物聯網智造基地
IOT SERVICE HUB

國產IC開發套件- HUB8735
示範案例-手勢辨識猜拳遊戲

指導單位：經濟部工業局

執行單位：財團法人資訊工業策進會

作者：陳淨騰

資訊工業策進會 Institute for Information Industry

大綱

- ❑ 方案介紹
- ❑ 功能說明
- ❑ AI Model 訓練過程
- ❑ 成果

方案介紹 1/3

HUB 8735
採用Ameba RTL8735

HUB 8735 Smart AI CAM是具備多功能影像處理的高度集成模組，內置NPU AI 運算引擎，加速處理AI模型以及802.11 a/b/g/n 雙頻Wi-Fi與BLE低耗電藍牙傳輸，可廣泛應用於各種結合影像識別或 AI運算之物聯網場域，適用於智能家庭，工業物聯網，智慧零售，健康照護或是車用電子等場景；多款Pre-trained AI models已最佳化在模組直接運行，可做 為AI教學之體驗工具；同時尺寸僅30.5x43.9mm，亦為可直接整合在產品設計中做為快速導入Edge AI應用的快製套件。

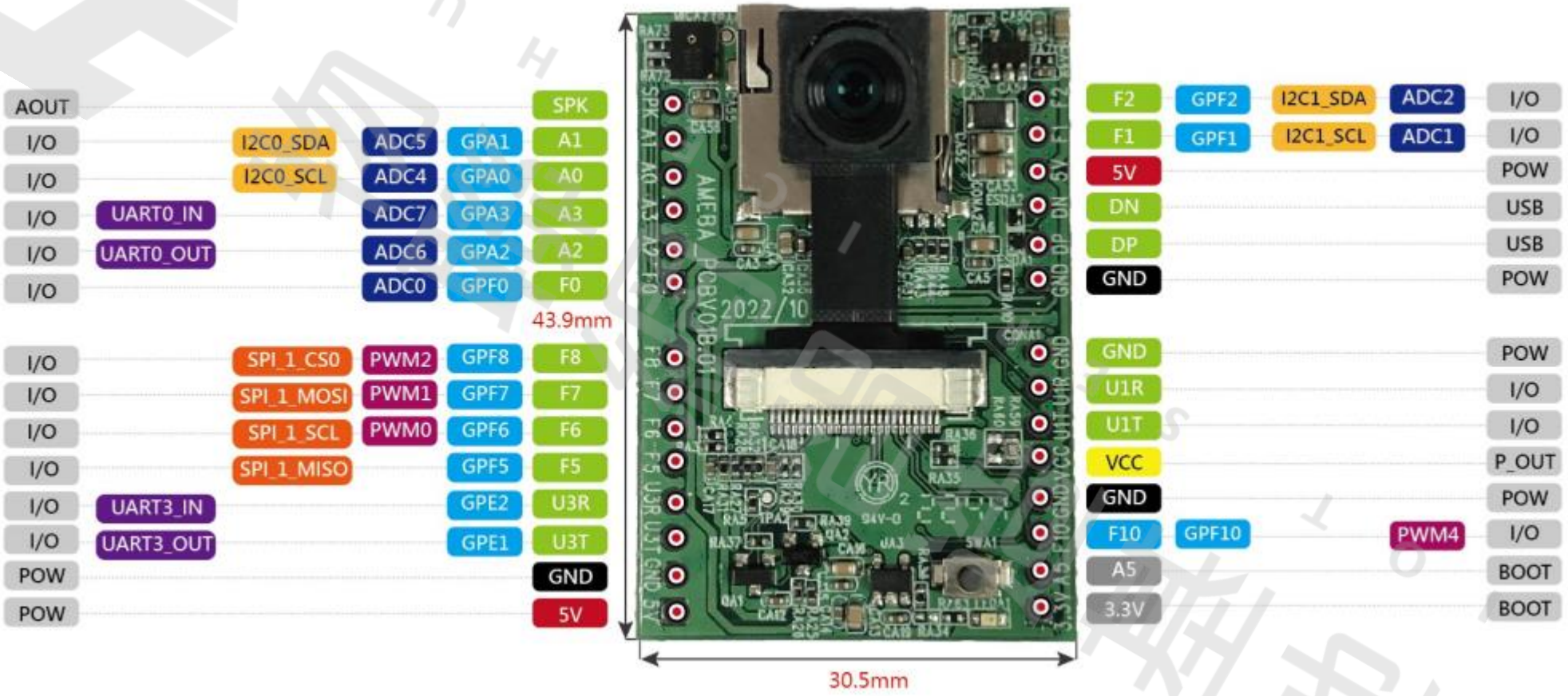
HUB 8735主要特色包括：

- 採用Ameba RTL8735
- 兼容Arduino開發特性
- 具備多功能影像處理的高度集成模組
- 內置NPU AI 運算引擎加速處理AI模型
- 提供802.11 a/b/g/n 雙頻Wi-Fi與BLE低耗電藍牙傳輸
- 可廣泛應用於各種結合影像識別或 AI運算之物聯網場域

