東吳大學 巨量資料管理學院專題實作

- 專案名稱 -

Horus :

結合人臉辨識、物件追蹤與 ReID 技術實作天眼系統

組員:

何彥南 Yen-Nan Ho

簡大爲 Ta-Wei Chien

盧煒中 Wei-Chung Lu

指導老師:

黃福銘 Fu-Ming Huang

目錄

HORUS :

結合人臉辨識、物件追蹤與 ReID 技術實作天眼系統

摘要

在電腦視覺的領域中，如何確認影像中的人物身份是一個重要的研究議題，發展到現在，許多相關研究都在身份確認的任務上有所貢獻，像是物件追蹤、人臉辨識。雖然兩者在各自的任務上已經有穩定表現，但實際的落地上還有許多具有挑戰性的場景，像是在影像中人不可能無時無刻面對鏡頭，而或者是捕捉到的臉部畫質不佳，導致臉部辨識無法有效達成任務；或是目標被遮蔽、目標出鏡頭後再回到鏡頭，導致持續追蹤在追蹤不連續的目標上有困難。鑑於以上問題，在近幾年有另一個研究問題出現了-「行人的重新識別（ReID）」，ReID 技術透過大量的行人的圖片並標記身分，藉此訓練 ReID 模型如何判定兩張照片是同個人，此技術使用整個人的特徵去做判斷，這樣做可以減少對臉的依賴，但缺點是無法分辨穿不同衣服的同個人，且在臉部資訊充足的狀態下準確度不會比臉部辨識高。也因在不同的情況下不同的技術各有優缺點，如何將相關技術結合並應用在實際場景，是本次專題的重點也是我們希望學習的。本專案的目標是結合臉部辨識、物件追蹤和 ReID 技術建構一個行人辨識系統 - Horus，並透過臉部辨識技術加入會員辨識機制。希望透過此專案之實作了解上述幾個

技術背後的運作方式與其相關限制，開發出一個可以實際落地的系統。

關鍵字: 臉部辨識 (Face recognition)、**物件追蹤** (Object tracking)、**行人重新識別** (ReID，Person Re-identification)、**深度學習** (Deep learning)、**資料庫系統** (database system)

一、緒論：

(一) 背景與動機

隨著深度學習與運算效能的進步下，許多問題都已經有效的解決了，像是臉部識別、物件追蹤和圖片分類。然而，這些技術都是以單面向的問題上去做發展的，這也導致在某些實際應用的場景上，這些既有的技術無法獨立達成任務，需要透過不同技術的結合才能實際應用。在人臉辨識的技術上，透過人臉的身份識別已經到了有很高的準確率，但不是所有場景中的人臉都是面相鏡頭的，而且在大部分的畫面中捕捉到的人臉較小，這些問題都是人臉辨識在實際應用上會遇到的困難。也因為如此，人臉部辨識能從一段影像中確認身份的片段會很短。在這種情況下，人臉辨識透過與物件追蹤技術結合，可以將臉部辨識到的身份近一步擴展到物件追蹤的片段上面，也就是說我們可以有效判斷身份片段的時間拉長了。但是物件追蹤技術也是有缺點的，當同個人在畫面中原先追蹤的人被擋住後在出現會導致誤判，又或者是同個人過一段時間在進入畫面，這時物件追蹤技術就無法確定兩段追蹤的人是否為同個身份。在這個問題上，ReID 的技術就派上用場了， ReID 技術透過大量以標記身分的人的圖片集訓練，並判斷該輸入圖片與圖片集中的哪些照片相近，並透過排名的方式確定身份。也因為重新識別主要是以整個人的作為特徵，整對臉部的依賴度也下降，所以我們可以透過 ReID 的方式去確認不同的物件追蹤片段之間的關係。只要其中一個捕捉到的照片可以使用臉部辨識確認身份，就可以擴展到透過由 ReID 串接的多個物件追蹤片段，此方法可以讓我們的身份辨識與追蹤系統捕捉到更多資訊。由此可知，在行人身份追蹤的場景下，使用單個技術時都會有各自的限制，還是必須結合其他技術使用才

能發揮更好的效果。如何結合不同技術，應用在實際的問題上是我們在本專案希望去學習的。

綜合上面所述，本專案希望結合臉部辨識、持續追蹤和重新識別技術，建立一套會員追蹤系統 - Horus，此系統使用臉部辨識技術精準確認會員身分，並使用物件追蹤與重新識別技術去對影像中各個人的片段做身分的確認。此外，本專案希望利用物件追蹤的準確性對同一個連續追蹤的行人資料進行自動資料標記，此作法可以收集更多的設置場域的資料，作為調整 ReID 的訓練資料，以此增強 ReID 之後的判斷。

本專案將有以下貢獻:

1. 結合臉部辨識的精準度與 ReID 以人體為特徵的並以物件追蹤為基底的方式建構系統，可以改善物件追蹤在某些情況下的問題。
2. 不僅增加追蹤資料的廣度(串接不同段的追蹤)，且可以增加資料的深度(透過人臉識別串接背後的會員系統)。蒐集龐大的資料後，可以做為後續分析的能量。
3. 實際將 ReID 技術導入到實際應用場景上，並結合物件追蹤和臉部識別技術，以後相關研究或是企業也可以參照我們提出的架構去實現。

(二) 系統設計概念

本系統之名字為「Horus」，Horus 是荷魯斯之眼的意思，荷魯斯之眼又稱真知之眼、全視之眼、埃及烏加眼，是一個自古埃及時代便流傳至今的符號，也是古埃及文化中最令外人印象深刻的符號之一。荷魯斯之眼顧名思義，它是鷹頭神荷魯斯的眼睛，代表全能全知之眼。我們希望能開發出一個媲美「天眼」的系統，故以全視之眼的概念命名，將把資料看清、把信息理解透徹做為我們系統的開發目標。 (部分摘錄自維基百科)

