基於 YOLOv5 與樹莓派之手勢辨識物聯網系統
Gesture Recognition IoT System Based on YOLOv5 and Raspberry Pi

黃樹林 游毓倫
Shu-Lin Hwang Yu-Lun Yu

明志科技大學電子工程系

摘要

本文提出基於YOLOv5的手式辨識方法，利用少量自定義的資料集進行訓練。它具有速度快、準確度高和模型尺寸小的特色。資料蒐集分為三個階段:1.在單純的白色背景下收集7個不同手勢，手勢擺放角度、距離都一樣各蒐集100張。2.複雜背景下拍攝少許手勢，手勢擺放角度、距離都不一樣各蒐集40張。3.單純拍攝背景和未偵測的手勢，以及經由Augmentor將原本資料集增強，產生每個手勢各500張。採取漸進式修正資料集訓練，根據實測辨識結果，在室內區域整體mAP@[0.5:0.95]達到0.909。另外以不在資料集內兩位實驗室同學進行手勢辨識，整體的mAP@[0.5:0.95]也有0.811。將手勢辨識模型轉換成edgetpu.tflite模型，搭載於Raspberry pi4配置Coral USB Accelerator邊緣裝置上，通過MQTT將webcam即時偵測手勢辨識結果發佈給Wi-Fi IoT晶片Esp8266。依不同手勢及架設的Flask網頁設定來控制家電產品，實現直觀的AI物聯網智慧控制裝置。

關鍵詞：手勢辨識、YOLOv5、MQTT、Flask、物聯網

ABSTRACT

This paper proposes a YOLOv5-based hand gesture recognition system using a small custom dataset. It is characterized by high speed, high accuracy and small model size. The data collection process involves three stages: 1. Capturing 100 images of 7 specific pre-defined hand gestures on a plain white background, maintaining consistent angles and distances. 2. Capturing 40 images of a few hand gestures in complex backgrounds with varying angles and distances. 3. Capturing images of backgrounds without any hand gestures and augmenting the above datasets using Augmentor to generate 500 images for each hand gesture. The training process employs progressive refinement of the dataset, achieving an overall mAP@[0.5:0.95] of 0.909 in indoor environments. Additionally, the model achieves an mAP@[0.5:0.95] of 0.811 when tested on two subjects not in the dataset. The hand gesture recognition model is converted into an edgetpu.tflite model and deployed on a Raspberry Pi 4 with Coral USB Accelerator for edge device deployment. The system uses MQTT to send real-time hand gesture recognition results from a webcam C3 to a Wi-Fi IoT chip Esp8266. According to different gestures and setting of Flask web pages to control home appliances, realize intuitive AI IoT smart control devices.

Keywords: Gesture recognition, YOLOv5, MQTT, Flask, Internet of Things.

1. 前言

人工智慧 (Artificial Intelligence) 和物聯網 (Internet of Things) 帶來前所未有便利和智慧化，深度學習則是 AI 領域的分支，通過模擬人類大腦學習大量的資料，而具備學習和辨識的能力。通過深度學習可以讓電腦從大量資料中學習和提取有用的特徵，進而實現圖像辨識、語音辨識、自然語言理解等功能。物聯網作為各種設備的橋樑，通過 Wi-Fi、藍芽等無線通信技術，兩者結合可以透過語音辨識、手勢辨識、網頁、APP 來控制家電、監控環境。然而深度學習需要大量資料集來訓練和參數調整來優化模型，對一般用戶有一定限制和門檻，本文以少量自定義的資料集訓練來達成準確率高手勢模型，部署在樹莓派上以及架設 Web 使用者界面，達成一個低功耗、速度快的 AI 物聯網系統。

深度學習受到許多的挑戰和限制像是 1.資料獲取:要如何獲得大量的資料是深度學習訓練模型的一大關鍵。2.時間成本:蒐集照片和標註照片以及訓練模型都需要大量的時間。本文的動機是提出一種解決深度學習中資料蒐集和時間成本高的問題

方法，能夠利用少量的自定義資料集訓練專屬的模型，並且可以部署在物聯網系統上。

本文研究目標是以收集少量自定義 7 種手勢資料集，採用 YOLOv5 物件偵測深度類神經網路技術，資料集漸進式修正訓練，達成準確率高、體積小的自定義手勢模型。部署在 Raspberry Pi 4 並配合 webcam，利用手勢模型即時偵測出影像中的手勢，回傳預測值及預測框，並且將預測值發布到 MQTT Broker，Broker 再將收到的訊息傳給訂閱 Topic 的 Wi-Fi IoT 晶片 ESP8266，再依據接收到的不同手勢以及架設 Flask 使用者界面，讓使用者不僅可以在家裡面使用手勢控制，也可以在戶外透過使用者界面控制家電產品，設計出一個無須穿戴裝置就能操控家電的 AI 物聯網裝置。