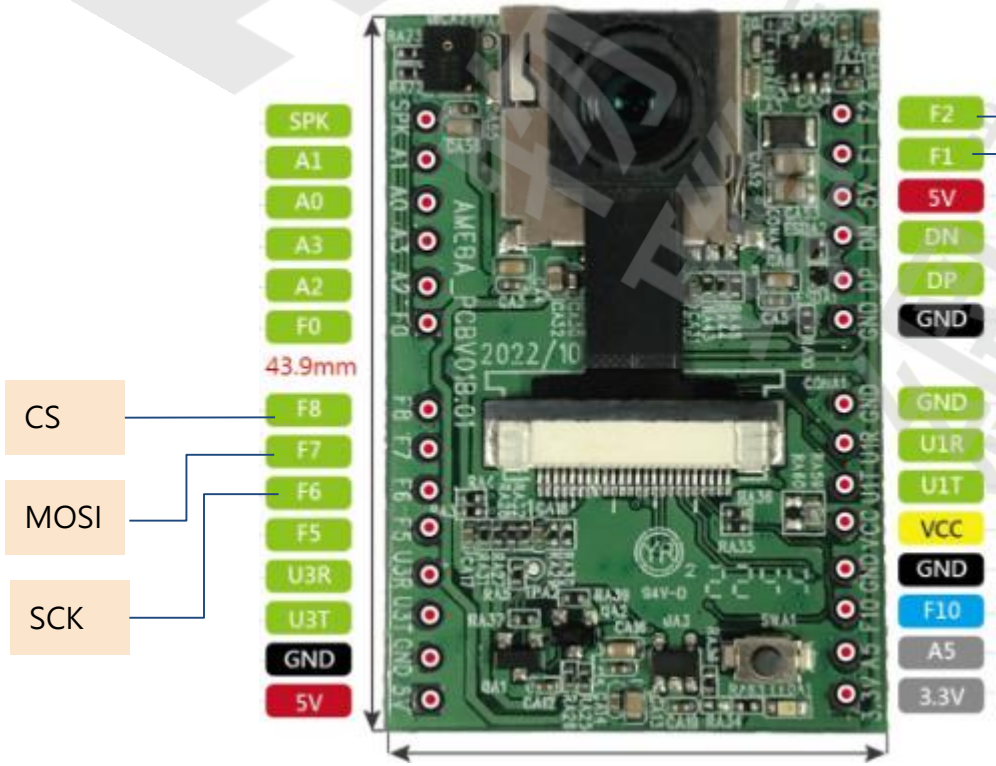
方案介紹 2/3

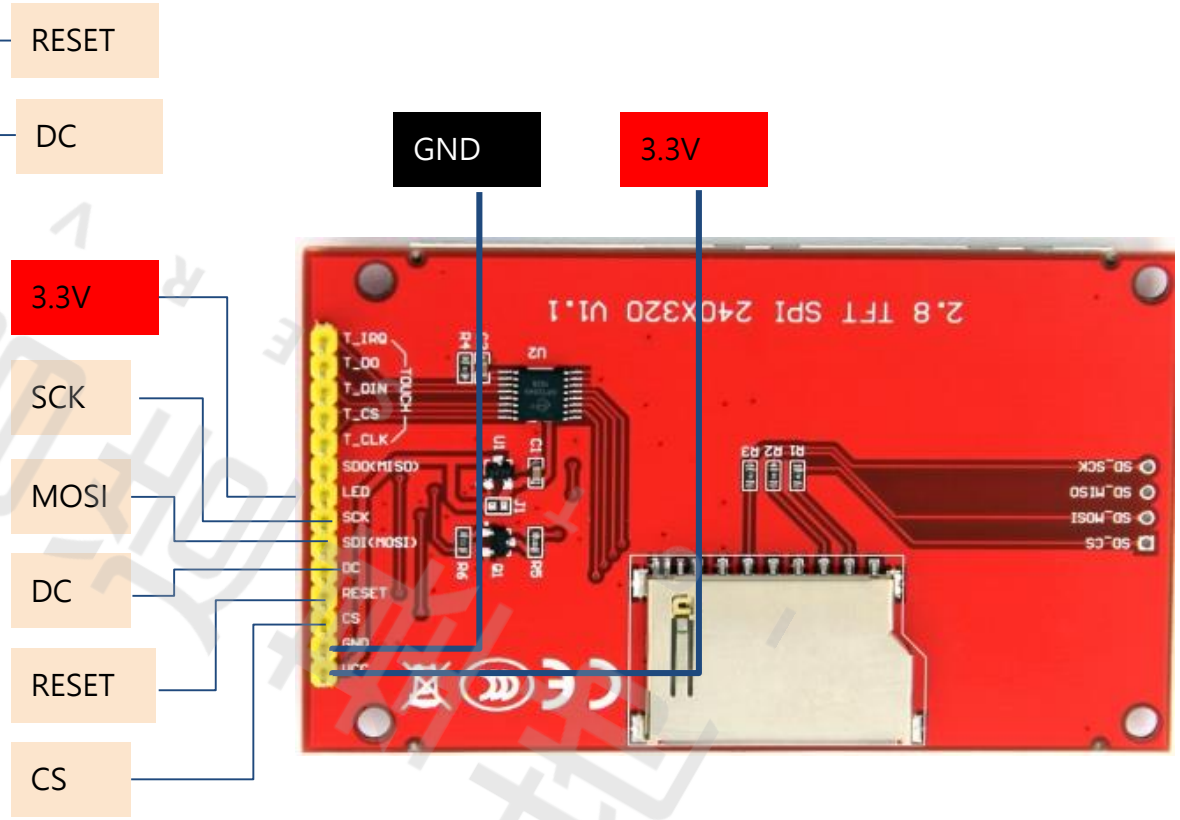
功能	
處理器	RTL8735 AIOT 國產晶片 / 32-bit Arm v8M, up to 500MHz
影像輸入	Full HD 1080P CMOS感測
語音輸入	內建MIC語音輸入
無線規格	802.11 a/b/g/n Wi-Fi, 2.4GHz/5GHz Bluetooth Low Energy (BLE) 5.1
影像規格	H.264/H.265
內建 AI	Integrated Intelligent Engine @ 0.4 TOPS
外部接口	SPI / I2C / PWM / UART / ADC / GPIO