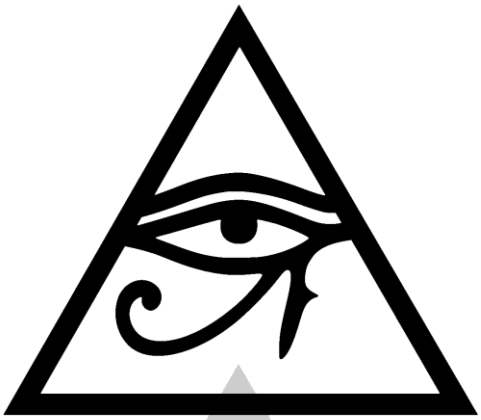


圖 1. Horus Logo (以 Adobe illustruator 繪製)

上圖為本系統的 Logo。三角形的圖樣除了代表我們三位組員外，同時也代表了本系統的架構設計 – 三位一體模組架構 (Trinity Module Structure)。三角形的三個頂點代表 Horus 系統中的三個核心模組：多物件追蹤模組 (Multiple Object Tracking Module)、臉部辨識模組 (Face Recognition Module) 、行人重新識別模組(Person Re-identification Module)，當三個模組皆具備，才能實現三角形中心的荷魯斯之眼。

(三) 市場需求分析

此技術有許多的應用場景，只要固定的攝影機，就可以對一定範圍內的人做追蹤。以下以兩個應用場景為例：

1. 學校

學校可以預先建立員工（學生）的臉部資料庫，這樣可以辨識並區分學生、員工與訪客的人，對不同群體做對應得追蹤。在許多對外開放的校園中，可以捕捉可疑人士動態，並搜尋可疑人士出現的時間與哪個監視器。在防疫期間，可以輔助管制出

入的人。此系統可以減少保全人力並加強校園安全，可以利用此系統建立之資料庫制定改善校園內人流的政策，成為智慧校園的一部份。

2. 實體零售店

可以辨識會員與分會員顧客，透過本系統，藉由監視器蒐集各個會員或非會員的資料，並建立資料庫。藉此資料庫可以讓公司分析一段時間內的會員與非會員動態，將動態特徵相似的店分成一群，以此對不同群的店推播對應的促銷活動。再者，透過本系統所建立的會員與非會員資料庫可以作為之後建立推薦系統的資料庫。此系統可以作為智慧零售生態系的一部分。

相關參考資料

- 日本電氣(NEC) - NeoFace
  - 人物特徵搜尋系統
- 訊連科技(cyberlink) - FaceMe
  - FaceMe® Security 解決方案
  - 訊連科技FaceMe AI臉部辨識核心 – 智慧零售應用(video)
  - 應用場景
- 技嘉科技(gigabyte)
  - 智慧臉部辨識解決方案

二、相關工作

(一) 多物件追蹤(Multiple Object tracking)

Multiple Object Tracking，最早是在 1988 年加拿大科學家 Zenon Pylyshyn 在研究動態環境中持續視覺注意力的實驗中提出，發展到現在其實以景發展到一個飽和的狀態。但是，Wenhan Luo 等人在針對 MOT 的問題上做的文獻檢討中[1]，有提到幾點未來方向[1]，像是在多個攝像頭間運作、使用深度學習、與其他電腦視覺任務一起使用，也因為如此，本專題才會想結合物件追蹤、臉部辨識與 ReID 三種不同的電腦視覺任務去建立 Horus 系統。

1. 任務簡介

MOT 主要分成物件偵測 (Object detection) 以及追蹤器 (tracker) 兩個部分，其中物件偵測就是從每個 frame 中捕捉目標物件的位置，再去做分類，像是追蹤人它就屬於二元分類，物件偵測的部分只要判斷是否是人就好。而追蹤器指的是前後 frame 與目標 frame 之間關係的預測，也就是說當我們使用物件偵測方法捕捉到每個 frame 的人出來，我們要知道 frame 之間那些框出來的人是同個人。

[1] MOT with video adaptation、MOT under multiple cameras、Multiple 3D object tracking、MOT with scene understanding、MOT with deep learning、MOT with other computer vision tasks

2. 應用場景

在電腦視覺的任務中 MOT 屬於中階任務 (mid-level)，所以大部分是輔助高階任務 (high-level)，而高階任務 (high-level) 則是比較貼近實際應用場景的，其中像是姿態估計、動作偵測、行為分析。應用上像是視訊監控、人機互動或是 AR 等領域，都會使用到 MOT 的技術。

3. 方法介紹 - JDE

JDE 為了解決多個任務學習問題 ，將 object detect 和 appearance embedding 融合在同一個網路裡一起訓練，構成 one-stage 的訓練方式，相較於一般的 two-stage 方法，JDE 的速度會更快，且模型的架構會更單純。圖 1 為 JDE 方法與 一般的 two-stage 方法 做比較。

