**「基於 YOLOv9 的食物物件偵測與熱量估算系統」**

第一作者姓名:陳辰

第一作者單位:中央大學資管2b

[brianchen989@gmail.com](mailto:brianchen989@gmail.com)

【摘要】

藉由拍照辨識食物，並以參照物計算體積並計算熱量

【關鍵詞】

圖片辨識,資料預處理,資料分析,資料庫,熱量計算

**1. 引言（Introduction）**

**1.1.背景：**

行動裝置與智慧型相機普及，使得以影像自動化辨識食物並估算熱量成為可行的健康管理工具。現有方法多依賴人工輸入或條碼，缺乏對餐盤中多種食物的自動化且即時評估能力。

**1.2.研究動機：**

自動化系統可減少使用者負擔、提高飲食追蹤的頻率與準確性；若能在手機端即時回饋熱量，對減重或慢性病管理有實際助益。

**1.3.研究目標與研究問題:**

**1.3.1.如何使用圖片辨識**

將圖片以檔案的形式放入並設定參數後能求得圖片結果和其他輸出參數

**1.3.2.YOLO和opencv比較**

Opencv分為定位和辨識兩個階段,而yolo只需一個階段

**1.3.3.YOLO如何使用**

放入資料集,修改檔案,輸入參數和想辨識的檔案並能得到輸出結果

**1.3.4.資料集如何訓練**

labelImg: open dir中放入自己想標註的圖片資料夾change dir中放想要儲存至的資料夾(快捷鍵:w創建矩形標註框d下一張)

roboflow:創建帳號，dataset,newdataset(此網站附有智能標註功能可以試試)

**1.3.5.公用資料集的使用和規範**

每個公用資料集（public dataset）都會有授權條款（license）。常見類型：

Permissive（允許商用與改作）：例如 MIT、Apache 2.0。

Copyleft / Share-alike：例如某些 Creative Commons（CC BY-SA），要求衍生作品相同授權。

Non-commercial（禁止商用）：例如 CC BY-NC。

資料特定條款：例如標註不能重新分發、需申請授權、僅限研究用途等。

**1.3.6.如何提高訓練準確度**

一 提升標註精準度

二 增加資料集

三 提高訓練輪數

四 提高參數中訓練解析度

**1.3.7.如何提高訓練速度**

一 提升硬體設備

二 用cpu協助處理

三 降低訓練輪數

四 降低參數中訓練解析度

**1.3.8.平面食物的熱量計算**

目前使用方法:

估算面積:取得 YOLO 偵測到的 Bounding Box，計算寬\*高得到像素面積。

估算重量:將像素面積乘以一個固定的係數(factor)來換算成公克。目前factor=0.002

**下方為還未解決的目標**

**1.3.9.有參照物的情況下如何判斷大小**

目前想法:藉由尺這個標的物能提升體積判斷準確性

**1.3.10.如何將其安裝至linebot**

之前學過aiagent想試試能不能以此為基底實作

**2. 相關工作**

**2.1. 物件偵測演算法比較:**

two-stage vs one-stage參考文獻: 深度學習-物件偵測:You Only Look Once (YOLO)

**2.2. 食物辨識與熱量估算相關研究回顧**

**2024-2025 重要研究成果摘要**

| 研究年份 | 核心技術 / 模型 | 研究貢獻與特點 |
| --- | --- | --- |
| 2025 | YOLOv8 + ONNX | 紐約大學開發的 AI 掃描器，強調即時性與輕量化，可在手機端離線運行。 |
| 2024 | Hybrid CNN-ML | 結合 CNN 特徵提取與機器學習（Random Forest/XGBoost）處理複雜的烹飪變數（如油量）。 |
| 2024 | 3D 立體深度學習 | 利用手機的 3D 感測器提升份量估算的精確度，解決傳統 2D 照片「看不出厚度」的問題。 |
| 2023 | Transformer-based | 優化輕量化模型，在提升辨識精確度的同時，降低對行動裝置運算資源的消耗。 |

**2.3. 資料集概況**

<https://www.kaggle.com/datasets/kmader/food41>

(僅參考 未使用)

<https://universe.roboflow.com/foodrecognitionprethesis/food-items-j9iud>

(僅參考 未使用)