方案介紹 3/3



方案接線圖





功能說明

手勢辨識猜拳遊戲應用範例可以使用機器學習技術來辨別玩家的手勢，並進行與 HUB 8735 對戰。在對戰過程中，TFT LCD 會顯示玩家的手勢，並顯示勝負結果。

由於手勢有不同的角度及手指的位置變化，加上不同玩家有不同的手勢習慣，因此手勢辨識是相當具有挑戰性的。在製作 AI 資料集時，需要調整各種手勢的誤差，以提高辨識的準確度。

AI Model 訓練過程 1/28

本次範例全程在 Google Colab 平台建立。

Colab 有以下特點

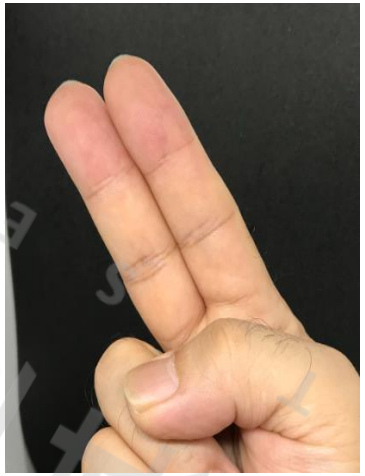
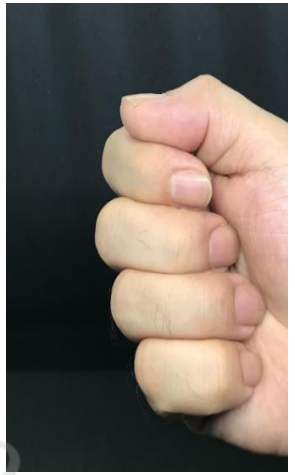
- 不必進行任何設定
- 可開發及訓練類神經網路
- 可進行 AI 研究
- 輕鬆共用

google Colab 入口網址:
https://colab.research.google.com/

AI 模型訓練過程 2/28- 資料集建置

本範例的資料集共有 3 個手勢，全部有 370 張圖片。全部皆自行拍照產生，以下是部分內容:



AI 模型訓練過程 3/28 -標註

使用 labelImg 標註軟體做識別特徵標註，這個軟體工具可以提供 YOLO (*.txt) 的標註格式。在此次範例分別標註不同的手勢 : 剪刀 石頭 布

AI 模型訓練過程 4/28- 導入 Darknet

導入 Darknet 前準備工作:

- my_dataset.zip 包含 370 張影像及 YOLO 標註檔
- g_obj.data　資料集設定檔
- g_obj.names　資料標籤名稱
- g_yolov4-tiny-custom.cfg　為自定義模型及參數
- g_train.txt　自定義資料訓練集檔案列表
- g_val.txt　自定義資料驗證集檔案列表
- test01.jpg~ test03.jpg 測試用影像

AI 模型訓練過程 5/28- 導入 Darknet

g_obj.data 資料集設定檔，classes 為有 三個類別
其餘設定為個資料的路徑

內容如下:

```
classes = 3
train = data/g_train.txt
valid = data/g_val.txt
names = data/g_obj.names
backup = /my_drive/g_yolov4-tiny
```

AI 模型訓練過程 6/28 - 導入 Darknet

g_obj.names 資料標籤名稱，設定各類別名稱。
內容如下:

Scissors
Stone
cloth

AI 模型訓練過程 7/28 - 導入 Darknet

g_yolov4-tiny-custom.cfg 自定義模型及參數
這個參數檔是從 darknet/cfg/yolov4-tiny-custom.cfg 修改而來

line 6 : batch=64

line 7: subdivisions=1

line 8: width=416 # 為32的倍數

line 9: height=416 # 為32的倍數

line 20: max_batches=6000 # 為 classes x 2000 ，範例類別為3

line 21: steps=4800,5400 # 為 max_batches 的 80% 與 90%

line 212,263: filter=24 # (classes + 5) x 3

line 220,269 : classes=3 #物件類別數量(手勢 :剪刀,石頭,布)

AI 模型訓練過程 8/28 - 導入 Darknet

g_train.txt　　自定義資料訓練集檔案列表，為資料集的路徑及檔案名稱。

注意：最後一筆資料不要換行，訓練過程有可能會出現錯誤。

部分內容列舉如下:

data/my_dataset/1.jpg
data/my_dataset/2.jpg
data/my_dataset/3.jpg
data/my_dataset/4.jpg
data/my_dataset/5.jpg
data/my_dataset/6.jpg
data/my_dataset/7.jpg
data/my_dataset/8.jpg
data/my_dataset/9.jpg
.....