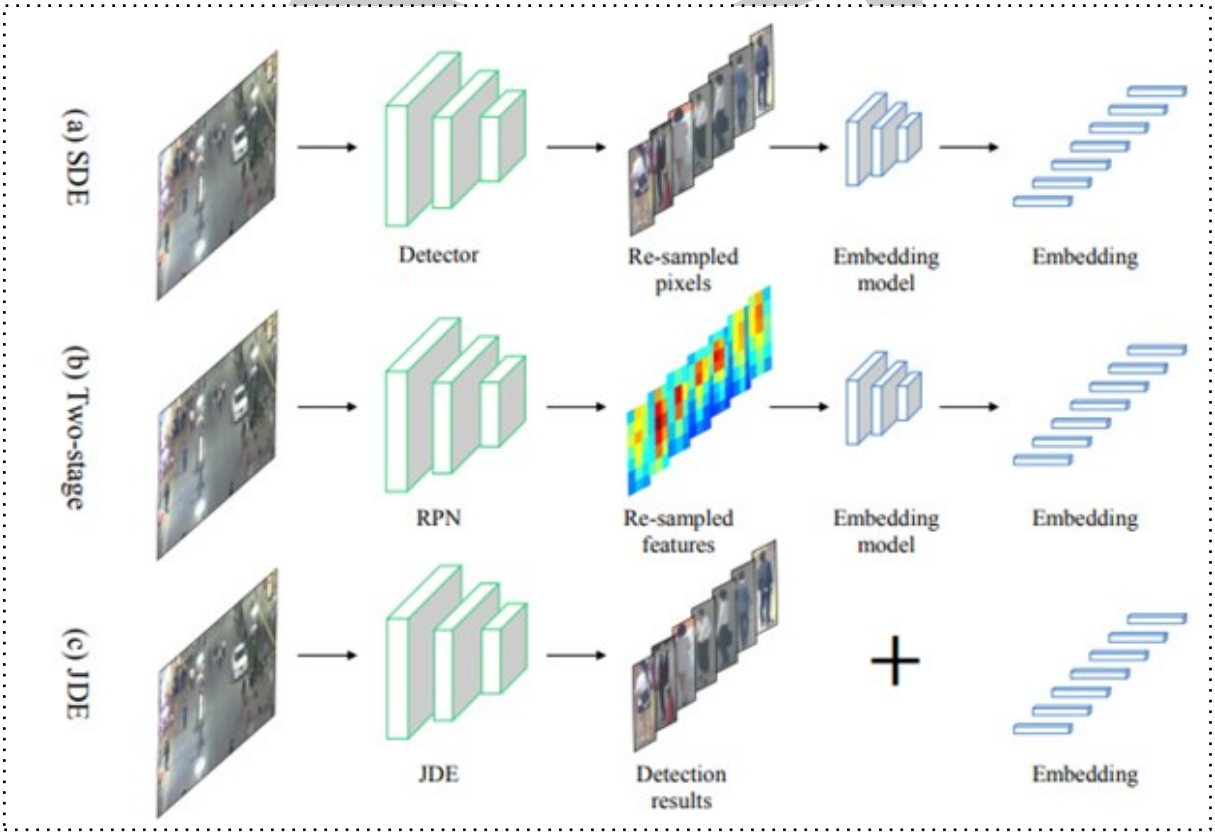


圖 2. JDE 方法與 SDE、two-stage 之模型架構比較

4. 相關難題 - 追蹤物件被遮蔽

相較於 MOT 的前身 SOT (Single Object Tracking)，MOT 多了確定目標的數量與目標之間互相干擾的因素，近幾年來也有許多方法已決此問題。其中，常被討論的問題是追蹤目標被遮蔽，或是當通個畫面中有兩個物件太接近甚至重疊時，會導致物件追蹤的結果會有交換 id 或是重複 id，交換 id 就是字面上的意思兩個 id 交換了，而重複 id 的就是一個人在同個畫面中出現第二個 id。圖2為使用 JDE [2] 測試此狀態的結果，可以看到其中紫色的框因在兩個人差肩而過後，被另一個人拉走了。

圖 3. 使用一般物件追蹤方法時兩人交錯問題示意圖

針對此問題其實已經有人提出解法，像 J. Peng 等人在 ECCV 2020 提出 Chained-Tracker (CTracker) [3]，如圖 3 可以看到 CTracker 使用軌跡紀錄去解決遮擋問題 ，但可惜的是官方沒有提供 pretrain model，因需要大量的運算效能才能訓練出模型，所以最後選擇使用有提供 pretrain model 的 JDE，在效率與準確度是上 JDE 也不會輸 CTracker 。

圖 4. Chained-Tracker 方法處理兩人交錯問題示意圖 (圖片來源)

5. 評估方式

關於 MOT 的評估方式如圖 4，其中箭頭向下代表越小效果越好，向上則越大效果越好。根據不同的評估指標有各自著重的點，大致可分成以下四類:

(引用至 MOT方法性能測評)

- 準確度 (Accuracy) :用於測量跟踪算法的準確程度。其中，IDs 統計 MOT 算法在不同目標切換的次數。MOTA 結合 false positive rate、false negative rate 和 mismatch rate，給出整體跟踪性能的相當合理的度量。
- 精度 (Precision) : MOTP (多目標跟踪精度)、TDE (跟踪距離誤差) 、OSPA 都被包含在內。這些描述了目標被精確跟踪的程度並通過邊界框重疊和 / 或邊界框間的距離度量。特別的，還有的考慮基數錯誤。

- 完整性 (Completeness) : 用於表示軌跡真值被跟踪的完整程度。MT、PT、ML、FM 都屬於此類。
- 魯棒性 (Robustness) : 用於評估 MOT 算法從遮擋中恢復出來的能力。分為 RS（從短期遮擋中恢復）和 RL（從長期遮擋中恢復）。

Metric	Description	Note
Recall	Ratio of correctly matched detections to ground-truth detections	↑
Precision	Ratio of correctly matched detections to total result detections	↑
FAF/FPPI	Number of false alarms per frame averaged over a sequence	↓
MODA	Combines missed detections and FAF	↑
MODP	Average overlap between true positives and ground truth	↑
MOTA	Combines false negatives, false positives and mismatch rate	↑
IDS	Number of times that a tracked trajectory changes its matched ground-truth identity (or vice versa)	↓
MOTP	Overlap between the estimated positions and the ground truth averaged over the matches	↑
TDE	Distance between the ground-truth annotation and the tracking result	↓
OSPA	Cardinality error and spatial distance between ground truth and the tracking results	↓
MT	Percentage of ground-truth trajectories which are covered by the tracker output for more than 80% of their length	↑
ML	Percentage of ground-truth trajectories which are covered by the tracker output for less than 20% of their length	↓
PT	1.0 - MT - ML	-
FM	Number of times that a ground-truth trajectory is interrupted in the tracking result	↓
RS	Ratio of tracks which are correctly recovered from short occlusion	↑
RL	Ratio of tracks which are correctly recovered from long occlusion	↑

