

Picollate 智能型視覺辨識相片分類 App—你是我的眼

專題組員：劉晏、黃亭雅、蔡羽涵

專題編號：PRJ-NTPUCSIE-104-006

執行期間：104 年 9 月 至 105 年 6 月

1. 摘要

影像分類常被應用在分析地理資訊，例如土地利用分類、河川流域影像分類、植被分類等。本專題利用此影像分類的技術研發出一款 App，協助使用者將相機拍攝的大量照片，自動分成五個類別，省去人眼分類整理所需花費的時間。將照片經過各種前處理，包含灰階、邊界偵測等，用來去除照片中的背景雜訊，再抽取出照片中的 feature(影像特徵)，將這些 feature 作為 Tree 的節點來建立眾多的 Tree，最後使用 Random Forest 分類演算法，將所有照片分類到正確類別。此外利用一鍵分享功能，隨機挑選類別與張數，方便使用者可直接上傳到社群網站與朋友們分享。

2. 簡介

本 App 可協助使用者將大量的照片分為五個生活常見的類別：人物、動物、食物、建築及其他類，可以在分類資料夾中分別查看每一類的照片，另外也考慮到為了方便使用者分享到社交平台，一鍵分享功能可勾選想要的類別與張數，由 App 隨機幫使用者挑選幾張照片分享到社交平台，省去挑選照片的麻煩。

3. 專題進行方式

3.1 App 功能使用說明

此 App 主要分為三大功能：

a.相機連線：強調使用者一拍完照即可上傳 server 端進行分類運算(圖 1~4)



圖 1：相機連線



圖 2：進入拍攝畫面



圖 3：拍攝完顯示照片的頁面，可以選擇將此照片刪除或上傳至 server 端

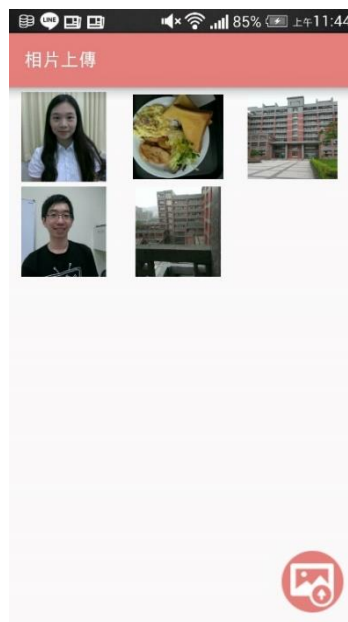


圖 4：相簿頁面，用來檢視所有拍攝過的照片並可於此挑選照片上傳至 server 端進行運算

b.分類相簿：server 端將運算結果回傳至行動裝置並顯示於此，將相片分類成五類別，幫助使用者

管理與查詢(圖 5~7)



圖 5：分類相簿



圖 6：系統將此照片分為人物類



圖 7：進入人物類相簿查看，並可於此頁面點選照片進行分享

c. 一鍵分享：省去使用者一張張選取照片分享的麻煩，系統自動幫使用者挑選照片分享(圖 8~10)



