Intelligent Integration of Enterprise

Project 3

資源回收自動糾察系統-

應用物件偵測模型

Automatic Detection of Recycled Waste in Trash Can

Group 5

109034535 葉子匯

指導教授：邱銘傳 教授

一、背景介紹

1. 前言

公共場所如公司社區與校園，大都有集中垃圾的區域，多數垃圾集中區域將垃圾桶與資源回收分類桶相鄰擺放。即使大力宣導資源分類，多數垃圾集中區由民眾自發性分類，**未經監督情況下，仍有不小比例的資源回收物品會被投入垃圾桶，迫使各單位需要雇用大量清潔員額外為垃圾分類**。

本研究由校園做起，觀察校園內垃圾與資源回收的分類方式，最大比例混合在垃圾桶中的回收物與餐飲相關:包含紙餐盒、飲料紙杯、寶特瓶等。然而紙餐盒與飲料紙杯除較乾淨不油膩的外，其餘多數因髒污比例大會被歸類為一般垃圾。依照校內分類標準，上述三類回收物僅寶特瓶必須投至資源回收桶，同時**學生族群每日消耗 500 個以上寶特瓶，為資源回收之重點項目，亦為本研究針對的回收物目標**。

本研究使用物件偵測演算法，發展自動偵測系統，偵測在投擲垃圾至垃圾桶的過程中是否有資源回收物(本研究針對寶特瓶)的出現。當偵測到回收物，發出警告以抑止亂丟回收物的現象，改善各區域分類回收物的狀況，提升環境品質。

本計畫自行蒐集用於模型訓練之垃圾桶圖

2. 5W1H

校園內垃圾集中區之資源回收分類，皆須由投擲者自發性將垃圾與資源回收物分開投擲至不同分類的桶子中。即使有規劃完善的資源回收桶與垃圾桶，在無監督情況下仍會有回收物被頻繁投入垃圾桶中。當垃圾清運時又需執行嚴格的資源回收分類，造成校方需額外雇用清潔人員協助做垃圾的二次分類。考慮到以上情形，本計畫先以 5W1H 進行問題定義。

5W1H 表格

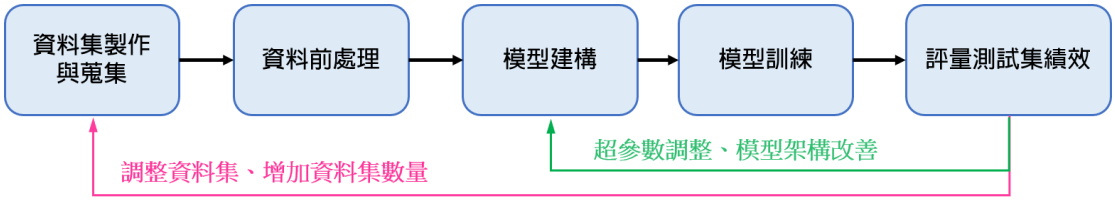
Who	校方、清潔人員、環保署
What	回收物的分類
When	1.投擲垃圾時，需投擲者自發性分類回收物，回收物投入垃圾桶狀況頻繁

	2.垃圾清運時，需進行嚴格的資源分類
Where	在垃圾集中區現場，可預防資源回收物的錯誤分類，減少清潔人力
Why	因人力成本高昂，無法雇用全天人力協助監督垃圾場的垃圾分類，但垃圾清運時須進行嚴格的資源分類，需雇用額外人力每天檢查並替垃圾桶垃圾做分類。
How	資料建立、深度學習、機器學習

透過電腦自動偵測系統全天候監視垃圾場之資源回收狀況，在資源回收物落入垃圾桶時給予投擲者警告，給予投擲者提醒與壓力需確實做到垃圾的回收分類。除了降低雇用清潔人員做額外垃圾分類之困擾，長期更能養成民眾為垃圾隨手分類的好習慣。

3.計畫流程

本計畫的流程如下圖所示。首先拍攝不同情境下的垃圾桶圖片，進行資料的前處理。接下來，建構模型並進行訓練，並且測試其績效成果，再根據結果進行超參數調整與模型架構的改善，或在資料集中新增更多情境之圖片，以達到偵測回收物的最佳效果。