AI 模型訓練過程 9/28 - 導入 Darknet

g_val.txt 自定義資料驗證集檔案列表，同樣為資料集的路徑及檔案名稱。
注意路徑是需自建 my_dataset 資料夾，可以換取不同資料夾名稱。

部分內容列舉如下：
data/my_dataset/50.jpg
data/my_dataset/51.jpg
data/my_dataset/52.jpg
data/my_dataset/53.jpg
data/my_dataset/54.jpg
data/my_dataset/55.jpg
data/my_dataset/56.jpg
data/my_dataset/57.jpg
data/my_dataset/58.jpg
data/my_dataset/59.jpg
.....

AI 模型訓練過程 10/28 - 訓練前準備

開啟 Google Colab 並新增建立記事本。
首先指令 !nvidia-smi ，確認 GPU 有上線

```
[ ]  !nvidia-smi

Thu May 18 11:11:16 2023
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 525.85.12    Driver Version: 525.85.12    CUDA Version: 12.0     |
|-------------------------------+----------------------+----------------------+
| 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 T4            Off  | 00000000:00:04.0 Off |                    0 |
| N/A   55C    P8    10W /  70W |      0MiB / 15360MiB |      0%      Default |
|                               |                      |                  N/A |
+-------------------------------+----------------------+----------------------+

+-----------------------------------------------------------------------------+
| Processes:                                                                  |
|  GPU   GI   CI        PID   Type   Process name                  GPU Memory |
|        ID   ID                                                   Usage      |
|=============================================================================|
|  No running processes found                                                 |
+-----------------------------------------------------------------------------+
```

AI 模型訓練過程 11/28 - 訓練前準備

如果 GPU 沒有上線，
請在左邊點取 "連線" 邊的倒三角形 ，再點取 "變更執行階段類型"
內選項 “硬體加速器” 更改成 GPU

AI 模型訓練過程 12/28 - 訓練前準備

連接 Google Drive 網路硬碟 ，並取名為 my_drive

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

    Mounted at /content/drive

[ ] !ln -s /content/drive/MyDrive/ /my_drive
```

AI 模型訓練過程 13/28 - 訓練前準備

從 Github 下載 Darknet

!git clone https://github.com/AlexeyAB/darknet

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

AI 模型訓練過程 14/28 - 訓練前準備

下載預設權重

!wget -N https://github.com/AlexeyAB/darknet/releases/download/darknet_yolo_v4_pre/yolov4-tiny.weights

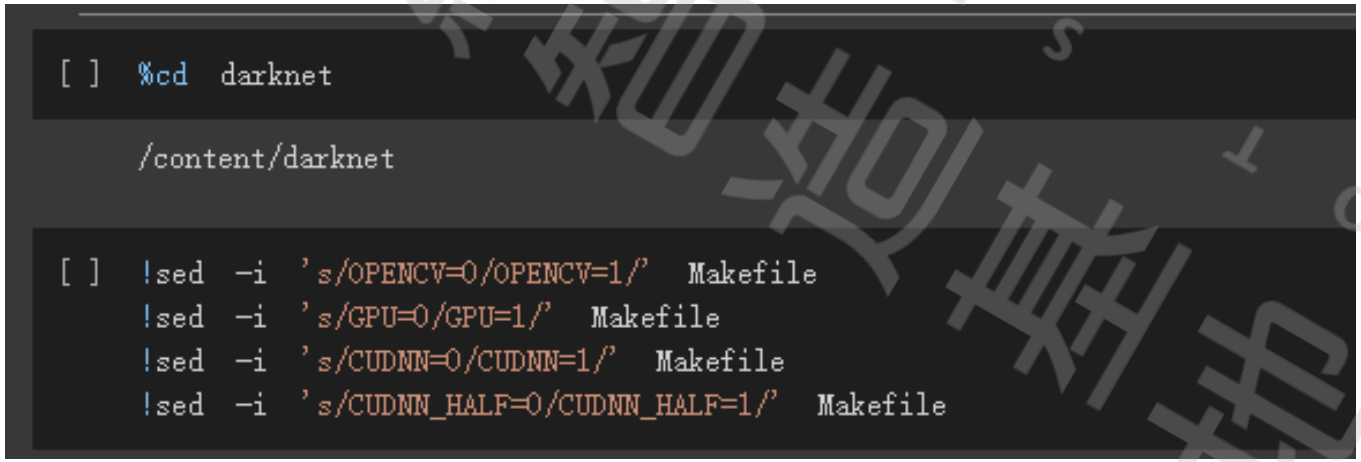
```
[ ]  !wget  -N  https://github.com/AlexeyAB/darknet/releases/download/darknet_yolo_v4_pre/yolov4-tiny.weights
```

AI 模型訓練過程 15/28 - 訓練前準備

修改 Makefile

將 GPU, CUDNN, CUDNN_HALF, OPENCV 設定為 Enable