2. 背景知識

2.1 影像辨識技術

傳統神經網路屬於全連接網路 (Fully Connected Networks)，每個輸入層皆連到隱藏層，每個隱藏層皆連到輸出層。全連接網路用於圖像辨識需要耗費大量的訓練參數和時間。在 1998 年由 LeCun et al. (1998) 提出比傳統神經網路還更好的 CNN (Convolutional Neural Network) 卷積神經網路，奠定之後深度學習深遠的發展。CNN 是一個利用卷積和池化步驟多次來提取圖片的特徵，最後在經由全連接層分類器來分類圖片的物件。不過原型 CNN 的模型太小導致訓練大數據集的圖片準確度不夠，特徵提取的不夠全面，因此，把卷積層和池化層拿掉並且替換成大型已經預訓練好 CNN 的模型，像是 VGG16Net、Resnet50、Mobile Net 等，並以這些架構為基礎，拿來做圖片特徵提取並延伸應用在其他的學習上，也就是深度學習中的遷移學習。物件偵測技術也因此延伸出了 One- Stage 偵測和 Two-Stage 偵測。本論文主要探討應用的是 One-Stage 的 YOLOv5。

2.2 YOLOv1-v4

YOLO (You Only Look Once) 物件偵測最早是由 Joseph Redmon et al. (2016) 提出。總共有 4 個版本，YOLOv1、YOLOv2、YOLOv3、YOLOv4，都是基於 YOLOv1 基礎上加以改良，也是當今深度學習最受矚目的類神經網路技術。YOLOv1 是第一個將繪製邊界框和識別類標籤的問題結合在一個端到端的物件偵測網路。YOLOv2 由 Joseph Redmon (2017) 提出使用新的網路架構(darknet-19)及結合新的技巧如:Batch Normalization、High Resolution Classifier、Convolutional With Anchor Boxes 等。

YOLOv3 由 Joseph Redmon (2018)提出引用多尺度 FPN 預測，主幹網路 darknet-19 加入類似 Resnet 的殘差結構，升級為 Darknet-53，引入殘差網路，可以使網路提取更深的特徵，分類損失使用 Binary Cross-Entropy Loss，分類器改用 Sigmod 函數做邏輯回歸，實現多分類目標偵測。

YOLOv4 由 Alexey bochkovskiy (2020) 提出提高處理速度，並提高模型偵測精度,降低硬體使用限制。使用新技巧 CSP (Cross-Stage-Partial-connections) 在 YOLOv3 主幹網路 Darknet-53 基礎上與 CSPNet 進行組合，提出 CSPDarknet-53，解決梯度重複，Mosaic 馬賽克資料增強參考 Cutmix 於 2019 年提出的數據增強方式。Cutmix 進行拼接時只使用 2 個影像，而 Mosaic 則使用 4 個影像隨機剪取、變換角度、佈置，使原有資料集可以更豐富化訓練。YOLOv4 在每個卷積層由 Mish 函數取代 Leaky-ReLU，達到更好準確性，加入特徵金字塔 PANet 代替 YOLOv3 的 FPN，提高小尺寸目標的偵測效果，而 IOU loss 改成了 CIOU。

2.3 YOLOv5

在 YOLOv4 版本出現後隔 2 個月發布了 YOLOv5 (2020)，大致上整個網路架構大同小異。最大改進是由原本的 Darknet 框架，改為 Pytorch 框架。YOLOv5 推出 4 個不同大小的 Weights，YOLOv5S、YOLOv5M、YOLOv5L、YOLOv5XL 等。其中 YOLOv5S 如圖 1，是 YOLOv5 版本中體積最小速度更快，號稱每個圖像推理時間最快 0.007 秒，每秒處理 140 張圖像，Weights 權重只有 YOLOv4 的九分之一 1.4MB，更適合部署在邊緣裝置上。

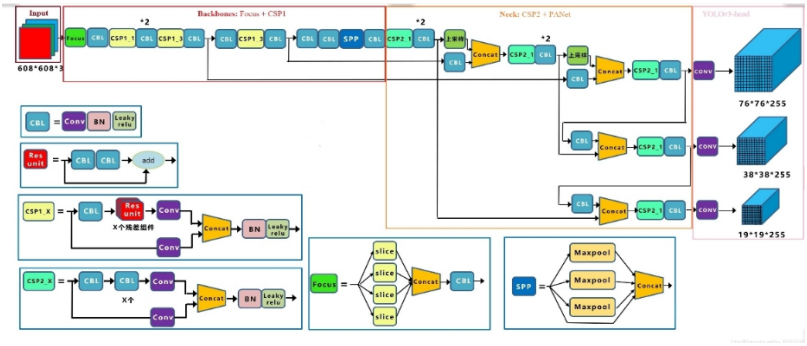

圖 1 YOLOv5S 架構圖

YOLOv5 在輸入端採用了 YOLOv4 的 Mosaic 馬賽克資料增強以豐富資料集。馬賽克資料增強技

術將四個不同圖像進行裁剪並拼成一張大圖像，在訓練時可以同時學習多個場景和多種目標。優勢是可以增加模型泛化能力，也能減少 GPU 計算需求。

自適應 Anchor 計算:YOLOv5 使用 k-means 分群演算法對資料集中的目標框進行分類，從而找出最佳 Anchor 大小和寬高比。在 YOLOv5 *.yaml 文件中已經有先預設 640*640 圖像大小 Anchor 的尺寸，在訓練前會對資料集的標註信息對默認 Anchor 進行最佳召回率，當大於等於 0.98 時不用重新計算 Anchor，小於 0.98 時會使用 k-means 重新計算。

