國 立 清 華 大 學

工業工程與工程管理學系

智慧化企業整合

Intelligent Integration of Enterprise

食品影像辨識

第六組

112034564 曾聖閔

112034554 陳　薪

112034556 莊傑宇

指導教授：邱銘傳 教授

中 華 民 國 一 一 三 年 六 月

目錄

一、簡介

1.1 背景與動機

本次專案以滷味店食品影像辨識為主題，旨在建立一個高度資訊化整合的系統，通過人工智慧技術幫助餐飲業者在食品管理和顧客服務上獲得更大的競爭優勢。隨著科技的進步，人工智慧在餐飲行業的應用越來越廣泛，例如智能點餐系統、食品品質檢測等。食品影像辨識技術，作為其中的重要一環，可以用來提高食品管理的效率和準確性，並且能夠提供更加個性化的顧客服務。

目前，許多餐飲業者仍然依賴於傳統的手動方式來管理食品，這種方式不僅效率低下，而且容易出現錯誤。隨著市場需求的變化和顧客期望的提升，如何提高食品管理的效率和精確性，並提供優質的顧客服務，成為餐飲業者面臨的重要挑戰。針對這一挑戰，我們計劃開發一個基於深度學習的食品影像辨識系統，能夠自動識別和分類各類食品，從而提升餐飲業者的管理效率和服務水平。

在這個專案中，我們將使用一個涵蓋多種類型食品的影像數據集，通過訓練深度學習模型，實現對不同食品的精確識別和分類。這樣的系統可以幫助餐飲業者自動化管理食品數據，減少人力成本，並且可以提供更準確的訂單預測，幫助管理者更好地規劃食品庫存和供應，避免供應不足或浪費的情況發生。

此外，我們還希望通過這個系統提升餐飲業者的整體運營效率。自動化食品分類和智能訂單處理系統可以使餐飲業者更快地響應顧客需求，提供更高效的服務。同時，通過數據分析技術，業者可以更好地了解消費趨勢和顧客行為，從而制定更加精準的營銷策略，提升顧客滿意度和忠誠度。

總結而言，本次專案的目標是通過建立高度資訊化整合的食品影像辨識系統，利用人工智慧技術實現食品的自動化分類和智能訂單處理，幫助餐飲業者提高運營效率，降低成本，並提供更優質的顧客服務。這樣的系統將為餐飲業者在競爭激烈的市場中提供更大的競爭優勢，幫助其實現持續增長和發展。

1.2 發想與目的

在構思這個專案時，我們注意到餐飲業在日常運營中經常面臨一些挑戰，包括食品管理的繁瑣過程和訂單處理效率低下。基於這些問題，我們認為可以通過食品影像辨識技術來實現自動化管理，進一步提升效率和服務質量。

首先，我們分析了餐飲業者的需求，確定了食品影像辨識技術作為解決方案的核心。通過使用影像辨識模型，我們可以自動識別不同類型的食品，並將其分類和記錄。這樣的自動化流程不僅能夠減少人工操作的時間，還能提高分類的準確性，減少人為錯誤。

其次，我們計劃利用分類模型和數據分析技術，對食品訂單數據進行深入分析和預測。通過分析歷史訂單數據，我們可以預測未來的訂單趨勢，幫助餐飲業

者提前準備，確保供應充足並減少浪費。這樣的預測能力不僅能提高業者的運營效率，還能有效控制成本，提升經營效益。

此外，透過這個資訊化整合系統，我們希望餐飲業者能夠更好地管理食品和訂單，提高整體運營效率。這樣的系統不僅能幫助餐飲業者在競爭激烈的市場中保持競爭優勢，還能提供穩定且優質的顧客服務，增強顧客滿意度和忠誠度。

總結而言，我們的目標是通過食品影像辨識技術和數據分析，幫助餐飲業者實現食品管理的自動化和智能化，提升運營效率和服務質量，並在市場競爭中獲得更大的優勢。這樣的系統將有助於餐飲業者持續發展並實現長期增長。

表 1 5W1H

What	食品影像及其分類結果。
Who	餐飲業者，包括滷味店、餐廳的管理人員。
Where	餐廳後台管理系統和櫃台結帳。
When	在需要對食品進行自動分類和管理時，以及在訂單高峰期進行快速處理時。
Why	傳統的手動分類方式效率低，容易出錯，無法滿足現代餐飲業對高效管理的需求。
How	利用深度學習模型進行食品影像識別，實現自動分類和管理。

二、資料集介紹

2.1 資料來源與選擇

在本專案中，我們選用了來自 Foodcam 的 UEC FOOD 100 Ver 1.0 資料集。這是一個專為食品影像辨識任務設計的開源資料集，旨在促進計算機視覺技術在餐飲業中的應用。選擇這個資料集的原因包括其涵蓋了多種類型的食品，標註信息詳細且準確，並且提供了高品質的影像資料，這些特點非常適合用於深度學習模型的訓練和測試。

2.2 資料集概述

UEC FOOD 100 Ver 1.0 資料集包含了豐富多樣的食品影像，主要特點如下：

1. 類別數量：資料集涵蓋了 100 種不同的食品類別，如米飯、麵食、麵包、披薩等，這些類別代表了各種常見的食品和菜餚。

圖 1 Data 範例圖

2. 影像數量：總共提供了 12740 張的食品影像，其中 11566 張影像中只有單一的食品項目，而 1174 張影像中包含了多個食品項目。
3. 影像分佈：資料集中，味噌湯類別的影像數量最多，共有 729 張影像；而蛋包飯類別的影像數量最少，只有 101 張影像。

2.3 資料前處理

在進行模型訓練之前，我們對資料集進行了全面的前處理，確保資料的一致性，並為後續的模型訓練和測試提供了堅實的基礎，具體步驟如下：

1. 圖像增強：為了提高模型的泛化能力，我們應用了多種圖像增強技術，如隨機旋轉、翻轉等。這些增強技術模擬了不同的拍攝條件，增加了數據的多樣。

```python
def augment_images(image_path, output_path, num_augmentations):
    os.makedirs(output_path, exist_ok=True)
    image = Image.open(image_path)
    base_name = os.path.splitext(os.path.basename(image_path))[0]

    transform = transforms.Compose([
        transforms.RandomRotation(degrees=(90, 270)),
    ])

    for i in range(num_augmentations):
        augmented_image = transform(image)
        augmented_image.save(os.path.join(output_path, f"{base_name}_aug_{i}.jpg"))
```

圖 2 圖像增強

2. 標準化對標籤(Label)中有關邊界框(Bounding Box)的資訊進行標準化處理。

```python
# 自動檢測圖像尺寸
with Image.open(img_path) as img:
    img_width, img_height = img.size

# 計算 YOLO 格式的標籤
x_center = (x1 + x2) / 2 / img_width
y_center = (y1 + y2) / 2 / img_height
width = (x2 - x1) / img_width
height = (y2 - y1) / img_height

label = f"{class_id - 1} {x_center} {y_center} {width} {height}\n"     # YOLO 的格式期望類別 ID 從 0 開始，所以 class_id -1
```

圖 3 標準化邊界框數值

3. 將原始數據集的目錄結構轉換為 YOLOv8 所期望的特定目錄結構，以利 yaml 檔進行數據的讀取。
4. 數據分割：我們將資料集分割為訓練集（80%）和測試集（20%），確保模型能夠在不同的資料集上進行有效的學習和評估，從而提高模型的泛化能力和準確性。

```python
# 設置好欲進行儲存的路徑
def create_directories(output_path):
    train_image_dir = os.path.join(output_path, 'train', 'images')
    train_label_dir = os.path.join(output_path, 'train', 'labels')
    val_image_dir = os.path.join(output_path, 'val', 'images')
    val_label_dir = os.path.join(output_path, 'val', 'labels')


    os.makedirs(train_image_dir, exist_ok=True)
    os.makedirs(train_label_dir, exist_ok=True)
    os.makedirs(val_image_dir, exist_ok=True)
    os.makedirs(val_label_dir, exist_ok=True)
```

圖 4 目錄結構轉換

三、模型介紹

本專題主要希望應用食物檢測的技術，為目標店家帶來競爭上的優勢，而在食物偵測的任務上，我們主要選擇 YOLOv8 作為深度學習的模型。因此，本節主要首先對 YOLOv8 進行基本介紹，後續則講解本組的資料預處理、超參數設計以及模型表現結果部分。最後，則講解本組的資料預處理、超參數設計以及模型表現結果部分。最後，由於 YOLOv5 與 YOLOv8 同為 Ultralytics 公司開發之產品，因此也對 YOLOv5 進行簡單的參數設計與實驗，觀察其表現與 YOLOv8 之間的差異。

