IIE Final Project 硬幣分類識別

108034548 卓妤庭

摘要

日常生活中，不論是店家清點錢幣、個人整理錢幣，時常要逐一辨識，很難快速區分分別種類，尤其遇到不熟悉的外幣時，更是不易分辨，因此本專題利用 YOLO 物件偵測方法，建立一個類神經網路偵測系統，幫助標記出大批硬幣中分別的種類，改善人為判斷時，時間的浪費及準確率。

關鍵字：類神經網路、YOLO、硬幣分類

一、緒論

1. 問題定義 5W1H

   Why：不論是店家清點錢幣、個人整理錢幣，時常要逐一辨識，很難快速區分分別種類，尤其遇到不熟悉的外幣時，更是不易分辨

   What：解決清點大量不同種類硬幣難分類的困擾

   Where：任何需要做硬幣分類的場所，例如：家中、店面、銀行等等

   When：需要做硬幣分類的時刻

   Who：舉凡有在使用實體貨幣，需要整理硬幣的人們

   How：利用類神經網路訓練並建立一個可以識別硬幣種類的偵測系統

2. 研究動機與目的

   藉由問題定義中 5W1H 的分析發現到在生活中，不論是店家清點錢幣、個人整理錢幣，時常要逐一辨識，很難快速分類，尤其遇到不熟悉的外幣時，分類硬幣更顯得不易，因此此為本次專題的主要動機，並期望可以建立一個識別硬幣種類的偵測系統，有助於解決清數硬幣的困擾。

二、文獻探討

1. YOLO 介紹

   (1) YOLO 介紹

   Yolo 系列(You only look once, Yolo)是關於物件偵測的類神經網路演算法，以 darknet 實作，也是 one stage 的物件偵測方法，就是只需要對圖

片作一次 CNN（卷積神經網路）架構便能夠判斷圖形內的物體位置與類別，因此提升辨識速度。實作這個架構的作者 Joseph Redmon 沒有用到任何著名深度學習框架，以輕量、依賴少、演算法高效率實行，在工業應用領域十分具有價值，例如行人偵測、工業影像偵測等等。

(2) 方法步驟

YOLO 執行的步驟大概可分為三個部分，如圖 2-1，分別為

I. 調整輸入的圖片大小 416*416

II. 執行一個卷積神經網路

III. 基於模型輸出的信心程度(Confidence)依據閾值和 Non-max suppression(NMS)得到偵測結果

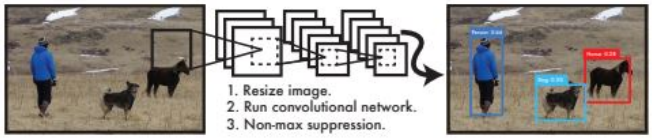

圖 2-1 YOLO 系統概念圖

(3) 偵測方法

YOLO 在物件偵測部分是將一張圖拆成很多個 grid cell，然後在每個 grid cell 進行 2 個 bounding box 的預測和該 grid cell 屬於哪個類別的機率預測，最後用閾值和 NMS 的方式得到結果。

先將一張圖片平均分成 S×S 格 grid cell，總共要偵測 C 個物件，每個 grid cell 必須要負責預測「B 個 bounding boxes」和「屬於每個類別的機率」，而每個 bounding box 會帶有 5 個預設值(x, y, w, h, and confidence)，其中(x, y)表示某一個物件在這個 grid cell 的中心座標，w, h 為此物件相對應的寬高，而 confidence 則是用來表示這個物件是否為一個物件的信心程度(confidence score)，confidence score 之計算方式為 $Pr(Object) \times IOU_{pred}^{truth}$，$IOU_{pred}^{truth}$計算方式為「兩個 bounding boxes 的重疊面積」除以「兩個 bounding boxes 的聯集面積」。因此 YOLO 最後輸出的 tensor 的大小是$S \times S \times (B \times 5 + C)$。

找到和分類物件的方式是由所有 bounding box 的信心程度經過閾值，先刪除確定不是物件的 bounding box，再用 NMS 把一些重疊的 bounding box 消除，重複執行後，剩下的 bounding box 就是選出來的物件，如下圖 2-2 之中上圖。而同時也輸出(S×S×C)個值，代表每個 grid cell 內每一類別的機率，這時取每個 grid cell 中機率最大的數值當作此 grid cell 的類別，如下圖 2-2 之中下圖。最後，結合選出的物件及對應 grid cell 的類別，即可判斷出物件所屬之類別。