自適應圖片縮放:在送入偵測網路中 YOLOv5 在推理進行改良。1.將圖片計算長寬除上預設的 640*640 輸入尺寸取寬或長的最小值。2.將圖片乘上最小值取得縮放後的尺寸。3.縮放後的長寬相減，再採用 np.mod 取餘數的方式後再除以 2 即可獲得上下兩端需要黑邊填充的數值，這樣可以縮減黑邊來提升推理速度，在訓練時還是依照常用的方法。

Focus 結構採用切片操作，在圖片進入 backbone 前，每一張圖片中每隔一個像素拿到一個值，這樣的操作可以使特徵信息不會流失，也可以提升小目標的準確率。

CSP 結構 (Cross Stage Partial) : YOLOv5 設計兩個 CSP，一個應用在主幹網路中 CSP1，另一個 CSP2 應用在 NECK 中。CSP1 (CSPDarknet53) 結構是 YOLOv5 主幹網絡核心部分，通過使用 CSP 模組對特徵進行分支處理。分支處理又分為兩個部分，第一部分接收原始輸入特徵圖提取圖像的低級特徵，例如邊緣、紋理等。第二部分則是對輸入特徵圖進行降維處理，減少計算量同時提取更抽象特徵。CSP2 結構通過 SPP (Spatial Pyramid Pooling) 對不同尺度的特徵圖進行特徵提取，更有效的捕捉不同尺度的上下特徵，提高物件偵測的性能。

BoundingBox: YOLOv5 採用 GIoU Loss。GIoU 是一種物件邊界框評估指標，不僅考慮預測框和真實框之間的重疊程度 (IOU)，還考慮兩個框之間的尺寸和位置差異。在 YOLOv5 中，GIoU 被用作損失函數的一部分，用於計算預測框和真實框之間的距離。通過最小化 GIoU 損失，模型可以更好對預測框進行調整，使其更貼近真實框的位置和大小。有助於提高物件偵測的準確性和邊界框的精確性。

2.4 LabelImg

LabelImg 是一種圖像標註工具軟體，用 Python 語言撰寫，圖形界面使用 PyQT，註釋格式有 YOLO、CreateML 以及 PASCAL VOC，能對影像進行矩形形式標註，針對框選出來的矩形命名為要辨識的名稱，主要用於物件偵測任務。(https://github.com/heartexlabs/labelImg)

2.5 Augmentor

Augmentor 是一個 Python Package，用於幫助機器學習任務的影像增強與生成。主要是一種資料增強工具，提供基本的影像預處理功能，在資料稀缺時來產生自己想要數量的豐富資料集。影像處理功能包含透視偏斜 (Perspective Skewing) 、彈性變形 (Elastic Distortions) 、旋轉 (Rotating) 、推移 (Shearing) 、裁剪 (Cropping) 、鏡像 (Mirroring) 。(https://github.com/mdbloice/Augmentor)

2.6 MQTT

MQTT(Message Queuing Telemetry Transport) ISO 標準(ISO/IEC PRF 20922)下基於發布/訂閱範式的消息協定，工作在 TCP/IP 協定上。MQTT 系統由伺服器與通信的客戶端組成，通常稱為"代理"。客戶可以是發布者或訂閱者。每個客戶端都可以連接到代理。適合具有限制頻寬的小型控制器以及在網路環境狀況糟糕的環境下，進而實現物聯網 AIoT 如圖 2 (陳響亮、李宜靜、陳文泉、李桂銘,2018)。

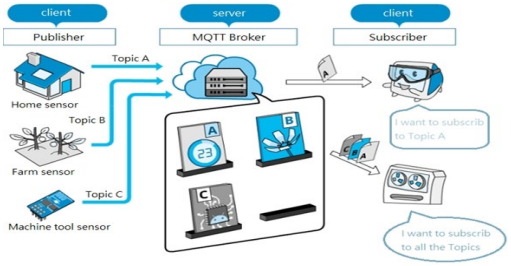

圖 2 MQTT 示意圖

2.7 Flask

Flask 是一個輕量級的 Python 網路應用框架，被廣泛用於開發 Web 應用和 API。它以簡單、易用和靈活著稱，非常適合小型和中型的項目。(https://flasksocketio.readthedocs.io/en/latest/)

3. 系統架構

本文物聯網系統架構分前端處理與後端處理，由 YOLOv5 訓練的手勢模型經模型轉換成 edgetpu.tflite 格式部署在樹莓派 4 上，搭配 Webcam、Coral TPU，利用 MQTT 訂閱/發布機制，控制微控制器 ESP8266 進行一系列家電產品控制，另架設 Flask 網頁提供使用者能夠及時監看家電產品的狀況如圖 3。