<https://universe.roboflow.com/yaman-e/food-ingredients-detection-qfit7>

(用來訓練基礎辨識)

[**https://universe.roboflow.com/frutas1-rx6mn/apple-hfvgr**](https://universe.roboflow.com/frutas1-rx6mn/apple-hfvgr)

(用來訓練蘋果辨識)

[**https://universe.roboflow.com/practicas-nlzno/ruler-**](https://universe.roboflow.com/practicas-nlzno/ruler-)**vylu**(用來訓練尺辨識)

**2.4. 與其他研究的比較：**

[**https://chtseng.wordpress.com/2020/09/10/%E9%A3%9F%E7%89%A9%E5%81%B5%E6%B8%AC%E5%8F%8A%E8%BE%A8%E8%AD%98/**](https://chtseng.wordpress.com/2020/09/10/%E9%A3%9F%E7%89%A9%E5%81%B5%E6%B8%AC%E5%8F%8A%E8%BE%A8%E8%AD%98/)僅有圖片辨識

<https://ithelp.ithome.com.tw/m/articles/10395445>

直接將圖片中食物丟入後端大語言模型計算熱量

**2.5.硬體設備:**

顯卡:rtx3050ti

CPU:Ryzen7 6800h 16核

記憶體:16g

**2.6.開發環境:**

主要環境:VScode

其他環境(協助用):colab,antigravity

**3. 方法**

**3.1.流程圖總結:**

**3.2 資料收集與標註:**

一開始是直接拍照，後來發現至少要百張以上便改用爬蟲抓網路上的照片

**3.3 資料預處理與增強:**

原本用labelIMG後來用Roboflow，最後時間不足改用公用dataset(Kaggle,roboflow universe).

**3.4 YOLOv9 模型架構與訓練設置**

**3.4.1. 模型架構**

\*\*基礎模型\*\*: YOLOv9

\*\*模型變體\*\*: `yolov9-c` (Compact 版本)

\*\*主要特徵\*\*:

\*\*PGI (Programmable Gradient Information)\*\*: 可程式化梯度資訊，用於解決深層網路中的資訊瓶頸問題，確保梯度能有效傳遞並更新參數。

\*\*GELAN (Generalized Efficient Layer Aggregation Network)\*\*: 廣義高效層聚合網路，結合了 CSPNet 和 ELAN 的優點，提供輕量級且高效的特徵提取能力。

\*\*設定檔\*\*: `models/detect/yolov9-c.yaml`

定義了 Backbone (骨幹網路) 與 Head (偵測頭) 的結構。包含輸入通道數 (ch)、類別數 (nc) 以及錨點 (anchors) 設定。

**3.4.2.訓練設置 (Training Configuration)**

基本放在remindme.md中

\*\*訓練腳本\*\*: `train\_dual.py` (支援雙流/輔助訓練邏輯)

\*\*預訓練權重 (Pre-trained Weights)\*\*: `weight/yolov9-c.pt` (載入預先訓練好的權重以加速收斂)

\*\*資料集設定 (Data Configuration)\*\*: `data/food\_datasets/data.yaml`

    包含訓練集、驗證集路徑。

    定義食物類別名稱與數量。

### 超參數 (Hyperparameters)

\*\*Epochs (訓練輪數)\*\*: `20` (訓練總共進行20個世代)

\*\*Batch Size (批次大小)\*\*: `4` (依據顯示卡記憶體調整，較小的值可減少記憶體佔用但可能影響 Batch Normalization)

\*\*Image Size (圖片尺寸)\*\*: `416` (輸入圖片被調整為 416x416 像素進行訓練)

\*\*Device (裝置)\*\*: `0` (使用第一張 GPU 進行訓練)

\*\*超參數設定檔\*\*: `hyp.scratch-high.yaml`

定義了學習率 (lr), 動量 (momentum), 權重衰減 (weight decay) 等詳細參數，針對從頭訓練或微調進行了優化。

**3.4.3. 推論/偵測設置 (Inference/Detection Settings)**

參數基於 `remind.md` 中的偵測指令整理：

\*\*偵測腳本\*\*: `detect\_dual.py`

\*\*客製化功能\*\*: 已整合 `img\_to\_calo` 模組，在偵測時會同步計算並顯示食物的估計重量與熱量。

\*\*權重來源\*\*:`runs\train\exp18\weights\best.pt` (使用訓練實驗 exp18 中表現最好的權重)

