- 請使用Project1 LabelImg工具取得的檔案進行訓練(包含資料集影像與同名的.txt座標文件)。
- 請使用Project1助教所提供的影片選擇一部進行測試。
  - 訓練影片與測試影片需為不同部影片。
- 將包含框選結果的影片上傳至youtube，於ppt中附上連結。
- 助教提供的參考程式檔包含：
  - relabel.py #可將所有物件類別ID重設為0
  - Project2.ipynb #YOLO訓練
- 參考程式檔連結：

<https://drive.google.com/drive/folders/1mRS-GsWC0db57vF2Thu4S7YbLFUEZDOq?usp=sharing>

# 作業繳交要求

- 作業繳交項目：
  - 小組報告(一組繳交一份就好)
  - 個人心得
- 本專案繳交期限至2022/06/08(三)23:59
- 超過時間遲交每隔一週（含一週內）分數打8折，採累計連乘方式
  - 舉例：  
遲交三天—以遲交一週計算  $\text{<遲交的項目分數>} * 0.8 = \text{該項目得到的分數}$   
遲交九天—以遲交兩週計算  $\text{<遲交的項目分數>} * 0.8 * 0.8 = \text{該項目得到的分數}$
- 遲交兩週以上作業不予補繳

# 作業繳交要求

- 小組報告需包含以下內容：
  - 檔名格式：**Project2\_第X組\_小組報告.pptx**
  - 小組成員名單。
  - 執行專案步驟。
  - 執行程式是否有遇到什麼困難，如何解決。
  - 寫出選擇使用的資料集為哪種類別(工人辨識、車輛辨識)及影片名稱(訓練與測試分別為哪部影片)。
  - 程式執行的結果(將結果影片上傳至youtube並附上影片連結)。
  - 討論分析(包含是否成功或失敗，可能原因及改進方法等等)。

# 作業繳交要求

- 個人心得需包含以下內容:
  - 檔名格式：**Project2\_學號\_姓名.docx 或.pdf**
  - 本次專案中個人所學、執行程式是否有遇到什麼困難，如何解決(150字起，未達150字會斟酌扣分)。
  - 小組分工表(含姓名、貢獻比例、工作內容)。



# 參考資料

- 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](https://colab.research.google.com/drive/12QusaaRj_lUwCGDvQNfICpa7kA7_a2dE)