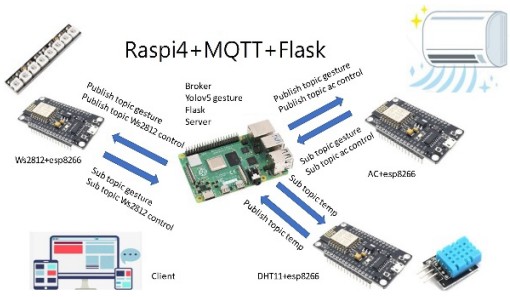

圖 3 物聯網系統架構

3.1 系統前端處理(HTML 網頁架構)

使用樹莓派 4 前端處理平台，運用 Opencv 及 Flask 內置的 Jinja2 模板引擎功能設計圖形化直覺的 HTML 網頁，透過動態 HTML 網頁顯示讓使用者更能清楚知道目前各設備運作狀況以及室內溫度、濕度，也可透過網頁進行一系列家電產品控制，架構圖如圖 4。

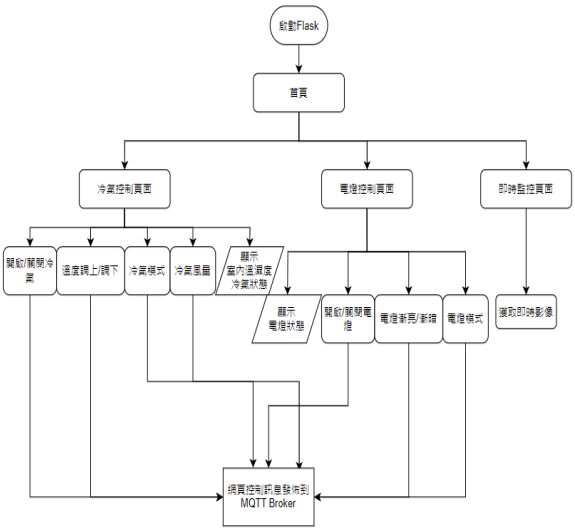

圖 4 前端網頁架構圖

3.2 系統後端處理

後端基於需要低功耗、長時間運行的需求，採用了樹莓派當作後端處理平台。在筆電進行訓練手勢模型後，將手勢模型引入到樹莓派中進行影像辨識，透過 Coral USB Accelerator 加速器來增強即時運算以及影像呈現，另外在 Flask 後端以 Python 程式撰寫，以 MQTT 訂閱/發布的純文字傳輸協議，將預測的值發布給訂閱端的 ESP8266 微控制器來達成控制家電產品，收到控制指令後 ESP8266 再發布控制結果到前端的網頁，即時呈現給使用者其流程圖如圖 5。

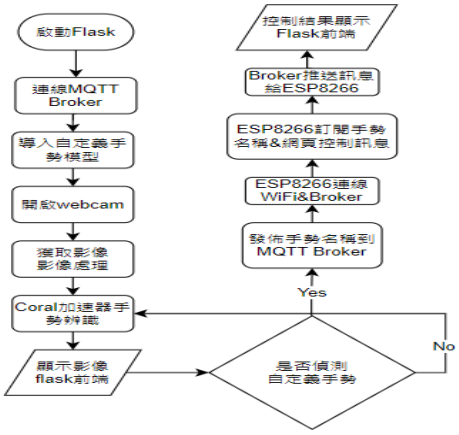

圖 5 後端流程圖

3.3 YOLOv5 自定義手勢模型訓練

手勢模型訓練分為 5 個部分:架設環境、蒐集資料、標示資料集與劃分資料集、調整訓練參數&訓練，最後訓練完會得到一個模型權重，如圖 6。

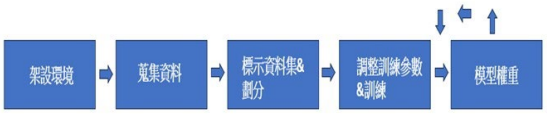

圖 6 手勢模型訓練流程圖

架設環境:本文使用 Anaconda3 創立虛擬環境在虛擬環境下安裝 Pytorch 架構的 YOLOv5，藉由 python 進行訓練模型。

蒐集資料:使用 iphone SE 拍攝畫素 4032x3024 7 種自定義手勢分別為 Down、Close、Right、Left、Open、Palm、Up 如圖 7。

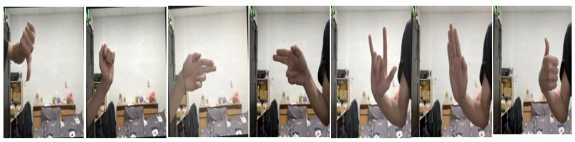

圖 7 自定義手勢 7 種

蒐集資料分為三個階段:

