

2019 Footprintku 校園達人秀

〈以 I 為名，AI 動起來〉

初賽報告

中華民國 108 年 12 月 20 日

隊名：_____ 圖片辨辨辨

1. 為維護比賽公平性，請勿於報告中洩露比賽隊伍之學校與教授名稱。
2. 敬請依此格式撰寫報告，若有不適用之項目，該項目請留空白，切勿刪除。
3. 內文字體大小以 12 字體，行距以 1.5 行距為原則。
4. 報告得以中文或英文撰寫。
5. 報告內容篇幅以 15 頁為限(不含封面、目錄、圖目錄、表目錄、摘要及參考文獻，摘要不得超過一頁)。
6. 請以 PDF 為檔案格式提交。

目錄

中文摘要	5
壹、 緒論	6
1-1、 研究背景與動機	6
1-2、 研究目的與方法	6
1-3、 研究流程與架構	6
貳、 背景知識與相關文獻	8
參、 研究方法	9
肆、 研究結果與分析	13
伍、 結論與未來展望	14
陸、 參考文獻	15

圖目錄

圖 一、本研究架構圖	7
圖 二、內容分割結果	10
圖 三、切割圖去框前	10
圖 四、切割圖去框後	10
圖 五、圓形框範例	10
圖 六、去圓框結果	10
圖 七、文字內容範例	11
圖 八、水平方向加總	11
圖 九、第一行字垂直方向加總	11
圖 十、第二行字垂直方向加總	11
圖 十一、第三行字垂直方向加總	12
圖 十二、第四行字垂直方向加總	12
圖 十三、字元分割結果	12
圖 十四、辨識範例圖	12
圖 十五、未加邊範例	13
圖 十六、加邊結果	13

↵

表目錄^u

表 一、調整參數結果.....	11
表 二、加邊辨識結果.....	12
表 三、字元切割結果.....	13
表 四、各圖執行時間.....	13

中文摘要

具影像辨識功能的演算法是在國內外都被列為重要的智慧工業與生活應用的重要技術。本次研究主要分為二階段，第一階段先取得表格內容跟標籤位置，並將兩者對應，第二階段以文字辨識方法取得表格跟標籤內容。

針對第一階段，我們使用 OpenCV 對圖像做前處理，並運用 Numpy 處理矩陣與計算。圖片大致分為兩種，以格線分開，或以間隔分開。以格線所分開的圖片可直接依框線切割，另外的圖像有固定間隔為特徵。利用這些特徵，能將各個座標格分開，並利位置對應其座標。

第二階段則利用 pytesseract 來辨識每個座標格的內容，並將其輸出到 excel。

關鍵字：ORC, pytesseract, 座標辨識, 圖片前處理, OpenCV, Numpy, Pillow, pytesseract, Pytorch

壹、緒論

1-1、研究背景與動機

隨著 AI 在人們日常生活中逐漸佔有重要的角色，圖像辨識也逐漸廣泛應用在生活中。例如手機的相機功能讓人們隨身攜帶紀錄影像，此時人們若想將所見的知識留存，圖片轉換為文字便能方便編輯與註記。因此若有優秀的圖片轉文字軟體，藉由 AI 處理可減少人們花在打字的時間，也方便資訊的流通與紀錄。

1-2、研究目的與方法

本次研究藉由圖像處理與字元解析，來實現多張圖像內的文字辨識。先辨識各種不同顏色、樣式的圖片，再取得圖像表格中的內容與其對應的座標，最後將其結果寫入 Excel 檔，並以提高辨識準確率為主要目標。團隊以 Python 作為工具，使用 openCV、Numpy、Pytorch、Pytesseract...等套件找出圖片特徵，對圖片做前處理、訓練模型，調整參數並反覆實驗，使辨識準確率提高。

1-3、研究流程與架構

在研究流程上，團隊先參考多種方法後，決定將工作分為兩部分同時進行，一部分處理圖片，二值化、切割、去噪...等前處理以取得表格內容，另一部分尋找適合辨識文字的套件與模型，最後將兩者整合並輸出。

本研究架構如圖一所示，在物件偵測的研究中，團隊找到主流的方法像是 YOLO、connected component、cv.findContours；原本團隊用於切割字串的方法，團隊也發現可以用在取得表格位置上。

