期末專題 結報

Facebook Image Search on Android

第 12 組
B95901146 鄭嘉勳
B96901068 詹智鈞
B96901138 林志瑜

Outline

Part 1

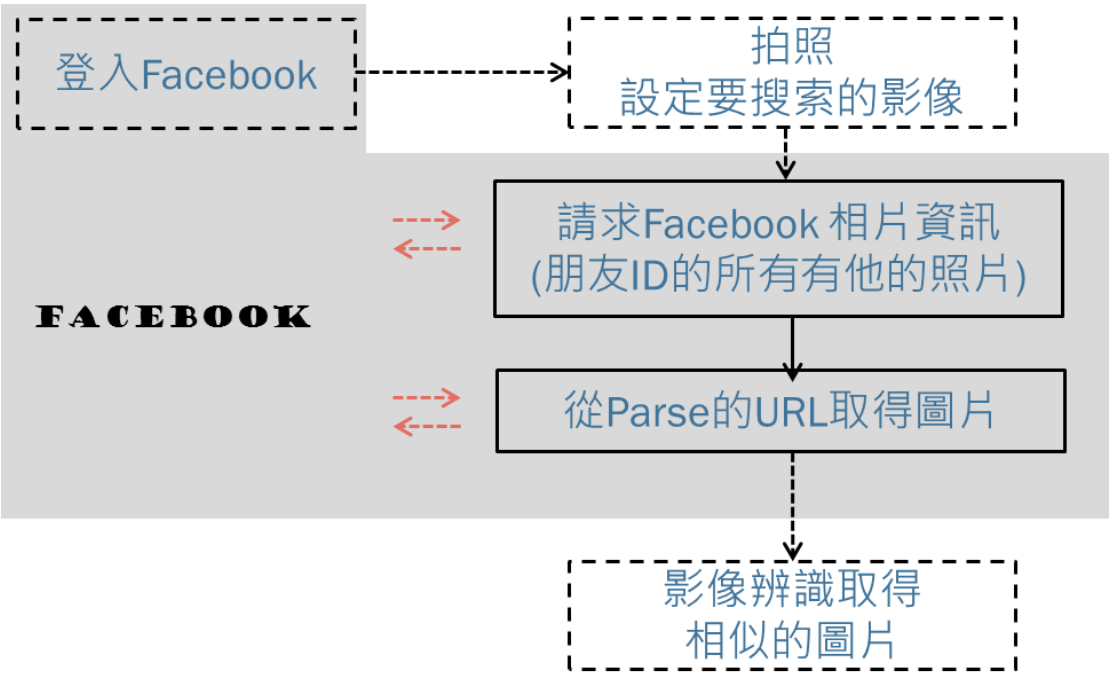
介紹

目標：我們使用Android(HTC desire)及Facebook API，讓使用者照相後可以根據影像處理找出本人或相似度最高者。

實作：首先透過程式拍一張想要來來搜尋的相片(或者也可以選擇SD卡內所保存的照片)，並經由Android臉部辨識功能來偵測出相片中人物的臉部。接著透過API的request，抓取包含朋友所有被Tag的照片，最後透過臉部的辨識來判斷與所選取的相片和照片中所Tag的圖上的人的相似程度。而相似度在設置的限制內的圖片與連結・・・・・・等，便會顯示在最後的畫面中，如果顯示介面的格數滿了，便會停止搜索，使用者可以刪除不相似的人，程式就會繼續搜尋下去。

Part 2

流程與挑戰



使用 Facebook 爲資料庫影像搜索流程圖

上圖爲程式大架構的流程圖，而細節與挑戰如下：

Facebook API & How to get data & Listener　志瑜

Facebook Developer 提供給想要開發 facebook 的相關程式的使用者 API，包括 Core APIs、Facebook SDKs、Advanced APIs。在這次的程式中，由於為了比較方便地來使用 Android 來完成從拍照到偵測的動作，選用的是 Core APIs 中的 Graph API、SDK 中的 Android SDK、以及 Advanced APIs 中的 Old REST API。本來想使用 PHP 或者 Java 來架 Server，但由於對於 PHP 不熟，以及 Java 的 API 使用十分的不易(目前 Facebook 已停止對於 Java API 部分的更新支援，目前的更新是由一些較為熱心的自由開發者所進行的)，因此便沒有採用。