3.1 YOLO 介紹

YOLO(You Only Look Once)是即時物件偵測中重要的技術。首次於 2015 年 6 月由 Joseph Redmon 和 Ali Farhadi 共同提出和開發的。隨後在過去幾年裡生成了許多版本，每個版本都做出了些許地改動，這些改動包括架構設計、損失函數的修改和邊界框調整等。

YOLOv5 是由 Ultralytics 公司在 2020 年年底開發的，其旨在提供更高效和更精確的目標檢測解決方案。目前，這項演算法在各種領域和應用中被廣泛使用，包括電腦視覺與圖像識別、自動駕駛與無人系統、醫療影像分析等。該演算法通過改進 YOLOv4 並採用新的模型結構和訓練技術，提高了速度和精度。與過往的版本不同，YOLOv5 一次性地推出了多個版本(如 s、m、l、x)，並使用 Python 在 PyTorch 框架下進行編寫。

YOLOv8 是 Ultralytics 公司在 2023 年 1 月開發的，與之前的 YOLO 版本相比，YOLOv8 引入了一些新的設計思想和技術，以提高模型的精度和速度。該演算法不僅可以在通用的目標檢測任務中表現良好，還可以應用於各種領域，如自動駕駛、工業檢測、物體識別等。該演算法在模型結構、數據增強等方面進行了優化，使得最終輸出獲得出色的結果。

3.2 YOLO v8 的優勢

YOLOv8 在多個方面進行了改進，旨在提高模型的性能和適用性：

1. 改進的網絡結構：YOLOv8 引入了一種新的網絡架構，這種架構比前幾代更加靈活和高效。它包括了一些新的模塊，如 FPN（Feature Pyramid Network）和 PAN（Path Aggregation Network），這些模塊能夠更好地處理不同尺度的特徵，提升了小目標的檢測能力。
2. 更高的計算效率：YOLOv8 使用了更加高效的卷積運算和優化演算法，使得模型能夠在保持高檢測精度的同時，顯著降低了計算成本和運行時間。這使得 YOLOv8 非常適合用於資源受限的設備，如嵌入式系統和移動設備。
3. 增強的多尺度檢測能力：YOLOv8 通過引入多尺度特徵融合技術，能夠同時

處理來自不同尺度的圖像特徵，這提高了模型在各種大小目標上的檢測能力，特別是在食品影像辨識這類場景中，能夠有效識別大小不一的食品物體。

4. 改進的損失函數：YOLOv8 使用了一種新的損失函數，該損失函數能夠更好地平衡目標定位和分類的準確性，從而提高了模型的整體檢測性能。這種損失函數還引入了 IoU 損失和邊界框回歸損失，進一步提升了檢測邊界的精確度。
5. 實時性能：YOLOv8 延續了 YOLO 模型的一貫特點，即能夠在保持高精度的同時實現實時性能。這對於需要快速響應的應用場景，如食品影像分類和實時物體檢測，是一個非常大的優勢。

3.3 架構比較

本節深入研究 YOLOv5 以及引進新一代模型 YOLOv8 的架構設計與相關內容。作為一種專注於目標檢測的模型，YOLOv5 的架構特點及其獨特優勢將被詳細探討，並呈現其對目標檢測任務所帶來的顯著進展。接著，借鑑於 YOLOv5 的優點與前瞻性研究，新興的 YOLOv8 被提出，其目的在進一步提升目標檢測的性能與效率。該篇章旨在全面探 討這兩種模型的架構特性，深入分析其優勢與差異性。

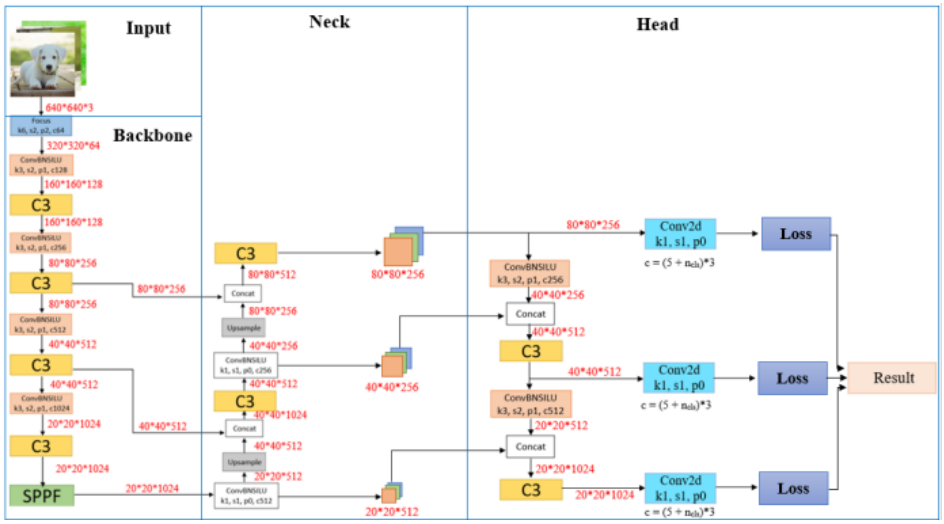


圖 5　YOLOv5 架構圖

(一) YOLOv5 架構

YOLOv5 目標檢測模型的架構是由 Input 層、Backbone 層、Neck 層以及 Head 層結構組成，如圖 5 所示。

1. Input 層匯入圖片後將會經歷 Mosaic 數據增強、動態邊界框計算、自動圖片縮 放等初步的圖片處理。

(1) Mosaic 數據增強： 每次讀取四張圖片後，接著進行隨機翻轉、縮放、剪裁、排佈和色域變化的方式進行拼接成 Mosaic 數據，為模型提供了更多樣化且豐富的資料。透過 整合這些圖片，模型能學習更多變和複雜的場景，進而提升對不同情況下的辨 識能力。

(2) 動態邊界框計算： 不同的數據集都對應著各自初始設定的長寬邊界框。這些邊界框被視為訓練過程中的參考基準，模型將基於初始邊界框輸出預測框，然後與真實邊界框 進行比較，計算兩者之間的差異。接著，通過反向更新模型參數，不斷疊代優化，這有助於提高模型檢測的效率與準確度。

(3) 自動圖片縮放： 在以往的 YOLO 版本中，演算法習慣性地將圖片縮放至相同尺寸後再進行檢測。然而，這種方法會導致輸入圖像兩側出現不同尺寸的黑邊，進而影響 模型推理速度，使得資料冗餘。相較於這種方式，YOLOv5 引入了一項新功能，能夠自動調整原始影像，使其周圍增加最少量的黑邊，以符合模型的需求。這個改進有效地減少了運算量，提高了目標檢測的速度，同時增加了模型的通用 性和適應性。

2. Backbone 層是將處理好的圖片進行特徵點提取。此架構採用了 Focus 結構、ConvBNSILU、C3、SPPF。

(1) Focus 結構： 透過圖像的切片處理和卷積運算，可獲得採樣的特徵圖。如 408*408*3 的圖片輸入至 Focus 結構後，該圖片經過切片處理轉變為 204*204*12 的特徵圖。接著，進行 64 個卷積核的卷積操作，最終得到尺寸為 204*204*64 的特徵圖。通過這樣的處理，可以更好地捕捉到圖像中的各種特徵，並且能夠更有效地進 行後續的分類、檢測和識別任務。

(2) ConvBNSILU： 對輸入圖像進行卷積操作、批量歸一化和引入非線性激活函數 SiLU。這對於提升模型的表示能力和學習圖像特徵的效率具有重要意義。

(3) C3： 此為 CSPDarknet53 中的特定模塊。透過多層卷積操作來處理輸入的特徵圖，有助於更準確地檢測和識別圖像中的目標物體。

(4) SPPF: SPPF 是一種擴展自 SPP 的技術，旨在更有效地利用不同尺寸的感受野 (Receptive Field)來捕捉圖像中物體的多尺度特徵。透過 SPPF，我們能夠更全面地在不同尺度下理解和學習圖像的細節特徵，有助於提

升物體檢測的精準度與清晰度。

3. Neck 層用了 Concat、Upsample、ConvBNSILU、C3，主要是將特徵圖進行特徵融合、尺寸變換、特徵增強等操作。

   (1) Concat： Concat（Concatenate 的簡稱）是一種特徵融合的方法。透過將兩個或多個數據結構按照某個特定維度或軸連接在一起，形成一個更大的數據結構，以更全面地捕捉和整合圖像中的信息。這有助於提升目標檢測的精確度和豐富度。

   (2) Upsample： 上採樣是一項用於將低解析度特徵圖提升至高解析度的技術，藉此促進跨 層級特徵融合。這項技術有助於增進神經網絡對目標的檢測和定位能力，使網絡能夠更有效地理解和捕捉不同尺度的特徵，進而提升目標檢測的精確性和性能。

