智慧影像辨識暨語音水果查價系統
Intelligent Fruit Price Inquiring System with Object Detection and Speech Recognition

謝佳勳 簡浩軒 黃允誠 江振瑞
國立中央大學 資訊工程學系
jordenisalegend@gmail.com, vistashor@gmail.com, fabin1790@gmail.com,
*通訊作者: jrjiang@csie.ncu.edu.tw

摘要

本論文提出一款智能水果查價系統，並以行動設備應用程式(APP)的方式實現。此應用程式使用YOLOv4技術搭配TensorFlow Lite實現透過手機攝像頭對水果的影像辨識，搭配Google Android Speech以語音辨識抓取使用者的語音指令，最後以Google Firebase Realtime Database作為後端資料庫系統，完成智能水果查價應用程式的各項功能。

關鍵詞：影像辨識、語音辨識、資料庫、智能查價、行動設備

Abstract

This paper proposes an intelligent fruit price inquiring system, which is realized as APP running on model devices. The APP utilizes TensorFlow Lite with YOLOv4 as the core technique for object detection. Furthermore, the APP employs Google Android Speech for speech recognition to input user commands, and uses Google Firebase Realtime Database as the backend database to realize the functions of the APP.

Keywords: object detection, speech recognition, database, intelligent price inquiring, mobile device

一、緒論

在現今手機已成為人們不可或缺的生活夥伴，我們可以透過身邊的智慧型手機完成生活周遭大大小小的事情。許多手機應用程式都能輔助日常生活瑣碎問題而為生活帶來便利，例如計算機、記事本、動態公車APP、各種通訊社交軟體等等。

然而家庭主婦買菜時貨比三家，耗時又耗心力，市面上卻缺乏一款為輔助這件日常瑣事量身訂做的蔬果時價查詢應用程式，我們便由此發想開始著手設計。

隨著智慧城市的蓬勃發展，智慧化的生活早已是人們所追求的目標，現在的應用程式多數追求即時快速、操作便利且具有實用性。為了達到上述目標，在本次專案中我們先著重在「水果查價」的部分，以此為主軸設計出結合影像辨識及語音查詢，操作簡易便捷的系統，並以行動設備應用程式(APP)的方式實現。

本論文所提的系統使用YOLOv4 [1]技術搭配TensorFlow Lite [2]實現透過手機攝像頭對水果的影像辨識，搭配Google Android Speech [3]以語音辨識抓取使用者的語音指令，最後以Google Firebase Realtime Database [4]作為後端資料庫系統，結合成為一款智能水果查價應用程式(APP)。

二、系統整體架構及流程

我們的應用程式以Firebase作為後端儲存水果時價資料，前端部分由影像辨識透過手機攝像頭以擴增實境的方式為所有出現在鏡頭前的水果加上標籤，同時常駐語音辨識。

使用者可隨時以語音輸入特定指令，程式便會查詢指定的水果時價並跳轉畫面顯示相關資訊。在顯示資訊的畫面中，使用者一樣能透過語音輸入指令返回影像辨識的畫面。水果查價應用程式的主要操作流程如圖1所示。

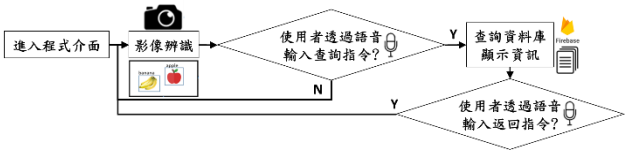

圖1、應用程式主要操作流程圖

三、相關技術

(一) YOLOv4

YOLO(You Only Look Once)為物件偵測(object detection)技術。YOLO第一個版本由Joseph Redmon於2015年6月提出，於2016年及2018年分別推出YOLOv2和YOLOv3，而YOLOv4 [1]則在2020年2月釋出。如圖2所示，物件偵測器大致上可分成兩種，分別為One-stage detector及Two-stage detector，兩種偵測器最主要的差異在於處理預測樣本的方式，