圖 8：一鍵分享



圖 9：點選想要分享的照片類別與數量，系統會從點選的類別隨機挑出照片分享



圖 10：選擇想要分享至的社交圈，成功分享

3.2 照片自動分類設計流程

專題程式設計流程如圖 11。

建立分類程式前的準備工作：選取適合的照片資料當作訓練依據。

訓練資料處理，第一步驟-先將照片分析所需的特徵做擷取，透過演算法-Random Forest 建立分類模型，依據新照片的特點，通過模型的分類，得到結果。

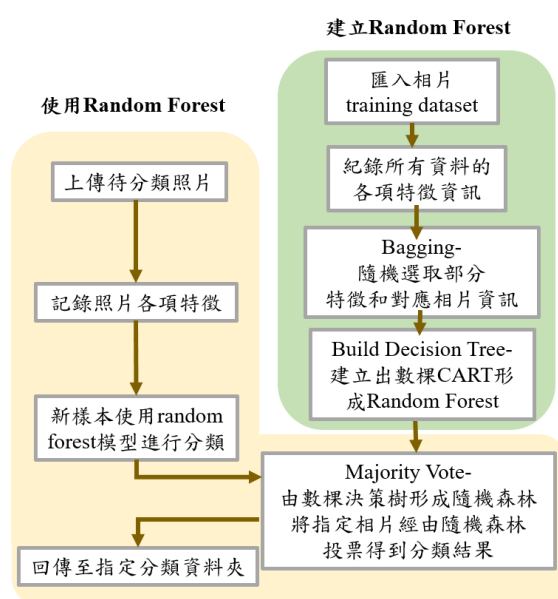


圖 11：自動分類程式流程圖

3.3 建立 Random Forest

訓練 Random Forest 模型當作分類五大類別分類依據，過程需經過前置作業及三步驟[1,2,7]：

前置作業-匯入訓練資料庫：

參考圖 12。

在分類程式撰寫之前，了解 Random Forest 需要準備的訓練照片資料類型，並不像一般的分類器需給予正負樣本，而是每一類的定義照片各 500 張，尺寸需統一為 480*600 之比例。

- 人物-使用 Flickr 的生活化人物照片(by iPhone6)。

(公開的人物 data 為 face detection 類型，畫面中僅存臉部並擁有乾淨的背景，偏離了我們所預期的生活化照片，故我們不予以採用，而用 Flickr 相簿之人物照片代替。)

- 動物-The Oxford IIIT Pet Dataset
- 食物-Food-101
- 建築-Zurich Building Image Database



圖 12：訓練照片資料出處

3.3.2 紀錄相片特徵擷取結果

Feature：人臉偵測(Face detection)
[ref 3]

將每張圖片經過特徵擷取後 (Feature Extraction)，再進行瀑布偵測 (Cascade Detection)，若每一層的檢查情況都符合，會將此特徵辨識為人臉並篩選出來，選擇的 Classifier 為 Opencv 2.4.9。

人臉偵測能有效地輔助判別人物

照片。

Feature：霍夫直線偵測轉換 (Hough Line Transform) [ref 4]

使用標準霍夫直線變換的函數：HoughLines。此函數能提供一組參數對 (θ, r_{θ}) 的集合來表示檢測到的直線。

利用霍夫直線偵測轉換，建築類照片平均來說線條較為明顯(圖 13)，相較於人物的線條數量(圖 14)，可以納入參考依據。

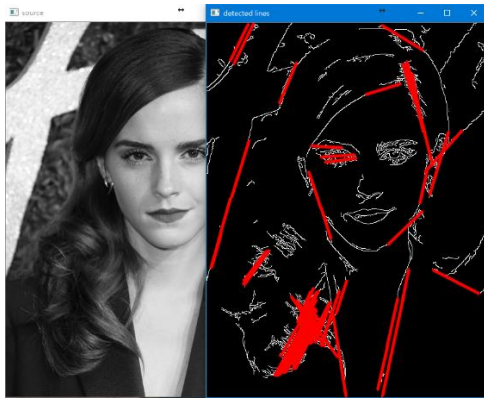


圖 13：(人物)霍夫直線轉換成果

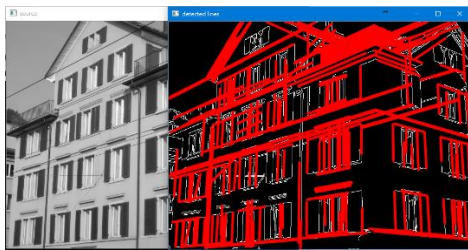


圖 14：(建築)霍夫直線轉換成果

Feature：霍夫圓偵測轉換 (Hough Circle Transform) [ref 5]

我們用和霍夫直線偵測同樣的概念，進行霍夫圓形偵測(圖 15)，圓方程式為 $(x-a)^2 + (y-b)^2 = r^2$ ，其中 (a, b) 為圓心座標， r 為圓的半徑，用這個三維數據組，讓 (a, b) 在影像座標內不斷改變

位置，找出所有可能的半徑 r ，最後當這三維數據組的點數，超過自定的閾值時判定偵測到圓。

使用 OpenCV 函式：

HoughCircles，將欲偵測的相片輸入，記錄偵測到圓的個數。經過影像的模糊前處理，食物類別的相片產生出團狀的食材聚集，所以霍夫圓偵測轉換可使食物類別的特徵突顯出來。(圖 16)