4. Head 負責生成目標檢測的預測結果

   (1) 目標類別預測： 預測層負責生成一個類別概率分布，用於表示每個檢測框所屬的目標類別。通常情況下，這個概率分布使用 Softmax 函數進行歸一化，確保每個檢測框僅 屬於一個類別。

   (2) 邊界框座標預測： 給出物體邊界框的位置，包括框的中心座標、寬度和高度，用以準確定位物體在圖像中的位置。

   (3) 置信度分數預測： 用於評估每個方框是否涵蓋目標物體的指標。通常呈現在 0 到 1 之間的數值，數值越高代表模型對於方框內包含目標的確信程度越高。

(二) YOLOv8 架構

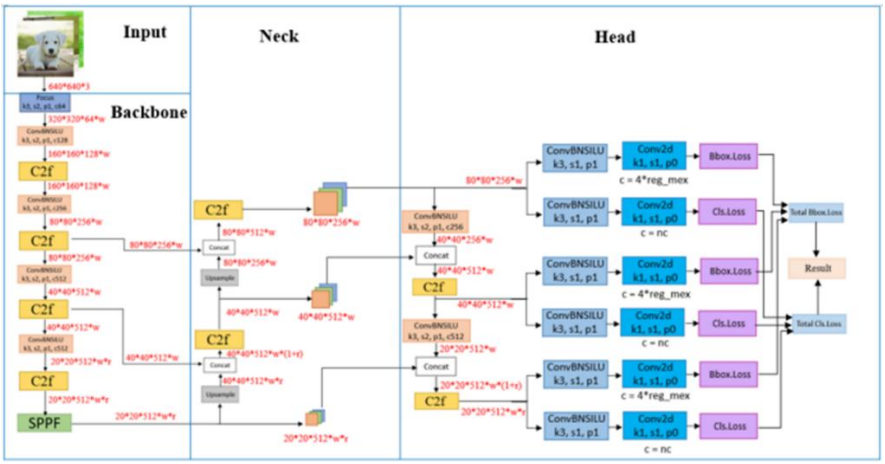


圖 6　YOLOv8 架構

YOLOv8，如圖 6 所示，與其前身 YOLOv5 在整體架構上保持了一致性，均包含 Input、Backbone、Neck、Head。只在細微上進行了改動，這些改動旨在進一步優化目標檢測性能。

1. Backbone

(1) 將卷積核從 6*6 調整為 3*3，以進行初步特徵處理。輸入圖像經過 3*3 的卷積 操作後，解析度從 640*640 被降低到 320*320。此操作旨在進行初步的特徵提取，同時對圖像進行解析度縮減。

(2) 將所有的 C3 模塊替換成梯度信息更豐富的 C2f 模塊，如圖 7 所示，進一步增加跳層連結。C2f 模塊的設計受到 C3 模塊和 ELAN 模塊的啟發，以確保 YOLOv8 在保持輕量化的同時，獲取更加豐富的梯度流信息。ELAN 模塊是 YOLOv7 所引入的一種模塊，其能夠補獲更為豐富的梯度信息，進而實現更高 的精確度和更合理的推理時間延遲。

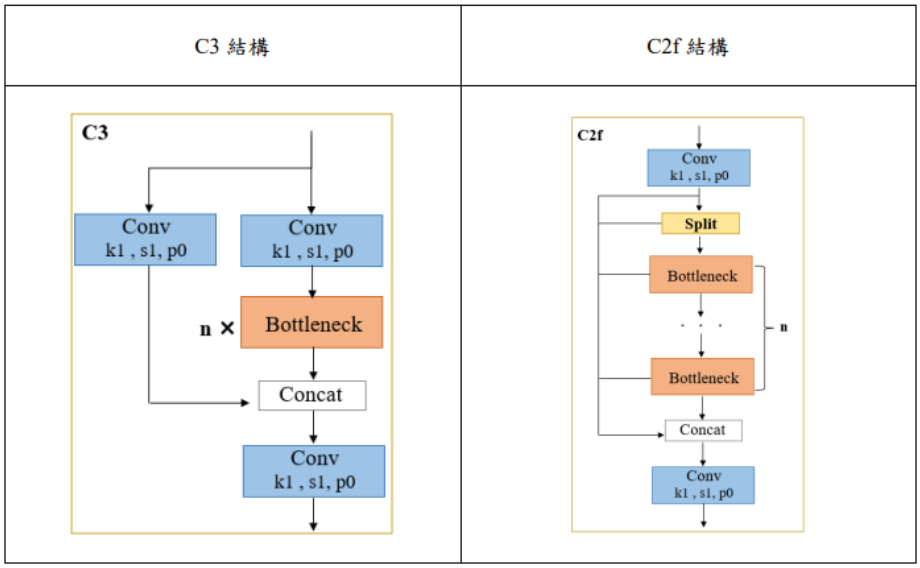


圖 7　C3 結構與 C2f 結構差異圖

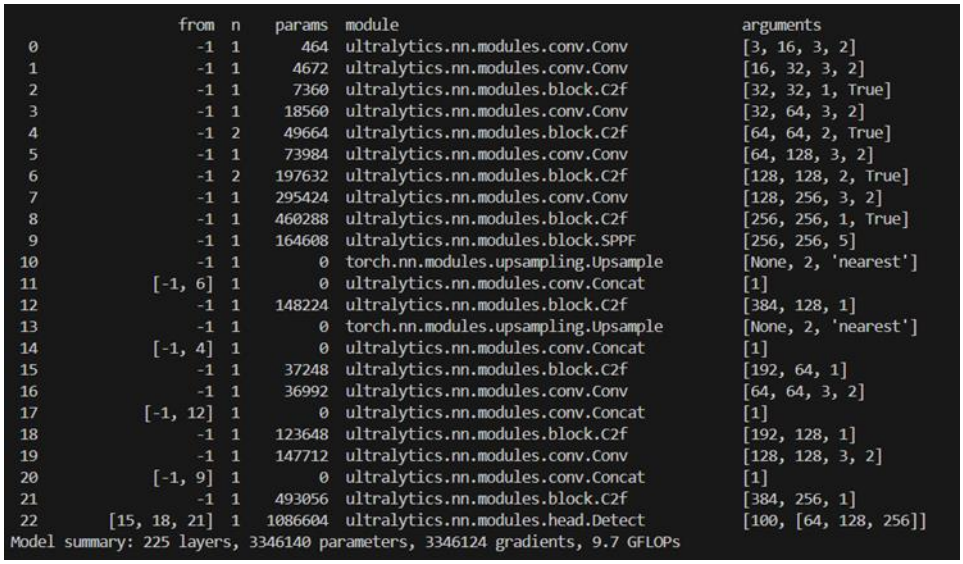
2. Neck

(1) 將所有的 C3 模塊替換成梯度信息更豐富的 C2f 模塊，進一步增加跳層連結。

(2) 取消兩個 1*1 卷積層(ConvBNSILU)的操作，改為直接將 Backbone 不同階段的輸出特徵通過 Concatenation(拼接)操作後，直接送入 Upsample 層。這種修改旨在簡化模型架構，省略特定層次的卷積處理，使得不同階段的特徵能夠直接被 整合和上採樣，以提高計算效率和減少參數數量。

3. Head 從原先的耦合變成解耦，將目標分類和偵測框架目標回歸兩個行動解耦，並分別進行預測，這種解耦合的操作有助於加速模型的收斂速度並提高目標檢測的精度。

3.4 YOLOv8 模型實際架構



	from	n	params	module	arguments
0	-1	1	464	ultralytics.nn.modules.conv.Conv	[3, 16, 3, 2]
1	-1	1	4672	ultralytics.nn.modules.conv.Conv	[16, 32, 3, 2]
2	-1	1	7360	ultralytics.nn.modules.block.C2f	[32, 32, 1, True]
3	-1	1	18560	ultralytics.nn.modules.conv.Conv	[32, 64, 3, 2]
4	-1	2	49664	ultralytics.nn.modules.block.C2f	[64, 64, 2, True]
5	-1	1	73984	ultralytics.nn.modules.conv.Conv	[64, 128, 3, 2]
6	-1	2	197632	ultralytics.nn.modules.block.C2f	[128, 128, 2, True]
7	-1	1	295424	ultralytics.nn.modules.conv.Conv	[128, 256, 3, 2]
8	-1	1	460288	ultralytics.nn.modules.block.C2f	[256, 256, 1, True]
9	-1	1	164608	ultralytics.nn.modules.block.SPPF	[256, 256, 5]
10	-1	1	0	torch.nn.modules.upsampling.Upsample	[None, 2, 'nearest']
11	[-1, 6]	1	0	ultralytics.nn.modules.conv.Concat	[1]
12	-1	1	148224	ultralytics.nn.modules.block.C2f	[384, 128, 1]
13	-1	1	0	torch.nn.modules.upsampling.Upsample	[None, 2, 'nearest']
14	[-1, 4]	1	0	ultralytics.nn.modules.conv.Concat	[1]
15	-1	1	37248	ultralytics.nn.modules.block.C2f	[192, 64, 1]
16	-1	1	36992	ultralytics.nn.modules.conv.Conv	[64, 64, 3, 2]
17	[-1, 12]	1	0	ultralytics.nn.modules.conv.Concat	[1]
18	-1	1	123648	ultralytics.nn.modules.block.C2f	[192, 128, 1]
19	-1	1	147712	ultralytics.nn.modules.conv.Conv	[128, 128, 3, 2]
20	[-1, 9]	1	0	ultralytics.nn.modules.conv.Concat	[1]
21	-1	1	493056	ultralytics.nn.modules.block.C2f	[384, 256, 1]
22	[15, 18, 21]	1	1086604	ultralytics.nn.modules.head.Detect	[100, [64, 128, 256]]