圖 5. 物件追蹤的評價指標表(圖片來源)

相關參考資料

- 初探物件追蹤 Multiple Object Tracking(MOT)
- Object Detection and Tracking in 2020
- MOT方法性能测评
- 链式跟踪算法 CTracker（ECCV 2020 Spotlight）
- 多目標跟蹤綜述：Multiple Object Tracking: A Literature Review

(二) 臉部辨識 (Face recognition)

臉部識別已發展多年，精準度已經飽和，該著重的方向應是使用何種方法去解決特定的問題，下面將會針對此部分做講解。主要參考與臉部辨識有關的文獻檢閱 [4-6] ，系統部分我們參考 [7-9]。

1. 任務簡介

Face Recognition 模組之任務主要能分成三部分，分別是人臉處理流程 (Face Pipeline)、人臉資料建置 (Face Enrollment)、人臉辨識 (Face Recognition)。

a. 人臉處理流程 (Face Pipeline)
   - 定義對輸入圖片的預處理工作
   - 使用 Python 開源套件完成圖片預處理工作，從圖片中獲取人臉特徵資料

b. 人臉資料建置 (Face Enrollment)
   - 將會員照片經過人臉處理後得到的資料做儲存
   - 此工作將累積人臉特徵資料並產生人臉資料集

c. 人臉辨識 (Face Recognition)

- 透過計算人臉特徵，得到臉與臉之間的差異
- 與人臉資料及內的資料進行運算、比對，最後得到身分認證

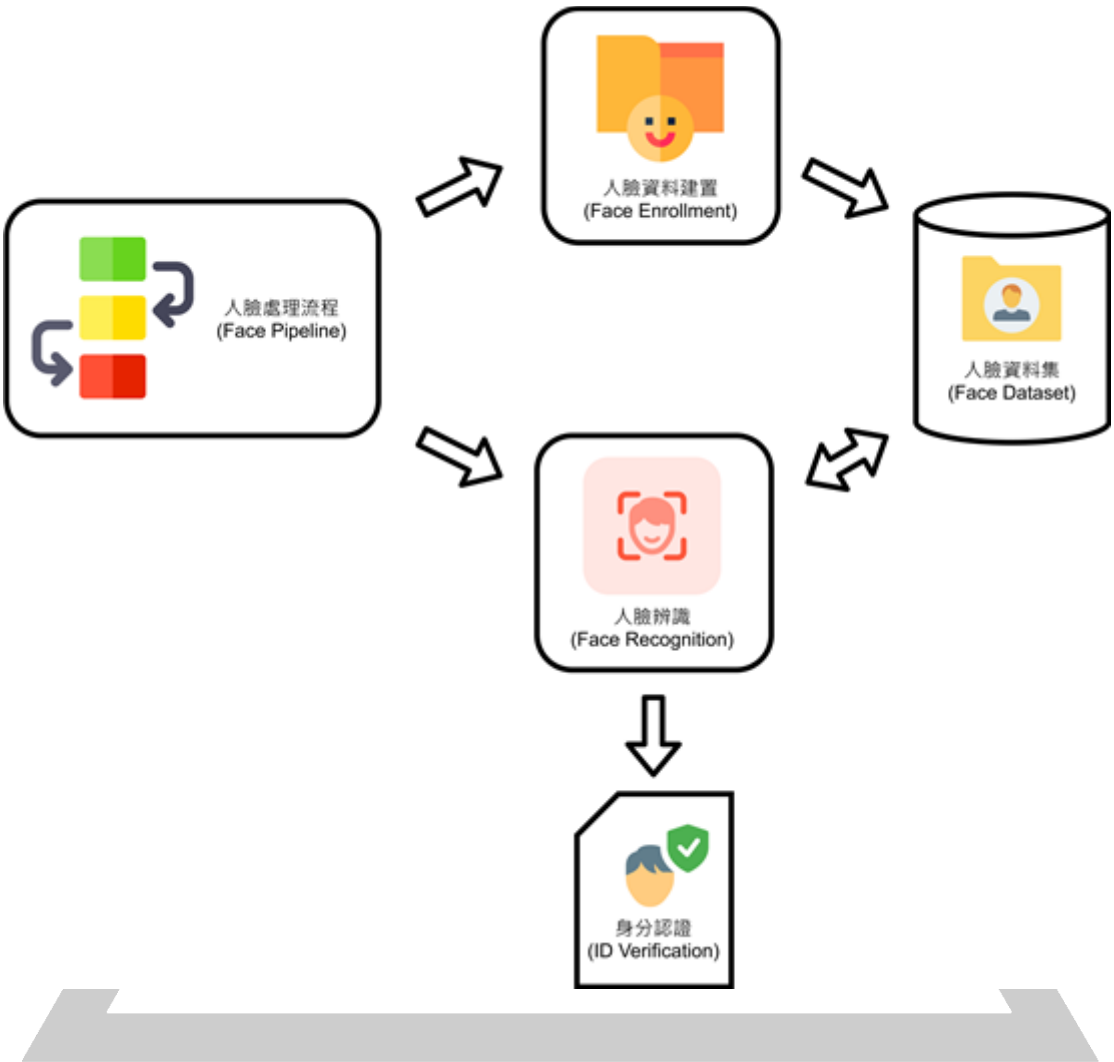


圖 6. 人臉辨識任務架構 (以 draw.io 繪製)

