深度學習影像偵測結果分析工具

Deep Learning Object Detection Result Evaluation Tool

作者：王人禾

系級：電子工程學系碩一

學號：M0906013

開課老師：王通溫

課程名稱：儀器人機介面設計與分析

開課系所：電子工程學系

開課學年： 109 學年度 第 二 學期

中文摘要

對於視覺神經網路的蓬勃發展，越來越龐大的運算需要移植到邊緣運算平台，需要更低的能耗表現，常使用量化(Quantize)的技術來降低資料寬度，這點讓資料搬運、儲存與運算的成本相較於單精度浮點數的成本更低。但是從中衍生的就是精確度損失的問題。

對於以物件框輸出作為偵測結果的神經網路，通常以該神經網路對於相同的 Ground Truth 資料集進行測試所得出的 mAP (mean Average Precision)來評估神經網路的準確性。mAP 的確能表示該神經網路的準確度，但是以實際應用上，仍然需要直接檢視結果來判斷。

這次研究是希望單純以物件框的在原始圖片中的位置來觀察神經網路的偵測結果，也就是更方便的比較不同神經網路和實現流程在實務上的成果。並且在自己的實驗中，需要比較相同測試資料在不同資料寬度和網路架構的偵測結果。或能透過這次做出的工具，在之後的深度學習網路的訓練中，增強神經網路訓練的不足或對結構做調整。

首先我先訓練出幾組權重，並透過 Quantization 將其轉換為 int8 的資料格式，在不同平台進行物件偵測，並取得各自的偵測結果。接著透過 PyQt5 設計一個 UI 來讓我想要觀察的資料能夠更容易的直接呈現。最後，這裡做出了一個能夠比較相同 Ground Truth 透過不同神經網路實現流程的偵測結果的工具。

關鍵字：OpenCV、 PyQT5、Vitis AI

Abstract

Because of the flourishing development of visual neural networks, in order to transplant more and more large calculations to the edge computing platform, lower energy consumption performance is required. And quantization is often used to reduce the data width. This makes the cost of data handling, storage, and calculations lower than the cost of single-precision floating-point data. But what derives from it is loss of detection accuracy.

For convolution neural networks(CNNs) that use bounding boxes as the detection result, the accuracy of the CNNs are usually evaluated by mAP (mean Average Precision) obtained by testing the CNNs on the same ground truth data set. This can indeed indicate the accuracy of the CNNs, but in practical applications, it is still necessary to directly inspect the results to judge.

This research hopes to observe the detection results of the CNNs by checking the position of the bounding boxes in the image, that is, it is more convenient to compare the practical results of different CNNs and implementation processes. And in my own experiment, I need to compare the detection results of the same test data in different data widths and network architectures. Perhaps through the tools made this time, in the subsequent deep learning network training, the deficiencies of neural network training can be enhanced or the structure can be adjusted.

First, I train a few sets of CNN weights, and convert them to int8 data format through quantification, perform object detection on different platforms, and obtain their respective detection results. Then design a UI through PyQt5 to make the data I want to observe more easily and directly presented. Finally, here is a tool that can compare the detection results of the same ground truth through different neural network implementation processes.

Keyword：OpenCV, PyQT5, Vitis AI

目 次

二、研究動機

由於在自己的實驗中，為了將 GPU 訓練得到的權重移植到 FPGA 或是其他邊緣運算平台，所以需要測試各種不同的神經網路以及不同資料寬度的下準確度，在之前的經驗中，儘管 mAP 在經過轉換後可以做到沒有非常大的差別，但是在實際上的物件框位置還是會有所差距，因此有了設計這個分析工具的想法。

2-1 YOLO

視覺深度學習模型中，YOLO 是近年主流的深度學習模型之一。在 2015 年公布第一版的論文。因為它是一個 one-stage 的物件偵測模型，所以它有好的執行速度，但是同時他又可以有不錯的準確度。