要進行開發前，首先必須向 Facebook 申請一個應用程式的開發授權，如右圖所示，申請完畢後可以拿到一個 API 金鑰(用於程式內的授權)以及應用程式密鑰(代表你擁有這個程式的開發及修改權限等)。

對於使用 API 的方式部分，Android SDK 有提供一些選擇。我們可以使用底下提供的 class 裡的 request 的方法來存取 Graph API 以及 Old REST API。

目前 Graph API 是 Facebook 主打的較新的 API，其想法是漸進式的取代 Old REST API，故目前 Old REST API 有些已經被取消支援並告知使用 Graph API 了。但就說明的完善性來說，Graph API 還有待加強，直接打上提供的網址而忘了使用 access_token 的話，會無法取得授權而失敗。

第一次使用程式登入的時候會詢問授權請求，這點可以在 class 內修改 permission 來達到目的，需要注意的是，如果 API 的 request 需要某些特定的權限，而程式一開始並沒有取得授權，request 時是無法拿到有效資料的。

每次 API 的 request 都會花上一點時間，故這裡是使用 Listener 及 Thread 的方式來進行處理。當跑某次 request 時，程式會新開一個 Thread 來處理這次的傳送請求及接收資料，我們可以透過 API 拿到使用者登入 ID 所有朋友的 ID 名單(Graph API 的 me/friends)，而透過這些 ID 名單，可以抓到所有這個朋友 ID 所被 Tag 的相片，即便那張相片不一定是在自己或者是這個朋友的相簿裡。(Old REST API 的 photos.get)，接下來我們再針對每張圖片接收那張圖片所有的 Tag 資訊(Old REST API 的 photos.getTags)，由於我們想要搜索的相片就有可能當中存在想要找的那個人，所以之後再將這些資訊做進一步的辨識處理。

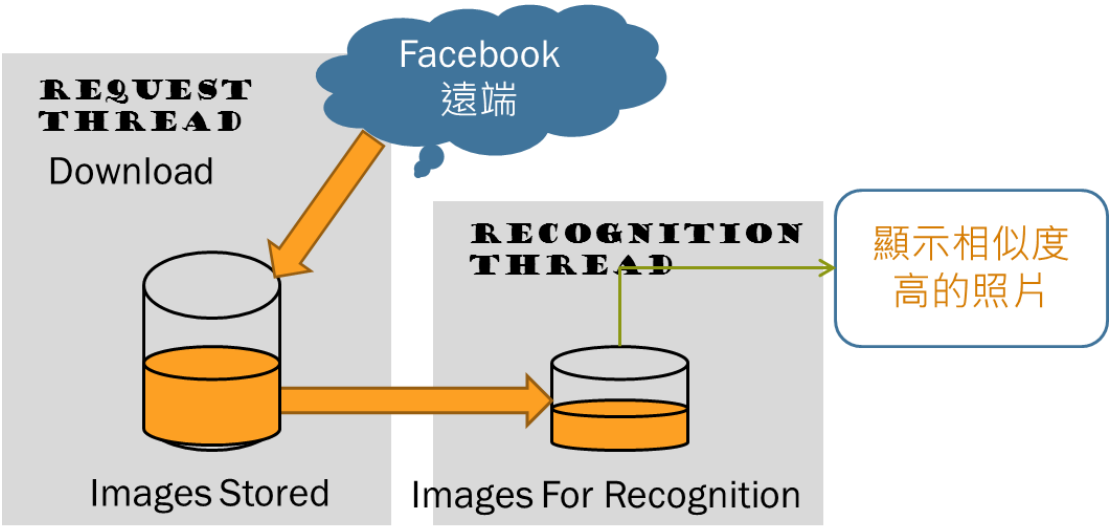
所有的 API request 的回傳都會是一個 String 型態的 response，而內容型態會像右圖：