Model summary: 225 layers, 3346140 parameters, 3346124 gradients, 9.7 GFLOPs

圖 8　YOLOv8 模型實際架構

3.5 YOLOv8 超參數優化

超參數設計與優化的部分，我們選擇了三種超參數進行實驗與評估，分別為迭代次數(Epoch)、批量大小(Batch Size)以及優化器(Optimizer)，並且使用網格搜索(Grid Search)的方法對每一組超參數組合，訓練模型並評估其性能。下表是我們所選擇的超參數以及超參數組合:

表 2 超參數組合表

Epoch	Batch Size	Optimizer
50	8	Auto
50	8	SGD
50	16	Auto
50	16	SGD
75	8	Auto
75	8	SGD
75	16	Auto
75	16	SGD

需特別解釋的部分是"Auto"優化器，該優化器是 YOLOv8 的預設優化器，具體來具體來說，當設定優化器為 auto 時，模型會自動選擇一個適合當前任務和硬體配置的優化器，而不是讓使用者明確指定某一種優化器（例如 SGD 或 Adam)。根據官網 Auto 會進行選擇的範圍包括 SGD, Adam, AdamW, NAdam, RAdam, RMSProp 等等。

而我們的實驗表現相關指標包括 Precision,Recall, mAP50, mAP50-95 以及另外使用 time 模組，檢測每個實驗組合的實驗時間。 mAP50, mAP50-95 兩種指標的定義分別為:

- mAP@0.5: 表示在 Intersection over Union (IoU) 閾值設為 0.5 時，計算出的平均準確率(Average Precision, AP)。這意味著在評估中，只要檢測框和真實框的 IoU 大於等於 0.5，就被認為是正確檢測。
- mAP@0.5:0.95 表示在多個 IoU 閾值下的平均準確率，這些閾值從 0.5 到 0.95，步長為 0.05。這意味著計算 IoU 閾值分別為 0.5, 0.55, 0.6, ... , 0.95 的 AP，然後對這些 AP 取平均。

  由以上定義可知mAP@0.5的標準相對寬鬆,因此一般來說其值會相對較大。

3.6 YOLOv8 實驗結果

Epoch	Batch Size	Optimizer	Precision	Recall	mAP50	mAP50-95	Time(hrs)
50	8	Auto	0.6377	0.6605	0.595	0.3101	1.72
50	8	SGD	0.6677	0.6853	0.5995	0.3146	1.667
50	16	Auto	0.6992	0.659	0.6054	0.3178	1.62
50	16	SGD	0.6458	0.6291	0.5879	0.301	1.635
75	8	Auto	0.667	0.6768	0.603	0.314	2.52
75	8	SGD	0.6621	0.6743	0.6023	0.3139	2.786
75	16	Auto	0.6833	0.6732	0.6018	0.317	2.681
75	16	SGD	0.6948	0.6688	0.5967	0.3148	2.616

經過 grid search 的綜合性評估後，我們選擇 epoch=50, batch size=16, optimizer=auto 的超參數組合，作為我們後續進行食物辨識任務的主要模型。原因是在八個超參數組合中，該組合分別在 Precision , mAP@0.5, mAP@0.5:0.95 三項指標中得到最佳表現。另外，其訓練時間 1.62 小時也是八個中最短的訓練時間，綜合上述因素我們認為最適合進行後續任務。而以下幾張圖表則為最佳超參數組合的最終表現。

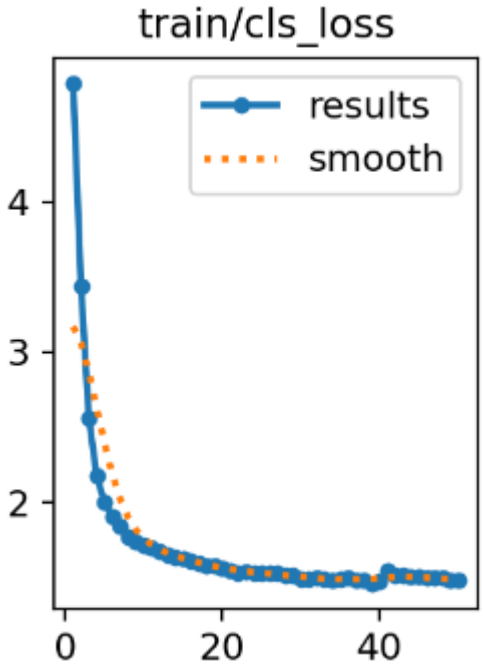


圖 9 訓練階段

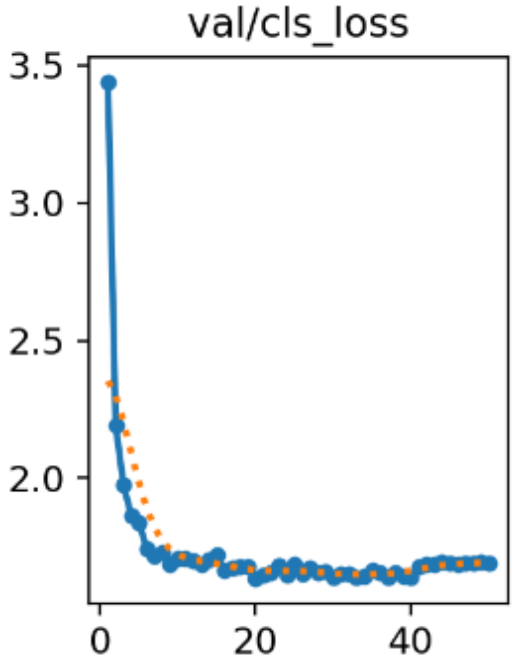


圖 10 驗證階段

首先，以上兩張圖分別為訓練以及驗證階段，損失函數的下降過程，這裡的損失函數則是分類損失(Classification Loss, CLS)。在辨識任務中，其定義為預測每個邊界框所包含的對象類別與真實類別之間的差異，是用來衡量模型在對每個邊界框進行分類時的準確性。而從上圖中可以看出:

- 隨著訓練進行，損失逐漸減少，這表明模型在學習並且性能在改善。
- 損失在前幾個 epoch 減少得很快，之後減少的幅度變小，屬於正常的收斂現象

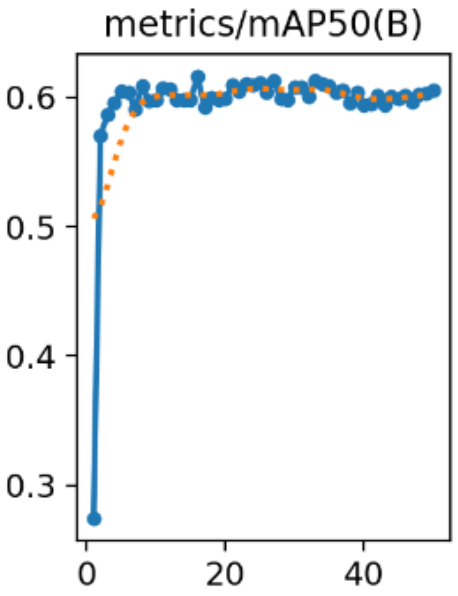


圖 11 mAP@0.5

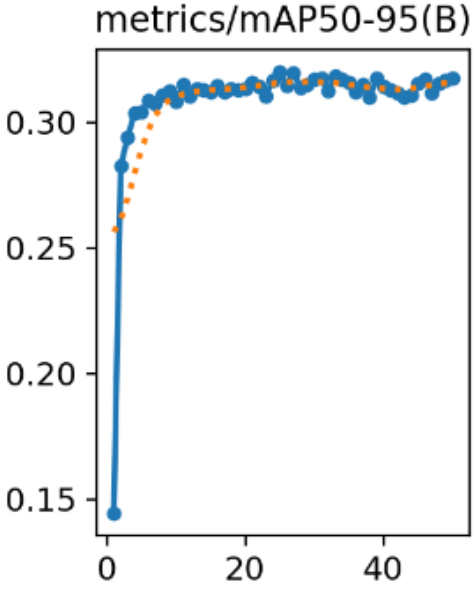


圖 12 mAP@0.5:0.95

接著，上方兩張圖分別為前面所介紹的 mAP@0.5 以及 mAP@0.5:0.95 指標，從兩張圖中可以看出:

- mAP@0.5 和 mAP@0.5-0.95 均顯示隨著訓練過程逐漸提高並趨於穩定，尤其在訓練初期階段就快速提升，表明模型的整體檢測性能的大幅度改善。
- mAP50 比 mAP50-95 高，因為更高的 IoU 閾值要求更精確的檢測框。

3.7 YOLOv5

Ultralytics 公司於 2023 年 1 月開發 YOLOv8，而在此之前，YOLOv5 為 Ultralytics 的最新產品於 2020 年 1 月開發，因此本專題雖然以 YOLOv8 作為食品辨識任務的主要模型，但除此之外也與 YOLOv5 進行比較。

3.7.1 YOLOv5 超參數優化與實驗結果

Epoch	Batch Size	Precision	Recall	mAP50	mAP50-95	Time(hrs)
50	8	0.6472	0.6534	0.6	0.3011	1.924
50	16	0.679	0.652	0.612	0.3148	1.714
75	8	0.6541	0.6775	0.6213	0.3058	2.912
75	16	0.6539	0.6362	0.598	0.2998	2.762

四、網頁服務設計

4.1 ER model

期中報告的網頁服務，提供使用者在操作訂單及會員服務，所應用到的 ER Model 如圖 X。

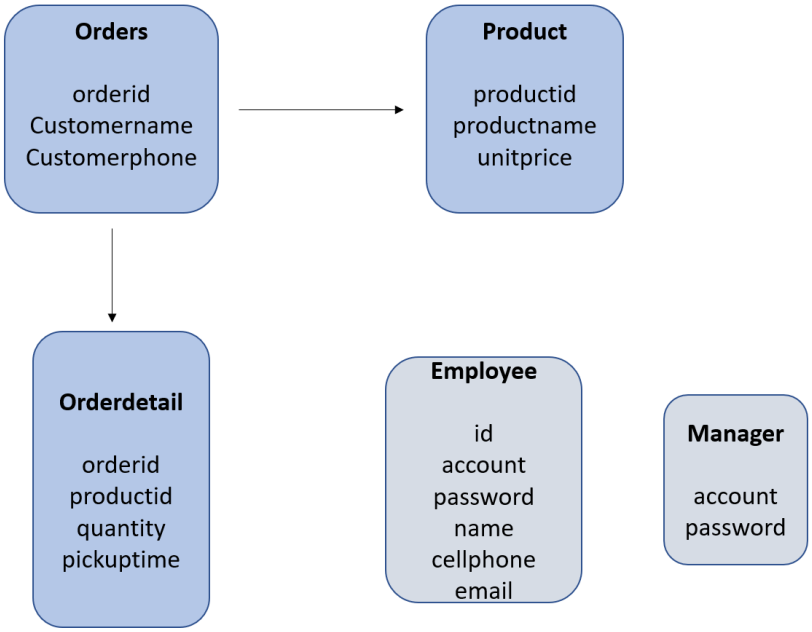


圖 13 會員系統 ER Model

4.2 網站架構圖

下圖為期中報告已設計供使用者使用之網頁設計圖。

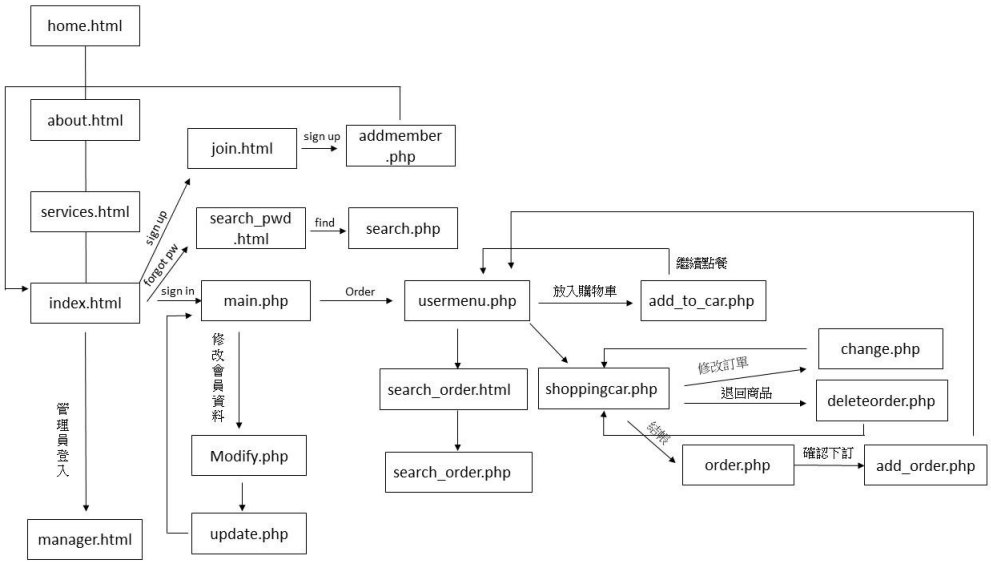


圖 14 前端網頁架構圖

圖 X 為後端管理員使用介面架構圖，供管理員修改訂單及會員資料。