(1) 在單純的白背景下收集 7 個不同手勢，手勢角度遠近一樣各蒐集 100 張。

(2) 複雜背景下拍攝少許手勢，手勢擺放角度遠近都不一樣各蒐集 40 張。

(3) 單純拍攝背景和沒有要偵測手勢及上述影像資料集經由 Augmentor 資料集增強，產生每個手勢各 500 張。

3.4 手勢評估標準

評估標準採用 YOLOv5 的 Comfusion Matrix 混淆矩陣、P 值 (Precision) 單一類準確率 、R 值 (Recall) 、mAP (Mean Average Precision) 。混淆矩陣將手勢辨識結果分類為四種，分別為 True Positive、True Negative、False Positive、False Negative，其在機器學習中模型評估好壞最常用的方法，是一個可視化工具，此四種結果定義如表 1。

表 1 混淆矩陣定義

True Positive(TP)	訓練樣本中有此手勢，正確辨識此手勢
True Negative(TN)	訓練樣本中未有此手勢，正確辨識未有此手勢
False Positive(FP)	訓練樣本中未有此手勢，卻辨識有此手勢
False Negative(FN)	訓練樣本中有此手勢，卻無法正確辨識有此手勢

P 值:單一類準確率 (Precision) ，指的是在所有預測為正樣本的樣本中，有多少是真正的正樣本定義如下。取值範圍是 0 到 1，越接近 1 表示模型在預測為正樣本的準確率越高。

準確率=正確預測正樣本數/所有預測為正樣本樣本數。

R 值:召回率 (Recall) ，指在所有實際正樣本中，正樣本有多少被找出來定義如下。取值範圍是 0 到 1，越接近 1 表示模型能夠更好的找到正樣本。

召回率=正確預測樣本數/所有實際樣本數。

mAP:計算各種類別精確度並平均，一般使用 IoU (Intersection over Union) 作為判別準則，IoU 定義如下。

IoU = Intersection (A, B) / Union (A, B)

其中 A 和 B 分別表示模型預測和真實邊界框，Intersection(A, B)表示 A 和 B 交集面積，Union(A, B)表示 A 和 B 聯集面積。mAP50:取 IoU Threshold 為 0.5，IoU 大於 0.5 預測為 TP 小於 0.5 預測為 FP。mAP@[0.5:0.95]:取 IoU 門檻值 0.5 到 0.95 每 0.05 一個間隔設定門檻值算一次 mAP 平均。

3.5 測試資料說明

測試資料總共有 7 種手勢，包含有距離較遠照片、距離較近照片、室內背景、教室背景、資料集增強、參雜除 7 種手勢以外的資料集等作為訓練樣本。在實驗中採用漸進式訓練，並且辨識照片準確度以及即時偵測在不同環境下準確度。本文主要目的是希望以少量資料集訓練，達到辨識效果較佳的情況，並且應用於生活當中。測試資料詳如表 2。

表 2 測試資料

總測試資料手勢	7 種 (Down、Fit、Right、Left、Open、Palm、Up)
其他手勢訓練樣本	5 種 (One、Two、Three、Four、Six)
拍攝地點	室內 (教室、家裡)
拍攝人數	3 人
檔案規格	JPG
樣本色彩	彩色
額外拍攝區域複雜背景	31 張
拍攝不在資料集內同學	2 人

4. 實驗結果分析

4.1 自定義白色背景手勢模型驗證分析

第一階段蒐集資料為單純白背景下收集不同手勢，角度遠近一樣各蒐集 100 張，總數為 700 張進行訓練，訓練前使用 Random 函數使其劃分比例為 (0.7,0.2,0.1) 作為訓練集、驗證集以及測試集。訓練 300 次，白色背景測試集 70 張影像測試結果如表 3，複雜背景測試集 31 張影像測試如表 4。

此實驗結果使用 YOLOv5 作者默認參數配置訓練，效果很好也有穩定的收斂，在驗證白色背景測試集 mAP 都有高於 0.9，在偵測上沒有誤判情形。但在複雜背景測試資料集，發現泛化能力不足，mAP 下降很多甚至最低到 0.559，誤判情形如圖 8，分析後嘗試解決方法為拍攝背景多元化手勢，並加入原先資料集漸進訓練。

表 3 白色背景測試集測試結果

Class	Label	P	R	mAP [.5:.95]
All	70	1	1	0.932
Open	10	1	1	0.926
Palm	10	1	1	0.957
Fit	10	1	1	0.914
Up	10	1	1	0.942
Down	10	1	1	0.921
Left	10	1	1	0.917
Right	10	1	1	0.947

表 4 複雜背景測試集測試結果

Class	Label	P	R	mAP [.5:.95]
All	32	0.94	0.836	0.748
Open	4	1	1	0.83
Palm	6	0.75	0.999	0.841
Fit	5	1	1	0.804
Up	4	1	0.75	0.708
Down	4	0.577	0.75	0.559
Left	5	1	0.6	0.7
Right	4	1	0.75	0.795