```
!sed -i 's/OPENCV=0/OPENCV=1/' Makefile
!sed -i 's/GPU=0/GPU=1/' Makefile
!sed -i 's/CUDNN=0/CUDNN=1/' Makefile
!sed -i 's/CUDNN_HALF=0/CUDNN_HALF=1/' Makefile
```



AI 模型訓練過程 16/28 - 訓練前準備

編譯 Compiler Darknet

! make

```
[ ]  !make
```

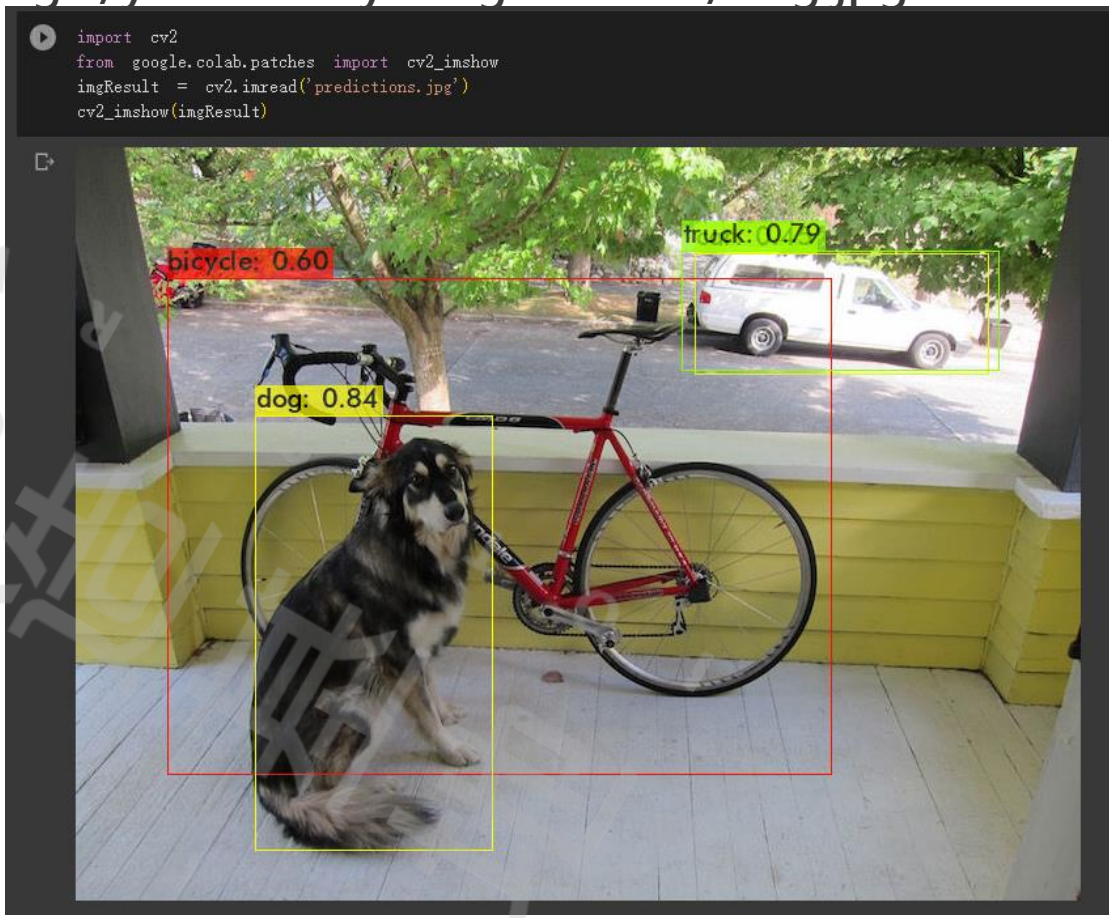
AI 模型訓練過程 17/28 - 訓練前準備

測試 Darknet，使用內建的 coco 資料集

! ./darknet detector test cfg/coco.data cfg/yolov4-tiny.cfg ../yolov4-tiny.weights data/dog.jpg

辨識完成後，編寫簡短的 python 來顯示結果
如左圖，即成功完成辨識。