前者針對特徵圖標示出預測結果，後者則需要先標出預測的 Region Proposal，再比對特徵圖以及 Region Proposal 以得到更精準的結果。

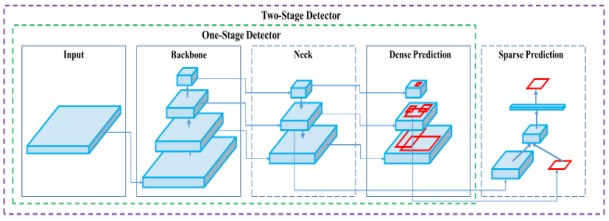

圖 2、偵測器結構圖[1]

若追求效能，模型結構以 One-stage detector 較佳，YOLOv4 便屬於此類。其結構大致上可分成四層[5]，如圖 3 所示 ，包含輸入層用來接收輸入之後送入主幹網路(Backbone)形成特徵圖，接著在當中根據輸入提取不同大小的特徵圖(在 YOLOv4 當中分成 76*76、38*38、19*19 三種大小)，而後再送到脖頸層(Neck)合併不同大小的特徵圖資訊，最後便能生成不同大小的 Prediction。

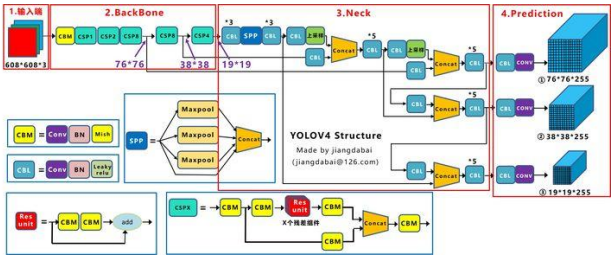

圖 3、YOLOv4 結構圖[5]

YOLOv4 的主幹網路結構使用 CSPDarknet53，而非 YOLO 系列常見的 Darknet 或是其他分類準確率高的網路結構(如 ResNet)。如圖 4 [6]所示，YOLOv4 使用 CPSDenseNet 取代 DenseNet，其中 CSP 代表 Cross-Stage-Partial connections。這是因為 YOLOv4 設計者認為計算量大的原因是網路中的梯度訊息重複過高導致[7]，因此 YOLOv4 便採用論文[7]的優化方向。，先將 Base layer 的特徵映射(Feature Mapping)分成經過卷積及不經過卷積的兩部分，之後再跨層次結合，減少計算量的同時也能盡量保持準確度。

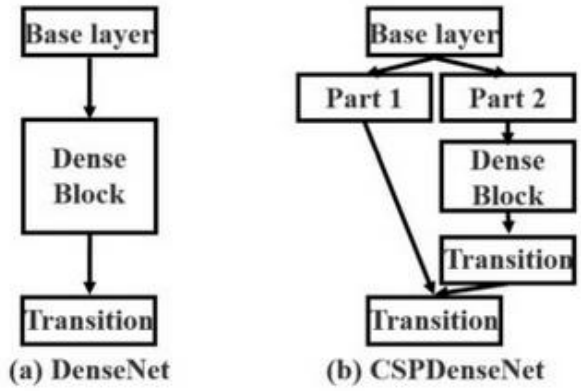

圖 4、YOLOv4 使用 CSPDenseNet 取代 DenseNet [6]

YOLOv4 的 Neck 則是使用 SPP(Spatial Pyramid Pooling)以及 FPN＋PAN(Feature pyramid networks、Path Aggregation Network)，如圖 5 及圖 6 所示。前者的好處是採用多核(1x1、5x5、9x9、13x13)的最大池化取得多種特徵圖之後 concatenate 成輸出，比起單核最大池化更能有效增加主幹網路的感受視野(Receptive Field)，也就是不同特徵圖的代表性資訊。除了物件檢測 Neck 層中常見的 FPN 結構以外，經由 SPP 融合完成的特徵圖再來需要經過 PAN 來產生預測的特徵圖，針對主幹網路中以及來自 SPP 不同大小的輸入可分別整合參數(aggregate parameters)，到 FPN 中能夠針對不同尺寸大小目標的檢測判斷(稱為 semantic augmentation)，而到 PAN 中則能夠加速偵測目標的定位(稱為 instance segmentation)。