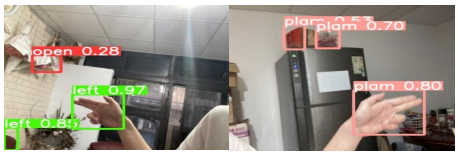

圖 8. 手勢誤判情況

4.2 區域性資料集訓練與驗證

資料集用第一階段白色背景 700 張照片，新增複雜背景拍攝 7 種手勢 280 張。測試是否可利用區域資料集拍攝複雜背景下的 31 張測試集，辨識率可以提升並且減少誤判情形。區域性資料集增加 98 張影像測試如表 5 和複雜背景測試集 31 張影像測試如表 6。

表 5 區域性資料集測試結果

Class	Label	P	R	mAP [.5:.95]
All	98	1	1	0.933
Open	14	1	1	0.952
Palm	14	1	1	0.96
Fit	14	1	1	0.896
Up	14	1	1	0.891
Down	14	1	1	0.911
Left	14	1	1	0.94
Right	10	1	1	0.947

表 6 複雜背景測試集測試結果

Class	Label	P	R	mAP [.5:.95]
All	32	1	1	0.88
Open	4	1	1	0.911
Palm	6	1	1	0.964
Fit	5	1	1	0.891
Up	4	1	1	0.8
Down	4	1	1	0.79
Left	5	1	1	0.908
Right	4	1	1	0.895

針對區域性資料集的訓練，明顯改善模型在複雜背景的辨識。新增區域性資料集的訓練模型，對

於複雜背景測試資料集測試結果，由 mAP 0.748 提升至 0.88，發現加入少許區域資料集，其辨識能力有所提升，誤判情形也有所改善。

即時偵測是本文主要目的之一，利用即時手勢辨識結果進行物聯網的操控。使用區域性資料集訓練的模型，設定置信度=0.5 時，發現即使訓練結果 mAP 都在 89-96%的高準確率。但是在即時偵測時，仍出現誤判問題，如圖 9 比出其他未標示手勢情況下會產生誤判，還有手勢角度稍微偏移的時候也會產生誤判，距離遠的時候會辨識不到手勢等。

圖 9. 誤判手勢

4.3 其他手勢及 Augmentor 資料集增強

在區域性資料集增加新增未標示手勢，以試圖改善手勢間的差異性。在訓練模型過程中發現雖然已有標示的手勢能夠明確的辨識。但是透過即時偵測，未知手勢會因為膚色相近，手勢間太過相似(例如比出數字 1-6)，從而導致模型的訓練結果再好，也會辨識誤判。靜態時影像模型能夠辨識成功的機率高於 80%，在動態影像手勢因稍微上下左右擺動誤判或是偵測不到。針對這兩個問題，嘗試透過未使用 labelimg 標記的影像，稱之為其他手勢。並利用 Augmentor 資料集加強來使有限的資料集透過角度的變換 45 度角、15 度角、30 度角，讓資料集的內容多元化。訓練集從原本的 980 張透過 Augmentor 以及其他手勢資料新增了 4124 張，驗證集 196 張新增了 977 張，測試集 98 張新增了 409 張。測試集影像測試結果如表 7 和複雜背景測試集 31 張影像測試結果如表 8。

表 7 測試集 409 張測試結果

Class	Label	P	R	mAP [.5:.95]
All	409	0.996	1	0.926
Open	68	1	1	0.931
Palm	47	0.988	1	0.914
Fit	51	0.981	1	0.885
Up	50	1	1	0.908
Down	68	1	1	0.933
Left	63	1	1	0.962
Right	62	1	1	0.951

表 8 複雜背景測試集測試結果

Class	Label	P	R	mAP [.5:.95]
All	32	0.976	0.971	0.909
Open	4	1	0.8	0.811
Palm	6	0.833	1	0.891
Fit	5	1	1	0.887
Up	4	1	1	0.911
Down	4	1	1	0.967
Left	5	1	1	0.953
Right	32	0.976	0.971	0.909

經過增加其他手勢、Augmentor 資料集增強方式訓練 300 次後發現誤判手勢情形有明顯改善如圖 10。即時偵測在置信度=0.9 時，誤判情形也有明顯改善。訓練次數一樣 300 次，當資料集增多時，第一階段到第三階段總體 mAP@[0.5:0.95]從最初的 0.932 微幅下降至 0.926，但相反的測試複雜背景的測試集時，mAP@[0.5:0.95]從 0.748 大幅上升至 0.909，另外不在資料集內的同學測試結果如表 9，整體 mAP@[0.5:0.95]也有 0.811。代表利用少張資料集加上 Augmentor 資料集增強可以成為沒有那麼多資料集時很好的代替方式，讓模型可以達到很好的辨識效果。

表 9 不在資料集內的同學測試結果

Class	Label	P	R	mAP [.5:.95]
All	49	1	1	0.811
Open	10	1	1	0.858
Palm	10	1	1	0.942
Fit	6	1	1	0.778
Up	8	1	1	0.828
Down	5	1	1	0.57
Left	5	1	1	0.876
Right	5	1	1	0.824