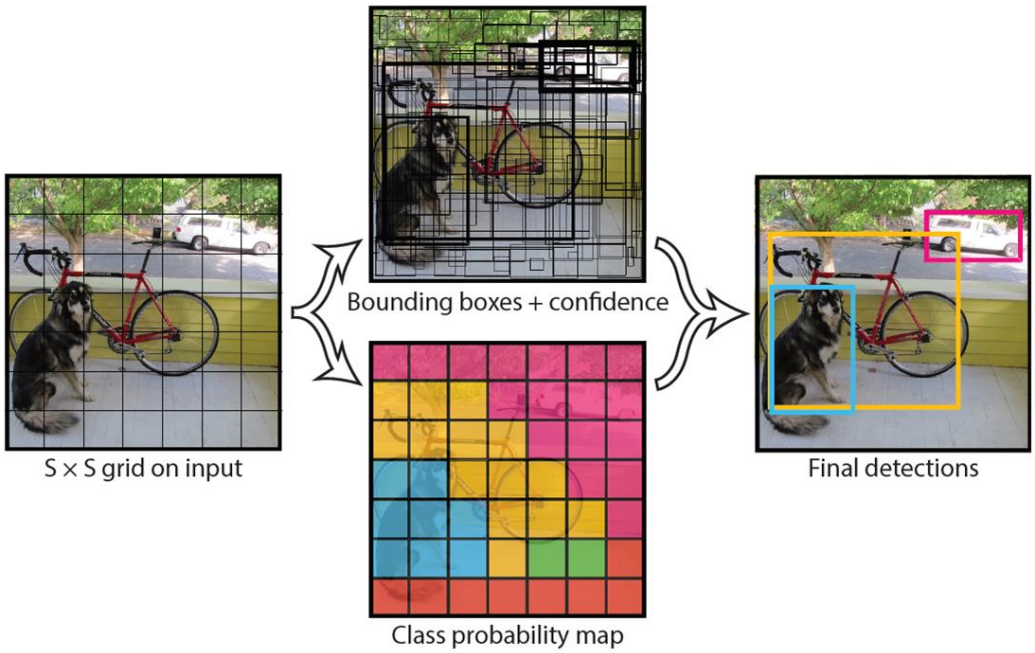

Figure 1 YOLO[1]運作方法

圖 1 是 YOLO 的作法：首先會將輸入圖片切成多個 Grid Cell，盡可能得到多個物件框的位置和信心程度，透過設定閥值，先過濾一部份的物件框。過濾完後，篩選重疊的物件框，重複執行直到完成所有類別。

同時還會有另外一個輸出是每個 Grid Cell 可能的物件類別，透過結合兩者，便能得到最後的預測結果。

2-2 mAP

一般而言神經網路藉由 mAP 評估神經網路的好壞，這個指標可以顯示這個神經網路的準確度。這個指標是透過 Precision、Recall 和 IoU (Intersection over Union)三個指標所得出。Precision 是指被判斷為某類的物件實際上也為該類物件的比例；Recall 則是某類物件被判斷成該類物件的比例。

$$Precision = \frac{TP}{TP + FP}$$

$$Recall = \frac{TP}{TP + FN}$$

IoU 則是用來評估物件框的精準度使用，它的定義很簡單，數學式的表示為預測物件框與 ground truth 物件框的交集除以聯集。

$$IoU = \frac{result\ \ bbox \cap ground\ \ truth\ \ bbox}{result\ \ bbox \cup ground\ \ truth\ \ bbox}$$

mAP 是由不同偵測類別的 AP(Average Precision)取平均獲得。AP 的計算是透過 IoU 設定閥值，若物件框和 ground truth 的 IoU 超過設定的閥值則判定為偵測到該物件，接著將所有物件框透過信心指數(Score)進行排序，依序進行 Precision 和 Recall 的計算，依此得出一個 Precision-recall Curve。AP 是將 Precision-recall Curve 拉平後，計算曲線下的面積。