\*\*信心閾值 (Confidence Threshold)\*\*: `0.30` (只顯示信心分數大於 30% 的偵測結果)

\*\*輸入來源\*\*: 圖片檔案 (例如 `.../train/images/....jpg`)

**3.4.4.功能擴充：熱量計算系統**

YOLOv9 的基礎上擴充了熱量計算功能：

1.  \*\*物件偵測\*\*: YOLOv9 識別食物類別 (如 Apple) 與位置 (Bounding Box)。

2.  \*\*像素面積計算\*\*: `weight\_calcu.py` 計算 Bounding Box 的面積。

3.  \*\*重量估算\*\*: `weight\_calcu.py` 使用係數 (目前設為 `0.002`) 將像素面積轉換為公克數。

4.  \*\*熱量查詢\*\*: `calculate.py` 根據食物名稱查詢資料庫 (`database.py`)，並依據重量計算總熱量。

5.  \*\*結果顯示\*\*: `detect\_dual.py` 將 "類別 + 信心度 + 重量 + 熱量" 標註在輸出圖片上。

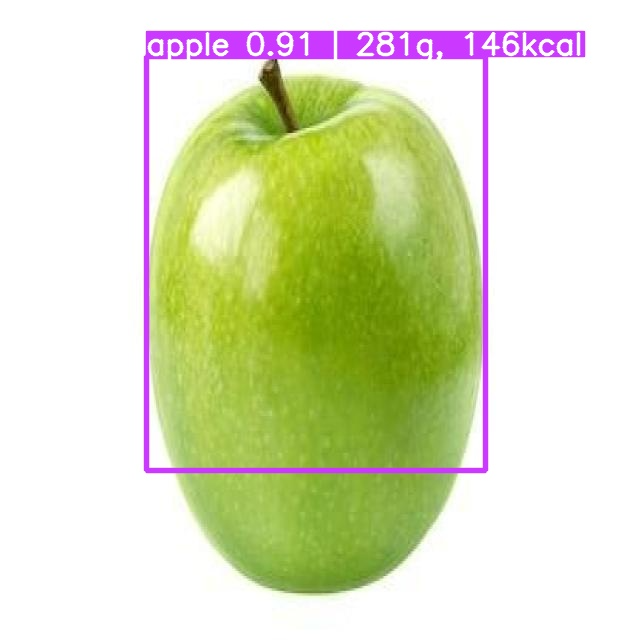
**3.5 份量/重量估計方法（若有）**

估算面積:取得 YOLO 偵測到的 Bounding Box，計算寬\*高得到像素面積。

估算重量:將像素面積乘以一個固定的係數(factor)來換算成公克。目前factor=0.002

**3.6 後續處理與輸出**

圖一中顯示1.食物種類2.信心度3.重量4.熱量

****

**(圖一)**

**4. 實驗（Experiments）**

**4.1. 實驗設計**

**4.1.1.圖片辨識準確度測試**

目的:希望評估 YOLOv9 模型在食物影像上的辨識能力，確認模型是否能正確辨識圖片中的食物種類與位置。

實驗方法:

一.使用訓練完成之 YOLOv9-c 模型（best.pt 權重）進行測試。

二.測試圖片來源包含：訓練資料集中未參與訓練的驗證圖片

三.網路隨機下載之資料集透過 detect\_dual.py 進行推論，並設定信心閾值（Confidence Threshold）為 0.30。

四.系統輸出每張圖片中偵測到的食物類別、Bounding Box 位置與信心分數。

實驗結果說明: 實驗結果顯示，模型對於訓練過之食物圖片具有良好的辨識能力，在單一食物且背景較單純的情況下，可穩定框出食物並給出合理的信心分數。然而，在光線不足或背景較為複雜時，仍可能出現信心度下降或誤判的情形。

**4.1.2.熱量計算準確度測試**

目的: 評估系統所提出之熱量估算方法，確認由影像推算之重量與熱量是否具有合理性

實驗方法:

一.使用 YOLOv9 偵測食物後，取得每個食物的 Bounding Box。

二.計算 Bounding Box 的像素面積（寬 × 高）。

三.透過設定之換算係數（factor = 0.002），將像素面積轉換為食物重量（公克）。

四.根據資料庫中所定義之「每 100 公克熱量值」，計算該食物的估計熱量。

五.將系統估算結果與常見食物實際熱量範圍進行比較。

實驗結果說明:

實驗初期使用簡易重量估算方式時，計算出的重量與熱量誤差較大，顯示單純以像素面積進行估算仍有明顯限制。  
在調整重量估算程式後，系統所計算之重量與熱量已大致落於合理範圍內，證明該方法具備基本可行性，但仍需進一步改良以提升準確度。

**4.1.3.訓練速度測試:**

目的:評估 YOLOv9 模型於本研究環境下之訓練效率，分析模型在有限硬體資源條件下的訓練速度與實用性。

實驗方法

一.本實驗使用 YOLOv9-c（Compact）模型進行訓練，以降低模型複雜度並提升訓練效率。

二.訓練環境為個人電腦，使用 NVIDIA 顯示卡進行 GPU 加速運算。

訓練參數設定如下：

>> --batch 4 `

>> --epochs 100 `

>> --img 416 `

>> --min-items 0 `

>> --data data/food\_datasets/data.yaml `

>> --weights weight/yolov9-c.pt `

>> --cfg models/detect/yolov9-c.yaml `

>> --hyp hyp.scratch-high.yaml

三.使用 train\_dual.py 進行模型訓練，並記錄每個 epoch 所需之訓練時間與整體訓練完成時間。

得到26epoches/3hrs

實驗結果:

使用 YOLOv9-c 模型可在有限的硬體資源下完成訓練，整體訓練時間維持在可接受範圍內。隨著訓練進行，模型能穩定收斂，且各 epoch 所需時間差異不大，顯示訓練流程具有良好的穩定性。

相較於較大型的模型版本，YOLOv9-c 在訓練速度與模型效能之間取得良好平衡，使其適合應用於本研究所提出之食物辨識系統。

**4.2 訓練細節與結果**

訓練細節

本研究採用 YOLOv9-c（Compact）模型作為主要架構，進行食物影像辨識模型之訓練。為兼顧訓練效率與辨識效能，訓練過程中採用較小模型規模，以適合一般個人電腦環境。

訓練資料集包含多張已標註之食物影像，並依照 YOLO 格式進行資料前處理與標註轉換。訓練過程使用 GPU 進行加速運算，相關訓練參數設定如下：

* 模型架構：YOLOv9-c
* Batch size：4
* Epoch 數量：20
* 影像輸入大小：416 × 416
* 訓練程式：train\_dual.py
* 其他超參數皆採用 YOLOv9 預設設定

在訓練過程中，系統會即時顯示每個 epoch 的訓練損失（Loss）變化，以觀察模型是否逐漸收斂。

**訓練結果**

訓練結果顯示，模型在訓練初期損失值下降速度較快，顯示模型能迅速學習影像中的基本特徵；隨著訓練 epoch 增加，損失值逐漸趨於穩定，代表模型訓練過程具備良好的收斂特性。