圖 10. 誤判手勢改善後

4.4 即時偵測模型

接著將訓練好模型放在 Raspberry pi4 上運行並查看辨識速度。發現雖然 YOLOv5S.pt 的模型只有 1.4MB 的大小，但是由於 Raspberry pi4 沒有 GPU 的情況下 FPS 只有 0.58，根本無法實施即時偵測。為解決這個問題，將 Raspberry pi4 搭配使用 google 的 Coral USB Accelerator，是一個能幫助沒有 GPU 邊緣裝置提升推理能力的 USB 硬體裝置。但其只支援 Tensorflow lite 格式，最大的 image input size 也不完全支援 640x640。因此把 yolov5s.pt 模型轉換成 image size 320x320 的 edgetpu.tflite，轉換過後模型大小由原本的 1.4MB 縮減成了 7.4KB，FPS 也從原本的 0.58 提升了 15.38 如圖 11。

在模型轉換的過程中為了要適用於低功耗設備上進行推理訓練，把 YOLOv5S float32 架構的神經網路層轉換 int8 量化架構的神經網路層。轉換的目的在於 float32 架構的神經網路層使用 32 位浮點數來表示模型的權重和計算，具有高的計算精度、較大的存儲空間和計算量，用於大多數的深度學習模型的訓練。相對來說 int8 量化架構的神經網路層使用 8 位整數來表示模型的權重和計算結果，具有較低的計算精度，優點是可以提升運算速度、節省存儲空間和較小的計算量，但是會損失準確度。

圖 11. 左邊是 YOLOv5S.pt，右邊是經過轉換的 edgetpu.tflite

4.5 樹莓派轉換模型與筆電效能比較

在 4.4 中雖然解決速度上的限制，但是模型因為轉換成了 edgetpu.tflite 導致其總體準確率 mAP@[0.5:0.95]從 0.835 下降至 0.221，其中是因為模型的大小縮減、架構改變，加上圖像輸入尺寸從原本的 604x604 降至 320x320，導致模型衰退如表 10。

表 10 測試集在不同實驗設備的結果

實驗設備	Label	P	R	mAP [.5:.95]
MSI NB	49	0.952	0.986	0.835
樹莓派 4	49	0.447	0.383	0.221

4.6 距離、解析度分析

距離測試:以圖 10 環境為例，使用 ASUS webcam 網路攝影機 C3，畫素為 640x480 拍攝距離 1 公尺、2 公尺、3 公尺時，每個手勢拍攝 10 張，共 7 個手勢 210 張作為測試集。在 1 公尺的測試集 mAP@[0.5:0.95]整體 0.419，在 2 公尺的測試集 mAP@[0.5:0.95]整體 0.028，因為模型的衰退在辨識手勢時距離超過 1 公尺基本偵測不到，如表 11。

表 11 距離測試實驗結果

Distance	P	R	mAP [.5:.95]	Distance
1M	0.919	0.471	0.419	1M
2M	0.0857	0.0857	0.028	2M

解析度分析:如圖 12 拍攝測試集是以樹梅派上面的 ASUS webcam 網路攝影機 C3 所拍攝，其畫素根據 OpenCV 默認畫素為 640x480。同樣用 iphone SE 拍攝 2 公尺的手勢，發現同樣距離使用 edgetpu.tflite 模型進行偵測，iphone SE 拍攝影像偵測的到。原因可能為本文模型皆是以高畫素拍攝進行學習，再加上 ASUS webcam C3 本身有廣角 78 度，導致低畫素加上廣角使影像越後面越顯得模糊。

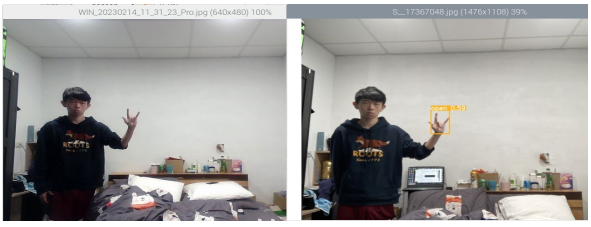

圖 12. 左邊 ASUS C3 webcam，右邊 iphone SE

表 12 本文資料集加上距離資料集的測試結果

Class	Label	P	R	mAP 50	mAP [.5:.95]
All	42	0.995	1	0.995	0.681
Open	6	1	1	0.995	0.646
Palm	6	1	1	0.995	0.709
Fit	6	1	1	0.995	0.618
Up	6	1	1	0.995	0.722
Down	6	1	1	0.995	0.717
Left	6	1	1	0.995	0.683
Right	6	0.964	1	0.995	0.671