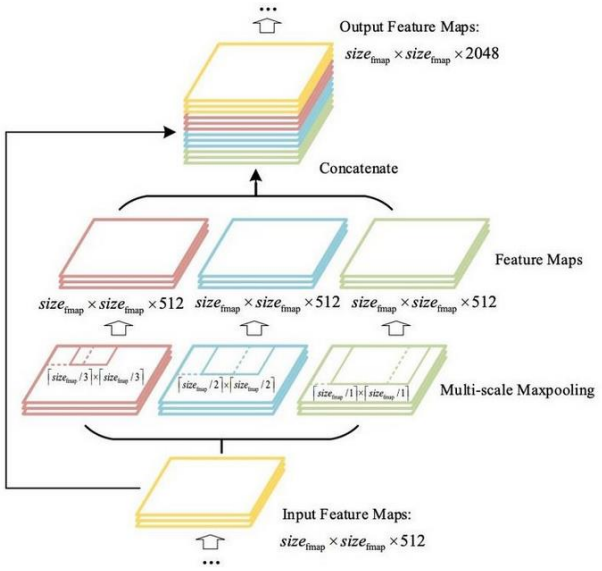

圖 5、SPP 示意圖[6]

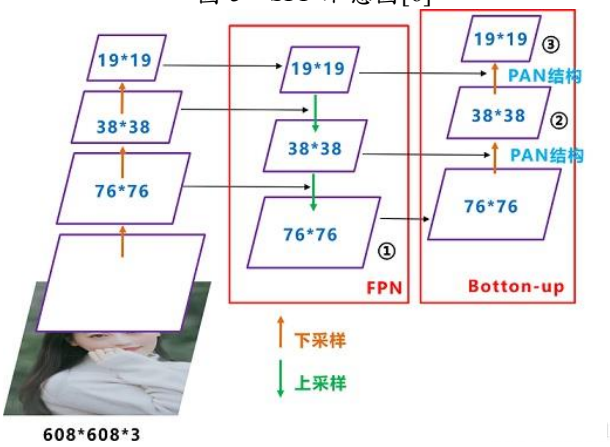

圖 6、FPN＋PAN 示意圖[5]

另外，YOLOv4 還有多項創新改良。首先，YOLOv4 在輸入端中的 Mosaic data augmentation 以 4 張圖片以不同比例合成為一張來訓練，可增進網路的 Robustness，並減少對 Mini-batch 的需求進而減少硬體需求。

再者，YOLOv4 在主幹網路中使用 CSPDarknet53 增強 CNN 學習能力，輕量化的同時保持一定準確度，

降低計算瓶頸與記憶體成本，且還能透過 Mish activation function 提升準確度。

Dropblock 則與 Dropout 相似，都是用來降低 over fitting 的可能。但採用 Dropout 對於卷積層來說影響並不大，因為卷積層通常是由卷積、Activation、Pooling 組成，尤其 Pooling 本來就是對相鄰像素取樣。由圖 7 可看出 Dropblock 拋棄一小區域的做法比起 Dropout 來說對卷積層的影響更大。

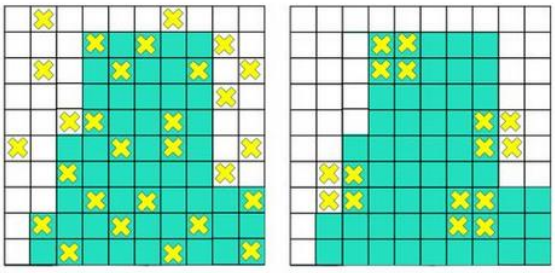

圖 7、Dropout VS Dropblock 差異示意圖[5]

Neck 中的 SPP 比起單純使用 k*k 最大池化的方式更能有效增加主要特徵的接收範圍，明顯分離出最重要的特徵。YOLOv4 論文中在 608*608 的圖形(COCO 資料集)測試時發現以 0.5%的額外代價可增加 AP50 2.7%。FPN + PAN 則從不同的主幹網路深度選擇資料內額外的 block 來增加 receptive field。

Prediction 層中訓練時的損失函數 CIOU Loss 考慮了預測框迴歸函數的三個重要幾何因素：重疊面積、中心點距離、長寬比，可提升預測框迴歸的速度及精度。

預測框篩選則由 NMS 改良為 DIOU + NMS [8]，由於 DIOU 也會考慮中心點距離，由圖 8 可以看出加上 DIOU 更能在物件高度重疊的情況下保留住正確的預測框。