計畫執行流程圖

二、模型訓練與績效

1. 資料蒐集

本研究使用監督式機器學習方法，故需一定數量之資料才能訓練績效良好的模型。由於目前無有關垃圾桶與資源回收相關之公開數據集，資料集內所有圖片由本研究自行拍攝而得。在白天與晚上拍攝小吃部與宿舍區域的垃圾集中區的垃圾桶，從垃圾桶正上方視角，分別搭配 4 種常見的寶特瓶以各種投擲角度進行拍攝。

1.1 原始資料集

原始資料集為 258 張 3456px*3456px 垃圾桶與投入寶特瓶的圖片，可依寶特瓶種類與數量不同分為以下 5 類。所有圖檔均使用 labelImg 軟體標記後，每張圖檔產生一標記檔(.txt 檔)。

透明圓形正常規格（單一數量）	含包裝圓形正常規格（單一數量）	含包裝方形正常規格（單一數量）	介於圓形方形中間之規格（單一數量）	多個寶特瓶
				
65 張	48 張	51 張	56 張	38 張

原始資料集圖片類別統計

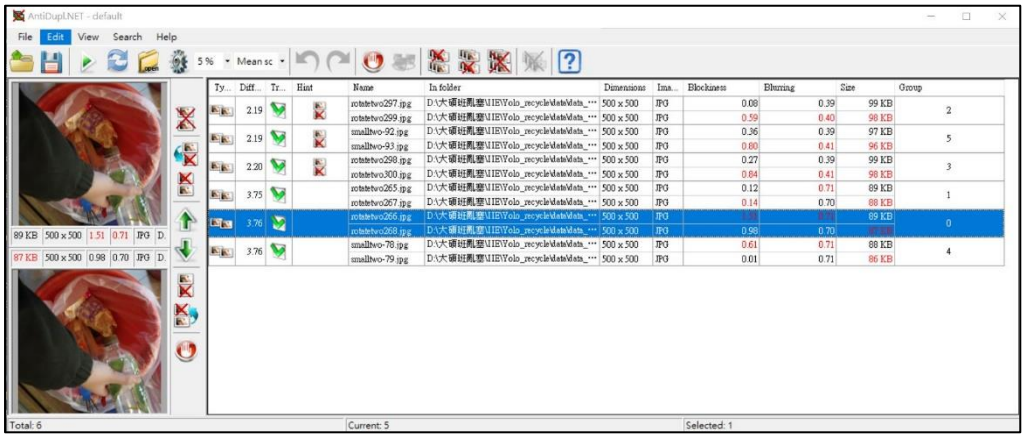
2. 資料前處理

本資料集為自製資料集，故資料前處理步驟較多。資料前處理步驟包括(1)資料清理、(2)資料尺度調整、(3)資料擴增、(4)資料標記、(5)訓練測試驗證集分割。

(1)資料清理

資料清理的主要目的為確認沒有異常與重複之資料。

首先於圖檔資料處理的部分，會先由人工檢查方式確認是否含有異常資料:全黑、空白、或背景模糊之圖片。接下來透過免費軟體 AntiDupl.NET 比較 blockiness、blurring 等圖片特徵找出重複圖片。



AntiDupl.NET 軟體畫面

(2)資料尺度調整

以物件偵測競賽資料集 Pascal VOC 為例，在每張圖片尺寸為 360px*480px 輸入物件偵測模型訓練的情況下，物件偵測模型已有良好的表現。為加快資料處理速度與縮短模型訓練時間，將原圖尺寸 3456px*3456px 調整為 500px*500px。

```
#>>>Resize
#===Read file in folder
onlyfiles = [f for f in os.listdir(path+'/data_org2')
             if os.path.isfile(os.path.join(path+'/data_org2', f))]
i=1
for _file in onlyfiles:
    img = cv2.imread('./data_org2/'+_file,cv2.IMREAD_COLOR)
    img_resized=cv2.resize(img,(500,500))
    cv2.imwrite(os.path.join('./data_small2', f'smalltwo-{i}.jpg'),img_resized)
    i+=1
```

資料縮小尺度之 python 程式碼

(3)資料擴增

原資料集圖片**數量不足 300 張數量過少**，使用 cv2.rotate()函式將原圖檔順時針旋轉 90 度與 180 度來擴增資料集。**原始資料集之 258 張圖片，擴增後數量達到 774 張**。