根據測試結果發現在進行漸進訓練模型時，會因模型輸入尺寸、影像大小改變進而影響對目標辨識和邊界框準確性，使用較大輸入尺寸進行訓練可以提高模型的特徵提取能力，讓模型更好地捕捉影像中特徵，尤其在小目標或細節場景中，最後將距離資料集加入到之前資料集上，設定輸入尺寸 640 訓練 300 次，將 best.pt 模型轉換為 edgetpu.tflite 模型，整體的辨識率 mAP@[0.5:0.95]為 0.681 如表 12。即時偵測上設置置信度 0.5 時，最遠可以辨識到 3 公尺，其中 0.5 公尺到 2.5 公尺之間最穩定。

5. 物聯網應用

本文設計 7 種手勢名稱分別為 open、palm、fit、up、down、left、right，其詳細功能如表 13。

表 13 手勢功能

手勢名稱	手勢功能
Open	開啟冷氣/燈條
Palm	切換冷氣/燈條模式
Fit	關閉冷氣/燈條
Up	調整冷氣溫度增加 調整燈條亮度增加
Down	調整冷氣溫度減少 調整燈條亮度減少
Left	調整冷氣風量
Right	燈條呼吸效果模式 調整冷氣模式

將訓練好的手勢自定義模型、MQTT Broker、Flask 搭載於樹莓派 4 上面，並配置一個 Coral USB Accelerator 及 webcam 網路視訊鏡頭，將即時影像傳給手勢自定義模型，經過運算回傳的手勢預測值名稱、準確度、物體偵測位置邊框，透過 opencv 函式庫顯示在影像上。手勢名稱、網頁控制主題訊息經由 MQTT 的 Publish 推送到 Broker，ESP8266 連接 WiFi、MQTT Broker IP、訂閱手勢名稱、網頁控制主題訊息，Broker 推送訊息給 ESP8266 達成物聯網控制，展示如圖 13、14、15、16。

圖 13. 居家物聯網主頁

圖 14. 冷氣控制界面

圖 15. 電燈控制界面

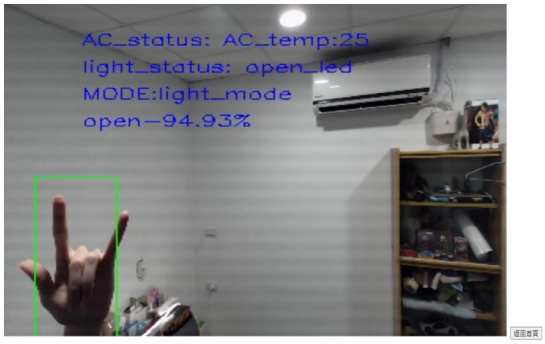

圖 16. 影像辨識界面

6. 結論與未來展望

本文以 Pytorch 框架 YOLOv5，蒐集少許資料，採取資料集漸進式訓練來達成準確率高，體積小的自定義手勢模型。最終實驗結果 mAP 達到了 0.909，不在資料集內兩位同學實驗結果 mAP 也達到 0.811。 實驗中發現在一個區域蒐集的區域照片不用太多，即可擁有良好的辨識能力。在蒐集資料過程中除了要蒐集識別的手勢，還要蒐集手勢的角度、距離、光線及定義手勢外的手勢才可在學習過程中避免產生誤判。在部署過程中，若疏忽 Coral edgetpu 型號要求、每台照相機不同解析度、廣角、模型輸入尺寸等問題，將導致訓練好的手勢模型準確率下降，針對設備所要求的輸入尺寸大小進行調整，透過遷移學習產生良好的辨識率。

未來將持續實驗蒐集多元化資料集、訓練參數調整、修改神經網路層、程式撰寫改良。最後延續本文手勢模型，完成多元化設備整合 AIoT 物聯網智慧控制系統。

7. 參考文獻

1.游云慈、蘇鈺涵、張昱筠 (2022)，「偵煙偵火在各 YOLO 模型的預測結果與應用」，東華 AI 通訊報雙月刊，第 4 期，頁 1-7。

2.陳響亮、李宜靜、陳文泉、李桂銘 (2018)，「MQTT 聯網架構實現工業物聯網閘道器」，TANET2018 臺灣網際網路研討會，頁 2167 - 2172。

3.洪岳君 (2019)，「ESP8266 智慧溫溼度監控系統」，樹德科技大學電腦與通訊系碩士論文。

4.Yann Lecun, Leon Bottou, Yoshua Bengio & Patrick Haffner (1998). Gradient-based learning applied to document recognition. Proceedings of the IEEE, Vol.86, Issue11.

5.Joseph Redmon, Santosh Divvala, Ross Girshick & Ali Farhadi (2016). You only look once: Unified, real-time object detection. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp.779-788.

6.Joseph Redmon & Ali Farhadi (2017). YOLO9000: Better, Faster, Stronger. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp.7263-7271.

7.Joseph Redmon & Ali Farhadi (2018). YOLOv3: An Incremental Improvement. arXiv:1804.02767

8.Alexey Bochkovskiy, Chien-Yao Wang & Hong-Yuan Mark Liao (2020). YOLOv4: Optimal Speed and Accuracy of Object Detection. arXiv:2004.10934v1

9.YOLOv5 (2020) , in https://github.com/ultralytics/yolov5