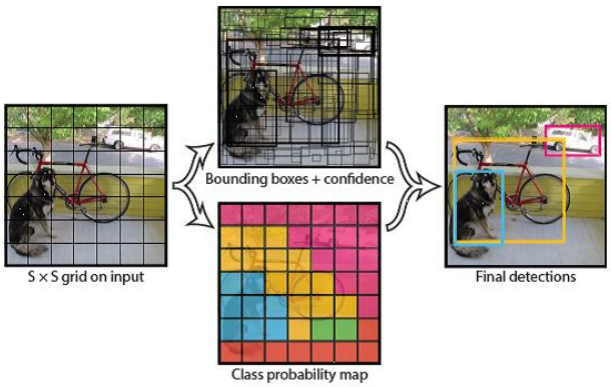

圖 2-2 YOLO 模型概念圖

三、研究方法

1. 資料集來源

資料集為自行拍攝的新台幣硬幣群圖片，包含 1 元、5 元、10 元、50 元之正反面，共 800 張照片作為訓練集；20 張照片作為驗證集；32 張照片、5 段影片做為測試集，如下表 3-1。

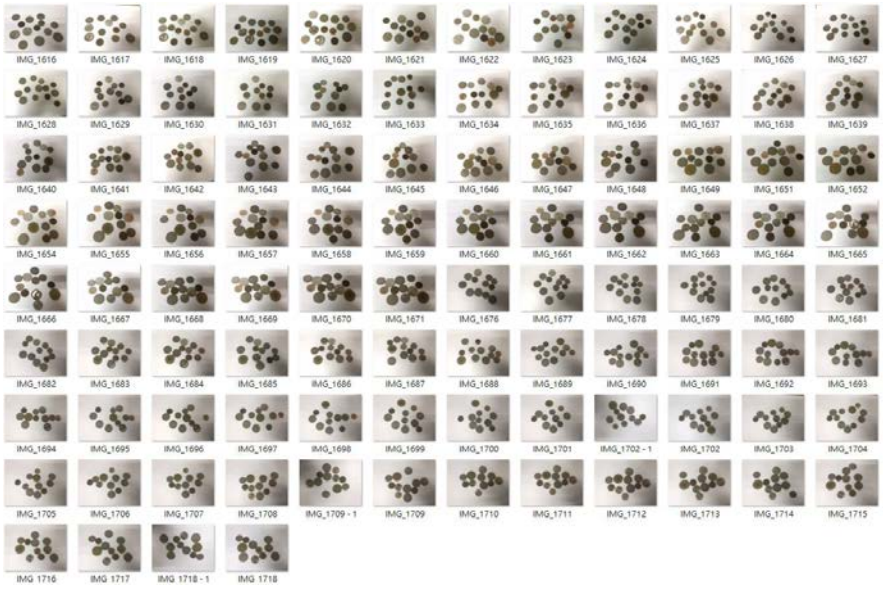

圖 3-1 訓練集照片

2. 訓練集標記

利用 labelImg 套件進行手動標記，將訓練集、驗證集的 100 張照片中的硬幣給予對應的標記項目，標記項目對照如下表 3-1，標記過程如下圖 3-2。這些標記項目也作為後續識別的類別。

表 3-1 標記項目對照表

項目	標記名稱	項目	標記名稱
1 元反面	NT_1_T	10 元反面	NT_10_T
1 元正面	NT_1_H	10 元反面(新款)	NT_10_T_2
5 元反面	NT_5_T	10 元正面	NT_10_H
5 元正面	NT_5_H	50 元反面	NT_50_T
		50 元正面	NT_50_H