2. 應用場景

人臉辨識之應用範圍非常廣泛，生活中的不同場景都能發展出不同的應用，若以發展智慧企業為例，則可將人臉辨識技術應用在人臉辨識門禁、教育訓練報到、出

缺勤管理等等；若以住家安全或管理安全做為出發考量，則可將辨識技術應用在社區陌生人進出管理或是機場安檢等等。

3. 臉部辨識方法

a. 找出畫面中的人臉:

在 2000 年左右，Paul Viola 和 Michael Jones 發明了一種在廉價相機運行的人臉檢測方法，該方法為 Viola–Jones object detection framework[2]，這個方法在開源工具 - OpenCV 中被實現為cvHaarDetectObjects( )。然而現在更有效的人臉偵測方法是一種叫做 Histogram of Oriented Gradients[3] 的方法。將人臉照片顏色轉至灰階後，透過計算像素彼此之間的深度，即顏色的深淺差異，將差異由淺到深的變化用向量來表示，透過此動作，所有像素都會被向量箭頭所取代，最後人臉照片能化成一張佈滿向量的直方圖，若直方圖的向量分布情形與人臉輪廓相似，則能找出人臉的位置。

[2] Viola – Jones object detection framework: https://en.wikipedia.org/wiki/Viola%E2%80%93Jones_object_detection_framework

[3] Histogram of oriented gradients: https://en.wikipedia.org/wiki/Histogram_of_oriented_gradients

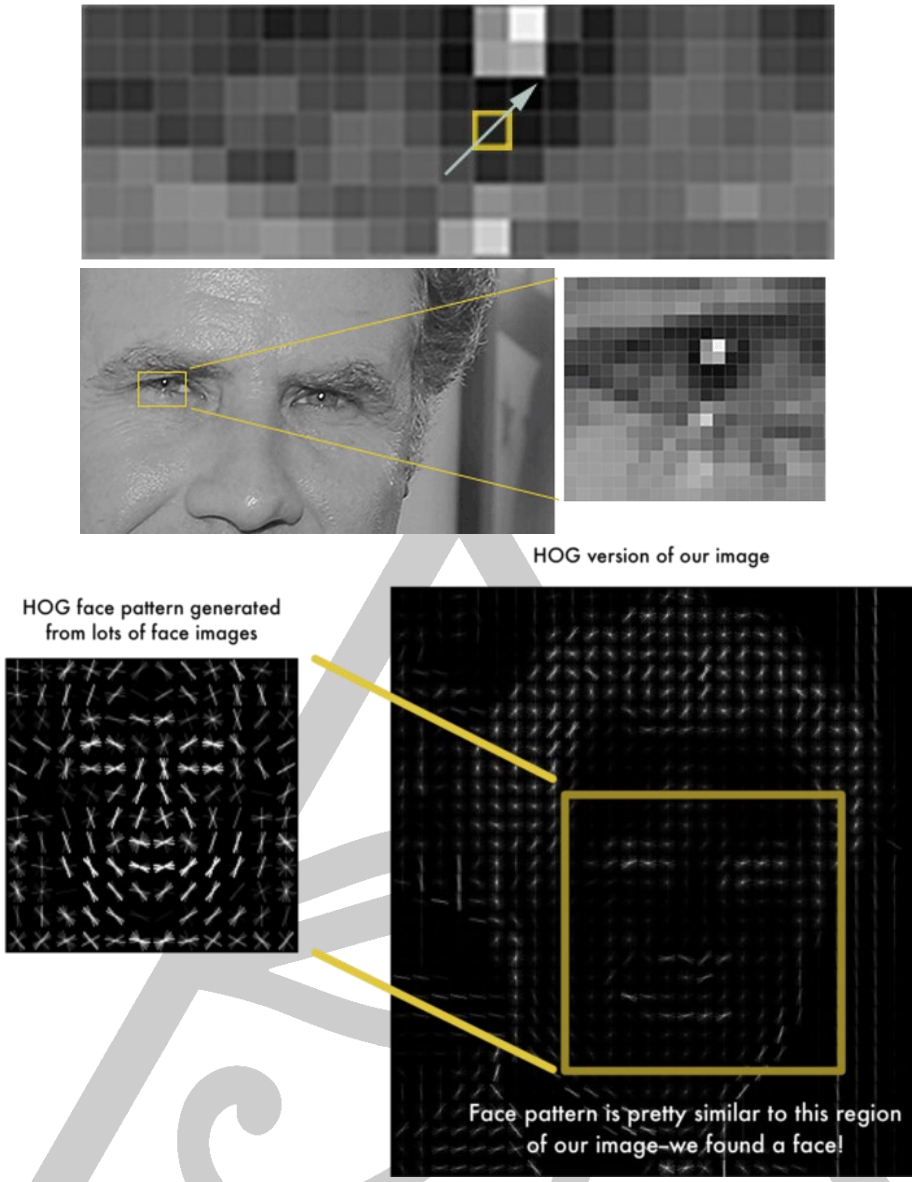


圖 7. Histogram of Oriented Gradients 人臉偵測概念圖 (圖片來源)

b. 識別不同角度的人臉

當人臉在三維空間中的角度不同，即便是出自同一個人的臉，對於電腦來說，每一張臉仍是不同的。為了解決這點，使用臉部特徵點估算 (Face Landmark Estimation) 是一種有效的方法。此方法的核心概念是找出 68 個人臉上普遍存在的特徵點，包括下巴、眼睛的外部輪廓、眉毛的內部轮廓等等，並搭配機器學習的方法訓練出一個能從任何一張人臉找出這些特徵點的模型。