$$\text{mAP} = \sum_{k=1}^{n} Average\ \ Precision\ \ of\ \ class\ \ k$$

Figure 2 兩個網路對相同圖片的偵測結果，設定不同 Score。

在圖 2 中，左側為 Score 設為 0.1 時的結果，右側為 Score 設為 0.5 的結果。紫色框和橘色框分別是兩種深度學習網路的偵測結果，半透明的青色則是 Ground Truth。以後方左邊第二名男性來說，紫色框更符合 Ground Truth 的位置，但是在 Score 設為 0.5 的時候並不在畫面中。

三、研究方法

這次的工具是希望看到神經網路的權重在 GPU 上使用 32 位元浮點數和 FPGA 上使用 8 位元定點數的偵測導致的差異。同時這個工具也可以直接比較兩種不同的模型的偵測結果。這一章會分兩個部分做說明，第一個部分是如何取得偵測結果，第二部分是如何利用偵測結果來分析。

3-1 影像偵測

第一個部分是取得偵測結果的方法。偵測結果除了透過 GPU 直接進行測試，還有一組是透過 Quantization 將 32 位元浮點數轉為 8 位元定點數，並透過 FPGA 進行測試的方法。

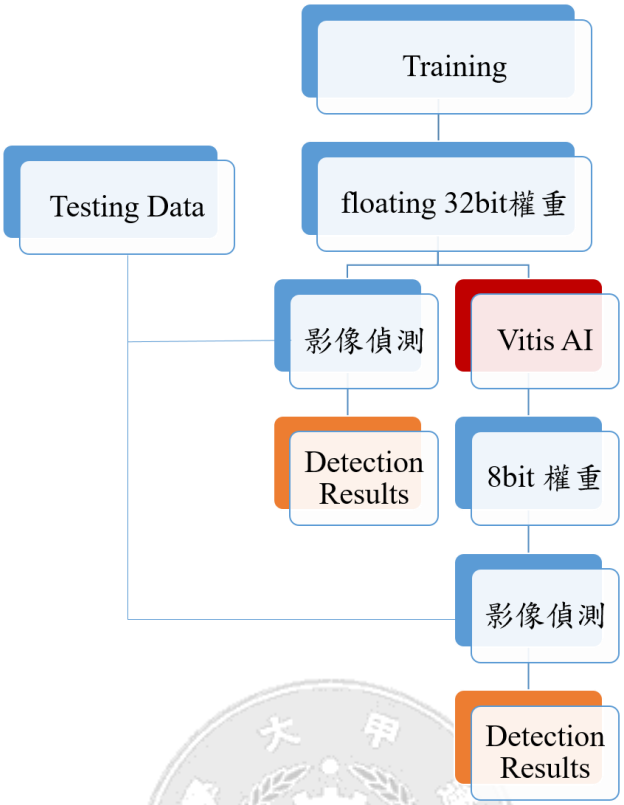

Figure 3　影像偵測結果取得流程圖。

如圖 3，神經網路經過訓練後可以取得 32 位元浮點數的權重，直接使用這組權重來進行影像偵測作為第一組測試資料，輸出的結果對應原始圖片產生相同名稱的文字檔，儲存於一個資料夾。

在 FPGA 的實現方式是透過 Xilinx 的 Vitis AI[3]，先將 32 位元的浮點數權重轉換為 8 位元，並且移植到 ZCU102 上進行影像偵測，和 GPU 儲存結果的方式相同，對應原始的圖片儲存相同名稱的文字檔於一個資料夾。

