IIE final project 辨識錢幣紙鈔

109034544 姜柏宏

摘要

錢是人生活中不可缺少的東西，人與人之間常常會有錢的來往，不管是買東西、交易、借還等行為都會有錢的來往，而這之中常常會出錯也很花時間，因此本專題提出了一個方法，利用 CNN 卷積神經網路訓練一個模型 YOLO，針對辨識台幣硬幣紙鈔，並在圖中框出該紙幣為何幣值，協助人類判斷錢幣總數，降低找錢錯誤機率以及減少判斷時間。

關鍵字:YOLOv4、CNN 卷積神經網路、錢幣判斷

一、 介紹

1. 問題探討 5W1H

   Why：錢幣是每個人日常皆會用到的物品，而常常也有算錯錢、找錯錢的情況，希望透過辨識來避免錯誤。

   What：快速算出紙鈔及硬幣數量，降低數鈔時間和避免找錢錯誤。

   Where：買東西、給錢、清點營收等。

   When：需要做硬幣分類的時刻

   Who：店員、業主等所有會使用到錢的人。

   How：利用卷積神經網路的方法快速辨識螢幕內錢幣數量。

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，不論是店家還是個人清點錢幣、整理錢幣，需要逐一辨識，很難快速分類也很容易出錯，因此為本次專題的主要動機，是建立一個識別硬幣紙鈔種類的辨識系統，有助於解決交易中金錢來往錯誤的困擾，以及減省時間。

二、 文獻探討

1. YOLO 介紹

   (1) 簡介

   YOLO 在從影像輸入到輸出預測結果僅靠一個 CNN 來實現利用，CNN 來同時預測多個 bounding-box 並且針對每一個 box 來計算物體的機率，而

在訓練的時候也是直接拿整張圖丟到 NN 中來訓練，這樣 end-to-end 的算法可以避免傳統 object detection 的必須分開訓練的缺點，並且大幅加快運算速度。

(2) 演進

2020 年 4 月，YOLO v4 發佈。其在 MS COCO 數據集上的精度達到了 43.5% AP，速度達到 65FPS，與 YOLO v3 相比分別提高了 10% 和 12%。而相較於其他 CNN 網路模型，其賣點就是超高的 FPS，但些微犧牲準確度。主要貢獻如下：

- 建立了一個高效強大的目標檢測模型，並且使用 1080Ti 或 2080Ti 的 GPU 就可以進行訓練。
- 驗證了 SOTA (State of the Art) 的 Bag-of-Freebies 和 Bag-of-Specials 目標檢測方法在檢測器訓練過程中的影響。
- 改進了一些 tricks 、SOTA 的方法，包括 CBN、PAN、SAM 等，使之更加高效，並能夠在單 GPU 上訓練。