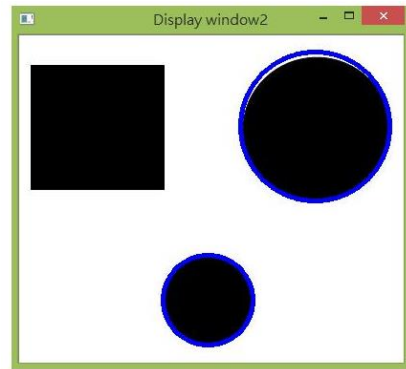


圖 15：霍夫圓偵測轉換成果



圖 16：(食物)圓形偵測成果

Feature：矩形偵測 [ref 4]

同樣基於霍夫直線偵測 (Hough Line Transform) 的概念，先找出相片中的直線，在計算出兩雙平行對邊，且相互垂直的四條直線，偵測為矩形，紀錄相片中矩形的個數。而建築物類別的主體結構以及窗戶通常為矩形，有利於矩形特徵的擷取(圖 17)。



圖 17：(建築物)矩形偵測成果

Feature：平均對比度(Contrast) [ref 6]

也可以稱之為相片的紋理，首先計算出相片中每個 pixel 對周圍 pixel 變化的估計值，稱之為 "Local Contrast"，假設 $p=(i,j)$ 在一個 $W \times W$ 的 pixel 矩陣中間，則

$$\text{Local_Contrast}(i,j) = \frac{\max_{p \in W \times W} (p) - \min_{p \in W \times W} (p)}{\max_{p \in W \times W} (p) + \min_{p \in W \times W} (p)}$$

而每個 pixel 的 Local Contrast 平均值稱之為 "平均對比度 (Contrast)"

Contrast =

$$\frac{1}{m \times n} \sum_{i=1}^n \sum_{j=1}^m \text{local_contrast}(i,j)$$

(n,m 分別為相片的長與寬)

記錄每張相片的平均對比度 (Contrast)，比較每一類別的數值，可以觀察到建築物類別的相片紋理較明顯，可以將他跟其他類別加以區分。(圖 18)

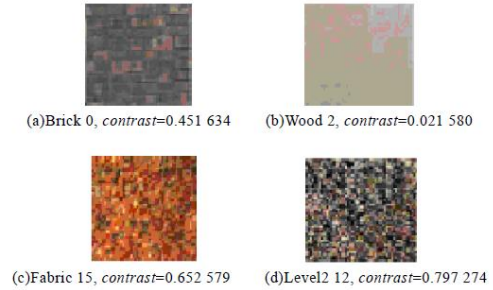


圖 18：平均對比度示意圖

Feature：哈希感知演算法 (Perceptual Hash Algorithm)

此演算法為 Google 利用以圖搜圖之方法，將顏色特徵擷取部分稍微做修改，改成每一步分的簡單紋理擷取。(圖 19)

此演算法著重在將原始圖片縮小後，紀錄由左上到右下的簡單紋理分布，在一類當中尋找跟該類圖接最相近之中心代表圖片，接著將訓練照片資料與各類別之中心圖片做比對，較為接近的就進行類別 tag。

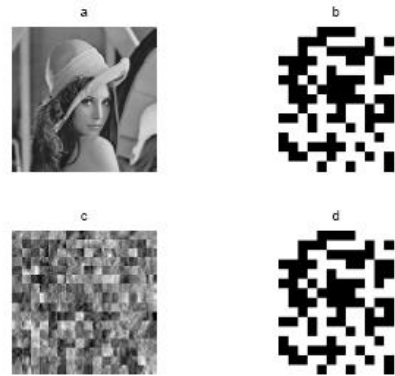


圖 19：簡單紋理哈希感知演算法示意圖

Feature：色彩平均(RGB)