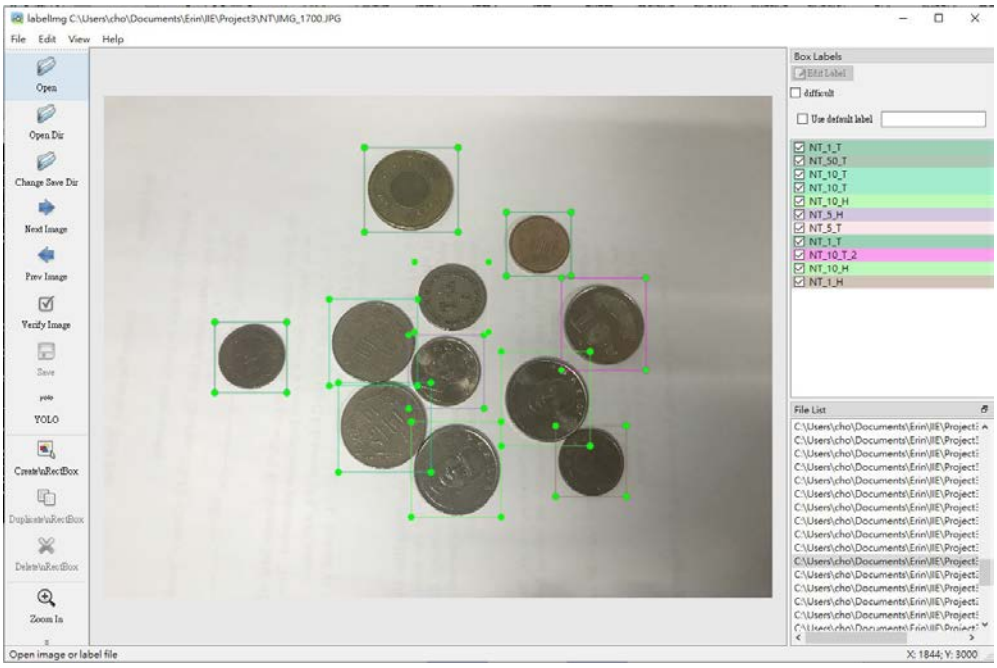

圖 3-2 labelImg 標記過程

3. 訓練- YOLO 之卷積神經網路

採用 YOLOv3 版本進行訓練。

在訓練圖片前，先將輸入圖的 bounding box 長寬進行正規化，使得 bounding box 的長寬會介於 0~1 之間。而 bounding box 的中心座標(x,y)也會介於 0~1 之間。YOLO 的卷積網路架構使用 darknet-53 結構，有 75 層卷積層且無全連結層，使得輸入的圖片不受限於固定的尺寸，任意輸入維度都可以在整個網絡上運行，整體架構如下圖。訓練過程中，每層之積活函數選擇 leaky rectified linear activation，損失採用 binary cross-entropy loss，如圖 3-3。

	Type	Filters	Size	Output
	Convolutional	32	3 × 3	256 × 256
	Convolutional	64	3 × 3 / 2	128 × 128
1×	Convolutional	32	1 × 1	
	Convolutional	64	3 × 3	
	Residual			128 × 128
	Convolutional	128	3 × 3 / 2	64 × 64
2×	Convolutional	64	1 × 1	
	Convolutional	128	3 × 3	
	Residual			64 × 64
	Convolutional	256	3 × 3 / 2	32 × 32
8×	Convolutional	128	1 × 1	
	Convolutional	256	3 × 3	
	Residual			32 × 32
	Convolutional	512	3 × 3 / 2	16 × 16
8×	Convolutional	256	1 × 1	
	Convolutional	512	3 × 3	
	Residual			16 × 16
	Convolutional	1024	3 × 3 / 2	8 × 8
4×	Convolutional	512	1 × 1	
	Convolutional	1024	3 × 3	
	Residual			8 × 8
	Avgpool		Global	
	Connected		1000	
	Softmax			