原圖

順時旋轉 90 度

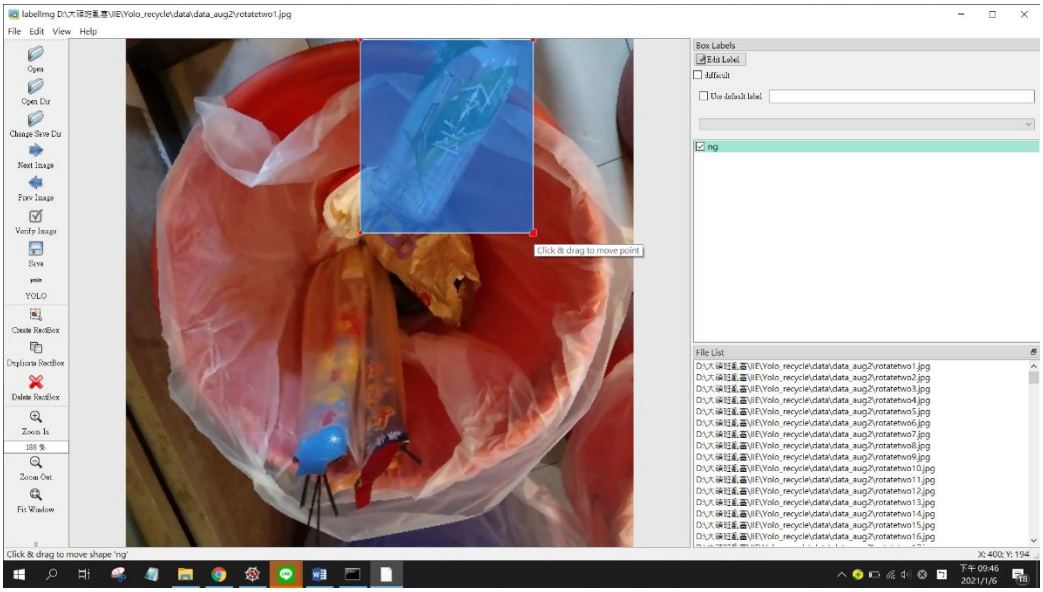
順時旋轉 180 度

```
#>>Data Augementation
#Rotation
#===Read file in folder
refiles = [f for f in os.listdir(path+'/data_small2')
             if os.path.isfile(os.path.join(path+'/data_small2', f))]
i=1
for _file in refiles:
    img = cv2.imread('./data_small2/'+_file,cv2.IMREAD_COLOR)
    #Rotate 90 degrees
    img_rotate1=cv2.rotate(img, cv2.ROTATE_90_CLOCKWISE)
    cv2.imwrite(os.path.join('./data_aug2', f'rotatetwo{i}.jpg'),img_rotate1)
    #Rotate 180 degrees
    img_rotate2=cv2.rotate(img, cv2.ROTATE_180)
    cv2.imwrite(os.path.join('./data_aug2', f'rotatetwo{i+1}.jpg'),img_rotate2)
    i+=2
```

資料擴增之 python 程式碼

(4)資料標記

使用 labelImg 軟體資料以人工標記出圖片中寶特瓶位置。



(5)訓練測試驗證集分割

透過 sklearn.model_selection. train_test_split 涵式，以亂數分配訓練集(65%)驗證集(20%)與測試集(15%)。

原始資料集	訓練集(65%)	驗證集(20%)	測試集(15%)
共 774 筆	525 筆	132 筆	117 筆

```
#>>Train test split
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(picfiles, txtfiles,
                                                    test_size=0.15, random_state=42)
X_train, X_valid, y_train, y_valid = train_test_split(X_train, y_train,
                                                    test_size=0.2, random_state=42)
#Copy files to certain folder
data_tr=[X_train, y_train]
data_ts=[X_test,y_test]
data_vl=[X_valid,y_valid]
for i in data_tr:
    for f in i:
        shutil.copyfile(path+'/data_aug2/'+f,path+'/data_2/train/'+f)
for i in data_ts:
    for f in i:
        shutil.copyfile(path+'/data_aug2/'+f,path+'/data_2/test/'+f)
for i in data_vl:
    for f in i:
        shutil.copyfile(path+'/data_aug2/'+f,path+'/data_2/valid/'+f)
```

切割訓練驗證與測試集 python 程式碼

3. 模型架構

本研究目標為建立即時探測目標物件(寶特瓶)之系統，故採用物件偵測模型為主架構。一般情形下投擲垃圾時，**垃圾於2-4秒內落入桶子中，為在垃圾落入前即偵測出回收物，模型偵測速度為本研究選擇模型架構的重要考量**。常見之物件偵測模型包括You Only Look Once(YOLO)、HSD為主的一階段模型(One-