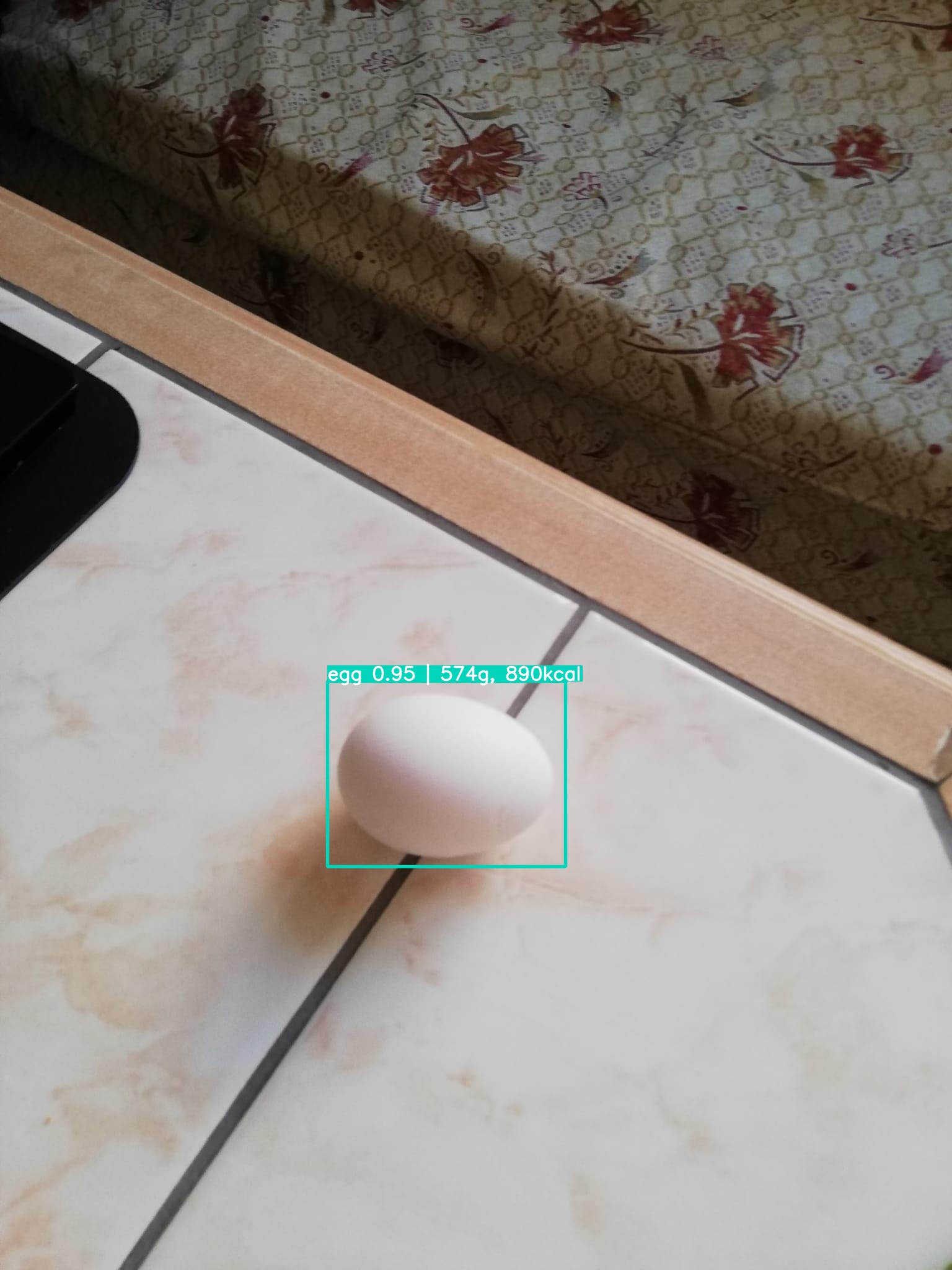
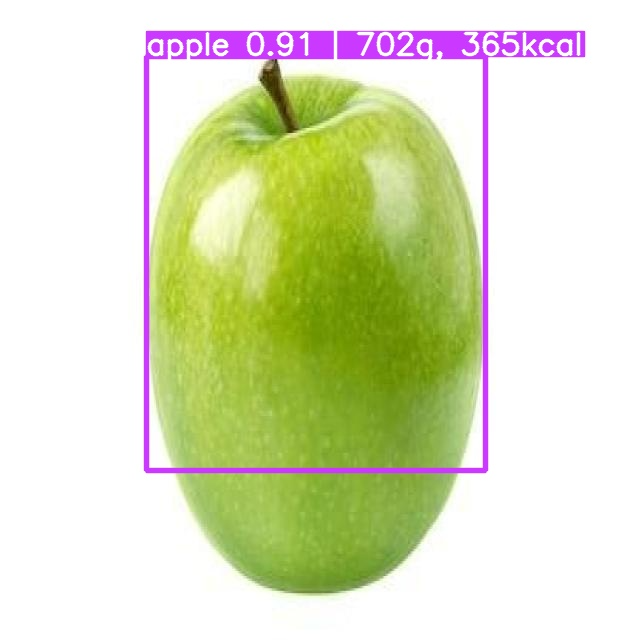
在完成 20 個 epoch 的訓練後，系統自動儲存辨識效果最佳之模型權重（best.pt），並用於後續圖片辨識與熱量估算實驗。實際測試結果顯示，模型能正確辨識訓練過之食物類別，並能在多數測試圖片中成功框出食物位置，顯示訓練成果具備實際應用價值。