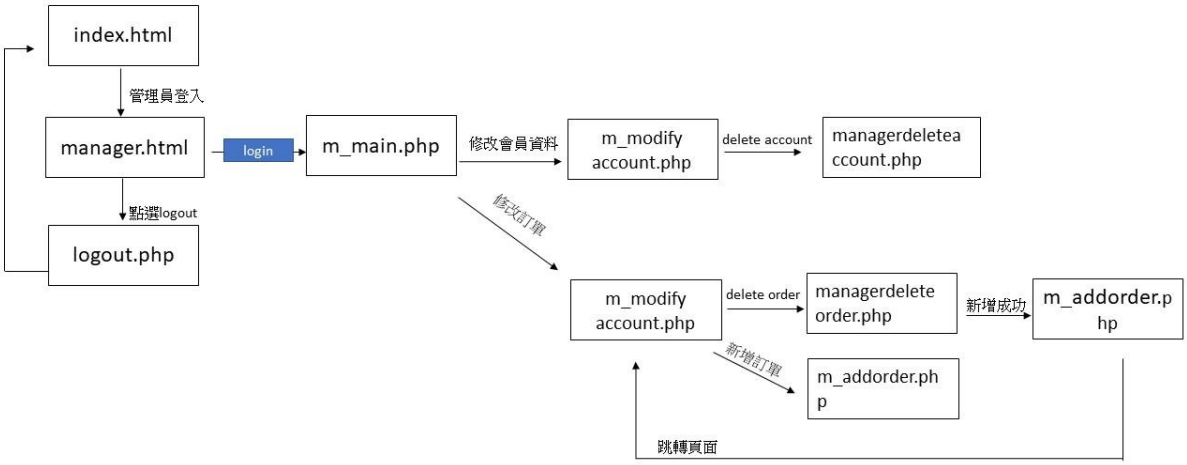


圖 15 後端管理員使用介面架構圖

圖 X 為本次報告新增後端連結 Detection Model 之架構圖，以及前端設計架構。

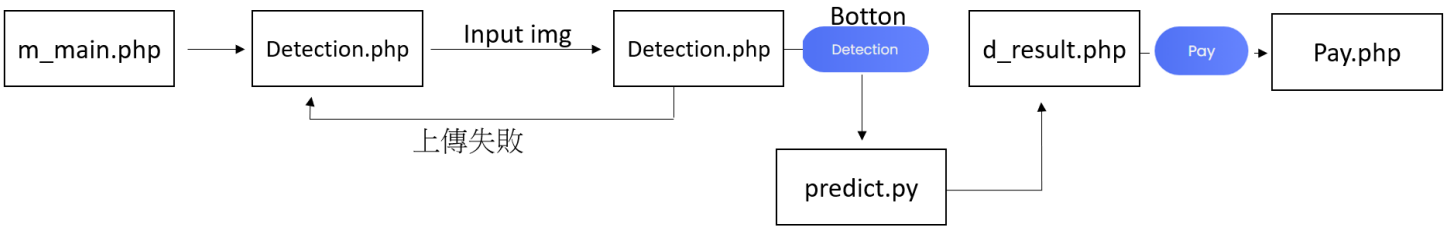


圖 16 Detection 架構圖

4.3 網頁功能介紹

針對偵測圖片的功能，主要提供給店家做使用，因此網頁設計皆以管理者角度設計，故偵測須從管理員登入開始。

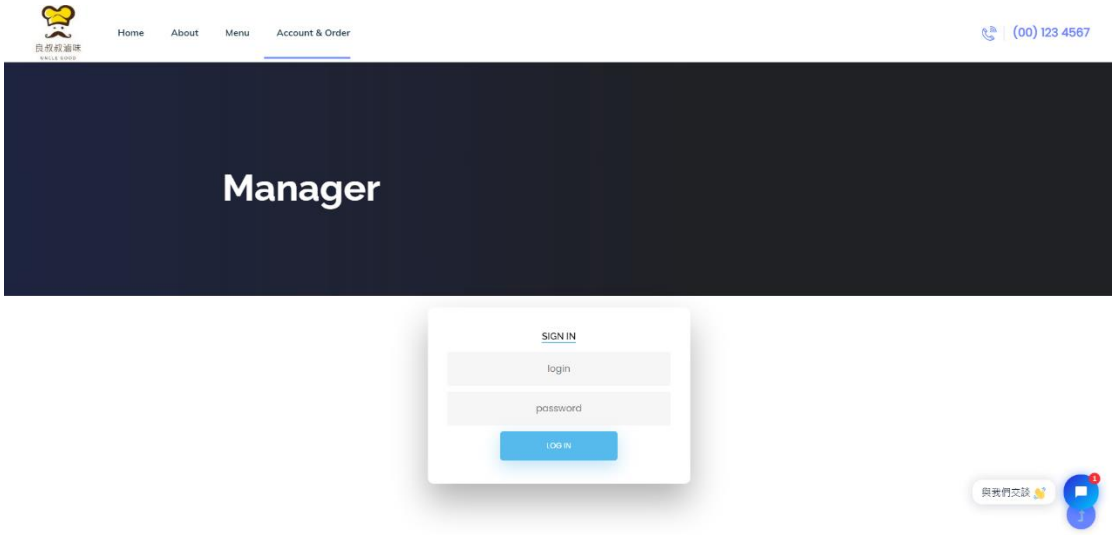


圖 17 管理員登入

登入管理員帳密後，即進入管理員操作頁面

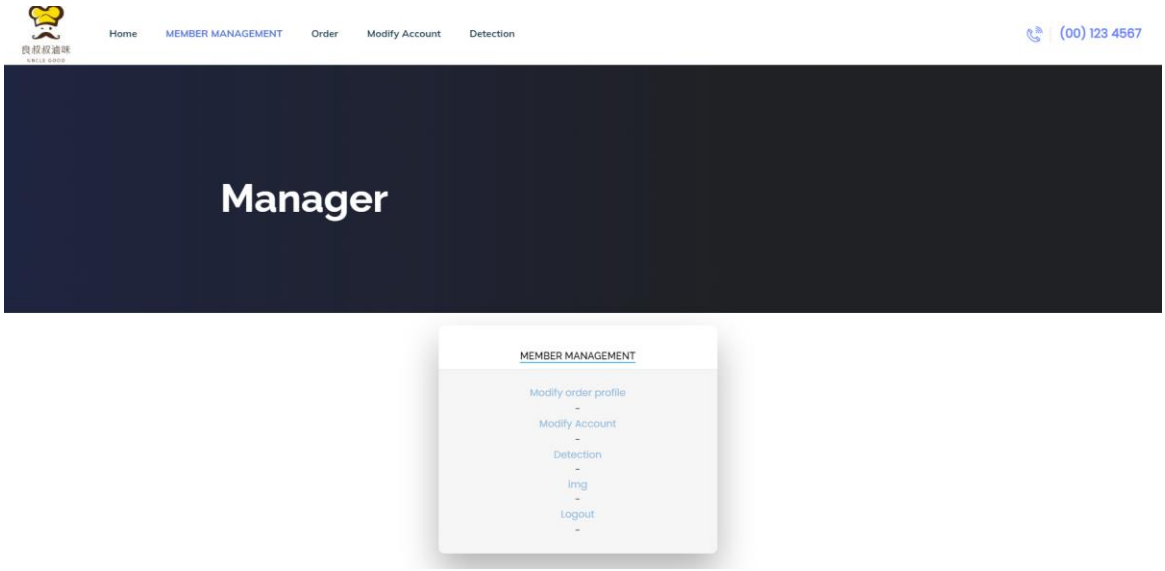


圖 18 管理員頁面

先前介紹過的管理會員帳密及訂單管理的網頁功能，提供管理者新增、刪除、修改客戶的帳號資訊、訂單內容，本次新增偵測圖片和管理圖庫的服務。

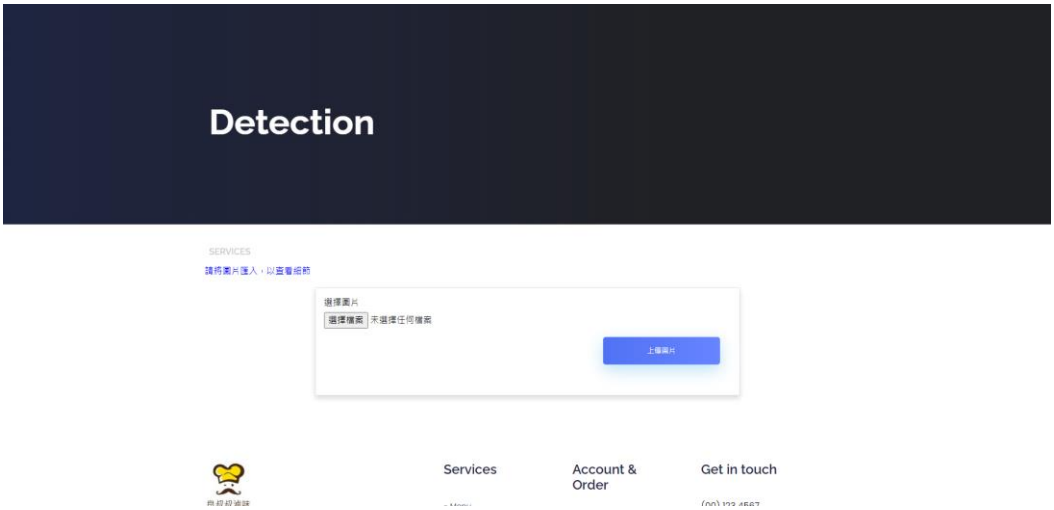


圖 19 Detection , upload image

將要進行偵測的圖片上傳。

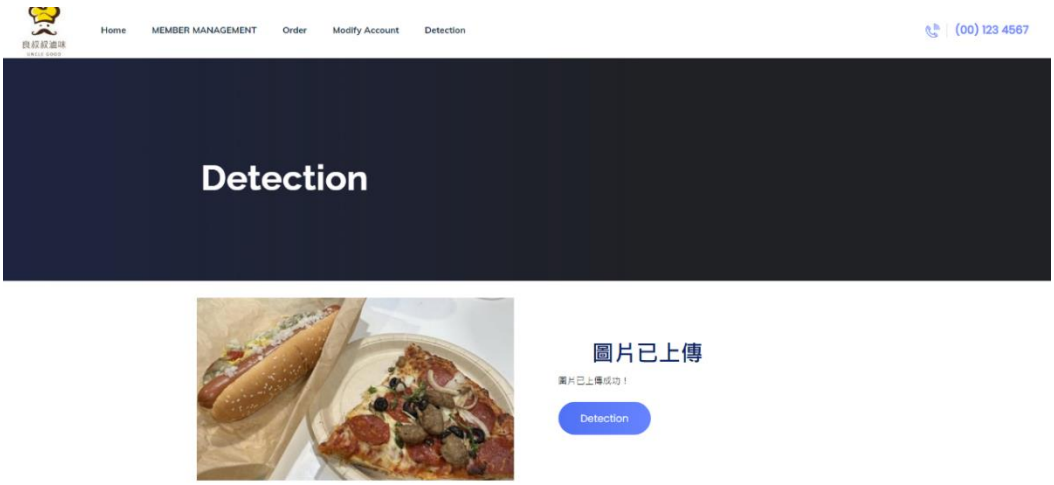


圖 20 上傳成功

上傳後，後端會偵測照片是否重複儲存並儲存至本地，若成功則顯示上傳之圖片及通知成功，再進行下一步偵測並輸出結果。

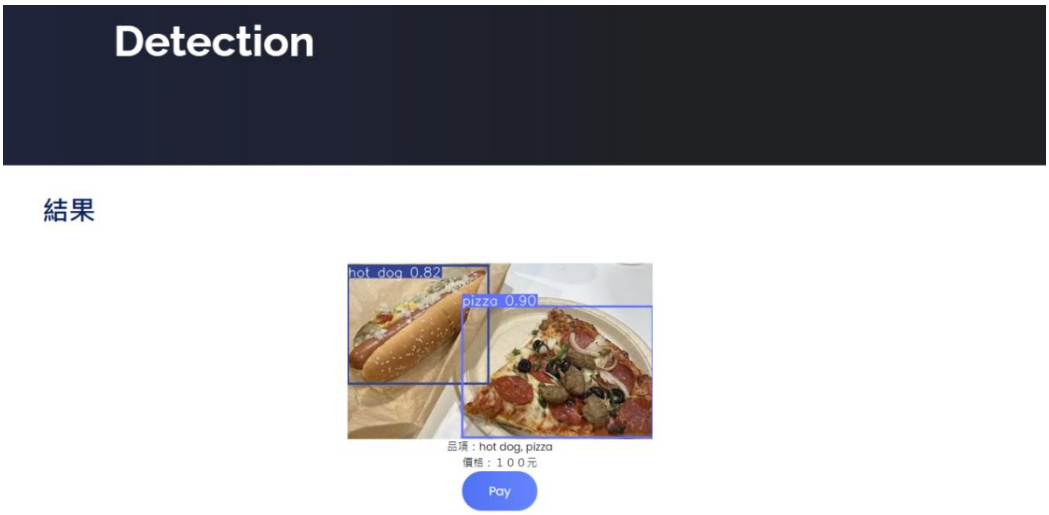


圖 21 分類結果

後端 yolo 會針對上傳的圖片進行預測，並輸出分類的類別，再透過給予的變數價格，計算後得到總額，消費者則可使用 Pay.php 提供的 QR code 進行付款。

