影像處理

IMAGE PROCESSING

Final project

系所： 資訊所

姓名： 張少鈞

學號： P76114545

簡介

本次專案目標在於訓練模型，用於工廠內的瑕疵檢測，能夠即時並準確的找到瑕疵與其種類，並將其瑕疵部位進行影像分割顯示出來。本次專案包含兩個部分:瑕疵偵測與影像分割。現今較為流行的檢測方法可以分為 one stage 與 two stage 兩總，前者為辨識出物體的同時進行影像分割；後怎則是先辨識出物體位置後，對範圍內的影像進行分割。這兩者方式的選擇在於使用者較重視執行時間還是精準度，在精準度要求上 two stage 的表現會比較好；反之，one stage 在執行時間上較具有優勢。本次選擇以 one stage 的方式來進行實作。

模型選擇

本次採用 yolo v7 來進行實作，yolo v7 於 2022 年 7 月推出，不管是在精準度或者是速度上都優於其他現有的物件檢測模型(如圖 1)，並且 yolo v7 也有提供影像分割的功能，只要輸入影像就可以同時得到 bounding box 與 mask 的結果。

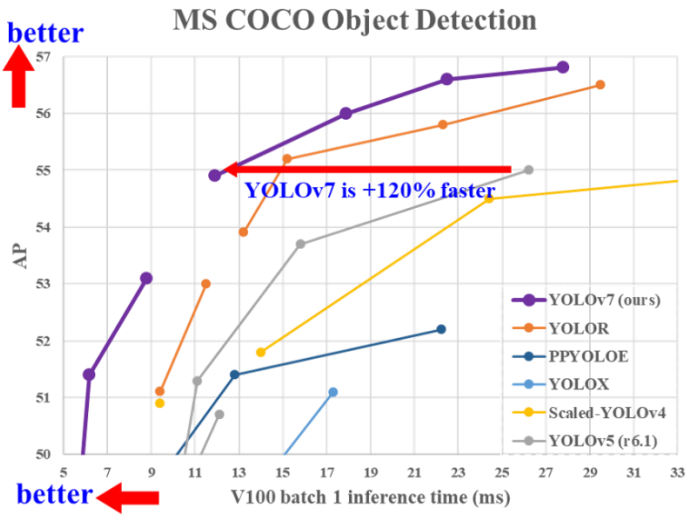

圖 1 執行速度與 AP 比較圖。

資料收集

本次採用課堂所提供的資料及進行訓練，檔案結構如圖 2 所示，分別有 Train 與 Val 兩個大資料夾，兩個資料夾內部均以瑕疵的類別將影像、標籤與分割結果分別儲存在 image、label 與 mask 資料夾下。

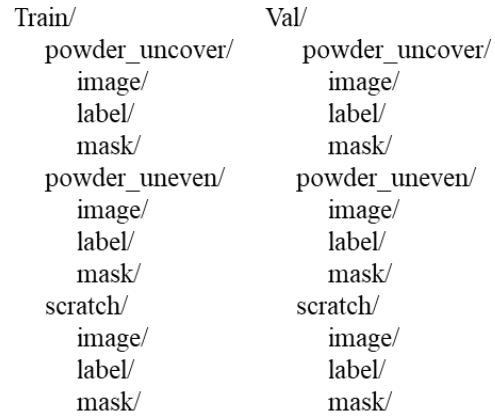

圖 2 原始檔案結構

原始的 label 檔為.json 的格式，本次專案會用到的為 shapes 裡的資訊，其代表本張圖片中所包含的瑕疵類別與影像分割後的形狀資訊，在開始訓練 yolo v7 之前要先將原始的 label 轉成 yolo v7 的格式。yolo v7 的輸入格式為.txt，.txt 中的每一行就代表該影像中的一個分割結果。每行的第一個數字代表瑕疵的分類，在本次的專案中定 0 代表 powder_uncover、1 代表 powder_uneven 而 2 代表 scratch，接下來的數字均代表影像分割完的結果，也就是 mask 的座標點，這些點都須以正規化後的結果來表現，因課堂後面所更新的資料均只包含 bounding box 的資訊，因此選擇使用第一版本的資料來進行轉換，轉換的結果如圖 3。