整體而言，本研究所訓練之 YOLOv9-c 模型在有限訓練次數與硬體資源條件下，仍能達到穩定且可用的辨識效果，證明其適合作為本研究系統之核心模型。

雖然訓練資料數量有限，模型仍能達到合理的辨識結果，顯示 YOLOv9 在小型資料集情境下具備一定的泛化能力。

**4.3 熱量估算結果（若有）**

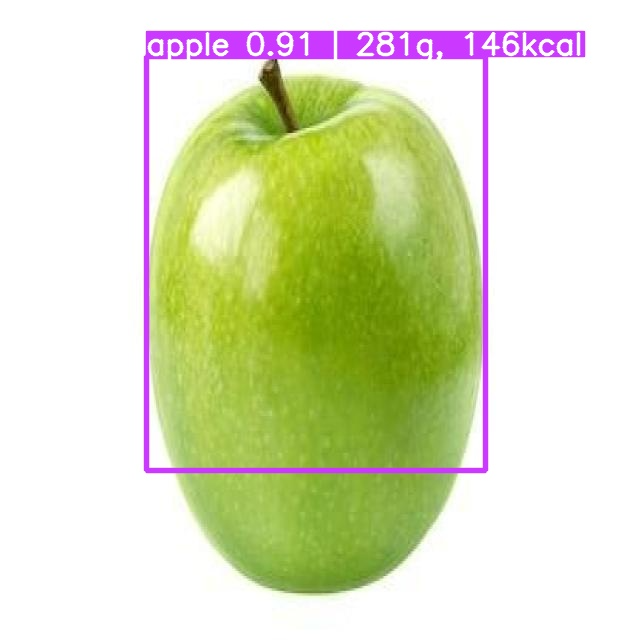
第一次:用簡易方法得到圖二,圖三

 ****

圖二 圖三

重量熱量皆錯誤，需改變重量計算腳本

2.重量估算修改後:大約正確如圖四，圖五

**** ****

圖四 圖五

**5. 討論（Discussion）**

本研究成功實作一套結合 YOLOv9 之食物辨識與熱量估算系統，並驗證其在影像辨識與基礎熱量估算上的可行性。

在圖片辨識方面，YOLOv9 能夠即時且準確地偵測食物位置，相較於傳統 OpenCV 需分階段處理的方式，YOLO 採用 one-stage 架構，使整體流程更為簡潔且效率更高。然而，模型辨識效果仍受到資料集多樣性與影像品質影響，當背景複雜或光線不佳時，辨識信心度可能下降。

在熱量估算方面，目前採用以 Bounding Box 像素面積推估重量的方式，雖可作為初步估算，但尚未考慮食物高度、厚度及拍攝距離等因素，因此仍存在誤差。此結果也呼應相關研究指出，僅使用 2D 影像進行份量估算具有先天限制。

此外，本研究嘗試加入參照物（如尺）作為未來改進方向，顯示系統具有良好的擴充性，未來可結合尺度校正或 3D 深度資訊，以進一步提升重量與熱量估算的準確性。

整體而言，本系統在「自動化食物辨識」與「即時熱量回饋」方面已具備實用雛形，適合作為飲食紀錄或健康管理應用之基礎。

本研究仍存在若干限制，例如資料集規模有限、食物種類尚未齊全，且熱量估算僅基於 2D 影像進行推算，未考慮食物厚度與拍攝距離等因素。未來若能結合更多資料或深度感測技術，將有助於提升系統整體準確度。

**6. 未來工作（Conclusion & Future Work）**

增加可辨認食物

增進模型效率

改進重量計算方式

實作成aiAgent,(lineBot)

**7. 學習心得與反思（Learning Reflection）**

其實一開始我只是因為興趣而開始嘗試圖片辨識，而這次的期末報告有給予了我更多的資源去實際操作，(這裡先要感謝老師們每次都耐心回答我的問題)這是我第一次實際去做圖片辨識，之前因為不知道怎麼樣開始誘惑是時間的不充足導致計畫一延再延。因為我又是一格完全的新手所以基本上都是一直遇到"實作"然後發生小問題，解決之後又發生了其他小問題，如果是在之前沒有大語言模型可以詢問時可能只能自己嘗試花費大量時間解決，但現在的科技剛好可以彌補這一點。因此我才能在時間內做出一個還有很多地方可以改善但可以實際運行的一套系統。