到研究的後半段，團隊決定選用 Pytesseract 作為辨識方法。為了提升準確率，除了調整 Pytesseract 的參數，還對圖片前處理部分進行改善，例如：調整文字的背景、將圖片切塊後重組、將圖片放大，最後將兩者整合並輸出。

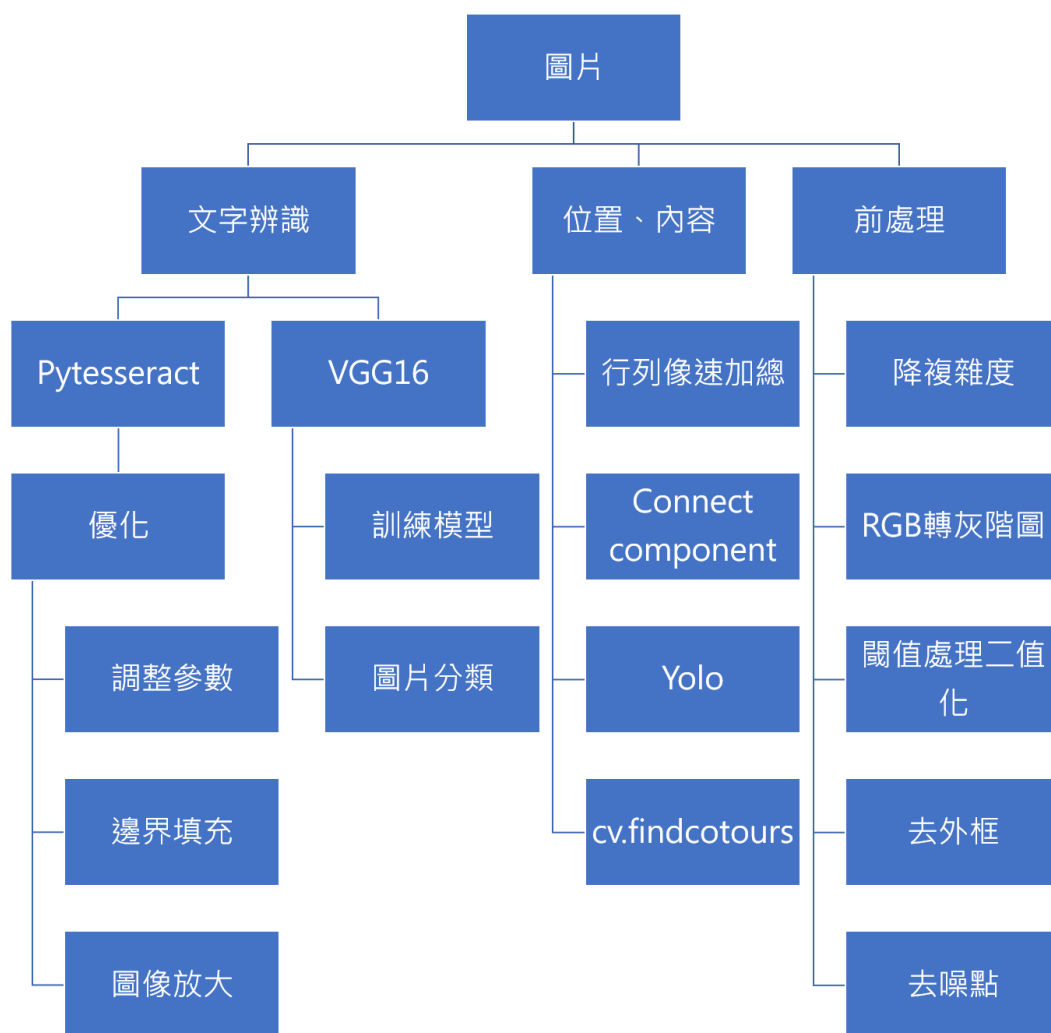


圖 一、本研究架構圖

貳、背景知識與相關文獻

影像處理，團隊使用 OpenCV 中的顏色空間轉換與自適應閾值函數；表格與內容的切割方面，則使用影像垂直與水平投影分析、連通域分析以及影像輪廓查詢等方法抓取物件。