紀錄圖片的中 RGB 三類色彩的比例，紀錄照片偏向的色系類別。

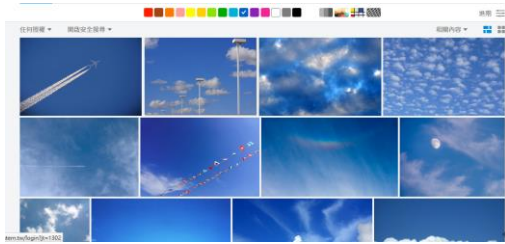


圖 20：天空顏色為 Blue

3.3.2 Random Forest- Bagging 隨機選取部分特徵及資料

Bagging 過程著重-隨機，也是 Random Forest 演算法的隨機來源，假設為袋中裝了全部的訓練樣本，從中隨機抽出部分的樣本作為下一步 decision tree 的訓練分枝依據，抽出後再放回，又稱此隨機抽抽出又放回的步驟為”bootstrap”。

模擬這樣的操作規則，演算法也對 feature 做 bagging 的動作，選取部分 feature 協助 decision tree 分枝，產生增加了 decision tree 的多樣性，不僅 data 不一樣連 feature 也不同。

其中，建立 decision tree 該隨機取多少樣本及隨機取多少的 feature 做參考，需要利用多次反覆測試取得最適合的參數值，讓相片分類更能清楚劃分。[ref 1,7]

3.3.3 Random Forest- 建立數個 CART

$$OOBAcc_k = \frac{\sum_{i=1}^n I(h_k(d_i) = y_i; d_i \in OOB_k)}{\sum_{i=1}^n I(d_i \in OOB_k)}$$

Forest 是由眾多的 decision tree 建構出的森林，且 decision tree 有多種類包含：ID3、CART、C4.5...等，演算法中選擇 CART 不剪枝，以下簡

略呈述使用 CART(Classification and Regression Tree)之特點：

1. 強調在每一個分枝上，與其他 ID3,C4.5...最大的不同是一次只能分枝成兩個子節點。
2. 利用 Gini(S)為吉尼係數

$$Gini(S) = 1 - \sum_{j=1}^n p_j^2$$

定義為 Pj 在 S 中的值組屬於類別 j 的機率。

不純度降低值：

$$\Delta Gini(A) = Gini(S) - Gini_A(S)$$

CART 中利用最大部純度降低值、或吉尼係數 Gini 最小作為節點先後順序的依據。

將 Bagging 的結果套入 CART 中，省略原始剪枝的動作，建立出數棵 decision tree 形成 forest 每一個 decision tree 可以產生出一個分類出的結果，到此步驟，random forest 可以算建立好分類模型。[ref 8]

3.3.4 OOB(Out-of-Bag)

在 Bagging 過程中會取出部分建立 decision tree，而剩下的部分就運用 OOB 原理做 decision tree 的評分標準，OOB 原理是依 Bagging 出的 decision tree 測試剩下沒有被選中的資料，比對正確的結果，若為判斷正確則可以把累積分數加一，最後累積較多分數者，就為判斷能力較好的 tree 可以被選進 random forest 中讓樹對於分類的幫助提高到最大。[ref 9]

3.4 使用 Random Forest

模型建立完成後可以匯入照片做測試分類，匯入後記錄上述 feature 做照片特徵的紀錄，將記錄好的資訊提供給模型需求做使用，3.4.1 要介紹的就是模型中的分類模型接獲新的資料時的判斷依據。

3.4.1 Random Forest-Majority Vote 隨機森林中每棵決策樹投票出最終類別

Random forest 中累積多棵 decision trees，將指定照片的所有資訊一一放入 decision tree 中每棵樹將會得到分類的結果，每筆分類結果視為一張票，投給一種類別，最後得到最多票的，就為該照片的答案(圖 21)。[ref 1,7]

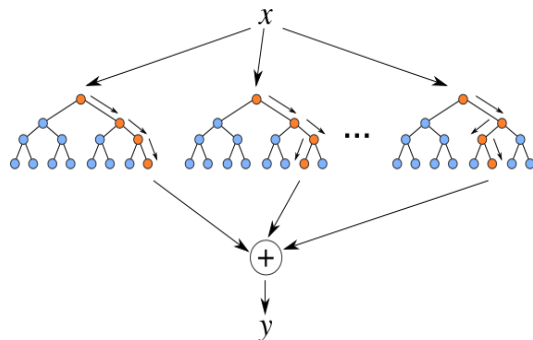


圖 21: Random Forest 每一棵 decision tree 共同投票出結果

3.5 系統架構與平台

分為 Client 和 Server 端，由 Client 端上傳欲分類的照片，Server 端接收到後進行分類運算，再將結果回傳給 Client 端。(圖 22)