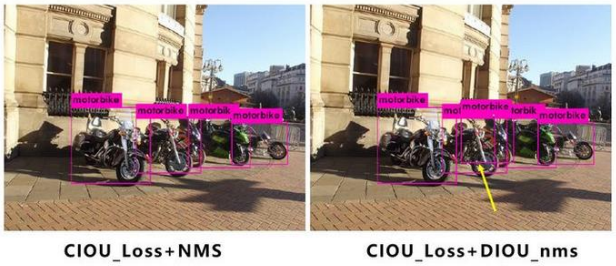

圖 8、有無使用 DIOU 結果比較圖 [8]

(二) Android Speech

Android Speech [3] 是 Google 提供給 Android 應用程式開發人員的眾多 API 之一，支援了語音辨識相關功能的實作。

Android Speech 主要以服務端依照各種事件在 Interface 上進行 callback，開發人員 override Interface 根據不同事件執行特定程式碼的方式實作。支援較高精準度的語音辨識、分辨音量大小，以及在服務就緒、開始辨識、辨識結束、取得結果等關鍵時間點執行特定程式等功能。

但 Android Speech 官方文件明確表示此服務並非針對常駐語音辨識的使用情境設計。為了應用 Android Speech 實現常駐語音辨識我們經歷了一些障礙。

首先我們發現 Android Speech 在開啟語音辨識後，即使沒有任何聲音輸入，一段時間過後會自動關閉，並回傳相關錯誤訊息，即 Timeout。為了實現常駐語音辨識，我們選擇在服務被關閉後直接重新啟動服務，不斷重啟來實現常駐。

接著繼續測試發現應用程式在開啟一段時間後，語音辨識便會失效。推論是短時間重複要求啟動服務，而被服務端視為 spam 拒絕繼續提供服務，因此我們在每次重啟之前加入短暫延遲，對使用者而言無感，但可以避免短時間重複發送請求。

剩下最後一個問題，Android Speech 在開啟語音辨識後僅有兩種方式結束語音辨識並回傳結果：第一，一段時間沒有音訊輸入 Timeout。第二，由程式要求結束語音辨識。如果要等待服務自動 Timeout 可能要等待許久，故一般都會採用由程式在適當時機要求結束的方式，例如按下按鈕開始語音辨識，語音輸入結束時使用者放開按鈕便結束語音辨識。

然而我們的應用程式希望常駐語音辨識，自動偵測關鍵字擷取指令，而不是讓使用者長按按鈕輸入語音。如果等待 Timeout，特別是在環境噪音很多的情境下，可能會持續等待而破壞使用體驗，因此必須找出一個結束語音辨識的適當時機。

解決方法如下：對於 Android Speech 服務端來說語音辨識一直是開啟的，但我們的應用程式會另外記錄一個「實際的語音辨識狀態」。設定一個較高的「語音門檻」及較低的「環境音門檻」，當音訊輸入音量高於語音門檻便進入「聆聽狀態」，而在聆聽狀態下若音量降低至環境音門檻以下且持續一小段時間，則直接結束語音辨識並取得結果。

最後透過特化的流程我們成功將 Android Speech 調整至實現了常駐的語音辨識技術，以偵測關鍵字的方式讀取使用者的語音指令，且一旦語音結束便自動執行後續動作。圖 9 展示此部分的流程：

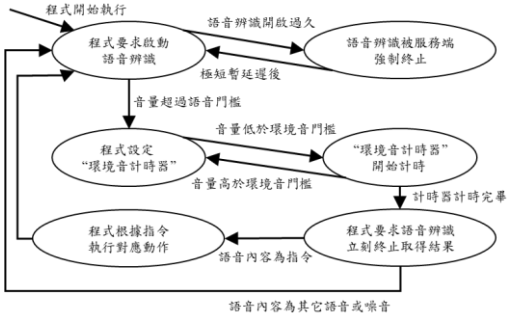

圖 9、Android Speech 改造為常駐語音辨識之狀態機

(三) Firebase

Google 在 2014 年開始支援 Firebase [4]的開發，Firebase 的定位在提供行動應用程式開發的後端及服務 BaaS (Backend as a Service)，讓行動應用程式的開發者能有一個不需要考量效能需求和功能驗證的後台。且 Firebase 支援即時資料庫管理、雲端儲存、數據分析、身分驗證等功能，提供開發者一個非常有效的開發平台。