這是一個使用 JSON 格式來形成的字串，我們必須利用 Util 底下提供的 parse 方法或者自己來寫一段 parse 的方法。

過程中會遇到的挑戰是由於回傳的問題，JSON 的 parse 沒有處理好的話會很容易 crash，而在進行一連串的 PhotoRequest 的時候，facebook 回傳的 response 有時會出現同一 response 內卻將下一張的 pid 放到前一張的 tag 資訊裡的狀況。這點經過多次確認後仍然無法理解在 request 的過程中究竟出了什麼狀況，最後只能以手動移位陣列來對齊，也因此會無法讀取到相簿最後一張相片的正確資訊。


```
[
 {
  "pid": "5935512127267816873",
  "aid": "5935512127239286972",
  "owner": "1381969109",
  "src": "http://photos-e.ak.fbcdn.net/hphotos-ak-snc3/hs122.snc3/16940_
  "src_big": "http://sphotos.ak.fbcdn.net/hphotos-ak-snc3/hs122.snc3/169
  "src_small": "http://photos-e.ak.fbcdn.net/hphotos-ak-snc3/hs122.snc3.
  "link": "http://www.facebook.com/photo.php?pid=30557609&id=1381969109"
  "caption": "",
  "created": 1264491372,
  "modified": 1264502979,
  "object_id": 1205310417871,
  "src_small_width": 75,
  "src_small_height": 100,
  "src_big_width": 453,
  "src_big_height": 604,
  "src_width": 97,
  "src_height": 130,
  "position": 1,
  "album_object_id": 1182571789419,
  "images": [
   {
    "height": 604,
    "width": 453,
    "source": "http://sphotos.ak.fbcdn.net/hphotos-ak-snc3/hs122.snc3/
   },
   {
    "height": 239,
    "width": 180,
    "source": "http://photos-e.ak.fbcdn.net/hphotos-ak-snc3/hs122.snc3
   },
   {
    "height": 130,
    "width": 97,
    "source": "http://photos-e.ak.fbcdn.net/hphotos-ak-snc3/hs122.snc3
   },
   {
    "height": 100,
    "width": 75,
    "source": "http://photos-e.ak.fbcdn.net/hphotos-ak-snc3/hs122.snc3
   }
  ]
 },
 {
```

Resource Management & Face Detection & Camera　嘉勳



在如何使得辨識順利進行並且持續取得資料，且不能超過硬體資源的限制，遇到了最大的阻礙，所以資料結構中要能設限制，不能一直開 socket；另一方面資料也不能儲存太多，想到的解決方法是開兩個有上限限制的 Pool。

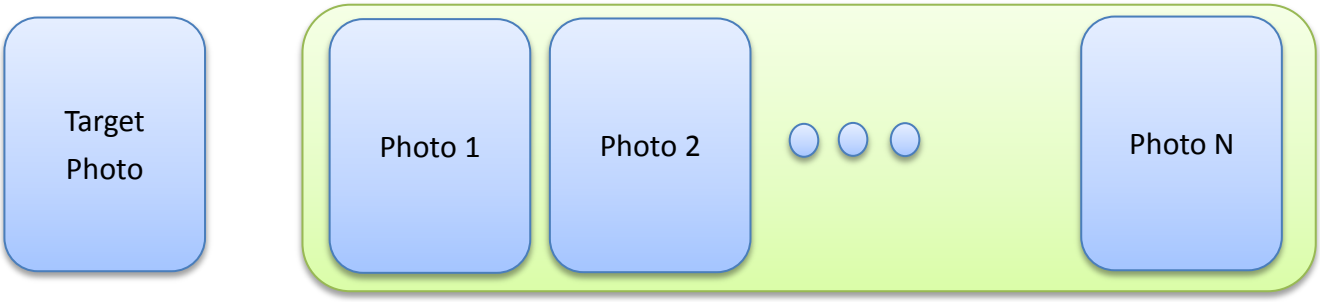
從向 Facebook 發出 request 到取得資料有延遲時間，所以要如何使資源利用最有效。一個 pool 負責存外界存進來的 Tag， URL 還有使用者資訊，可以避免等待收取資料；而內部的 pool，再從前一個 pool 拿資料來辨識。