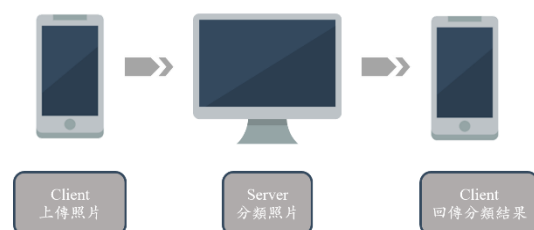


圖 22：系統架構

3.6 分工表

劉晏	1.Random Forest 建置 2.尋找可用 Feature 3.架置 Server 4.App 功能：相機連線、上傳至 Server 端
黃亭雅	1.尋找可用 Feature 2.App 介面設計 3.App 功能：分享至社交平台
蔡羽涵	1.尋找可用 Feature 2.App 功能：分類相簿、一鍵分享

4. 主要成果與評估

4.1 系統效能

統計 4 個類別分別測試 3 張照片，共 12 張照片的分類時間，平均一張花費 4.88 秒。每張照片所花費時間如表 1。

表 1：12 張照片測試時間

測試 1 (人物)	測試 2 (動物)	測試 3 (食物)	測試 4 (建築)	測試 5 (人物)	測試 6 (動物)
2.17	2.88	1.36	8.42	1.34	4.77

測試 7 (食物)	測試 8 (建築)	測試 9 (人物)	測試 10 (動物)	測試 11 (食物)	測試 12 (建築)
10.30	7.84	3.15	9.65	4.47	2.23

4.2 分類精確度

每一類別使用 500 張照片進行精確度的測試，準確度及測試所使用的 Random Forest 方法與數量如表 2，每類的分類結果數量總表如表 3。

表 2：測試每類 500 張照片之精確度

類別名稱	人物類	動物類	建築類	食物類
照片張數	500 張	500 張	500 張	500 張
精確度	87%	87%	90%	76%
RF 樹之總數	50*0.8=40(篩選後結果)			
樹 data 數目	1000 張(training data 共 2000 張)			

表 3：分類結果數量總表

Input Result	人物類	動物類	食物類	建築類
人物類	435	43	41	10
動物類	32	435	72	7
食物類	16	14	380	31
建築類	14	7	5	451
其他	3	1	2	1

4.3 主要成果評估

- a. 將分類演算法架置在 server 端為降低 App 在手機上的計算量，最後決定將演算法的部分放置在 server 端，讓使用者藉由上傳照片的方式，由 server 端接收照片進行分類運算，接著再回傳分類結果給使用者。

- b. 使用者介面(圖 1~10)
因應現在普遍的人都在使用智慧型手機，以及利用手機相機記錄生活，故將使用者介面做成 App，讓使用者可以隨時隨地，只要有網路即可將照片做良好的整理，也可以避免手機

記憶體不足的問題。為來希望能夠將 App 的功能做得更加完善及美觀。

- c. 分類演算法正確性
現在只有每類蒐集 500 張照片進行運算，而分類結果的精確度還不盡理想，未來希望能夠蒐集更大量的 Dataset 進行訓練，分類演算法可望更為精進與準確。

5. 結語與展望

5.1 結語與心得

在這一年的專題中，大部分時間主要著重在分類演算法的部分，所以 App 功能並未實做得很完整，另外也因為時間有限，無法進行更大量 Data 的訓練，讓分類的準確度更提升。但仍然在這之中學到了許多，比如分工合作、團隊溝通，也能夠自己學習原先不熟悉的事物，虛心向老師及實驗室的學長姐發問尋求協助，讓我們獲益良多。

5.2 未來展望

a. 客製化分類器(圖 23)

目前只有四個類別可供使用者分類，未來希望能夠讓使用者自行定義新的類別，只需要提供一些訓練樣本(照片)，就可以藉由我們的訓練方式，建立新的 Random Forest，提供給更多人使用。