(3)架構-骨幹

Yolov4 在主幹網絡 Backbone 採用 CSPDarknet53 網絡結構，主要有三個方面的優點，如圖 2-1：

優點一：增強 CNN 的學習能力，使得在輕量化的同時保持準確性。

優點二：降低計算瓶頸

優點三：降低內存成本

Yolov4 的 Backbone 中都使用了 Mish 激活函數，而後面的網絡則還是使用 leaky_relu 函數，如圖 2-2。

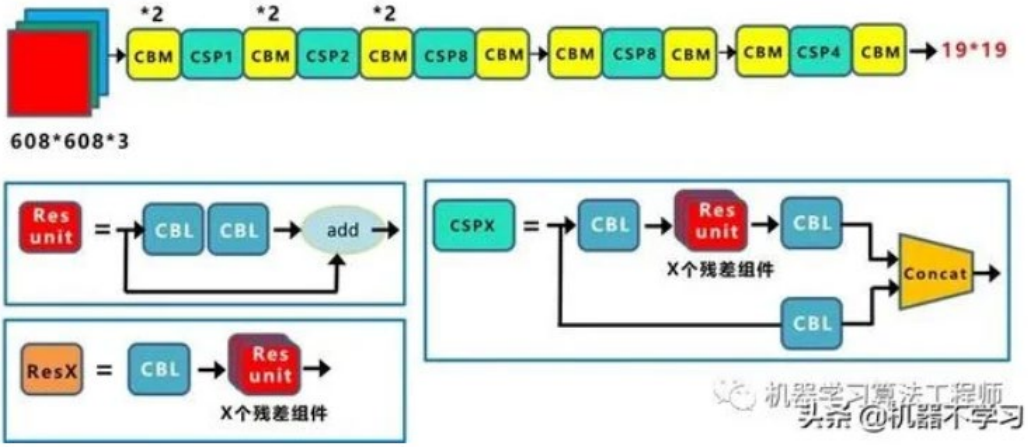

圖 2-1

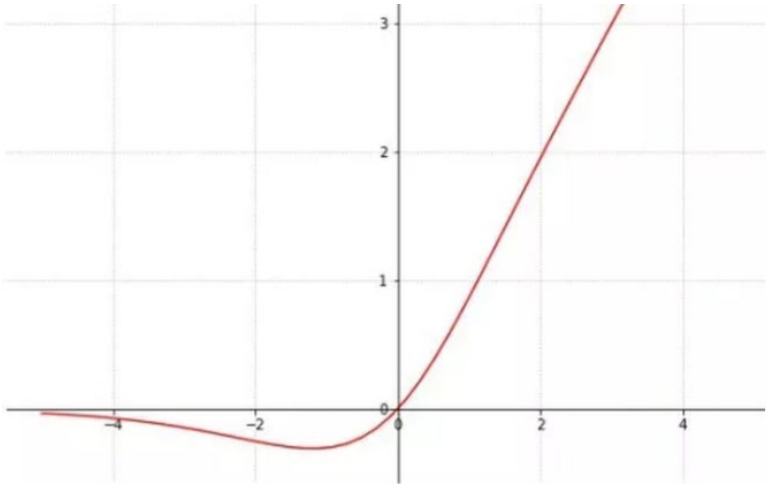

圖 2-2

(4)架構-droplock

Yolov4 中使用的 Dropblock，其實和常見網絡中的 Dropout 功能類似，也是緩解過擬合的一種正則化方式。

因為卷積層通常是三層連用：卷積+激活+池化層，池化層本身就是對相鄰單元起作用。而且即使隨機丟棄，卷積層仍然可以從相鄰的激活單元學習到相同的信息，因此，在全連接層上效果很好的 Dropout 在卷積層上效果並不好。

所以從右圖 Dropblock 的研究者則乾脆整個局部區域進行刪減丟棄，如圖 2-3 表示。

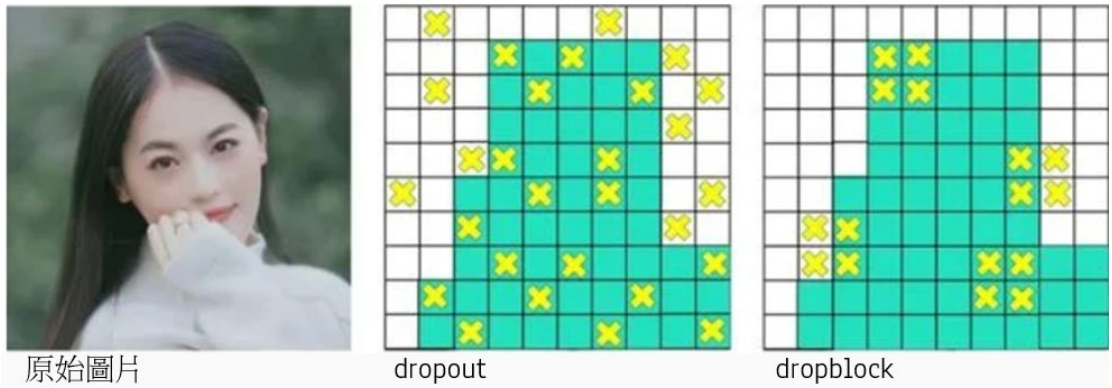

圖 2-3

三、 研究方法

1. 資料集來源

資料集為自行拍攝錢幣紙鈔共 189 張，內含多張紙鈔以及多個硬幣，包含 1 元、5 元、10 元、50 元、100 元、200 元、500 元、1000 元，並將所有資料集分為訓練集 151 張、驗證集 38 張，如下圖 3-1。

