

Digital Image Processing Final Project:

電機所 N26110833 陳居廷

Problems

Metal laser melting manufacturing (MAM) is a widely used technique in various industries, including the medical, manufacturing, aerospace and boutique industries. However, during the manufacturing process, defects may occur due to thermal stress or hardware failure. In order to improve the quality of the product, it is important to detect and segment defects during the manufacturing process. Our project aims to implement a system for detecting and segmenting three specific types of defects in MAM: powder uncover, powder uneven, and scratch.

Methods

1. Detection

在本次實作中我使用了 detectron2 框架中提供的 faster-RCNN 模型來做為 object detection 的模型架構。Faster R-CNN 是一種先進的 object detection 模型，它是由 R-CNN、Fast R-CNN 和 Faster R-CNN 三種模型發展而來。它採用了兩個子網路組成，一個是基於 CNN 的特徵提取網路，另一個則是基於 RPN 的候選區域生成網路。在特徵提取子網路中，通過 CNN 提取圖像特徵。在候選區域生成子網路中，通過 RPN 生成候選區域，再利用 RoI pooling 將候選區域轉換為統一的尺寸，最後使用全連接層對候選區域進行分類和回歸。Faster R-CNN 的主要優勢在於能夠在一個網路中同時實現特徵提取和候選區域生成，減少了訓練的時間。另外也能達到較高的準確性，已在多個資料集上獲得 state-of-the-art 的結果。而在程式中，我使用了 COCO Detection 在 detectron2 中已經訓練好的模型，透過指定在檔案中的參數並使用自己的資料集來訓練模型。最後再用測試集來驗證模型的準確性，並用於物體檢測的應用。

2. Segmentation

在 segmentation 實作的部分我使用了 detectron2 框架中提供的 Mask R-CNN 模型來做為 Image Segmentation 的模型架構。Mask R-CNN 是一種基於 Faster R-CNN 的模型，它除了能偵測物體的邊界框外，還能夠生成物體的 pixel-wise mask。它採用了兩個子網路組成，一個是基於 CNN 的特徵提取網路，另一個則是基於 RPN 的候選區域生成網路。在訓練模型時，我們使用了從 mask 目標的輪廓點取得的照片來訓練模型。透過指定在檔案中的參數並使用自己的資料集來訓練模型。最後再用測試集來驗證模型的準確性，並用於物體檢測及 mask 預測的應用。

3. FPS

我使用了 Python 中的 time 模組來計算 FPS (Frames per Second)。以我的了解 FPS 指的是在偵測和分割的過程中，每秒能夠處理的圖像數量。這能夠幫助我們了解實際應用中的偵測和分割的速度。我會在偵測和分割開始之前記錄一個初始時間戳記。然後在偵測和分割完成之後，再記錄一個結束時間戳記。最後，我會用這兩個時間戳記的差值來計算 FPS，這樣就能得到在一段時間內，處理的圖片的數量。這就是我使用 Python 中的 time 模組來計算 FPS 的過程。

4. IOU

Intersection over Union (IoU) 是一種用於評估兩個框 (bounding box) 重疊的程度。用來計算 gt_box 和 pred_box 交集與聯集的比值。利用 gt_box 和 pred_box 的坐標。通常這些坐標是指框的左上角和右下角的坐標 ($x_{min}, y_{min}, x_{max}, y_{max}$)。再來計算交集，需要找到左上角和右下角坐標的最大值和最小值，找到左上角坐標(x_{min}, y_{min})是 $(\max(gt_box_x_{min}, pred_box_x_{min}), \max(gt_box_y_{min}, pred_box_y_{min}))$ ，右下角坐標(x_{max}, y_{max})是 $(\min(gt_box_x_{max}, pred_box_x_{max}), \min(gt_box_y_{max}, pred_box_y_{max}))$ 如果左上角坐標(x_{min}, y_{min}) > 右下角坐標(x_{max}, y_{max}) 表示沒有交集，IoU 為 0。否則交集面積就是 $(x_{max} - x_{min}) * (y_{max} - y_{min})$ 再來計算框的聯集，聯集面積 = $gt_box_area + pred_box_area - 交集面積$ 。最後， $IoU = \frac{交集面積}{聯集面積}$ ，但是 predict 和 gt 數量可能會不同，需要進行 NMS，當預測多個框時，NMS 可以幫助減少重疊框的數量。NMS 基本上通過從預測框中選擇具有最高得分的框，並在附近的框中計算重疊，從而去除具有較低得分的重疊框。