Stage)及RCNN架構為主的二階段模型(Two- Stage)。其中以一階段模型之偵測速度較二階段模型快速，以YOLOv3為例，其在預測MS COCO資料集 (test-dev 2017)物件位置與類別之準確率AP達到31.0%的水準下，每秒最高可以35FPS(Frame Per Second)偵測，超越二階段模型Fast-RCNN於AP 39.8%之最高偵測速度9.4FPS。**因YOLO物件偵測模型的偵測速度優勢，本研究選定YOLO為模型架構。**

3.1 YOLO模型架構簡介

YOLO模型至今已更新到第四版(簡稱YOLOv4)，其最初架構在2016年由REDMON, Joseph等提出，包含24層捲積層與2層全連接層，設計end-to-end CNN架構使用多個bounding box同時預測圖片各區域並計算物體出現機率，大幅提高模型運算效率。於YOLOv2中採用以bounding box中心點為基準之anchor box設計，讓單一圖片區域可預測多個物體。2018年YOLOV3於架構上有突破性的發展，使用殘差學習(Residual Learning)與FPN(Feature Pyramid Network)方法提升對於小物件的偵測準確度。2020年提出之YOLOv4以YOLOv3為基礎，加入跨階段局部網路(Cross Stage Partial Networks, CSP)架構,大幅減少計算量,同時還得到了更高的準確率。在四版本的YOLO架構中，本研究採用YOLOv4之模型架構。

3.2 YOLOv4模型

YOLOv4模型之特點有三:

1.單一圖片區域內採用多組anchor box再計算物件出現機率進行比對找出物件位置與大小

2.採用多種物件偵測演算法之結構組合提升偵測績效

3.將資料輸入YOLO架構前提出兩種新方式擴增資料，提高模型泛化能力。

以下分別依此三項特點詳述。

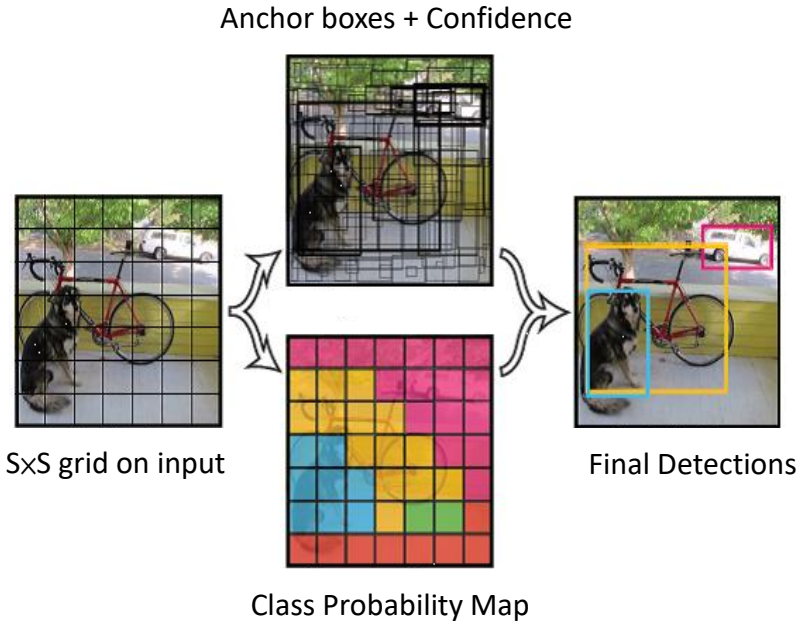
YOLOv4 模型訓練運作主要可分為四大步驟，其中 anchor box 的設計使模型有效預測物體的位置與類別。

1.調整輸入圖片尺寸為統一大小，並切割為 S×S 格座標格子(grid cell) 。每個座標格子預設 B 個以座標格子中心為中心點的 anchor box 用來框選物體位置。每組 anchor box 中紀錄的數值有:框中包含物體的信心程度(Confidence)與 C 個物件類別機率。