- 顏色空間轉換(cvtColor)與自適應閾值(Adaptive thresholding)：

顏色空間轉換，可將 BGR 三通道的彩圖轉為灰度圖。比起傳統的方法 global threshold，adaptive thresholding 決定閾值的方式會考慮其附近的像素，並將大於閾值之像素令為 255，小於閾值之像素令為 0，也因此閾值的選取相當重要。

- 影像垂直與水平投影分析：

在文字辨識時為了分割字串或字元，會先將像素以行或列的方向加總，得其特徵；由於文字行間會有間隔，加總後間隔處為零，就可獲得字串所在位置，字元同理。

- 連通域分析(connected component)：

為圖論內經典方法之一，將連接起來的像素視為同一個物件，opencv 也有支援的函式。

- 影像輪廓查詢(cv.findContours)：

為 opencv 內的一個函數，可以用來偵測圖片內的物件輪廓，並將其存為一個物件。

在文字辨識方面，Pytesseract 套件是支援 python 的 tesseract，而 tesseract 是一套開源的 OCR 軟體，目前由 google 管理，這個模型已經經由多種字型的 dataset 訓練過，是相當成熟的模型，目前已更新到第四版，並被利用在多種應用上。VGG 為 Deep learning 中的一大經典模型，他主要的貢獻是將 CNN 透過較小的卷積堆疊使模型能夠變得更“深”，是目前 CNN 圖像分類的主流模型之一。

參、研究方法

3-0、 分析圖片特性

先檢視題目中的每張圖片，找尋各個圖片的特徵與可能影響獲得目標物件的因素。

3-1、 預處理

對於要偵測以及辨識的圖片，只留下其重要的特徵，可以減少資訊量。對於文字辨識來說，不需要顏色資訊，因此可以將彩圖轉為灰階圖降低複雜度。然而灰階圖可再藉由閾值處理將背景及物件分為 0 與 255，減少運算量。傳統的二值化方法為利用單一的閾值，會忽略局部的變化，例如陰影、顏色的影響，故團隊利用自適應閾值(adaptive threshold)方法對圖片二值化，更適合內容複雜度高的圖片。

3-2、 取得布局位置

在座標判斷以及切割部分，利用影像垂直與水平投影分析來獲取座標格與標籤的位置資訊，由於圖片經過像素行列方向加總，空白間隔處值為零，先將不為零即有物件的位置設為切割上界，再選定下個為零的位置設為切割下界，可以獲得內容分布的位置(如圖二)。這個方法的優點在於計算量小，極具效率。

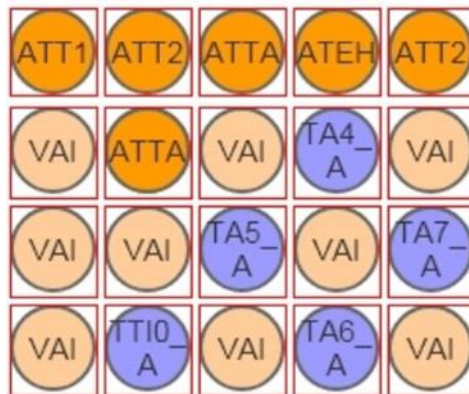


圖 二、內容分割結果

3-3、消除框線

在文字辨識中，多餘的框線會影響偵測跟辨識的結果，故選擇將框線去除。首先利用矩陣水平及垂直加總，框線在水平及垂直加總後，會產生極大值，即可得到位置資訊，並令其為零消除框線。(如圖三、四)

在競賽的圖片中，除了方框外，還有圓框的資料(如圖五)，框線一樣會影響內容的辨識。從前一步驟內容分布可得到小張圖片，再利用圓的特性，將圓周消除(如圖六)。

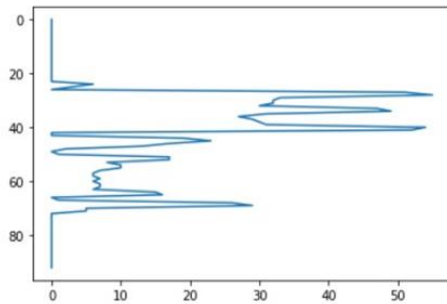


3-4、切字元

字句通常整齊排列且會有分行的特性，每行的字元又可以垂直的空白分割，利用此特性並可將字元分割。以圖七為例，透過水平方向的加總，以選定閾值可將圖分為四個區塊(圖八)，得到內容依序為第一行字、第一行底線、第二行字以及第二行底線；再將分割後的四張圖片個別垂直加總(如圖九~十二)，將字元與字元之以零為閾值分割，其結果如圖十三。