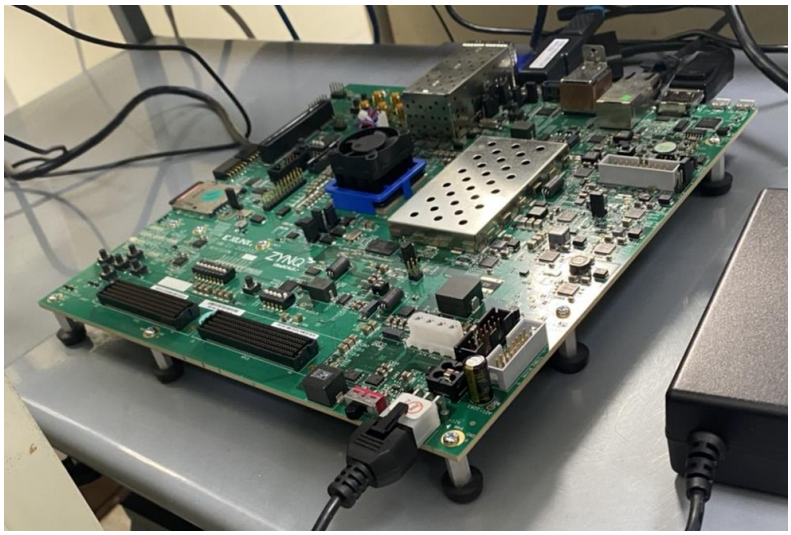

Figure 4　ZCU102

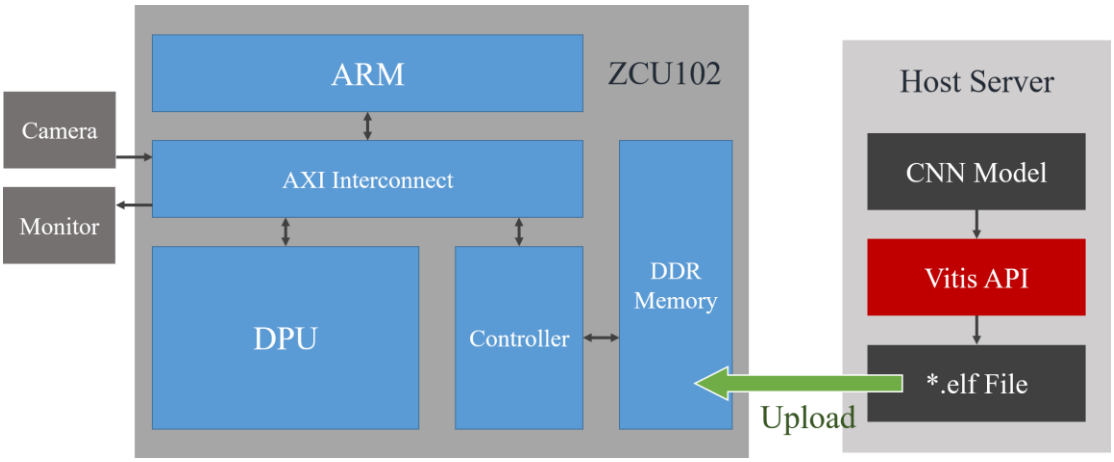

Figure 5 使用 FPGA 進行偵測的流程

圖 5 左半部是 FPGA 的部分，透過 Xilinx 的 Vitis AI API[3]可以將深度學習模型轉換至 ZCU102 上的 DPU 運行。DPU 是 Xilinx 設計的一個 AI 運算加速器，它可以在不同的 FPGA 上設定不同的參數來分配其運算平行度，並且支援它在 FPGA 上的 AI 運算。