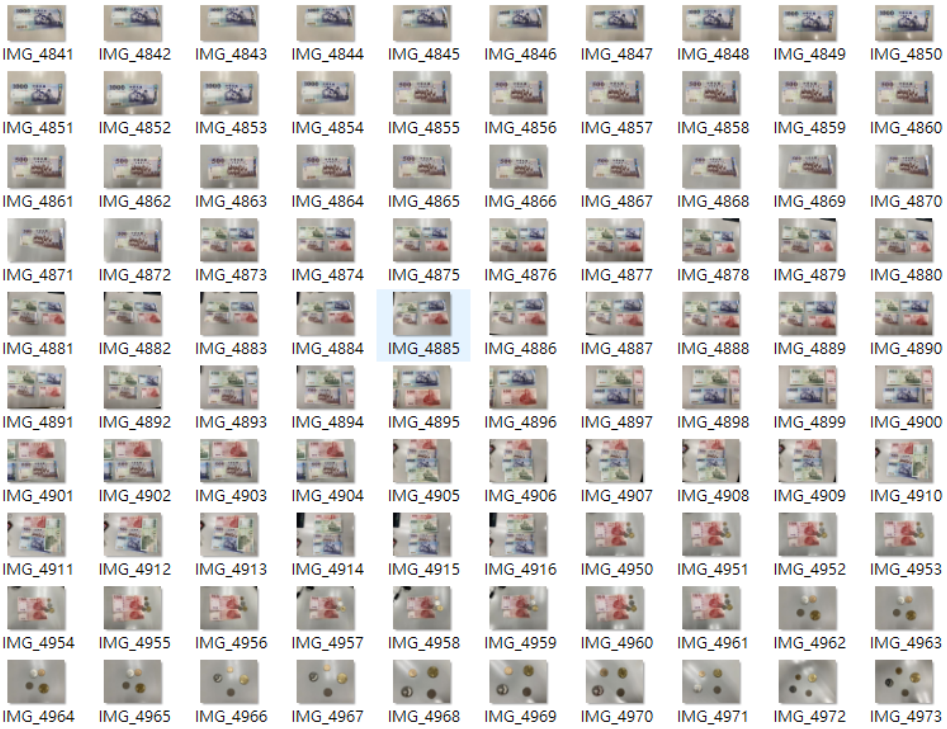

圖 3-1

2. 資料標記

利用 labelIMG 軟體進行手動影像標記，分邊將 151 張訓練集和 38 張驗證集給出相對應的標註檔，先將圖片檔案轉成較低畫質方便後續讀取如下圖 3-2，再用軟體手動標記如下圖 3-3，標記出檔案 txt 檔案，其內容包含該圖片標籤類別，以及在圖片中相對位置座標，如下圖 3-4。

C: > Users > user > Desktop > python > transport.py > ...

```python
import glob
import os
from PIL import Image

img_path = glob.glob("C:/darknet-master/build/darknet/x64/data/obj/*.jpg")
path_save = "C:/darknet-master/build/darknet/x64/data/obj/*.jpg"
for file in img_path:
    name = os.path.join(path_save, file)
    im = Image.open(file)
    im.thumbnail((640,640))
    print(im.format, im.size, im.mode)
    im.save(name,'JPEG')
```