5. AP50

我是計算 gt_box 與 pred_box 的 IoU 超過 0.50 的 pred_box 的平均精確度。計算方法是對每個 pred_box 計算其與 gt_box 的 IoU 值。如果 IoU 值大於等於 0.5, 將該預測框算作正確計入, 否則算作錯誤標記。 計算每個 pred_box 的精確度(precision) , 並對所有預測框的精確度取平均。

6. Dice coefficient

在這裡我是使用 $2|A \cap B| / (|A| + |B|)$, A 和 B 分別表示 pred_mask 和 gt_mask , $|A|$ 和 $|B|$ 表示集合 A 和集合 B 的元素個數 , $A \cap B$ 表示集合 A 和集合 B 的交集。

7. GUI

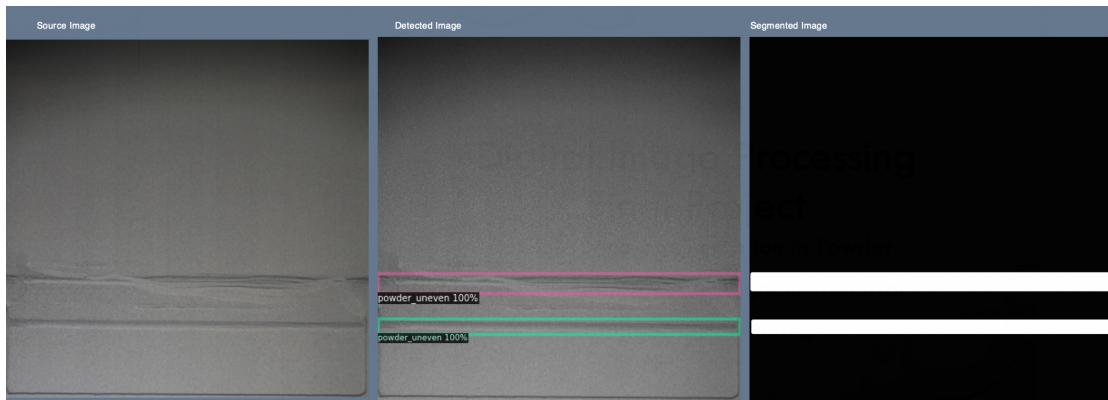
在本次實作我使用 PysimpleGUI · PySimpleGUI 是一個跨平台的 Python GUI 庫 , 它提供了一個簡單易用的 API , 可用於創建各種類型的圖形用戶界面。 PySimpleGUI 使用 tkinter、wxPython 或 PyQt 作為底層 GUI 框架 , 但是我在設計上較為陽春。