我使用 Tensorflow 進行 GPU 上的物件偵測，使用的模型架構有兩個，一個是比較輕量的模型，另一個是 YOLOv3[2]。

3-2 分析工具

使用 PyQT5 進行主要的 UI 設計，物件框使用 OpenCV 完成繪製。使用的程式語言主要為 Python。

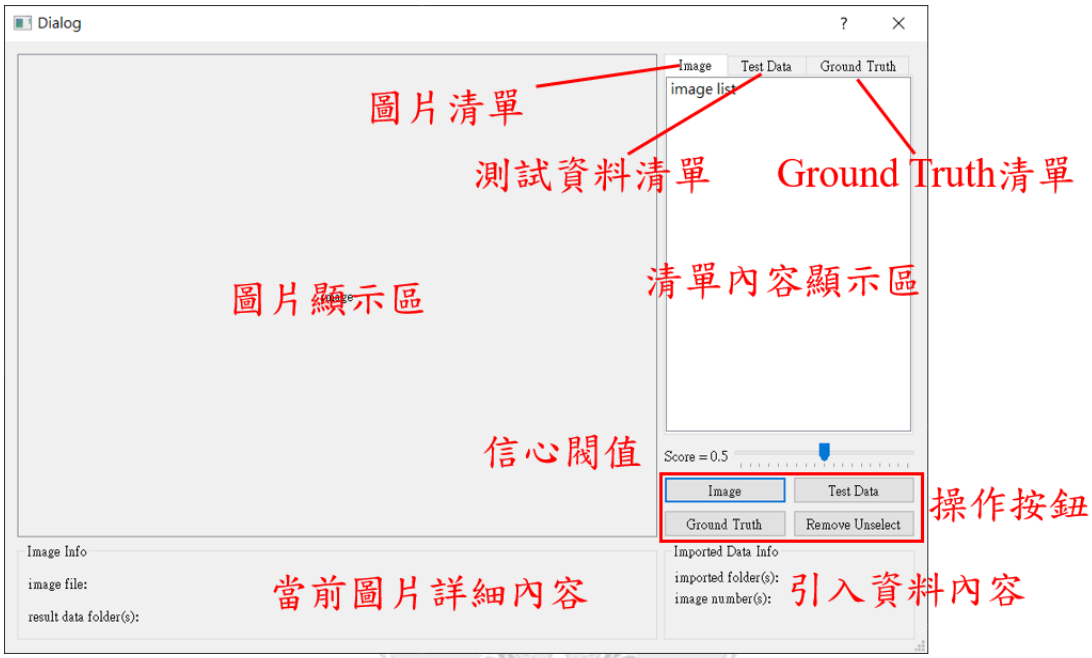

Figure 6　UI 介面詳細說明

在圖 6 中列出了這次的 UI 的介面，在下面詳細說明：

1. 圖片顯示區：這個區塊可以透過選取圖片清單中的圖片來顯示，並且如果在測試資料中有選取的圖片的話，也會將物件框畫出。
2. 圖片清單：選取圖片清單會在下方列出選擇的資料夾中的所有圖片。可以直接點選清單中的圖片，顯示於左側。
3. 測試資料清單：這個清單會列出透過上 3-1 中的方式產生的測試結果的資料夾，透過勾選來決定是否顯示於圖片上。呈現方式是矩形，依照不同的測試資料有不同顏色。
4. Ground Truth 清單：這個清單會列出 Ground Truth 的資料夾，透過勾選來決定是否顯示於圖片上，呈現方式是矩形半透明遮罩。
5. 信心閥值：可以透過信心指數進行篩選，過濾信心指數較低的物件框。
6. 操作按鈕：和上方的清單有對應的名稱來加入對應的資料。右下角的 Remove Unselect 則是可以刪除測試資料清單和 Ground Truth 清單中被取消勾選的資料。
7. 當前圖片詳細內容：分為兩部分，當前圖片名稱與路徑、測試資料與路徑。測試資料包含 Ground Truth 和偵測結果，並且會在最後面顯示測試