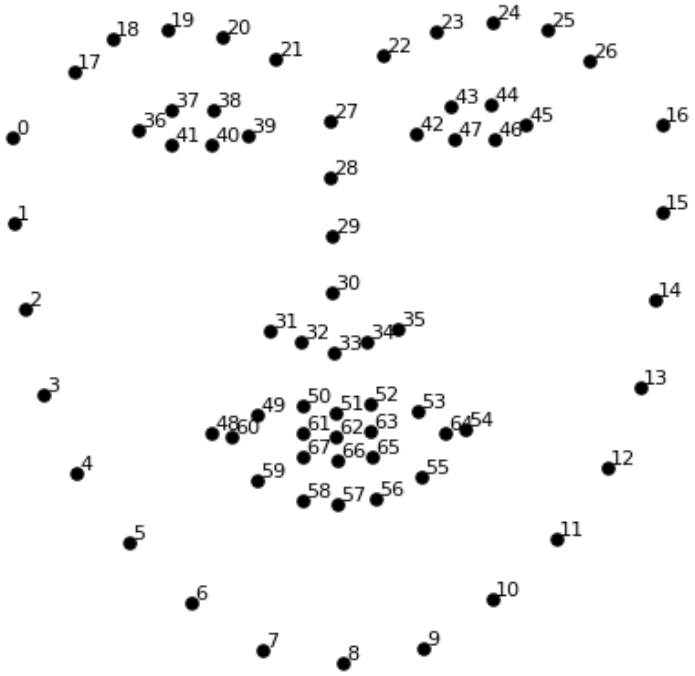


圖8. 人臉之 68 特徵點示意圖 (圖片來源)

c. 將人臉進行編碼

如何區分不同的人臉是人臉辨識的核心問題。最簡單的人臉辨識方法，是將未知的人臉照片，進行上述的臉部特徵點估算，得到面部特徵後，再與已標記過的人臉進行面部特徵運算比較，若特徵相似，則能預測兩張人臉應出自同一人。但是這樣的方法並非一個有效的方法。若要辨識人臉資料龐大，實務上是無法一一與每張標記過的人臉進行運算比較的。目前較好的方法是使用 Triplet Loss[4] 的概念去訓練卷基神經網路模型，此方法的概念並非訓練模型去辨識圖片中的物體，而是透過訓練，使模型有能力將臉部照片輸出成一個帶有 128 個特徵值的向量。訓練方法大致如下:

1. 讀取一張已標記的人臉照片
2. 加載另一張此人物的照片

[4] Triplet Loss: https://en.wikipedia.org/wiki/Triplet_loss

3. 加載另一張非此人物的照片
4. 依輸出之結果調整神經網路
5. 確保第一張和第二張產生的輸出值接近，而第二張和第三張產生的輸出值略有不同

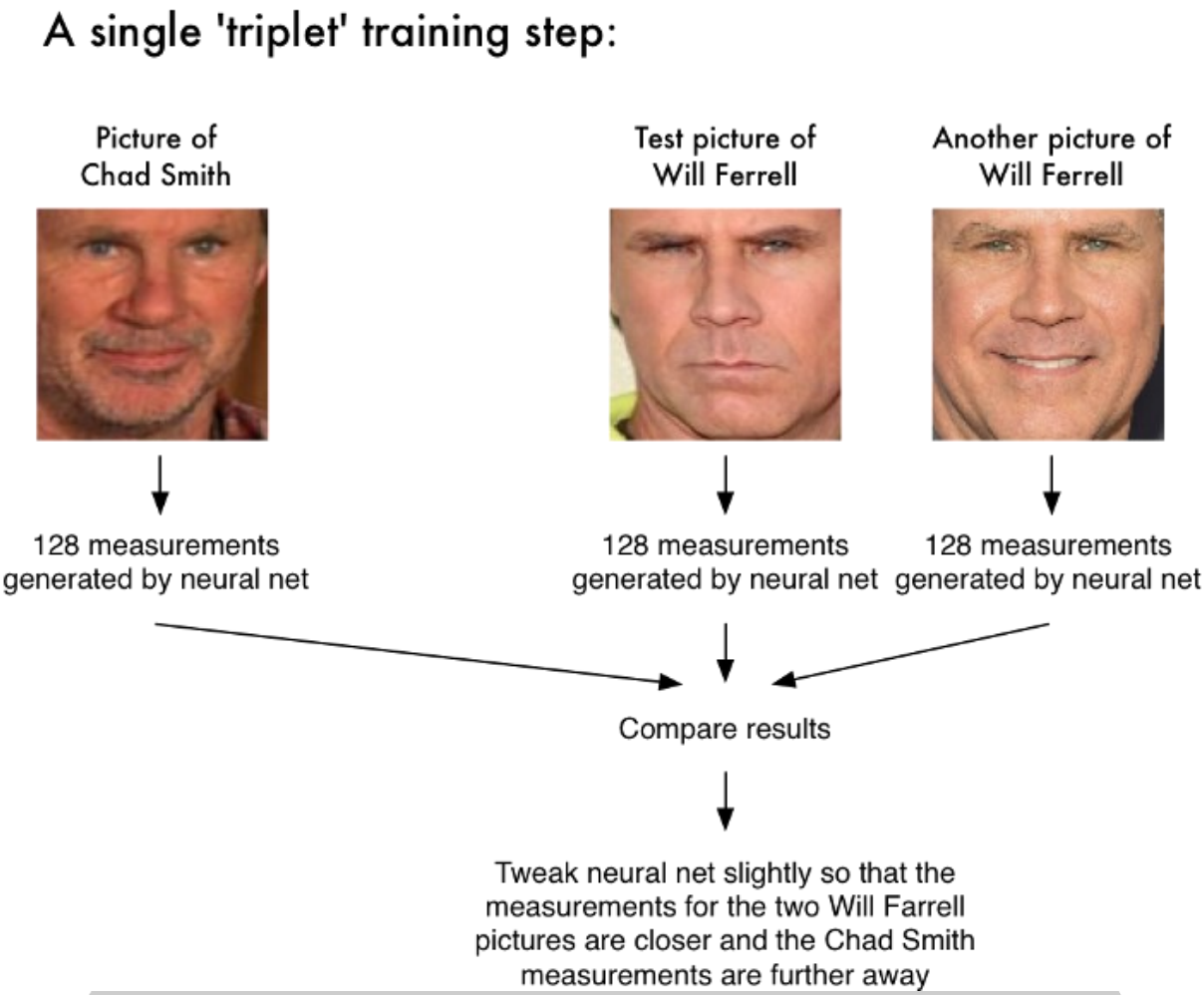


圖 9. Triplet Loss 訓練概念圖 (圖片來源)

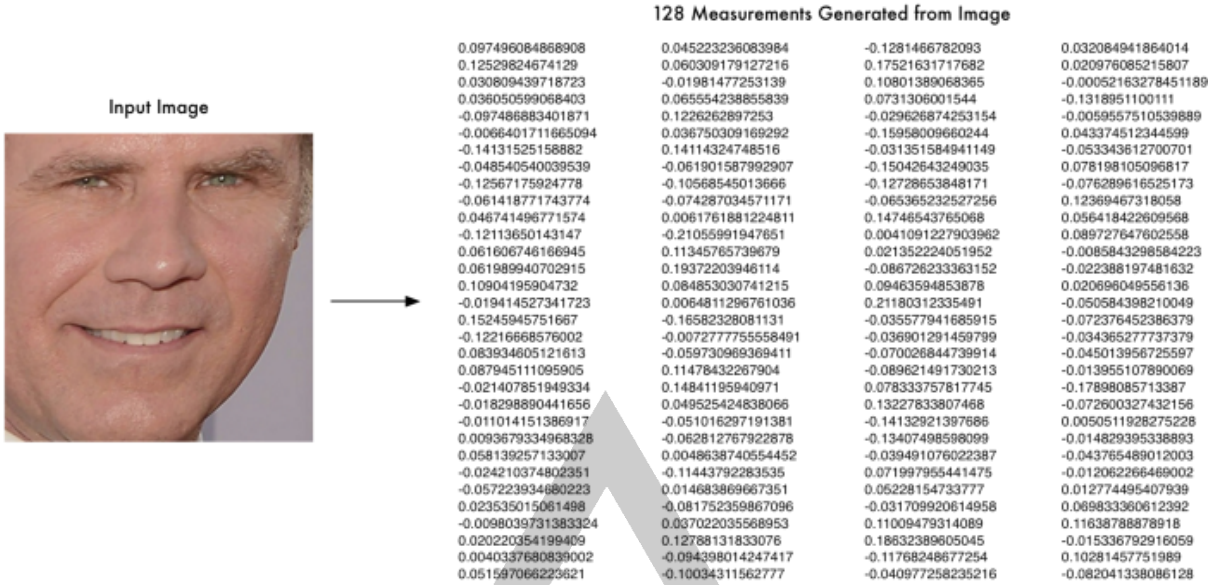


圖 10. 圖片編碼示意圖 (圖片來源)

透過臉部編碼，我們能夠得到將每張人臉轉化成 128 個特徵值，透過對這些特徵值的運算，能夠有效提升人臉辨識的效率與準確度。

4. 相關難題 - 人臉處理流程 Face Pipeline

所有的圖片資料都需經過人臉處理流程 (Face Pipeline) 才能進行人臉辨識，而人臉處理流程大致能分為以下四階段:

1. 人臉偵測 → ⊗ 人臉偵測模型無法偵測到照片中的人臉
2. 人臉裁切 → ⊗ 裁切後的人臉，人臉偵測模型無法偵測出畫面中的臉部座標
3. 人臉角度計算 → ⊗ 裁切後的人臉，人臉角度模型無法偵測出臉部特徵
4. 人臉編碼 → ⊗ 人臉編碼模型無法將圖片進行編碼

上述任一階段若失敗，該次辨識輸入之圖片就無法完成辨識。輸入圖片的像素大小、解析度、照片中的光線、人臉的清晰程度、人臉之角度，都有可能造成辨識過程失敗，造成人臉資料的流失。

5. 評估方式

Face Recognition 模組之評估方式主要為以下三指標:

- 人臉偵測準確率
- 人臉識別準確率
- 辨識花費時間

相關參考資料

- Facial recognition 2020 and beyond – trends and market
- Face Recognition for Beginners
- 人臉辨識在夯什麼！商機在哪裡
- Adam Geitgey - Machine Learning is Fun! Part 4: Modern Face Recognition with Deep Learning

(三) 行人重新識別(Person re-identification)

Person re-identification (ReID)，行人的重新識別。ReID 的問題在 CVPR 2006 第一次被 N. Gheissari 等人首次提出，作者提出使用人的整體外觀作為辨識的相同

行人圖片的方法[9]。也因為使用整體外觀做判斷依據的特性，所以本專案才會想導入 ReID 方法，去增加 Horus 系統的實用性。

1. 任務簡介

ReID 是影像檢索底下的子問題，目標是透過演算法搜尋圖片庫中目標圖片，此方法可以跨鏡頭的辨識同個人。下圖為 Zheng 等人所發表的論文 - Person Re-identification Past, Present and Future 中 reid 的架構圖[10]。作者有提到以目前的研究來看，ReID 模型主要是在後面部分的檢索問題，大多都是以預先標記好 id 的行人資料去訓練模型。還未有研究是結合偵測行人 (detected pedestrians) 和行人重新識別 (Person re-identification)。