而主要的 bottleneck 在辨識部分，愈高解析度的圖有愈高的辨識率，所以決定使用大圖，載下來的圖示大約爲 700x500 畫素，所以運行結果 1 分鐘約 40 張圖。假如取中圖速度可以提升一倍，但辨識效果較差。

而辨識部分需要取得欲要辨識的對象，並且框上人臉，這方面 Android API 有提供 Face detector，但缺點是假如圖片畫素高就必須花較久的時間，所以 detector 只能用來框 target bitmap。相反地 data-base 方面 Tag 的大頭圖是用 heuristic 的方法取得，一個人在圖片中就截半徑 40px 的矩形，而人多的時候則減少，而一張圖假如多於五人則不取圖，因爲圖太小也看不清楚。

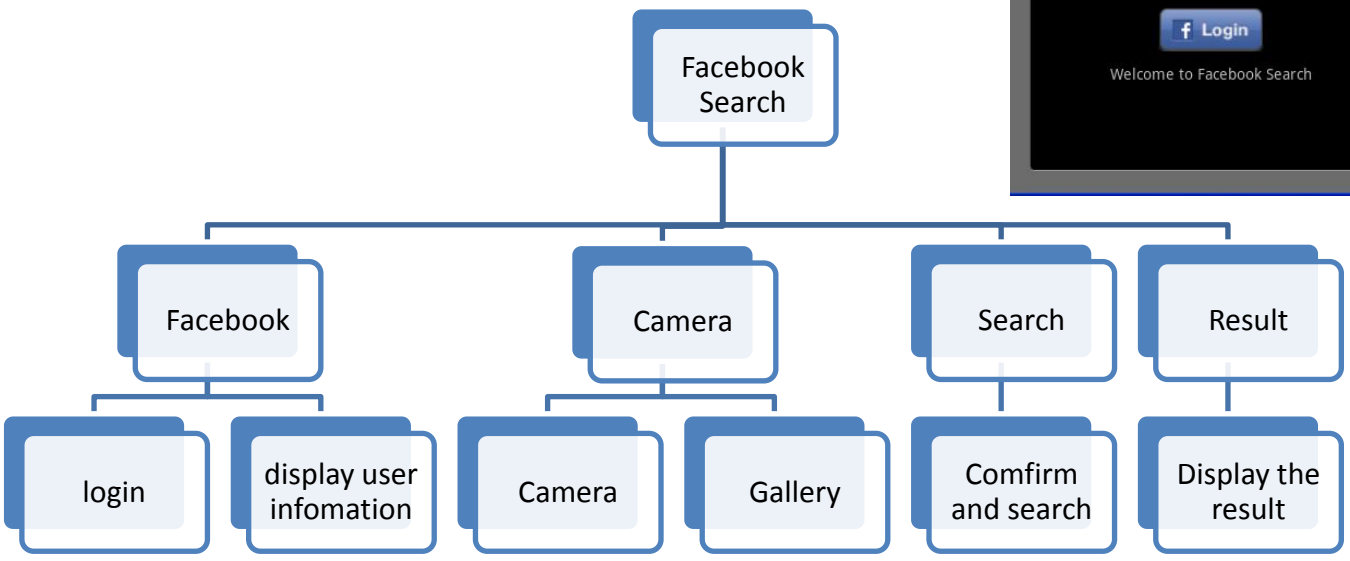
Interface & Face Recognition 智鈞

辨識部分，我們改寫網路上現成的 code，其實作如下圖表示:



將 target photo 逐一與下載到的 photo 進行比對。每一次比對，可以得到 N 個相似度值(以 distance 表示，越小表示其相似度越高)。排序選出 distance 最小的，如果其值小於我們自己所定的 bound，則將之顯示於 result 頁面。

介面的部分我們採用 tab host 的方式來實現。我將我們的程式分成四個部分，如圖所示。第一個 tab，我們提供 facebook 登入的功能。登入以後，會顯示登入者的大頭貼、姓名和狀態。第二個 tab，我們讓使用者能夠選擇要辨識的照片。第三個 tab，我們讓使用確認所選的照片，並加以辨識。第四個 tab 簡單顯示其搜尋結果。



Part 3

心得

志瑜:

在溝通上及整合上一定要仔細的互相說明清楚，由於是共同開發，如果有彼此改到對方的程式碼的時候應當事先知會，以免發生認知上的差異從而導致錯誤。

WBS 的系統對於並非是正職工作以及碰到不容易切細劃分時間 task 的網多來說，幫助上其實並不是很明顯，因爲處理 code 的 bug 很容易卡住，所以到處都會是 milestone 挖阿阿阿啊!

從一個想法到實際的寫成程式中間過程常會發現原先沒有考慮到許多問題，而我在找資料的能力上自覺還有很大的有待加強的空間，在找尋資源時應該要能夠思考更多可能搜索到東西的關鍵字。

在這次的程式撰寫中，我練習到了不少平常根本沒有碰過的東西，也更多的試著去學習在短時間內了解別人的程式碼究竟在寫什麼，我想這個訓練對於以後應該也很有幫助。就像之前隆哥請來在 Google 工作的學長所說的，有時候往往是拿了說明文件和 Example，接下來就是讀懂以及依樣畫葫蘆，對於這次從頭開始了解 Facebook API 的操作方法來說，的確是很好的註解。

Facebook 所有的 API request，程式授權，以及 Handle Class 的撰寫。

隆哥好棒喵!

感覺這種實驗就是修來逼自己看到 Deadline 來臨時榨出自己的能力極限以及燃燒肝臟，當然，也是短時間內讓自己學會原本不會的東西的好方法。(像本來我根本沒有碰過 java 的…)

門口的牌子好像應該要做牢固一點才不會掉下來 XD

這學期感謝隆哥的陪伴以及鼓勵，也希望以後的網多能夠更好。(例:像我座位這台的 BenQ 螢幕看一段時間眼睛真的就會很累，比起隔壁的 ViewSonic 的品質好像有點差距@@")

嘉勳:

期末專題中希望結合 Facebook 提供的社群資料，使得使用者能在手持裝置中取得，並且提供進一步的服務，所以學習到了資料庫 API 的使用方式，Facebook 是前幾大的社群網站，但是 API 所提供的方法仍有小 BUG，要的資料與得到的順序並不相同，不過其 API 包裝的方法類似 URL parse，如果寫資料庫相關的程式可以仿效。

另外因爲 Android 手機雖然在手機中算是性能不錯，但跟個人電腦或是伺服器相比，性能較差，所以常常 Memory overflow，花了很多時間去設定適合的 bound，所以更好的辦法是實際運算還是得用伺服器，而手機端只負責 request/display。

主要負責的部分是相機功能,如何讀 SD Card data，儲存待辨識資料的結構，以及 Thread 自動偵測顯示數目判斷是否要繼續執行。

整個學期以來，網多實驗是一個有題目後，大家就發揮想像力，去呈現心中的想法，別於一般的必修課，沒有主要的授課內容是它的優點也是缺點，不過可以把它當作是整合所有已知知識並與同學交流的地方,可以見識到不同的ideas，而各種平常見不到的設備，如：觸控螢幕，智慧型手機，各種 SDK 與 Library，如何使用也是需要時間自己去熟悉與活用。