資料在圖中有多少物件框。

8. 引入資料內容：顯示有多少圖片、多少筆測試資料。

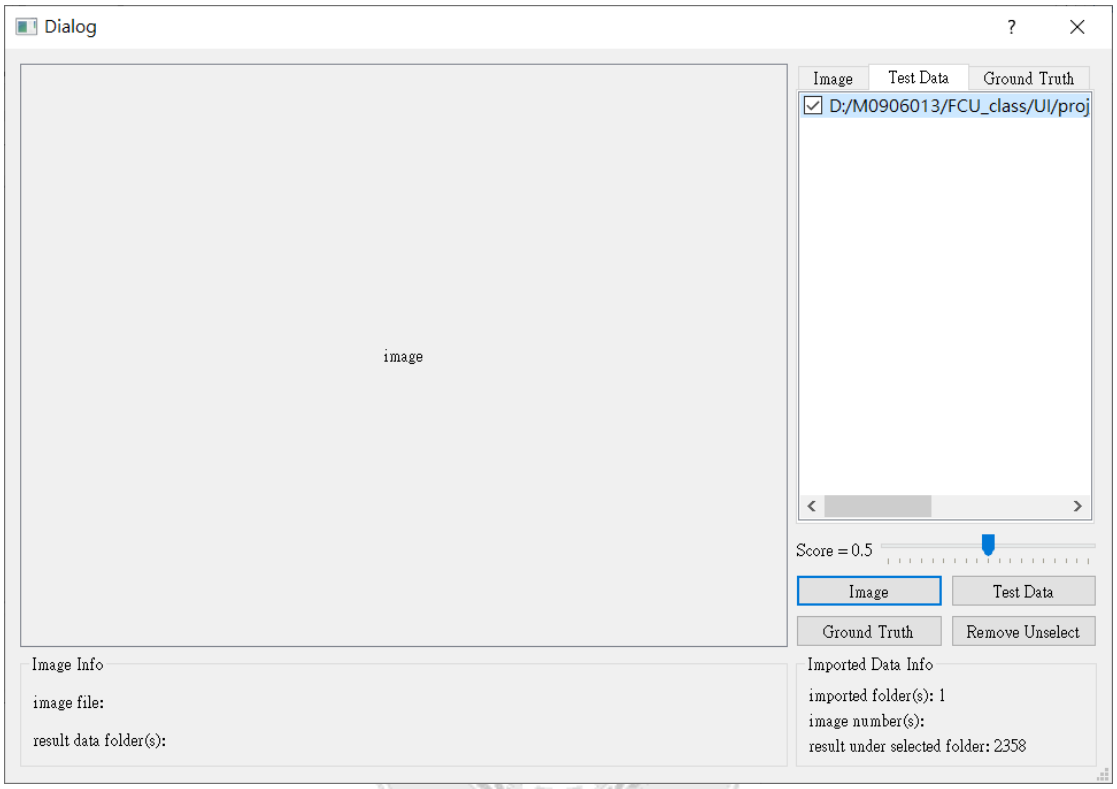

Figure 7　UI 介面

圖 7 在測試資料的清單中選擇的話，會在引入資料內容區塊中顯示出這個測試資料中有多少檔案。可以和圖片數量做比較，用來檢查是不是選錯資料夾。

四、結果

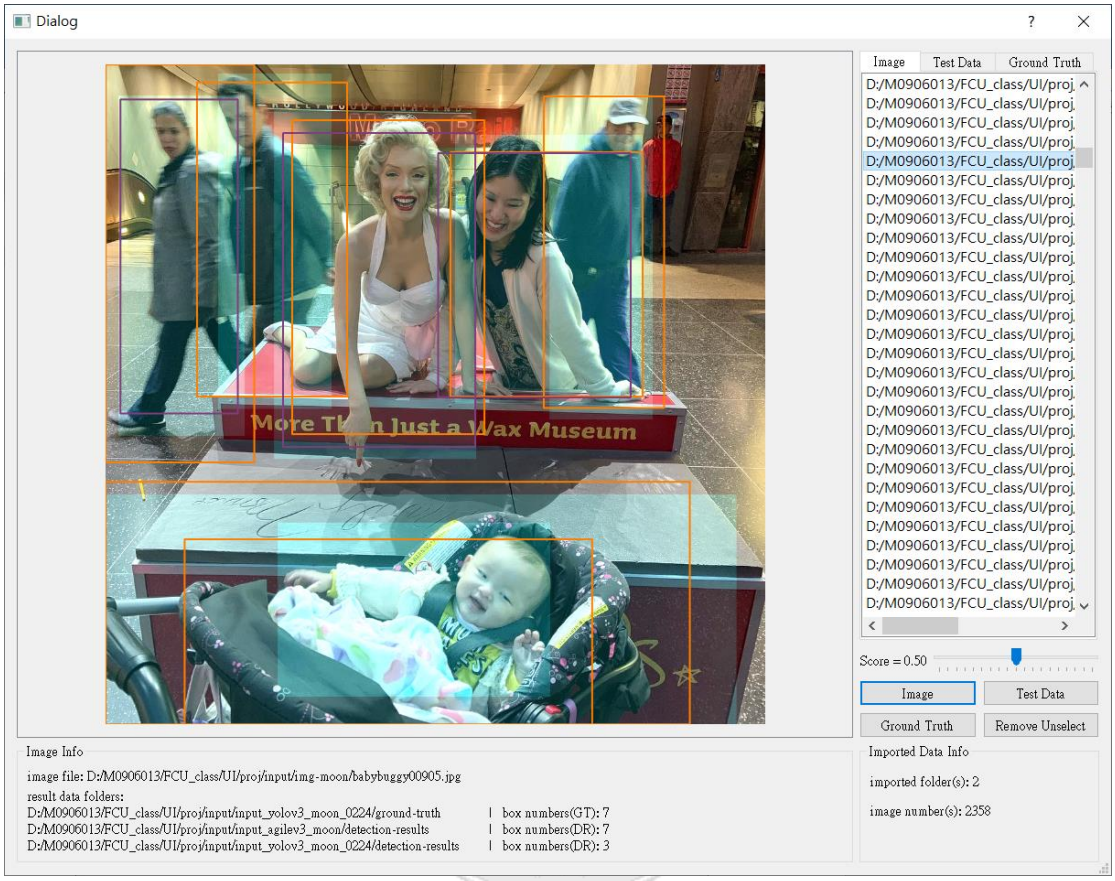

Figure 8 使用中畫面

圖 8 是顯示一張圖片的畫面，青色的遮罩是 Ground Truth，橘色和紫色分別是兩筆測試資料的物件框。圖片資訊的部分有顯示在圖中有幾個物件框被繪出，GT 是 Ground Truth，DR 是測試結果 Detection Result。這組測試圖片有 2358 張圖片。

五、結論與討論

我設計的分析工具可以快速地檢視多筆偵測結果的不同處，也可以看到偵測結果和 Ground Truth 的比較。但是仍有改進空間。

首先是 Ground Truth 的呈現方式，原先使用的是黃色遮罩，但是在楓葉林的照片或傍晚的場景很容易辨識不清，這是因為圖片本身的顏色就偏黃，現在的顏色若在清晨的場景會降低辨識效果。

接著是測試結果的物件框，目前是使用隨機的顏色，所以不能明確知道哪一個顏色是哪一筆測試資料所繪製，只能從物件框數量判斷，這也是需要改進的地方。

整體而言，它是一個很基本的分析工具，並且可以進一步改良並完善功能。

六、參考文獻

[1] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You Only Look Once: Unified, Real-Time Object Detection,” in 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, Jun. 2016, pp. 779–788. doi: 10.1109/CVPR.2016.91.

[2] J. Redmon and A. Farhadi, “YOLOv3: An Incremental Improvement,” arXiv:1804.02767 [cs], Apr. 2018, Accessed: Aug. 31, 2021. [Online]. Available: http://arxiv.org/abs/1804.02767

[3] “Vitis AI User Guide,” p. 171, 2020.