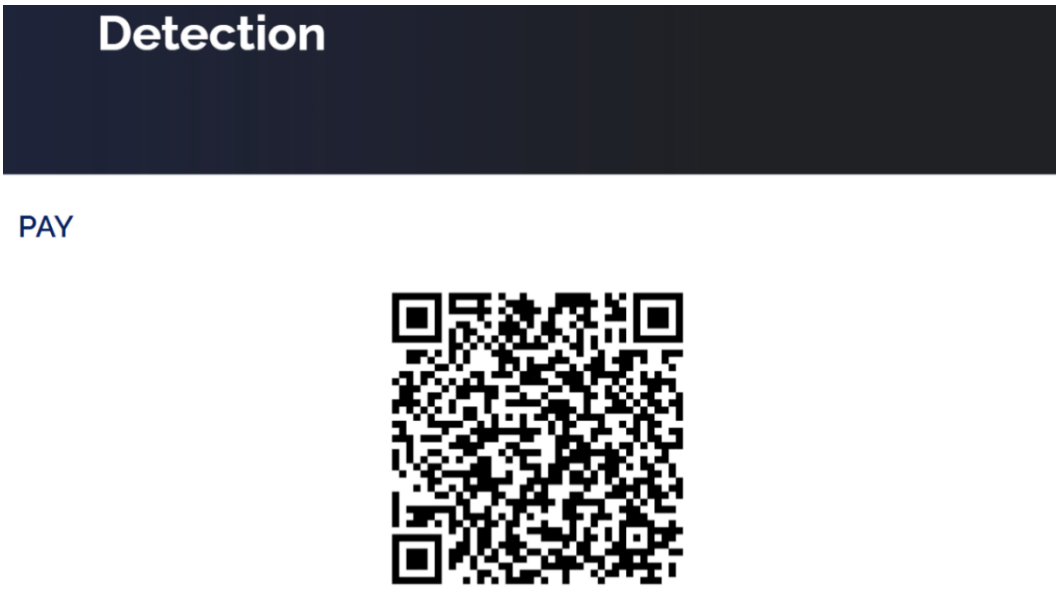


圖 22 付款頁面

為模擬實際應用上，容易為了餐點價格引起糾紛，故本功能提供偵測前的照片先上傳至本地或雲端，管理員可透過 img.php 的頁面進行核對消費者所享

用的餐點內容，以免後續不必要的紛爭。

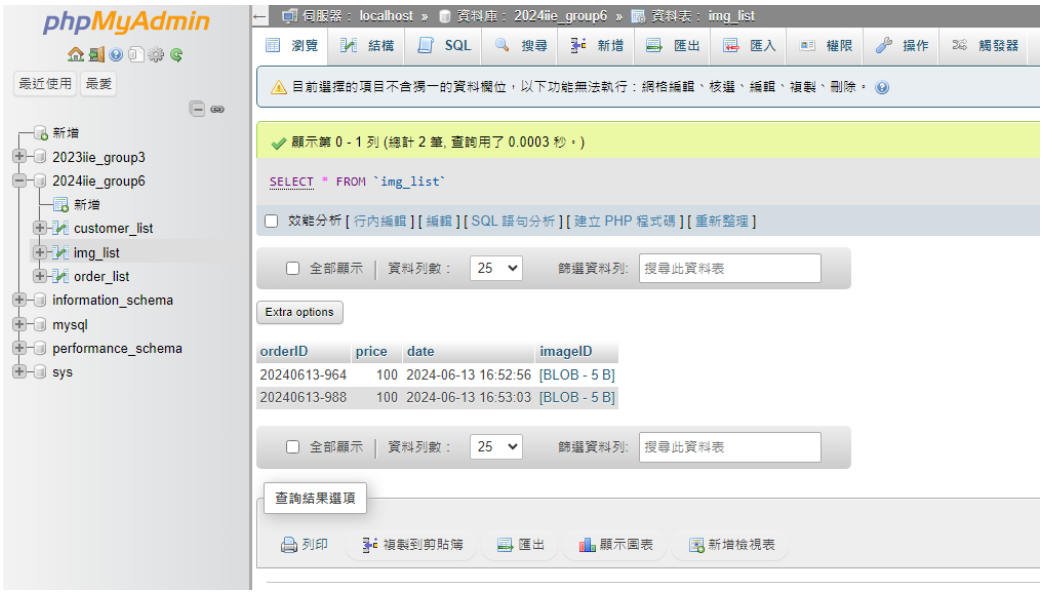


圖 23 圖片上傳資料庫

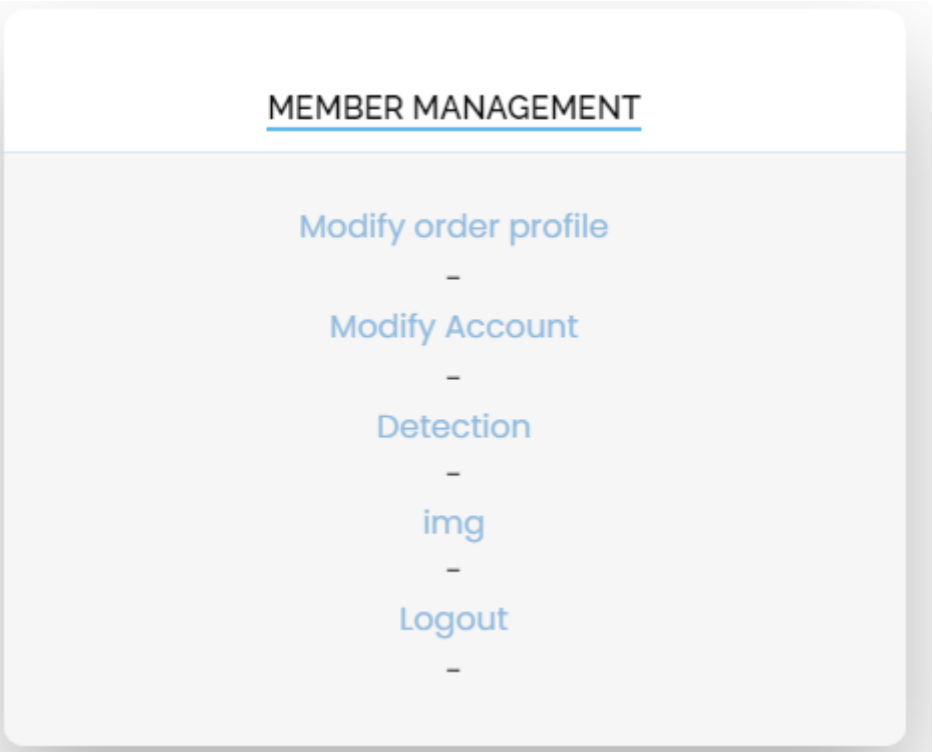


圖 24 管理員介面- img

管理員進入管理圖庫後，網頁上會顯示用餐時間以及偵測的圖片，管理員可透過此頁面進行刪除或新增的功能，提供此功能主要為針對顧客餐點疑慮進

行核對使用。點選 img 的超連結則會顯示餐點內容。

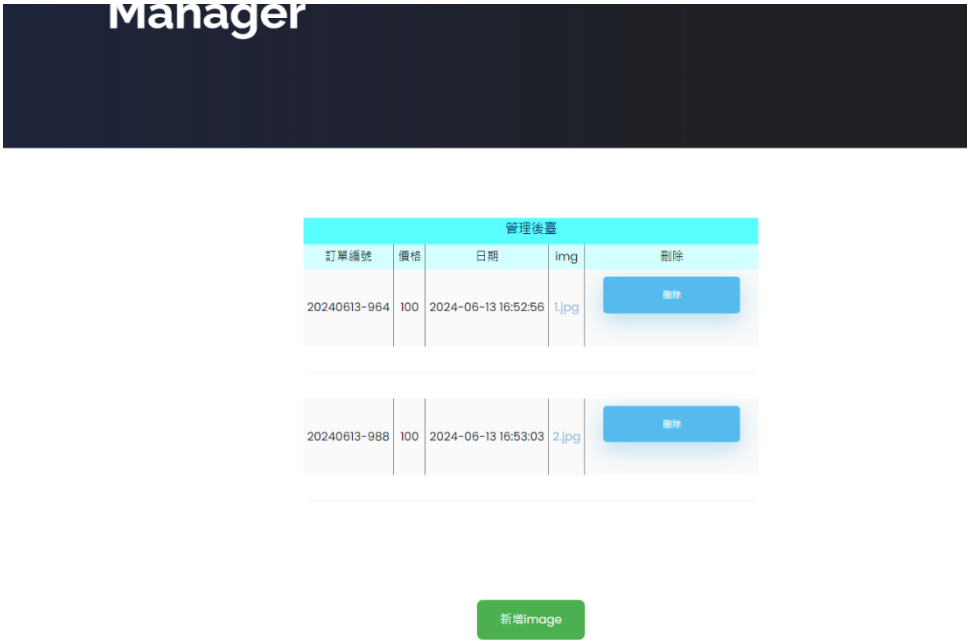


圖 25 圖庫管理

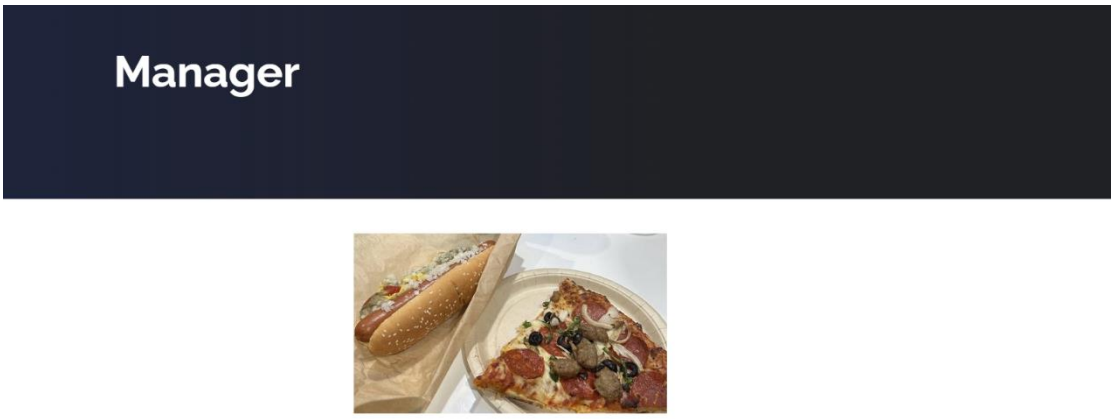


圖 26 核對餐點

4.4 AMPPS

當我們需要在 PHP 程式碼中執行圖片辨識時。由於缺乏服務器端的 Anaconda 環境的詳細位置，我們決定將這項任務轉移至本地端。這樣做可以確保我們能夠正確配置和使用 Python 環境，並且能夠導入所需的 Python 包來完成圖片辨識工作。因此，在 PHP 中，我們會通過適當的方式調用本地的 Python 環

境，以確保辨識過程順利執行並達到我們的需求。

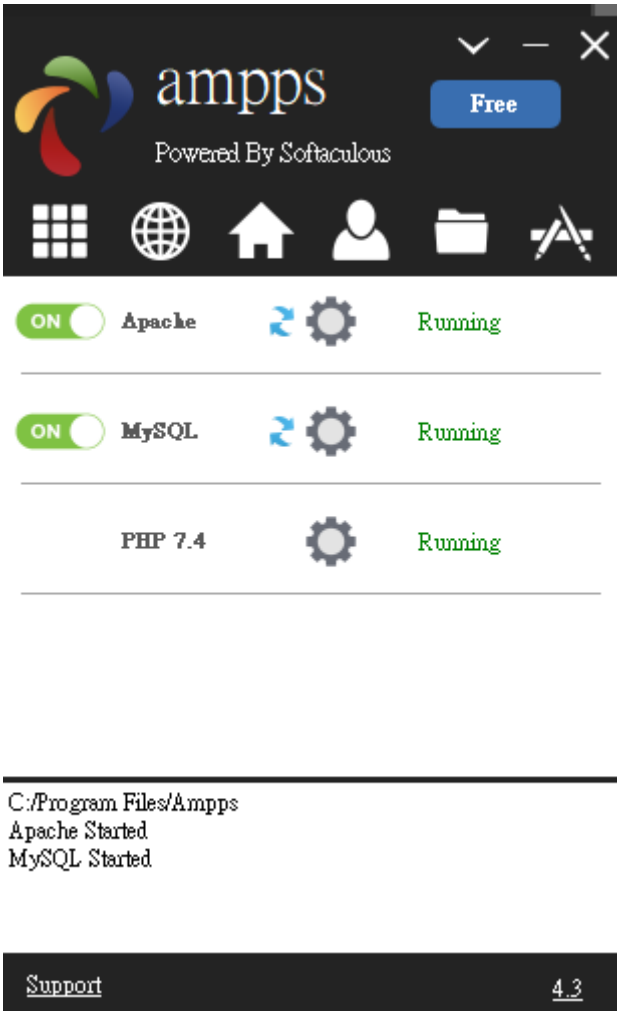


圖 27 AMPPS

五、結論與未來展望

5.1 結論

本專案通過結合深度學習模型和網頁應用，實現了食品影像辨識的自動化，並在此基礎上開發了一個完整的餐飲管理系統。系統具備以下主要功能：

1. 食品影像自動分類：通過使用 YOLO v8 進行食品影像的分類，我們能夠準確識別和分類多種食品。
2. 食品價格計算：系統根據識別結果，自動計算食品的價格。這不僅提高了效率，還減少了人工計算的誤差。
3. 生成付款 QR 碼：系統根據計算出的價格，自動生成對應的支付 QR 碼，方便顧客使用手機進行快速支付，提升了顧客的購物體驗。