Results

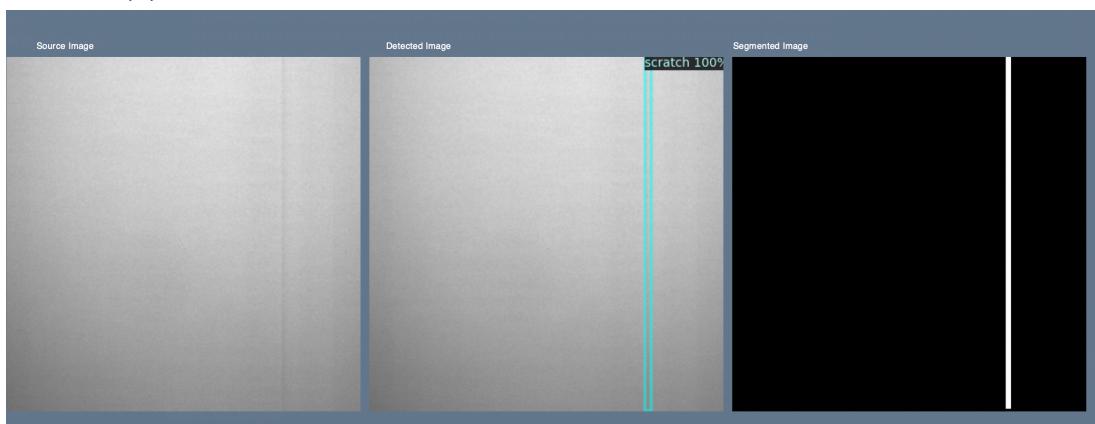
(1) Powder uncover



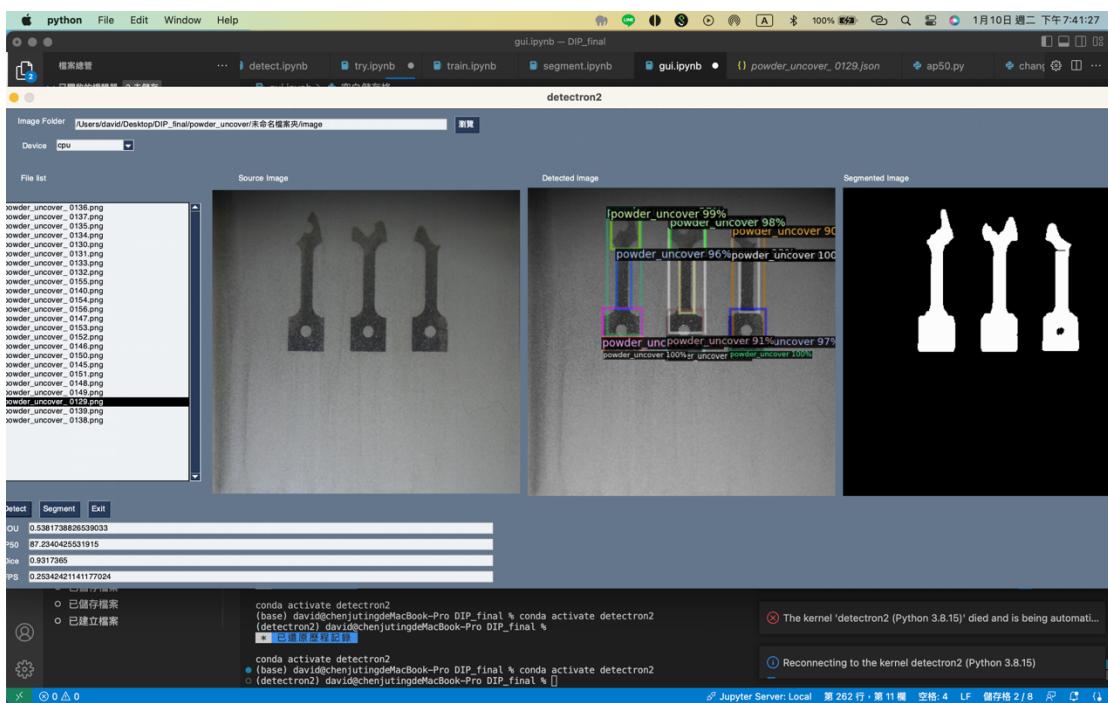
(2) Powder uneven



(3) Scratch



GUI layout



Discussions

在預處理數據時，首先，uncover 和 uneven 有些文件名重複，遇到了 label 中的邊界框的問題，以及 label 和原始圖片中之間的差異。具體來說，一些 bbox 的方向不同，且許多 json 文件中的標簽名稱也不正確。此外有些 mask 只有幾個 instances，但 json 中對應的 ground truth 却給出了更多的 instance，以上的問題造成在預處理時在圖片和 json 檔的更改遇到很大的困境，對於後續的各種計算上也因此遇到很多麻煩。

Conclusion

這次我使用 Detectron2，來快速建立一個模型，此框架相當的方便，包含多種如 Faster R-CNN·RetinaNet·Mask R-CNN2 等等算法來進行 object detection，並且有一些可以做後處理的方法和支援多種平台，這次另外最大的挑戰是實作 python 的 G U I，由於對於 G U I 是完完全全的新手，因此在實作上花費了許多時間，也在這次的期末專題中學習到了很多技能。