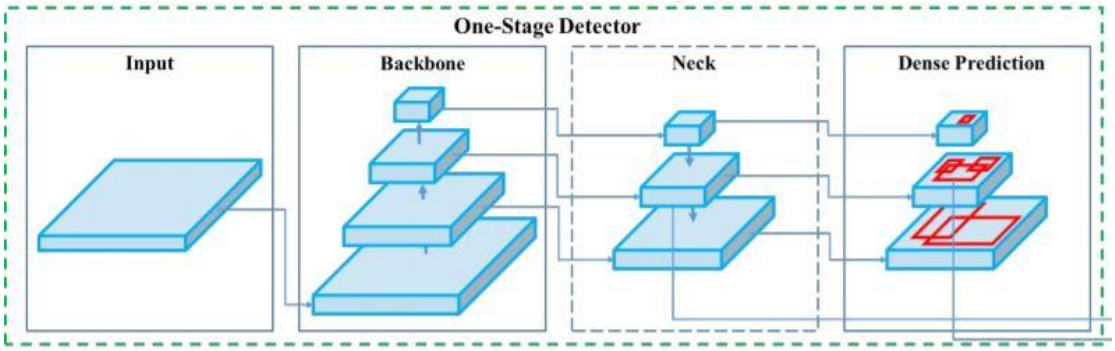
2. 執行一組 CNN 卷積神經網路調整 $S\times S\times(B*5+C)$個 anchor box 數值。

3. 模型輸出 anchor box 信心程度(Confidence)，再依據閾值(thresholds)和非極大值壓抑法(Non-max suppression)篩選得到最佳 anchor box。

4. 依據損失函數比較實際物件框選位置與 anchor box 預測值，反向修正模型權重。



架構設計上 YOLOv4 基於一階段物件偵測模型之結構，將不同偵測模型結構導入此基本架構。其主架構包含 Backbone、Neck 與 Dense Prediction 三部分。BackBone 主架構為 CSPDarknet53 網路，Neck 主架構為 SPP 加 PAN，HEAD 主架構為 YOLO HEAD 架構。



同時作者提出一創新的資料擴增方法 Mosaic 與加入 2019 提出的 CutMix 資料擴增方法，增加資料集變化程度，提升模型泛化能力。

Mosaic 作用為將四張圖像依不同比例拼接為單一圖片，而 CutMix 會擷取部分圖像 A 區域覆蓋至圖像 B 上。下圖為兩種擴增方法產出之圖像。

Mosaic 擴增方法

CutMix 擴增方法

4.模型績效比較與改善

本研究採用物件偵測常用之指標 MAP 與影片預測情形改善模型，當模型偵測效果不佳，依照兩大方向進行改善，分別為在資料集中新增更多情境之圖片或進行模型超參數調整。

方向一:在資料集中新增更多情境之圖片
增加不同類型寶特瓶、寶特瓶角度、寶特瓶模糊下的情形、更換環境(室內/室外/白天/夜晚)、增加不含寶特瓶之圖片等。 不同類別　　無寶特瓶　　模糊情形 
方向二:本研究調整之模型超參數
(1) anchor box(以下簡稱 ABox)之數量: 5 組、9 組 (2) 預設 anchor box 之尺寸: 預設之 9 組 anchor box 長寬，由資料集之標記進行 kmeans 分群後的長寬 (3) 資料擴增方法: 是否使用 Mosaic 與 CutMix (共 4 種組合)

YOLO 模型中參數調整之部分程式碼:

temp
recycle.data
yolov4-recycle...