```
import cv2
from google.colab.patches import cv2_imshow
imgResult = cv2.imread('predictions.jpg')
cv2_imshow(imgResult)
```



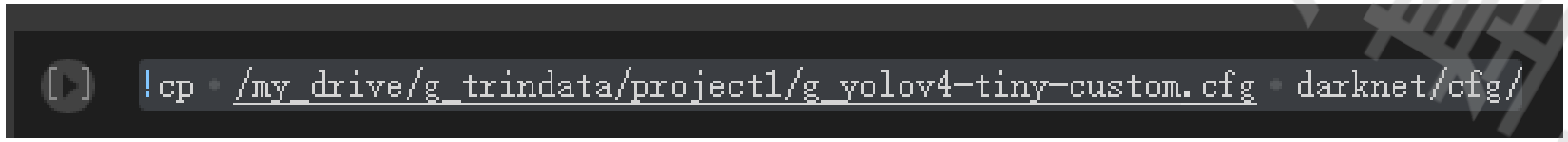
AI 模型訓練過程 18/28 - 訓練前準備

回到 content 並下載 預權重 yolov4-tiny.conv.29

!wget -N https://github.com/AlexeyAB/darknet/releases/download/darknet_yolo_v4_pre/yolov4-tiny.conv.29

複製 g_yolov4-tiny-custom.cfg 至 darknet/cfg 內

!cp /my_drive/g_trindata/project1/g_yolov4-tiny-custom.cfg darknet/cfg/



AI 模型訓練過程 19/28 - 訓練前準備

複製其餘資料集設定檔 至 darknet/data 內

```
!cp /my_drive/g_trindata/project1/g_obj.names darknet/data
!cp /my_drive/g_trindata/project1/g_obj.data darknet/data
!cp /my_drive/g_trindata/project1/g_train.txt darknet/data
!cp /my_drive/g_trindata/project1/g_val.txt darknet/data
```

AI 模型訓練過程 20/28 - 訓練前準備

解開 資料集 zip 檔案並放入 darknet/data 內

!unzip /my_drive/g_trindata/project1/my_dataset.zip -d darknet/data

```
[ ]  !unzip  /my_drive/g_trindata/project1/my_dataset.zip  -d  darknet/data
```

AI 模型訓練過程 21/28 - 開始訓練

準備就緒,開始訓練

! ./darknet detector train data/g_obj.data cfg/g_yolov4-tiny-custom.cfg ../yolov4-tiny.conv.29 -map -dont_show

```
[ ]  !./darknet  detector  train  data/g_obj.data  cfg/g_yolov4-tiny-custom.cfg  ../yolov4-tiny.conv.29  -map  -dont_show
```

約莫訓練 1~2 小時左右，訓練成果會在預設資料夾 g_yolov4-tiny 內生成 *g_yolov4-tiny-custom_final.weights* 為最終的權重檔

AI 模型訓練過程 22/28 - 測試結果

可以先行測試驗證訓練成果，透過以下辨識指令來辨識 test04.jpg

!./darknet detector test data/g_obj.data cfg/g_yolov4-tiny-custom.cfg ../g_yolov4-tiny-custom_final.weights /my_drive/g_trindata/project1/test04.jpg

```
[ ]  !./darknet detector test data/g_obj.data cfg/g_yolov4-tiny-custom.cfg ../g_yolov4-tiny-custom_final.weights /my_drive/g_trindata/project1/test04.jpg
```

AI 模型訓練過程 23/28 - 測試結果

同樣我們透過簡短的 python 程式碼來顯示結果

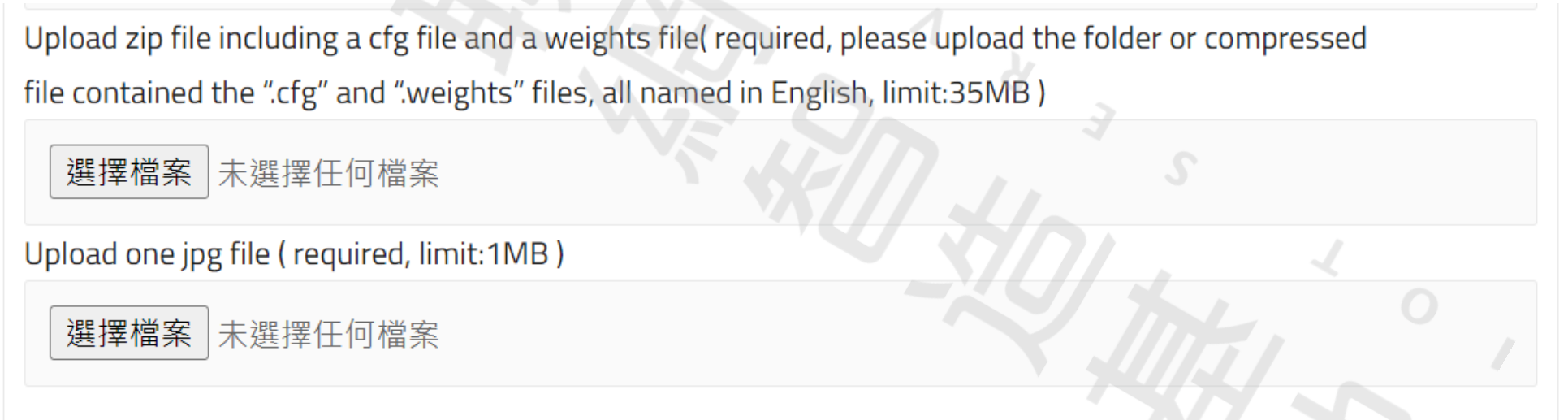
```
[ ] import cv2 # 導入OpenCV函式庫
    from google.colab.patches import cv2_imshow # 導入Colab.patches函式庫