很感謝組員的互相幫忙，因爲常常會手殘或秀逗，就需要其他人的意見與 debug 的協助，當然最感謝的是助教，助教出現在實驗室的頻率真高，會常來關心並提供解決辦法。

智鈞:

我們的作品最後仍然受限於效率問題，是我們最可惜的地方。原本打算在 server 上才執行搜圖與辨識的功能，但這個部分 api 上支援的不太好。後來才改在 api 支援較好的 android 平台上開發。在作法上面，我們先搜尋了照片的 URL，再下載辨識。後來才想到，可以用手機來搜 URL，將 URL 傳回 server 載圖這種作法，可惜爲時已晚。

我這次負責的部分包括整體 GUI 介面的設計和人臉辨識的部分。一開始我打算用 picasa 來辨識人臉，花了一星期的 search 後才發現 picasa 把這個功能關掉了，也沒有提供 api。最後我在網路上找到網友自行寫的 code，於是就決定直接改寫使用。

在不斷嘗試的過程，我學到心態是很重要的。常常自己的部分作好了，整合時卻有 bug 產生，這非常令人沮喪。必須與組員不斷溝通才能逐漸有所突破吧！如果組員聽不太懂我的想法，通常我就會放棄解釋了。但經過一段時間我發現還是得努力溝通。努力溝通下，我們也才能獲得共識。另外，不斷嘗試不放棄也是我所學習到的。一開始有點擔心我們不能做出來，也擔心做得不好，因爲從未寫過如此大的程式。做 Final 有點像在跑馬拉松，感覺已經不行了，只能說服自己，終點就在眼前，要不斷加油。直到最後 demo，雖然仍然沒有表現得 ok，但也說得過去，這都得感謝助教的幫助和組員們的合作。

在 demo 的時候，我覺得自己許多想法都被觸發了。自己一個人做，或者一組人員閉門造車,點子都不如所有的人一同展覽能夠感受得多。若是以後有機會，希望能有更多的嘗試。

對於助教隆哥一學期辛苦的帶領,後只能說非常感謝。隆哥一直很照顧我們，特別我們這組座位在隆哥座位的旁邊，有什麼問題都可以直接問。而且隆哥人很親切，都會主動來關心我們。我們也不用擔心自己的問題太蠢會被笑，隆哥都會盡其所能的回答我們。每個星期，隆哥甚至固定寄給大家通知信，提醒我們該注意和要完成的事。另外，隆哥這學期也給我們安排了兩次的演講。分別是葉丙成

教授來跟我們介紹 WBS 專案管理系統和在 Google 工作的 B93 屆學長跟我們分享其工作的心得。兩次演講都很有收穫，可見隆哥的用心。

最後，想跟學弟妹們推薦這門好課。不論本來的基礎如何，只要肯花時間就能學到很多。另外關於網多實驗需要改進的地方，個人認爲硬體設備雖然已經很好，但還可以再更完善。以我爲例，我使用的螢幕顯示效果較差，長時間看下來，眼睛很痛。另外書籍提供方面也有一些缺，特別是在 android 手機程式設計方面的參考書真的很少，如果經費許可，我覺得可以在這方面改善。

Part 4

參考資料

[1] http://www.htc.com/hk-tc/product/desire/specification.html
HTC desire specification

[2] http://developer.android.com/
Android & Facebook reference

[3] http://developer.android.com/resources/samples/ApiDemos/index.html
All API demos provided by android with source code

[4] http://zh-tw.facebook.com/
Facebook homepage

[5] http://stackoverflow.com/questions/477572/android-strange-out-of-memory-issue
Bitmap memory overflow issue