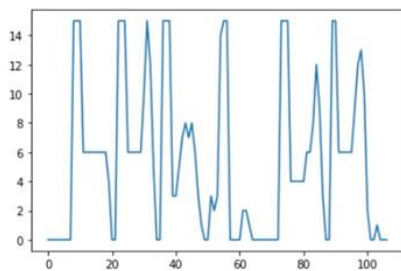
EBK1_DB
L_T_1

圖 七、文字內容範例



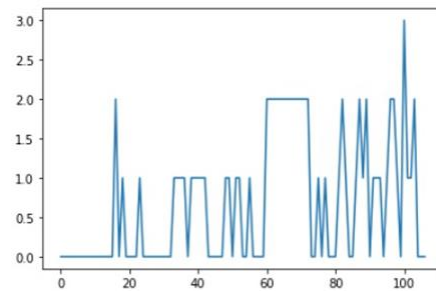
[(27, 42), (44, 48), (51, 66), (68, 70)]

圖 八、水平方向加總



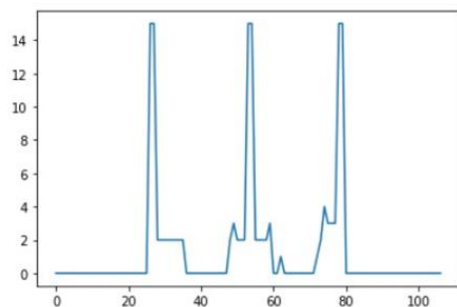
[(8, 20), (22, 34), (36, 49), (51, 57), (73, 87), (89, 101)]

圖 九、第一行字垂直方向加總



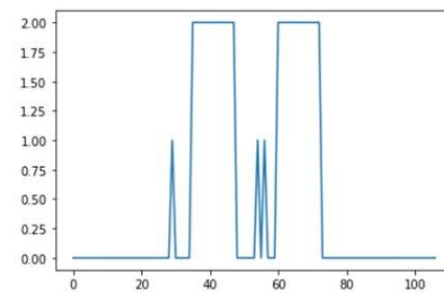
[(60, 73)]

圖 十、第二行字垂直方向加總



[(26, 36), (48, 60), (73, 80)]

圖 十一、第三行字垂直方向加總



[(35, 48), (60, 73)]

圖 十二、第四行字垂直方向加總

EBK1_DB EBK1_DB
L_T_1 L_T_1


圖 十三、字元分割結果

3-5、 辨識

- 調整參數

Pytesseract 有多種參數可以設定，根據不同情況使用，可以提高正確率。

表 一、調整參數比較

 圖 十四、辨識範例圖	未加參數(預設) (config='-psm 3 ')	輸出: 3_1_1_1_10
	config='-psm 7 digits'	輸出: BLLRXp1

- 邊界填充

部分原圖測試無法辨識出任何結果，在經過邊界填充後，Pytesseract 即可輸出(如表二)。合理推測與其模型訓練方式有關，一般來說訓練字元模型時，不會恰好切齊字元的邊界，故經過邊界填充可得較準確之結果。

表 二、邊界填充比較

原圖	 圖 十五、未加邊範例	輸出:
處理後	 圖 十六、加邊結果	輸出: JK

- 圖像縮放

團隊發現若將圖片放大後辨識效果較好，故將的圖片長寬都增加 20 倍，增加其辨識率。

肆、研究結果與分析

為得到模型的訓練與識別樣本，需針對不同的圖片做適合的前處理，例如彩圖變化多，需將圖片分割後再做灰階與二值化處理；框線對於座標對應內容的擷取相對容易，但在字元切割或是識別時就需再調整(如表三)。

表 三、字元切割結果

二值化圖片	去除圓外框	再分割	
		ATT2	
		TT10'	'_T'

圖片所得輸出結果是否正確需自行比對，針對圖像或是模型做改進後，較無法及時知道修正多少結果。團隊首先鎖定較多字元與較易辨識的兩種圖片，查看準確率尋找問題，再持續修正使準確率均達六成以上，數張達九成以上。