圖 3-2

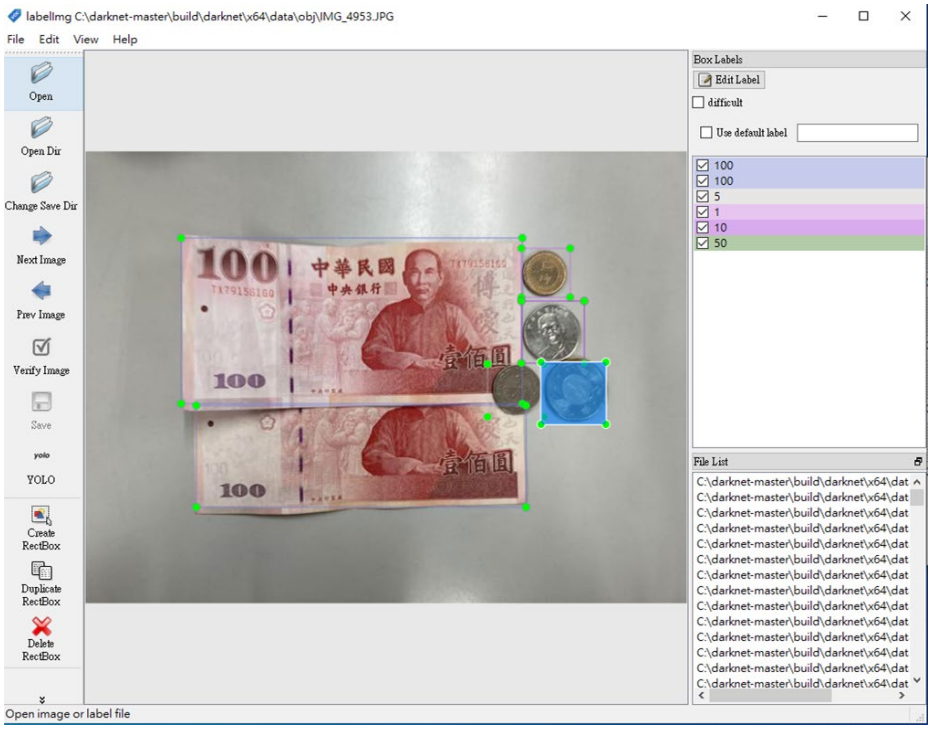

圖 3-3

圖 3-4

3. 訓練模型

將準備好資料送入模型訓練，並使用如圖 3-5 表示之超參數以及模型架構，即可開始訓練。

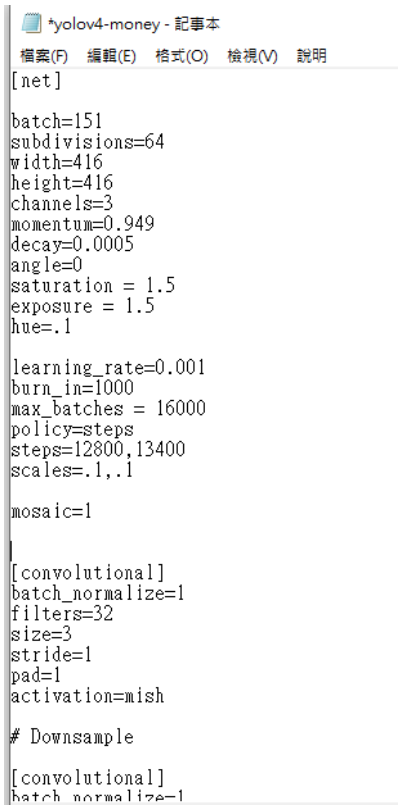

圖 3-5

4. 訓練過程

訓練過程如圖 3-6 會有一張顯示 loss 跟 map 的圖表每一個 iteration 就更新一次，也會顯示進行到了多少 iteration 及預估結束時間。

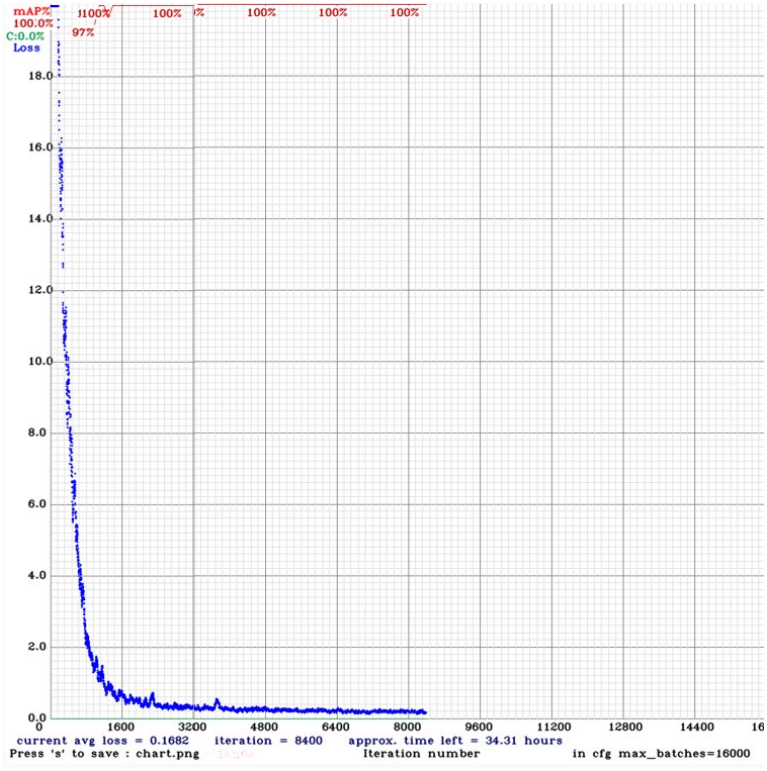

圖 3-6

四、 結果呈現

1. 訓練、評估結果

- 訓練 8400 個 iteration
- 平均 loss 為 0.1682
- 驗證集 mAP 為 100%
- 共耗時 40 小時左右訓練

2. 推論結果

五、 結論

1. 研究限制
   - 訓練數不足，本次訓練共用了 151 張訓練集、38 張驗證集，但較完善之模型訓練張數應為上千張。
   - 設備限制，因顯卡內存不夠大，訓練速度慢，故每一次嘗試模型需花上半天測試。
   - 樣本變異，因錢幣紙鈔新舊、髒污程度不同造成訓練誤差。

2. 未來展望
   - 提高訓練數，增加更多角度更多款式之訓練數，讓模型可適應各種錢幣。
   - 即時辨識，設計出軟體可串接手機或穿戴式裝置修改程式使其可串接手機或穿戴式裝置即時辨識，提供給店家，甚至是視力不佳需要輔助工具等人士。
   - 即時運算，透過辨識框加總所有辨識框之標籤，可即時算出該圖片內含有多少錢。

六、 參考資料

- YOLOv4 大神 AlexeyAB

https://github.com/AlexeyAB/darknet

- 深入淺出 Yolov3 和 Yolov4

https://kknews.cc/zh-tw/tech/3yzxlza.html

- YOLOv4 建置流程

https://wings890109.pixnet.net/blog/post/68926387-yolov4%e5%bb%ba%e7%bd%ae%e6%b5%81%e7%a8%8b