我一開始僅認為套用現成模型即可完成辨識任務，但在實際操作後才發現，資料集品質、標註方式與訓練參數設定，對模型效果皆有顯著影響。

在資料蒐集與標註階段，曾嘗試自行拍攝圖片進行訓練，但由於資料量不足，導致模型表現不佳，後續改採公用資料集並搭配 Roboflow 工具，才使訓練結果逐漸穩定。此過程讓我體會到資料在深度學習中的重要性，模型並非唯一關鍵。

此外，在熱量估算部分，原先期望能以簡單方式準確計算食物重量，但實作後發現僅使用 2D 影像進行估算存在先天限制。透過不斷調整計算方式，雖能使結果趨於合理，但也更清楚認知到理論與實務之間的落差。

而在這次的過程中我知道重點不是你夠不夠厲害，而是你有沒有勇氣去嘗試,去試錯，有沒有耐心去解決，去學習。如果以上皆具備了，雖然不一定能成為最厲害的，但能像我一樣享受這部段探索的過程。

**8. 參考文獻（References）**

【python】opencv 2小時初學者教學 ｜ 影像辨識 ｜ 影像處理 ｜ 人臉辨識 ｜ 電腦視覺

https://www.youtube.com/watch?v=xjrykYpaBBM

深度學習-物件偵測:You Only Look Once (YOLO)

https://chih-sheng-huang821.medium.com/%E6%B7%B1%E5%BA%A6%E5%AD%B8%E7%BF%92-%E7%89%A9%E4%BB%B6%E5%81%B5%E6%B8%AC-you-only-look-once-yolo-4fb9cf49453c

【YOLOV9】YOLOV9 實作指南：從零開始的手把手教學 | You Only Look Once | 沒有困難的理論只有快速的實作教學 | 使用自己的資料集訓練辨識

<https://www.youtube.com/watch?v=tMwyxKttZd0>

Day06：Roboflow實作資料集（一）：標註入門

<https://ithelp.ithome.com.tw/m/articles/10373564>

快速上手YOLO：利用 Roboflow 和 Ultralytics HUB 完成模型訓練與管理(上)

https://medium.com/@andy6804tw/%E5%BF%AB%E9%80%9F%E4%B8%8A%E6%89%8Byolo-%E5%88%A9%E7%94%A8-roboflow-%E5%92%8C-ultralytics-hub-%E5%AE%8C%E6%88%90%E6%A8%A1%E5%9E%8B%E8%A8%93%E7%B7%B4%E8%88%87%E7%AE%A1%E7%90%86-%E4%B8%8A-37acd110a8a0

參考dataset

Food Images (Food-101)

<https://www.kaggle.com/datasets/kmader/food41>

Food Items Computer Vision Model

<https://universe.roboflow.com/foodrecognitionprethesis/food-items-j9iud>

FOOD INGREDIENTS DETECTION Computer Vision Model

https://universe.roboflow.com/yaman-e/food-ingredients-detection-qfit7

apple

https://universe.roboflow.com/frutas1-rx6mn/apple-hfvgr

Ruler Computer Vision Dataset

<https://universe.roboflow.com/practicas-nlzno/ruler-vyluh>

watermelon

<https://universe.roboflow.com/ikun-klgdt/watermelon-lyddt>

strawberry

<https://universe.roboflow.com/-fqcxu/strawberry-p7nq9>

pineapple

<https://universe.roboflow.com/yadhu-krishnan-k-a-tmpt5/pineapple-xooc7>

peach

<https://universe.roboflow.com/talovpracticas/peach-kkgym>

fruit

<https://universe.roboflow.com/fruit-5mit8/orange-kxh32>

kiwi

[https://universe.roboflow.com/ikun-x51rs/ kiwi -p3s1h](https://universe.roboflow.com/ikun-x51rs/kiwi-p3s1h)

grape

https://universe.roboflow.com/frutas1-rx6mn/grape-gayym

食物偵測及辨識

https://chtseng.wordpress.com/2020/09/10/%E9%A3%9F%E7%89%A9%E5%81%B5%E6%B8%AC%E5%8F%8A%E8%BE%A8%E8%AD%98/