本系統的應用使得餐飲業者能夠更有效地管理食品和訂單，提高了工作效率，減少了錯誤，並改善了顧客服務品質。通過自動化的食品分類和價格計算，系統有效地降低了營運成本，並提高了營運效率。此外，生成付款 QR 碼的功能使得支付過程更加便捷，進一步提升了顧客的滿意度。

5.2 貢獻

1. 提升工作效率：本系統通過自動化食品識別和價格計算，顯著提升了餐飲業者的工作效率，減少了手動操作的時間和成本。
2. 降低錯誤率：通過使用深度學習模型進行食品分類，減少了人工操作中可能出現的分類錯誤，確保了數據的準確性和一致性。
3. 改進顧客服務：系統自動生成支付 QR 碼，簡化了支付過程，提升了顧客的購物體驗，提高了顧客滿意度和忠誠度。

5.3 局限性

1. 資料依賴：模型的準確性依賴於訓練資料的品質和多樣性，對於未見過的食品類別或新的影像資料，模型的識別準確性可能會有所下降。
2. 模型更新需求：隨著食品類別和樣本的變化，模型需要定期更新和重新訓練，以保持其準確性和適用性。
3. 計算資源要求：深度學習模型的訓練和運行需要較高的計算資源，對於一些

小型餐飲業者，可能需要額外的技術支持和硬體投入。

5.4 適用性

1. 餐飲業：系統主要適用於餐飲行業，尤其是需要進行大量食品分類和價格計算的餐廳、外賣店和食品配送中心等。
2. 零售業：系統可以擴展應用於超市和便利店等零售場景，實現商品的自動分類和價格計算，提升運營效率。

5.5 未來展望

1. 擴展資料集：

    - 增加資料集的規模和多樣性，涵蓋更多類型的食品和更廣泛的拍攝條件，以進一步提高模型的準確性和泛化能力。
    - 引入更多的食品類別，並擴展至不同地域和文化的食品，使系統能夠應用於更廣泛的市場。

2. 優化模型性能：

    - 採用更先進的深度學習模型，進一步提升分類的準確性和效率。
    - 模型進一步調整，根據特定的應用場景進行定制化優化，以達到最佳性能。

3. 新增更多功能：

    - 開發更豐富的支付功能，支持多種支付方式和貨幣，滿足不同市場的需求。
    - 增加營養資訊和健康建議功能，為顧客提供更加全面的食品資訊和個人化建議，提升用戶體驗。