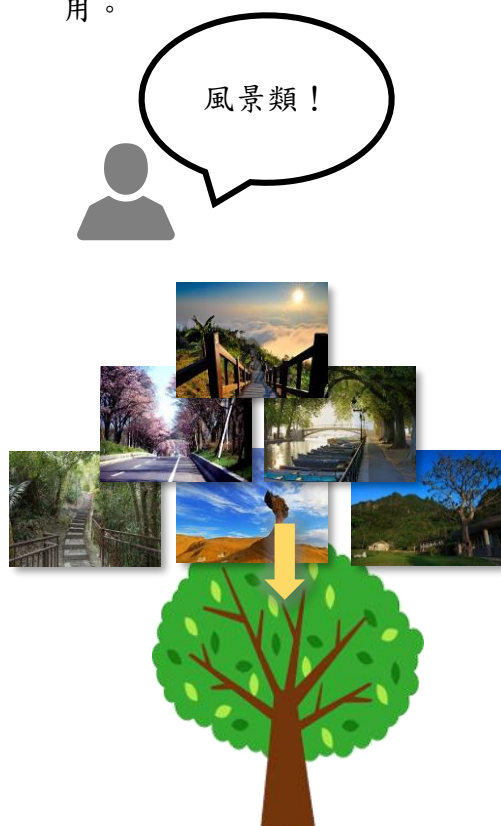


圖 23：客製化分類示意圖

b. 與可攜式相機結合

目前市面上有許多可攜式相機，例如：GoPro(圖 24)、小蟻運動相機(圖 25)，都有每隔幾秒自動拍攝的功能，也會造成大量照片無法整理的情形，這時便可利用 Picollate 進行分類整理，讓使用者在觀看相片以及分享時更為便利、簡潔。另外提供 Real-time 捕捉畫面功能，在拍攝前先設定想要的類別，每拍攝一張就上傳至 Server 進行分類，若結果為想要的類別則保留照片，反之則刪除，避免記憶體空間的浪費。



圖 24：GoPro

(圖片來源：<https://goo.gl/nyZLtJ>)

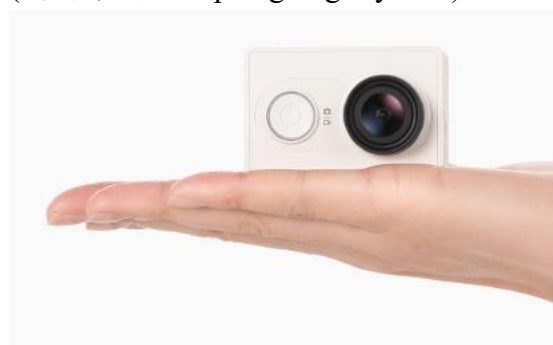


圖 25：小蟻運動相機

(圖片來源：<https://goo.gl/JqObb4>)

6. 銘謝

2015 Intelligent Multimedia System
Lab, IMSLab.

國立臺北大學資訊工程研究所
智慧型多媒體研究室

on Pattern Recognition, Challenges in
Machine Learning 1 (2011): 223-230.

7. 參考文獻

[1] L. Breiman, "Random forests,"
Machine Learning, vol. 45, no. 1, pp.
5–32, 2001.

[2] A. Bosch, A. Zisserman, and X.
Munoz, "Image classification using
random forests and ferns" *In ICCV*,
pp. 1-8, 2007.

[3]
[http://bugworkshop.blogspot.tw/2015/
09/opencv-24-houghlinesp.html](http://bugworkshop.blogspot.tw/2015/09/opencv-24-houghlinesp.html)

[4] Opencv, Face Detection using
Haar Cascades
[http://docs.opencv.org/master/d7/d8b/t
utorial_py_face_detection.html#gsc.ta
b=0](http://docs.opencv.org/master/d7/d8b/tutorial_py_face_detection.html#gsc.tab=0)

[5] <http://goo.gl/OOSDCg>

[6] 王莉. "基於 FRD 的圖像紋理
情感語義提取." *計算機工程* 35.20
(2009): 212-215.

[7] Keep Learning 之機器學習之隨
機森林
[http://backnode.github.io/pages/2015/
04/23/random-forest.html](http://backnode.github.io/pages/2015/04/23/random-forest.html)

[8] 國立聯合大學 資訊管理學系
機器學習課程(陳世杰教授)
Decision tree learning 上課講義

[9] Dahinden, Corinne. "An improved
random forest approach with
application to the performance
prediction challenge datasets." Hands-