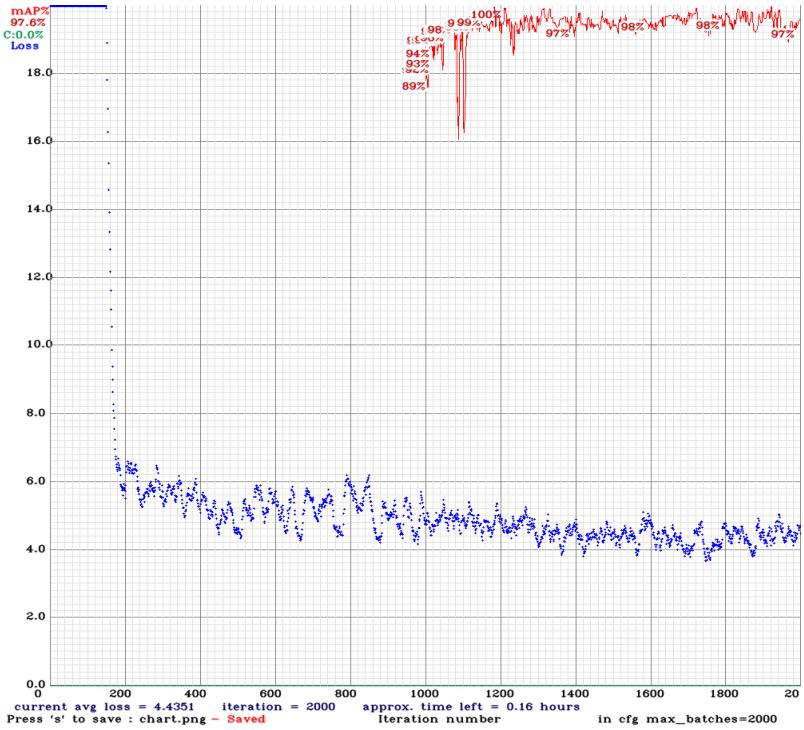
temp × | recycle.data × | yolov4-recycle.cfg ×

```
[yolo]
mask = 0,1,2
anchors = 12, 16, 19, 36, 40, 28, 36, 75, 76, 55, 72, 146, 142, 110, 192, 243, 459, 401
classes=1
num=9
jitter=.3
ignore_thresh = .7
truth_thresh = 1
scale_x_y = 1.2
iou_thresh=0.213
cls_normalizer=1.0
iou_normalizer=0.07
iou_loss=ciou
nms_kind=greedynms
beta_nms=0.6
max_delta=5


[route]
layers = -4

[convolutional]
batch_normalize=1
size=3
stride=2
pad=1
```

純文字 ▾　Tab 寬度：8 ▾　第 1 行第 1 字 ▾　插入

Model A-使用原始資料集與預設超參數:

由下圖可知模型訓練狀況良好

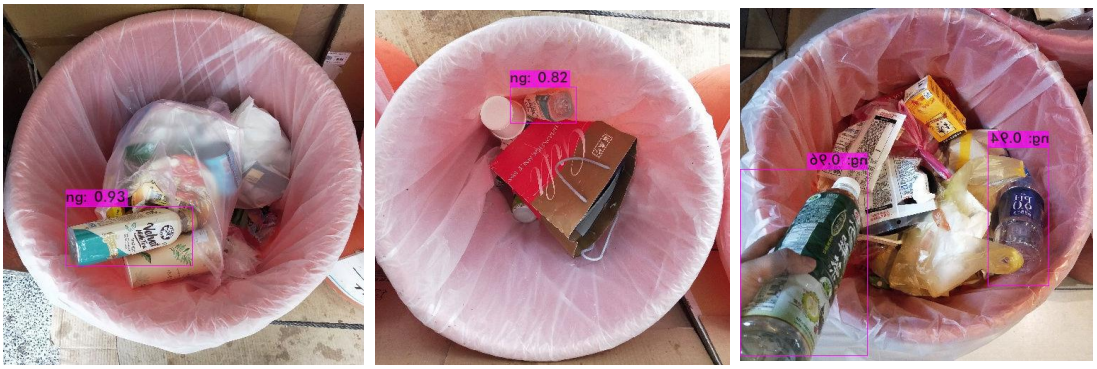


ModelA 訓練過程損失函數變化圖

預測原始資料集之測試資料的 MAP 達 95.49%，且無論寶特瓶是否完整出現階能被偵測，表現良好。

```
 calculation mAP (mean average precision)...
 Detection layer: 139 - type = 28
 Detection layer: 150 - type = 28
 Detection layer: 161 - type = 28
120
 detections_count = 231, unique_truth_count = 142
class_id = 0, name = ng, ap = 95.49%      (TP = 133, FP = 11)

 for conf_thresh = 0.25, precision = 0.92, recall = 0.94, F1-score = 0.93
 for conf_thresh = 0.25, TP = 133, FP = 11, FN = 9, average IoU = 73.52 %

 IoU threshold = 50 %, used Area-Under-Curve for each unique Recall
 mean average precision (mAP@0.50) = 0.954889, or 95.49 %
Total Detection Time: 3 Seconds
```