    imgResult = cv2.imread('predictions.jpg') # 讀入結果影像
    cv2_imshow(imgResult) # 顯示結果影像
```

驗證結果是可以正常的辨識

AI 模型訓練過程 24/28 - 轉檔

在與 HUB8735 應用之前，必須透過 Realtek 專責網頁先行轉檔。

將 權重檔(g_yolov4-tiny-custom_final.weihts 及 g_yolov4-tiny-custom.cfg 檔案 ，使用 zip 壓縮打包。
開啟 https://www.amebaiot.com/zh/amebapro2-ai-convert-model/ 網頁，填入 zip 檔案，及選擇任一張驗證圖片並上傳。



AI 模型訓練過程 25/28 - 轉檔

約莫 1 ~ 5 分鐘，會自動回覆至登記註冊 e-mail 信箱裡。並將 nb 檔案下載。

開啟檔案總管至

C:\Users\<User>\AppData\Local\Arduino15\packages\ideasHatch\hardware\AmebaPro2\4.0.4-build20230601\variants\common_nn_models

並更名置換 yolov4_tiny.nb 檔案。

AI 模型訓練過程 26/28 - Arduino 程式設計

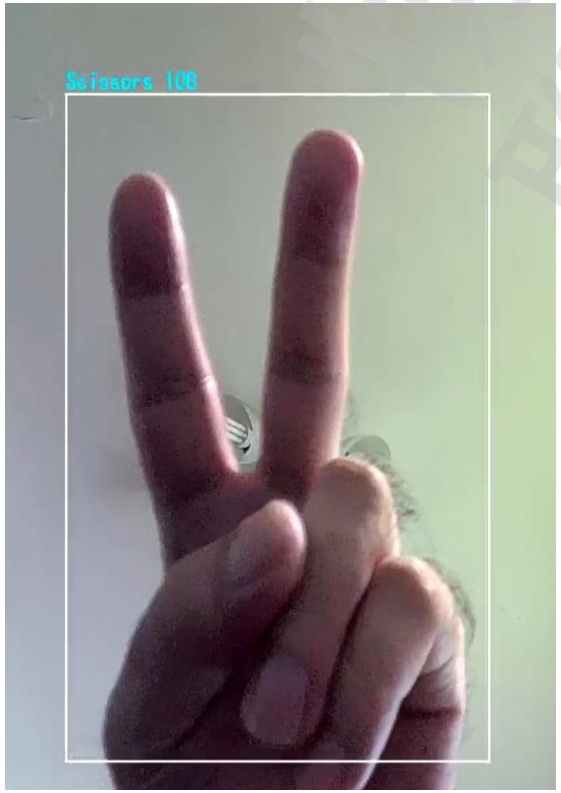
利用範例 ObjectDetectionCallBack 修改及新增功能

首先將 ObjectClassList.h 設定 3 個手勢名稱

```
ObjectDetectionItem itemList[3] = {
{0, "Scissors",          1},
{1, "Stone",             1},
{2, "cloth",             1}};
```

AI 模型訓練過程 27/28 - Arduino 程式測試

將程式上傳至 HUB8735，以下是 Video stream 接收至 VLC 軟體來驗證辨識能力



AI 模型訓練過程 28/28 - Arduino 程式設計

再透過主程式在判斷手勢時，會在