本論文所提的應用程式使用 Firebase 中的 Realtime Database 來儲存由農委會 API [9]取得的資料。Realtime Database 與很多開發者常用的關聯式資料庫 SQL 有很大的不同，在使用前並沒有任何初始化的動作，也沒有建立 table 欄位的問題，是一種 no SQL 資料庫。

Realtime Database 本體結構是一個巨大的 JSON tree，資料之間的層次通過 JSON 的巢狀來體現，這種結構的好處是可以很簡單的通過節點清楚查詢到指定內容，方便分散和減少查詢擴充，且改變的時候不必針對 database 設定。

Realtime Database 的 SDK 功能包括「監聽」，一旦開啟應用，當資料庫更新時可以即時的同步到主機上，且支援離線功能、同步速度快，這種特性也相當契合即時通訊應用。

此外，Realtime Database 支援 IOS、Android、甚至 JavaScript 同步到網頁上等等，應用層面廣，也有利於日後擴充平台。

四、應用程式實作成果

以下展示本論文所提應用程式的各項功能：

(一) 進入讀取介面

開啟應用程式時，影像辨識的模組需要一段時間初始化，此時顯示進入畫面，如圖 10 所示。

圖 10、應用程式啟始進入畫面

(二) 影像辨識

讀取完畢後，進入應用程式。此時顯示影像辨識介面，中央為手機攝像頭的畫面，並自動給所有偵測到的水果加上標籤，如圖 11 所示。

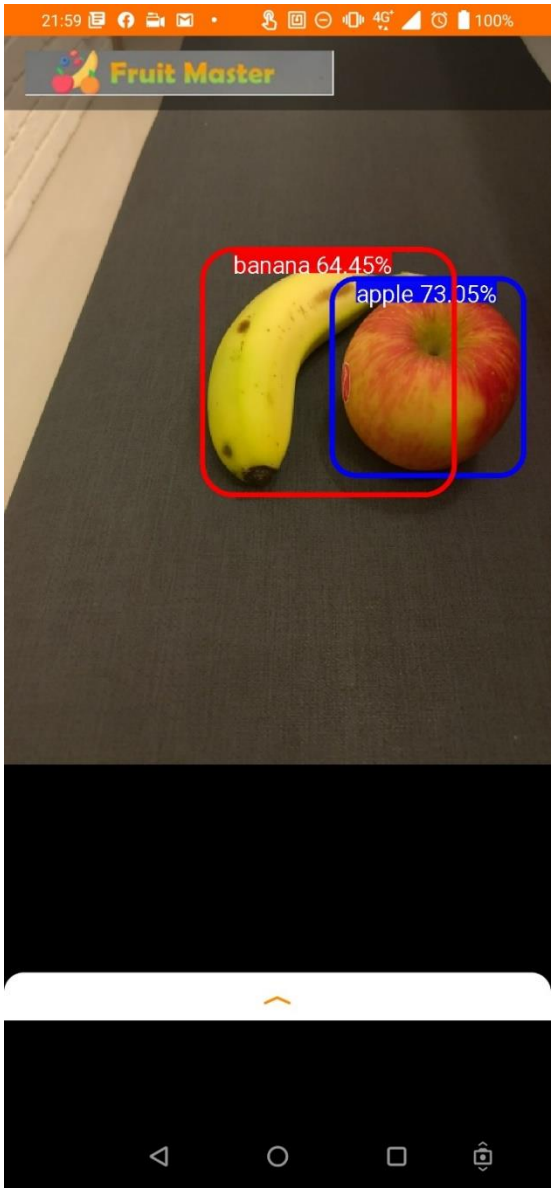

圖 11、應用程式影像辨識畫面

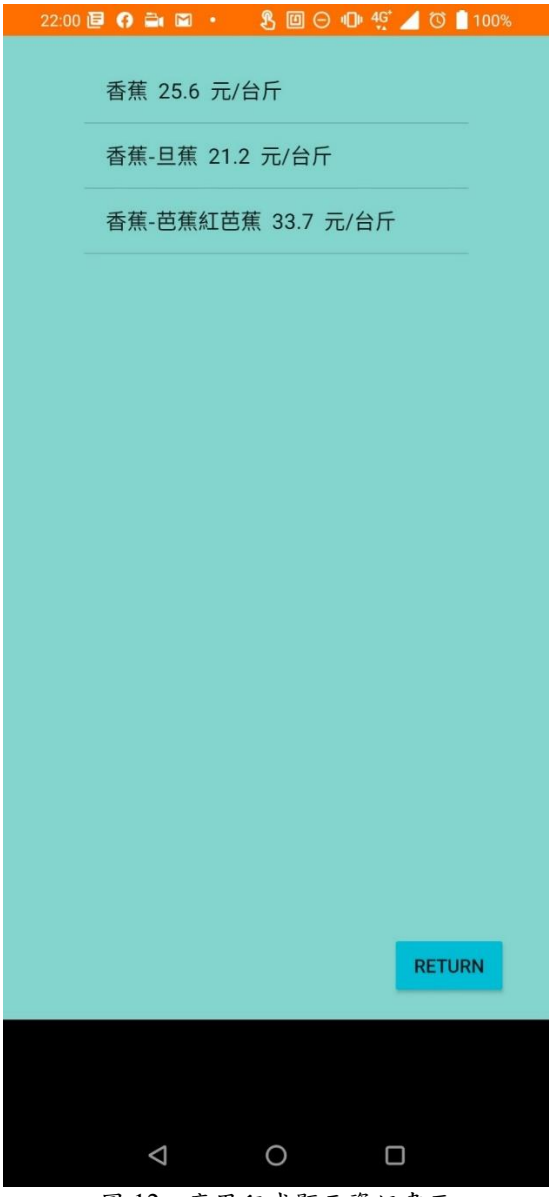

圖 12、應用程式顯示資訊畫面

(三) 語音輸入查詢

此時使用者說出「查詢北部的香蕉」，應用程式便向資料庫取得相關信息並顯示，如圖 12 所示。在此介面說出「返回」或是手動點擊返回按鈕都能夠返回影像辨識介面。

五、結論

我們所提出的智慧影像辨識暨語音水果查價系統結合 YOLOv4 物件偵測方法來達到即時快速的識別水果種類，並且透過 Android Speech 實現常駐語音輸入達到操作便利，最後搭配 Firebase 儲存由農委會 API 取得的水果價格資料，完成快速的水果交易資訊查詢。

參考文獻

[1] Alexey Bochkovskiy, Chien-Yao Wang, and Hong-Yuan Mark Liao, “YOLOv4: Optimal Speed and Accuracy of Object Detection,” *Cornell University arXiv*, Computer Science, Computer Vision and Pattern Recognition, 23 April 2020.

[2] TensorFlow Lite for Android with Java, *TensorFlow official website*, API, TensorFlow Lite, https://www.tensorflow.org/lite/api_docs/java/org/tensorflow/lite/package-summary, last accessed in July 2021.