YOLO 輸出 ModelA 預測測試集績效之畫面



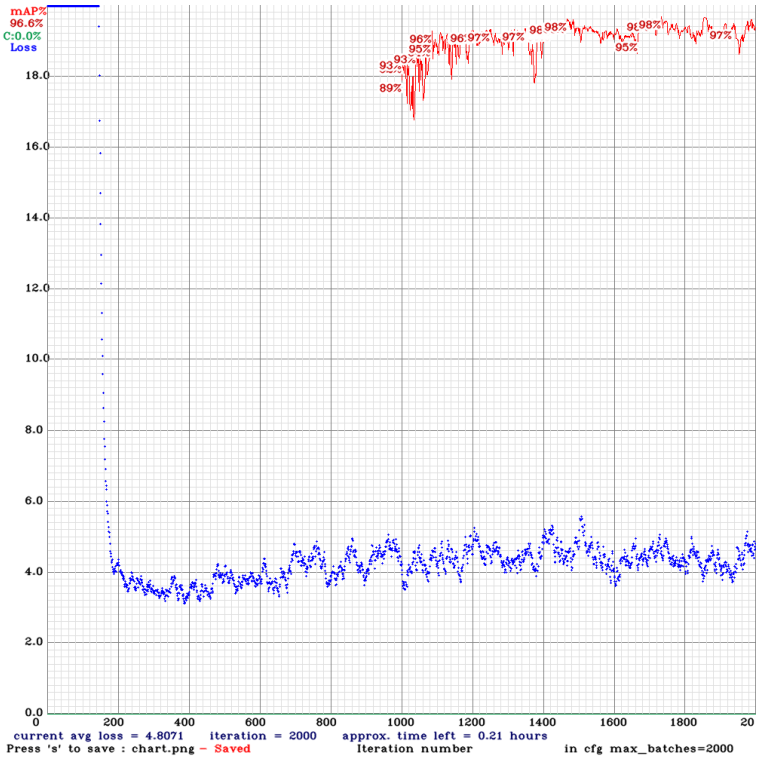
測試集預測狀況

然而當加入原始資料之 4 種寶特瓶外種類的寶特瓶後，發現模型泛化能力不佳，決定擴增資料集。增加不同類型寶特瓶、寶特瓶角度、更換環境(室內/室外/白天/夜晚)、增加不含寶特瓶之圖片，成為增圖資料集。

	原始資料集	增圖資料集
寶特瓶種類	4 種	6 種
無寶特瓶之圖片	無	30 張
寶特瓶移動模糊之圖片	無	30 張
擴增後總數	774 張	1242 張

原始資料集與增圖資料集比較表

ModelB -使用增圖資料集與調整 anchor 數量:



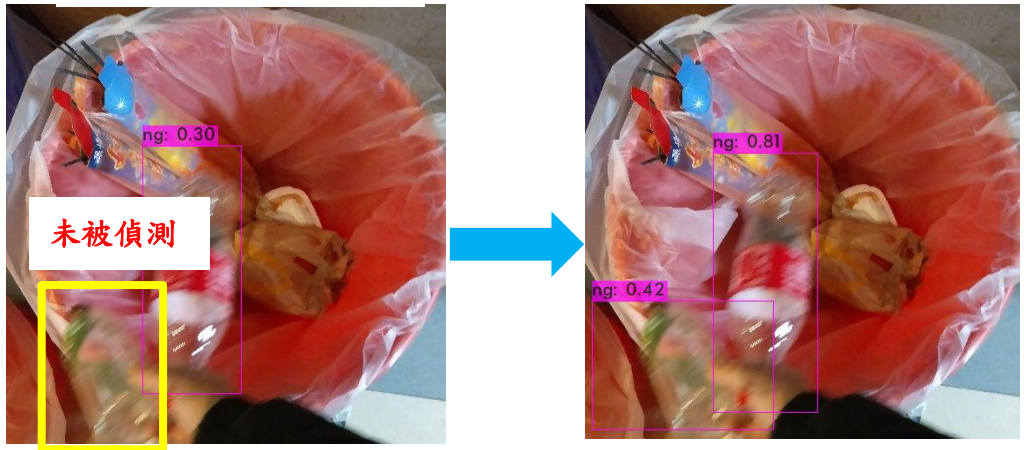
ModelB 訓練過程損失函數變化圖

```
 detections_count = 522, unique_truth_count = 244
class_id = 0, name = ng, ap = 96.03%      (TP = 235, FP = 17)

 for conf_thresh = 0.25, precision = 0.93, recall = 0.96, F1-score = 0.95
 for conf_thresh = 0.25, TP = 235, FP = 17, FN = 9, average IoU = 73.84 %

 IoU threshold = 50 %, used Area-Under-Curve for each unique Recall
 mean average precision (mAP@0.50) = 0.960318, or 96.03 %
Total Detection Time: 4 Seconds
```