```
ObjDet.getResultCount()
```

這裡成功識別後，產生對應的 Index 。此時就會得知玩家的手勢，再將結果顯示於 TFT LCD。

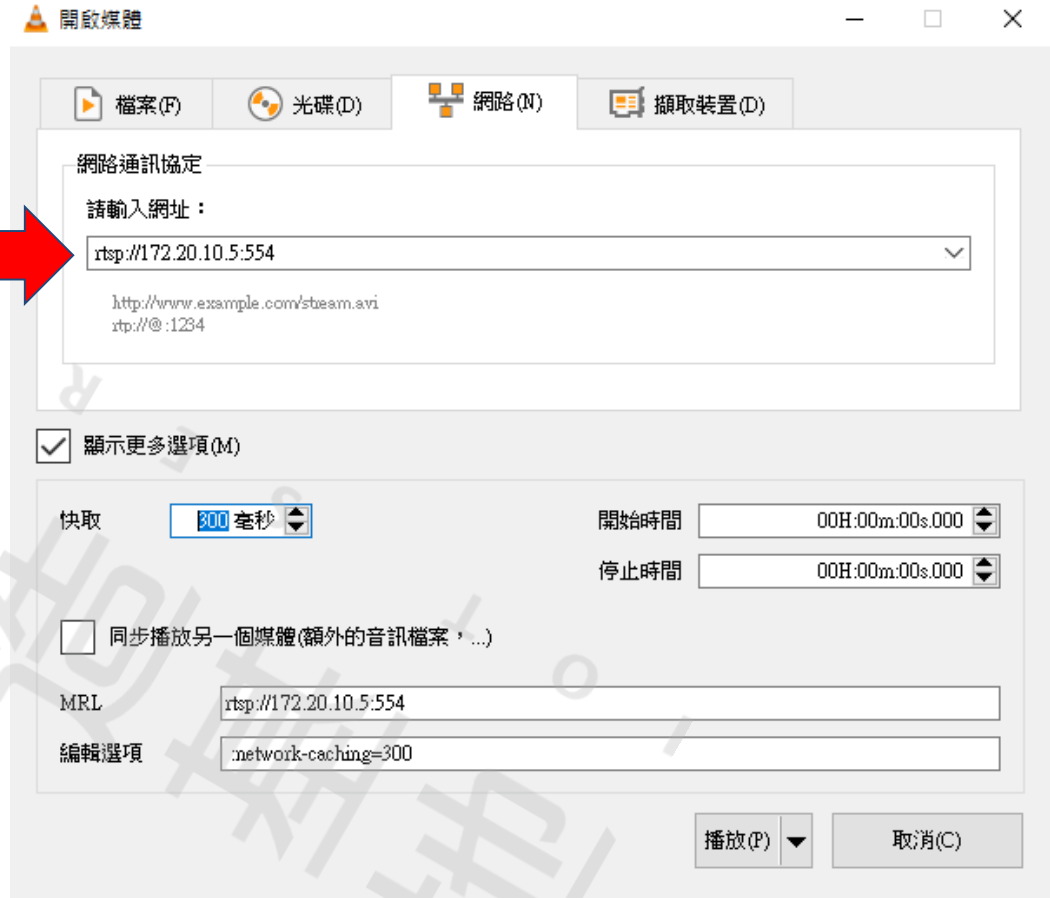
```
if(checkNum == 0)
{

  tft.drawBitmap(160,85,150,150,scissors);

}
else if (checkNum == 1)
{

  tft.drawBitmap(160,85,150,150,rock);

}
else if (checkNum == 2)
{

  tft.drawBitmap(160,85,150,150,paper);

}
```

AI 模型訓練過程 - VLC 使用說明

```
-> [Driver]: set group key to hw: alg:4(WEP40-1 WEP104-5 TKIP-2 AES-4) keyid:1
->
-> Interface 0 IP address : 172.20.10.5
-> fwin(1),enc_en(0),IQ_OFFSET = 0x3200
```

HUB8735 啟動後，注意在Serial Monitor 會顯示 IP Address。

這個 IP Address 為 Video Stream 位址，如左圖的方式填入 *rtsp://youraddress:554*



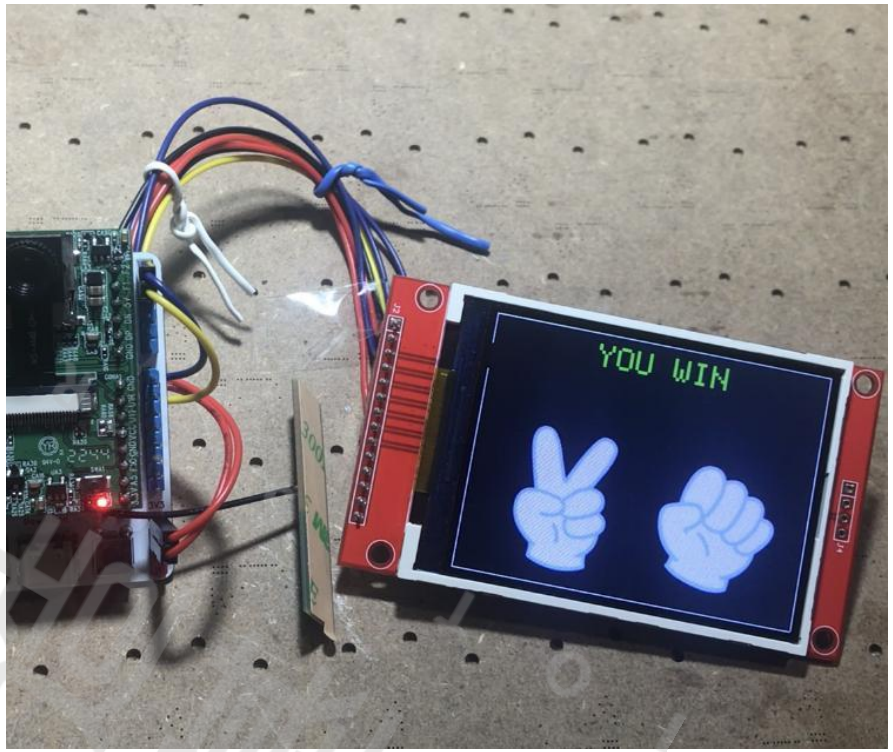
成果

最後，再透過亂數產生器生成 3 個手勢變化與玩家對戰。

```
myNum = random(3);
```

將結果輸出至 TFT LCD 。

```
tft.setForeground(ILI9341_GREEN);
tft.print("YOU WIN");
```



參考程式碼 :

https://github.com/cold63/Hub8735_Project/tree/main/finger-guessing-game/src

Thank you

物聯網智造基地
IOT SERVICE HUB