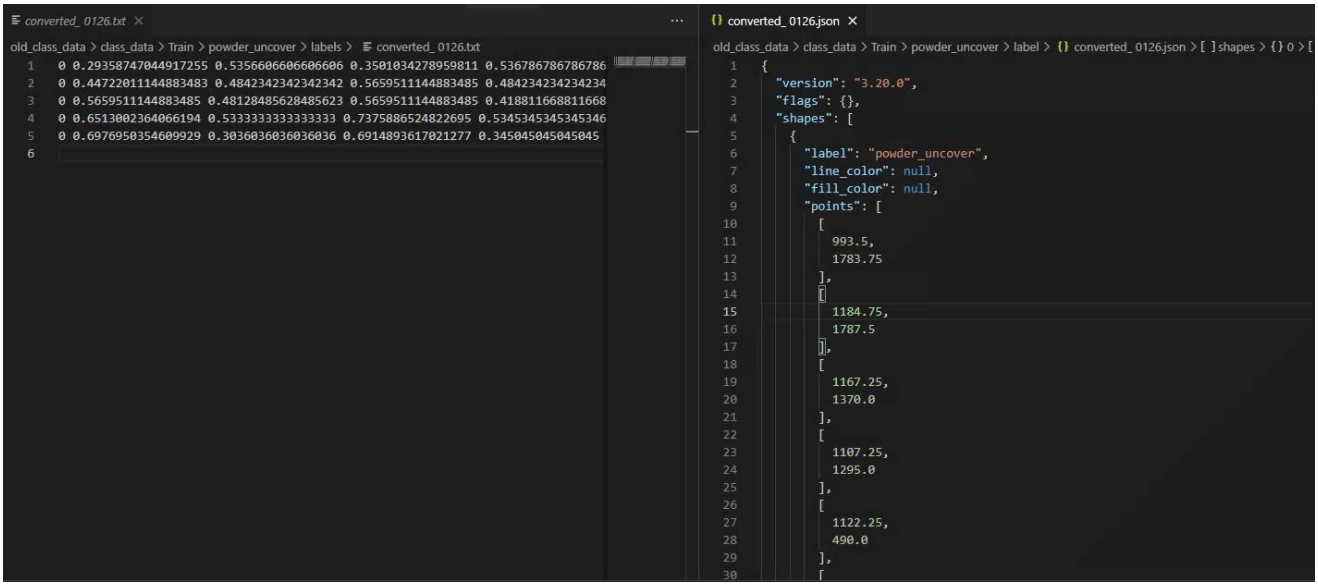

圖 3 label 資料轉換前(右圖)與後(左圖)比較

模型訓練

本次採用的 yolo v7 版本為 u7， 訓練的方式採用 4 fold cross-validation(如圖 4)來進行，將原始的訓練資料拆成 4 等分，每一等分輪流當驗證集，因此模型整體會訓練 4 次，每一次的 epoch 設定為 300，並從中取最好的結果進到下一輪繼續訓練。而第一輪會以 yolov7.pt 當作 pre-training 來進行輸入。整體的訓練結果如圖 5。

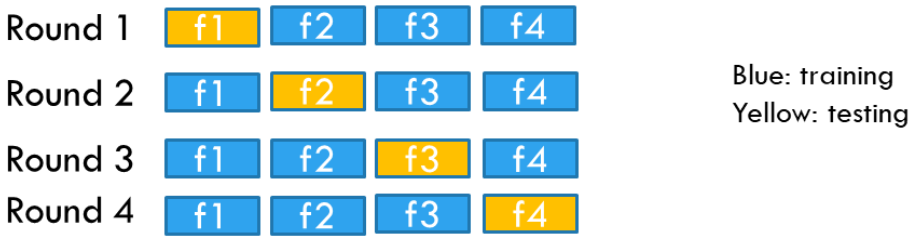

圖 4 4 fold cross-validation

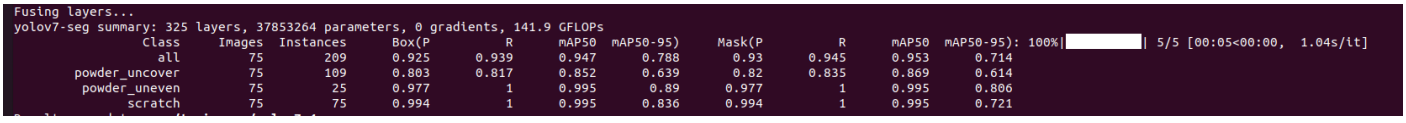

圖 5 模型訓練結果

結果評估

本次製作 UI 來讓使用者方便觀察結果(如圖 6)，UI 整體分為 5 個部分，其功能分別為:

1. Load 按鈕點擊後可以選擇與偵測的資料夾，所選擇的資料夾內必須包含 image、label、mask 這三個資料夾，分別包含原始影像、標籤結果以及影像分割結果。選擇好資料夾後點擊 Detect 按鈕就會開始進行偵測，全部偵測完後就會將結果展示於 UI 上。
2. 在此區塊將會顯示出原始影像(左)以及偵測後的結果(右)，偵測的結果將 detection 與 segmentation 合在一起進行顯示，方便使用者觀察。
3. 顯示出目前的圖片張數。
4. 左邊會顯示出各瑕疵分類再原始影像的真實數量，右邊則會顯示各瑕疵分類的偵測數量。
5. 此部分會顯示出 4 種參數來讓使用者評估模型的好壞，分別為:
   1). FPS:
   當前影像的處理時間取倒數。
   2). IoU:
   當前影像的 Intersection-Over-Union(IoU)數值，若偵測到的瑕疵大於 1 個，則為平均 IoU。
   3). AP50:
   計算 IoU 大於 0.5 的平均精準度(Average precision, AP)。
   4). Dice:
   也稱為 F1-Score，主要在計算真陽性率(True Positive Rate, TPR)和陽性預測值(Positive predictive value, PPV)的調和平均數(Harmonic mean)。

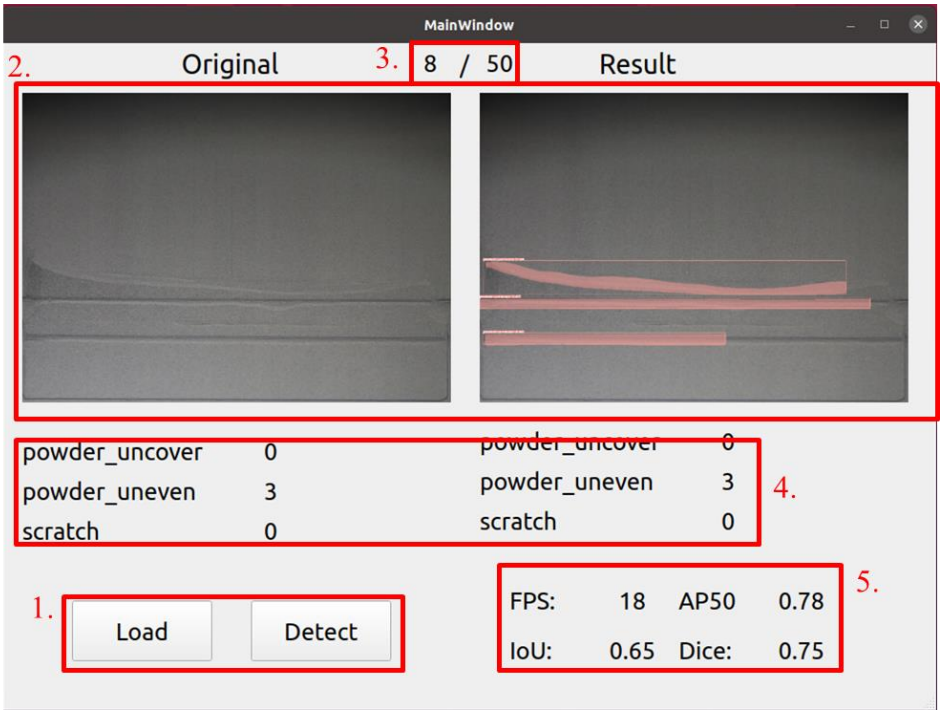

圖 6 UI 介面。

使用測試集來評估模型的準確度，將各平均項數據整理成表格 1。

表格 1 測試集各瑕疵分類數據表

Average	IoU	Dice	AP50
powder_uncover	0.651255	0.686920	0.672324
powder_uneven	0.663487	0.796257	0.905030
scratch	0.681030	0.778840	0.867627

總結

本次專案採用 yolo v7 以 one stage 的形式來進行物件偵測與分割，為了與 two stage 進行有效的比較，這邊引用其他同學採用 yolo v5+unet 的結果來進行比較。

表格 2 yolo v5+unet 結果表

Average	IoU	Dice	AP50
powder_uncover	0.73116	0.66962	0.61791
powder_uneven	0.88171	0.88247	0.98444
scratch	0.91178	0.89227	1.0

透過觀察表格 1 與表格 2 可以發現，yolo v7 的實作結果在 IoU、Dice 與 AP50 的表現上面均不如 yolo v5+unet，其中最特別的是理論上 v7 在物件偵測上要比 v5 好，但整體的 IoU 卻是 v5 比較高，推斷可能是不同的 label 造成此項結果，如單純使用 v5 進行物件偵測，輸入的 label 即為 bounding box，而我們直接採用 yolo v7 進行影像分割，因此輸入的 label 從原本的 bounding box 變為 mask 的結果。但在速度上 yolo v5+unet 平均 fps 落在 1~2 之間，但我所使用的 yolov7 平均落在 17~18 之間，在速度上有很明顯的差別。