YOLO 輸出 ModelB 預測測試集績效之畫面

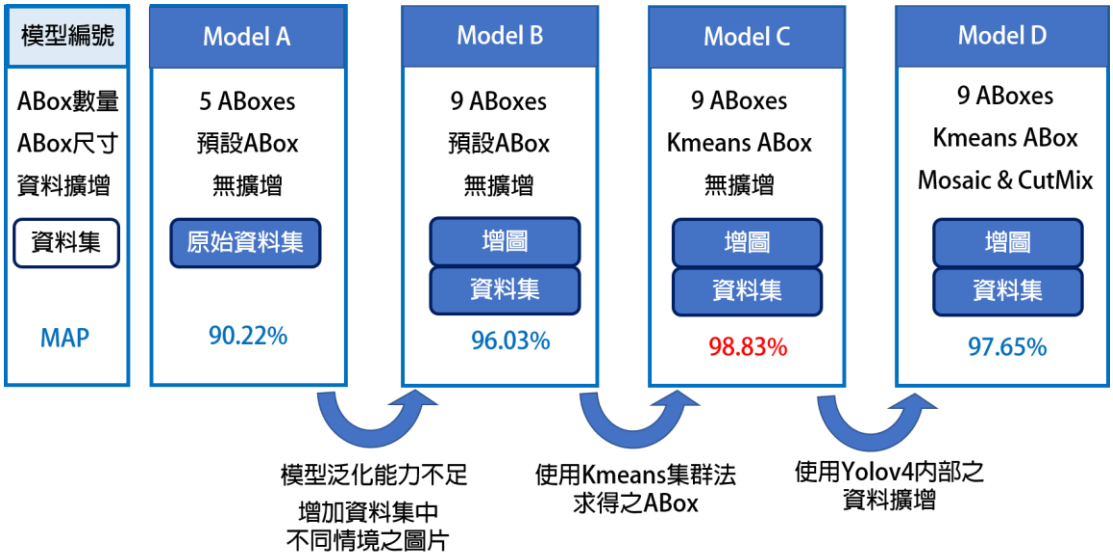
經過擴增資料集，預測表現顯著提升。若將增圖資料集之測試集放入 ModelA 預測僅能達到 MAP 90.22%，經過增圖資料集訓練之 ModelB 能達到 MAP 96.03%，績效提升 6.4%。

ModelA 預測結果　　ModelB 預測結果



ModelC&ModelD 模型超參數調整:

接下來 ModelC 與 ModelD 利用增圖資料集進行超參數調整後較佳之模型，研究發現超參數的調整僅帶來績效小幅度的提升。**最佳模型為使用 Kmeans 方法分類資料集標記檔之結果與採用 Mosaic&CutMix 資料擴增方法之模型。**上述 4 個模型其績效比較如下圖。



三、結論:

本研究提出創新的資源回收監督方法，目前於前段物件偵測模型訓練階段。透過自製垃圾桶資料集之原始資料集訓練物件偵測模型已達 MAP 90%以上的偵測績效，再透過增加資料集之圖片數量可以顯著提升 6.4%之模型績效。相較之下，模型之超參數小幅度提升模型績效，最終達到最佳模型績效 98.83%。本研究目前限制為只能針對寶特瓶偵測，影片測試時發現當桶子內垃圾眾多容易有誤判其他垃圾為寶特瓶之失誤發生，加上實際投擲垃圾速度甚快，仍須克服物體移動時畫面模糊的問題。

四、未來展望:

1.增加其他類別回收物資料，擴大訓練集數量
2.考量物體快速移動拍攝資料集克服上述影片偵測時之限制
3.架設其他角度之攝影機，並訓練辨識模型，交叉比對不同角度攝影之結果，降低誤判率

參考文獻:

(1) Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You only look once: Unified, real-time object detection. In Proceedings of the IEEE conference on

computer vision and pattern recognition (pp. 779-788).

(2) Alexey Bochkovskiy, Chien-Yao Wang, Hong-Yuan Mark Liao(2020). YOLOv4: Optimal Speed and Accuracy of Object Detection.