圖 3-3 卷積神經網路架構圖

4. 執行程式碼

由於 YOLO 的程式碼過於繁雜是由多個執行檔串連而成，篇幅過大，這邊僅附上卷積神經網路程式碼的前、後段。如圖 3-4。

```
[net]
# Testing
 batch=1
 subdivisions=1
# Training
batch=64
subdivisions=16
width=608
height=608
channels=3
momentum=0.9
decay=0.0005
angle=0
saturation = 1.5
exposure = 1.5
hue=.1

learning_rate=0.001
burn_in=5000
max_batches = 500200
policy=steps
steps=400000,450000
scales=.1,.1

[convolutional]
batch_normalize=1
filters=32
size=3
stride=1
pad=1
activation=leaky

# Downsample

[convolutional]
batch_normalize=1
filters=64
size=3
stride=2
pad=1
activation=leaky

[convolutional]
batch_normalize=1
filters=32
size=1
stride=1
pad=1
activation=leaky

[convolutional]
batch_normalize=1
filters=64
size=3
stride=1
pad=1
activation=leaky

[shortcut]
from=-3
activation=linear

# Downsample

[convolutional]
batch_normalize=1
filters=128
size=3
stride=2
pad=1
activation=leaky

[convolutional]
batch_normalize=1
filters=64
size=1
stride=1
pad=1
activation=leaky
```

```
748  [convolutional]
749  batch_normalize=1
750  size=3
751  stride=1
752  pad=1
753  filters=256
754  activation=leaky
755
756  [convolutional]
757  batch_normalize=1
758  filters=128
759  size=1
760  stride=1
761  pad=1
762  activation=leaky
763
764  [convolutional]
765  batch_normalize=1
766  size=3
767  stride=1
768  pad=1
769  filters=256
770  activation=leaky
771
772  [convolutional]
773  size=1
774  stride=1
775  pad=1
776  filters=1818
777  activation=linear
778
779
780  [yolo]
781  mask = 0,1,2
782  anchors = 10,13,  16,30,  33,23,  30,61,  62,45,  59,119,  116,90,  156,198,  373,326
783  classes=601
784  num=9
785  jitter=.3
786  ignore_thresh = .7
787  truth_thresh = 1
788  random=1
```