[3] Google android.speech, *Android Developers official website*, Docs, Reference, Android Platform, https://developer.android.com/reference/android/speech/package-summary, last accessed in July 2021.

[4] Google Firebase Realtime Database, *Firebase official website*, Docs, Build, https://firebase.google.com/docs/database, last accessed in July 2021.

[5] 江大白, “深入浅出 Yolo 系列之 Yolov3&Yolov4&Yolov5 核心基礎知識完整講解,” https://zhuanlan.zhihu.com/p/143747206, last accessed in July 2021.

[6] Jonathan Hui, “YOLOv4,” Medium, https://jonathan-hui.medium.com/yolov4-c9901eaa8e61, last accessed in July 2021.

[7] Chien-Yao Wang, Hong-Yuan Mark Liao, Yueh-Hua Wu, Ping-Yang Chen, Jun-Wei Hsieh, and I-Hau Yeh, “CSPNet: A New Backbone that can Enhance Learning Capability of CNN,” *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 14-19 June 2020.

[8] Zhaohui Zheng, Ping Wang, Wei Liu, Jinze Li, Rongguang Ye, and Dongwei Ren, “Distance-IoU Loss: Faster and Better Learning for Bounding Box Regression,” *Cornell University arXiv*, Computer Science, Computer Vision and Pattern Recognition, 19 November 2019.

[9] 農委會農產品交易行情 API, *農業開放資料服務平台*, API 專區 API 文件, https://agridata.coa.gov.tw/apidocs.aspx, last accessed in July 2021.