由於 Python 語言特性，發現打包成執行檔後，發現執行時間需拉長，考量主辦提供執行時間有限，為避免執行檔超過時間故作調整，所以內容較多的圖片無法完整輸出，較為可惜。表四為 10 張圖片實際測試所需執行時間。

表 四、各圖執行時間

	python 完整程式執行時間(秒)	exe 執行檔測試時間(秒)
FPK_1	40	73
FPK_2	534	172
FPK_3	28	99
FPK_4	183	177
FPK_5	47	94
FPK_6	142	161
FPK_7	90	114
FPK_8	36	53
FPK_9	27	63
FPK_10	81	95

伍、結論與未來展望

本研究主要利用像素加總切割方法、圖片前處理、與 Pytesseract 辨識解決本次影像辨識問題，其優點如下：

- 減少訓練時間

訓練模型是相當費時費力的，並且不同結構，不同 dataset，都會影響辨識出來的結果，團隊使用 Pytesseract 進行辨識，使得此實驗是每個人都可重複的，不會因為 model 的不同，而影響實驗結果。

- 有效率的位置取得

團隊運用影像垂直與水平投影分析來取得位置，善用表格對齊的特性，使得這方法相當適合此次的切割，加上其計算複雜度低、運算量很小，能在極短的時間得到結果。

相關缺點如下：

- Pytesseract 的限制

Pytesseract 是針對文件辨識用，對於特殊的需求，例如本次辨識的圖片中字元有 overline 的案例，需以其他方式判別。雖模型相對成熟穩定，但遇到辨識錯誤率高的情況，要修改 Pytesseract 的模型就相對困難。

- 前處理複雜

為了要讓 Pytesseract 有更好的結果，團隊嘗試多種圖像前處理，有些看似能夠改進的方法，對 Pytesseract 卻不一定有效。

考量訓練模型也需得到適合數據，這次團隊將研究分為兩部分同時進行，但圖片處理遇到較多情況，由於圖片樣式多樣，且適合人眼識別的數據不見得適合模型，花費較多時間前處理圖片。深度學習尚有多個經典卷積神經網路模型，這次只使用 VGG 作測試，時間允許則能夠嘗試更多模型訓練。

陸、 參考文獻

- [T. Lung-Yu, “影像垂直/水平投影分析,” [線上]. Available:
1 <http://honglung.pixnet.net/blog>.
]
- [M. B. Dillencourt, H. Samet 且 M. Tamminen., “A general approach to
2 connected-component labeling for arbitrary image representations,” *Journal of*
] *the ACM (J. ACM)*, pp. 253-280, April 1992.
- [“OpenCV 3.0.0-dev documentation,” [線上]. Available:
3 [https://docs.opencv.org/3.0-beta/modules/imgproc/doc/structural_analysis_and_](https://docs.opencv.org/3.0-beta/modules/imgproc/doc/structural_analysis_and_shape_descriptors.html#connectedcomponents)
] [shape_descriptors.html#connectedcomponents](https://docs.opencv.org/3.0-beta/modules/imgproc/doc/structural_analysis_and_shape_descriptors.html#connectedcomponents).
- [“OpenCV 2.4.13.7 documentation,” [線上]. Available:
4 [https://docs.opencv.org/2.4/modules/imgproc/doc/structural_analysis_and_shap](https://docs.opencv.org/2.4/modules/imgproc/doc/structural_analysis_and_shape_descriptors.html?highlight=findcontours#findcontours)
] [e_descriptors.html?highlight=findcontours#findcontours](https://docs.opencv.org/2.4/modules/imgproc/doc/structural_analysis_and_shape_descriptors.html?highlight=findcontours#findcontours).
- [K. Simonyan 且 A. Zisserman, “ Very Deep Convolutional Networks for
5 Large-Scale Image Recognition,” 於 *International Conference on Learning*
] *Representations* , 2015 .
- [“虫数据,” [線上]. Available: <http://chongdata.com/articles/?p=32>.
6
]
- [“OpenCV-Python Tutorials,” [線上]. Available:
7 https://docs.opencv.org/master/d7/d4d/tutorial_py_thresholding.html.
]