圖 3-4 YOLO 卷積神經網路程式碼

四、研究結果

1. 訓練、驗證結果

經由模型訓練得到結果如下，執行 interation 3300 次，績效指標 mAP 為 88.6%，損失 loss 為 0.057。

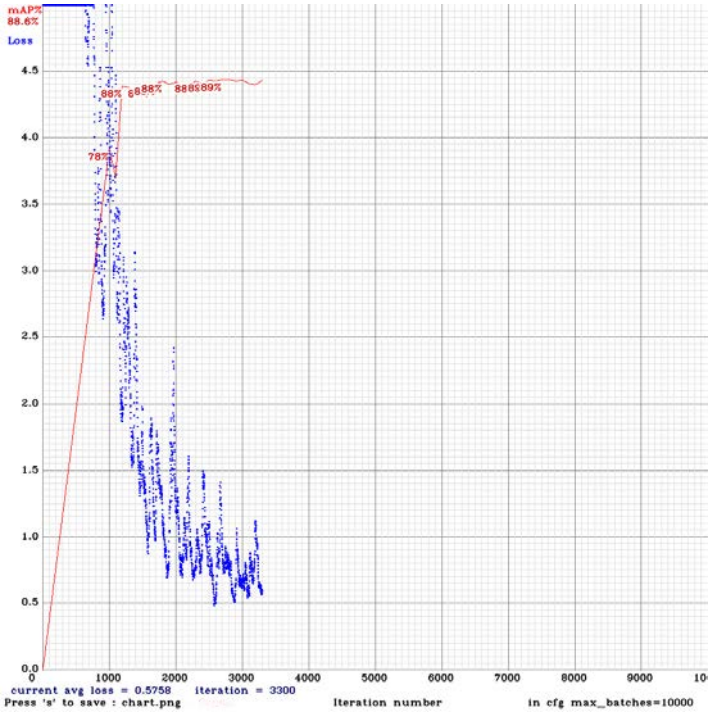

圖 4-1 訓練之績效及損失成果

2. 測試結果

由上述訓練結果，將測試集放入系統中，得到各類別之準確率皆大於80%，總體績效指標 mAP 為 77.71%，precision = 0.76，recall = 0.82，如圖4-2，測試結果照片如圖 4-3、圖 4-4，影片成果請參考簡報檔。

```
 calculation mAP (mean average precision)...
36
 detections_count = 1486, unique_truth_count = 523
class_id = 0, name = NT_10_T_2, ap = 97.67%       (TP = 53, FP = 21)
class_id = 1, name = NT_10_T, ap = 89.06%         (TP = 47, FP = 30)
class_id = 2, name = NT_10_H, ap = 80.03%         (TP = 54, FP = 24)
class_id = 3, name = NT_5_T, ap = 91.90%          (TP = 80, FP = 10)
class_id = 4, name = NT_5_H, ap = 56.61%          (TP = 8, FP = 1)
class_id = 5, name = NT_1_T, ap = 82.36%          (TP = 68, FP = 14)
class_id = 6, name = NT_1_H, ap = 91.52%          (TP = 65, FP = 36)
class_id = 7, name = NT_50_T, ap = 94.29%         (TP = 33, FP = 0)
class_id = 8, name = NT_50_T_2, ap = 0.00%        (TP = 0, FP = 0)
class_id = 9, name = NT_50_H, ap = 93.68%         (TP = 19, FP = 0)

 for conf_thresh = 0.25, precision = 0.76, recall = 0.82, F1-score = 0.79
 for conf_thresh = 0.25, TP = 427, FP = 136, FN = 96, average IoU = 66.47 %

 IoU threshold = 50 %, used Area-Under-Curve for each unique Recall
 mean average precision (mAP@0.50) = 0.777105, or 77.71 %
Total Detection Time: 2.000000 Seconds
```

圖 4-2 測試結果績效

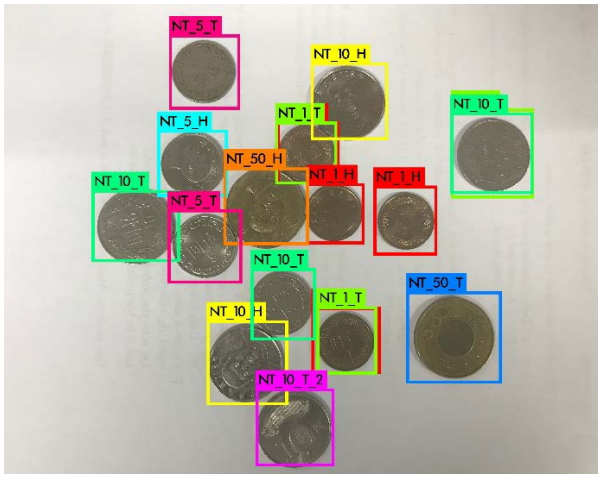

圖 4-3 測試結果照片

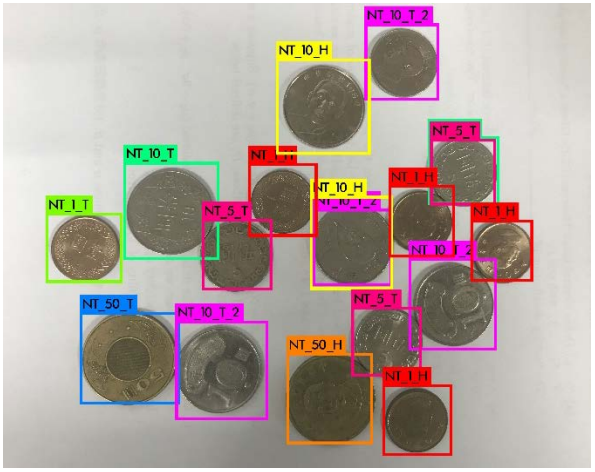

圖 4-4 測試結果照片

五、結論

1. 本次研究限制

由訓練和測試的結果發現，由於為自行拍攝的照片，在資料集的數量上仍稍嫌不足，因此在訓練的類別辨識上受限於樣本過少無法達到百分百精準。而在辨識硬幣方面的限制為，錢幣的本身變異過大，包括髒污程度影響硬幣圖像的浮現程度，進而使得模型不管在訓練或測試上皆會受到汙漬顏色影響判段結果。

2. 未來方向

本次專題僅針對台幣硬幣作訓練及系統建置，因此若此次專題內容實施效果不差，日後可以針對各國錢幣進行標記後訓練，使得此系統更加完善實用。另外在樣本數上可以再進行擴大，增加準確率。而污漬顏色影響的問題，日後可以在增強圖像前處理增加對比度，使得硬幣之字樣刻痕更加顯現，幫助判斷。

六、參考文獻

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).