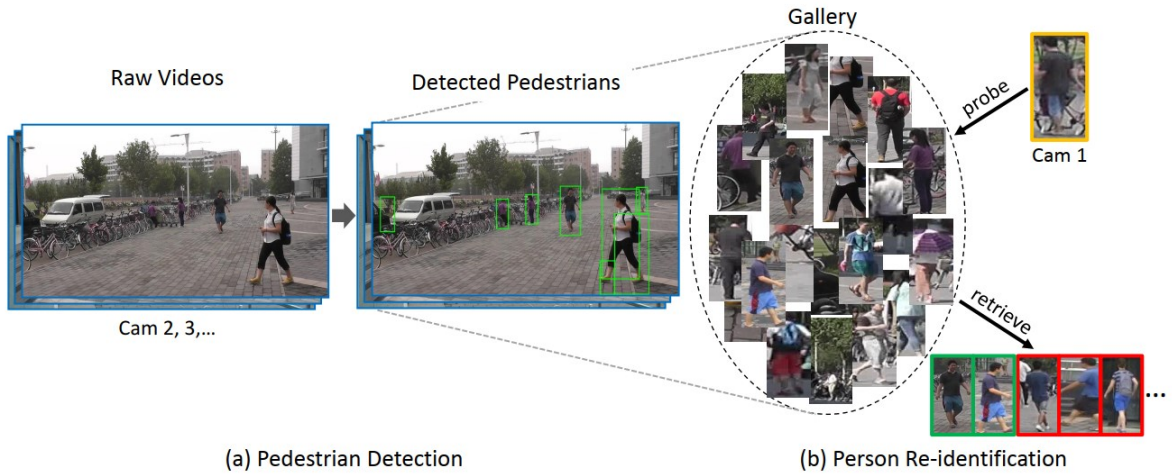


圖 11. ReID訓練架構圖 (圖片來源)

由圖 11 可以看到，右上單張的圖片是 probe (也稱作 query) 為我們要判斷的目標，而 gallery 就是我們的行人照片集，透過 ReID 模型可以從 probe 與 gallery 中分別提取兩者特徵並計算相似度。最後會根據相似度產生右下的排名，檢索出最可能

為 probe 的行人照片，最終模型好壞會依照排名內的行人圖片命中率去評估模型好壞。

2. 應用場景

在賣場、學校和城市規劃等實際場景下，大多數的影像都是不含人臉的，即使出現了人臉部分也因為距離關係極其模糊，這時強大的人臉辨識技術也束手無策。ReID 技術利用步態模式的捕捉、身體特徵等更為全面的信息來識別同個人，不僅可以彌補臉部的不足，在實際應用上可以與臉部辨識或是其他電腦視覺技術結合，更進一步的擴展其相關應用。

Mang Ye 等人在 2020 發表的『Deep Learning for Person Re-identification: A Survey and Outlook』[11]，這份關於 ReID 的文獻檢討中將 ReID 的任務分成 CLOSED-WORLD 和 OPEN-WORLD，基本上主要差在是否為 end-to-end，CLOSED-WORLD 指的是在已標記好的行人照片資料底下去做模型的訓練，注重在使用已標記資料加強部分任務的預測效果，並強調有足夠且單純的資料。而 OPEN-WORLD 是指從原始資料照片或影片到預測結果都包含，也就是 end-to-end的意思，強調無標記和非監督式、開放的環境、多元資料且有干擾的資料。兩者其實都已經有許多相關研究，雖然 OPEN-WORLD 比較貼近實際應用，但是在效果遠遠不及 CLOSED-WORLD，所以本專案是使用 CLOSED-WORLD 的方式訓練模型，並應用在OPEN-WORLD，且希望之後可以發展成自動從 OPEN-WORLD 的環境中篩選適合的照片自動標記，以此資料去 finetune ReID 模型，以此增強 Horus 系統在不同環境底下的掌握度，下圖為CLOSED-WORLD 和 OPEN-WORLD 的比較表。

TABLE 1: Closed-world *vs.* Open-world Person Re-ID.

Closed-world (Section 2)	Open-world (Section 3)
✓ Single-modality Data	Heterogeneous Data (§ 3.1)
✓ Bounding Boxes Generation	Raw Images/Videos (§ 3.2)
✓ Sufficient Annotated Data	Unavailable/Limited Labels (§ 3.3)
✓ Correct Annotation	Noisy Annotation (§ 3.4)
✓ Query Exists in Gallery	Open-set (§ 3.5)

圖 12. CLOSED-WORLD vs. OPEN-WORLD (圖片來源)

3. 系統建置

圖 13 為 Mang Ye 等人 提出的 ReID 實際執行的架構圖[11]，本系統也是參照此架構去設計的，他將整個過程分成以下五步:

1. Raw Data Collection
2. Bounding Box Generation
3. Training Data Annotation
4. Model Training
5. Pedestrian Retrieval

Fig. 1: The flow of designing a practical person Re-ID system, including five main steps: 1) *Raw Data Collection*, (2) *Bounding Box Generation*, 3) *Training Data Annotation*, 4) *Model Training* and 5) *Pedestrian Retrieval*.

圖 13. ReID 系統架構圖 (圖片來源)

4. 深度學習網路訓練方法

Khawar Islam 將 ReID 訓練網路與目標整理成圖 14 [12]，可以看到，其實 ReID 模型實際上在做的是基於一個 distance 演算法為基準去判斷 ReID 模型所抽取出來的特徵之間的距離，並透過訓練去 maximize 不同 id 照片之間的距離，和 minimize 同 id 之間的距離。可以看到圖左邊的原圖片特徵分布是散亂的，透過 ReID 模型可以將原圖片的特徵轉換，轉換後的特徵分布如圖右邊可以讓同 id 的人聚集在一起，這也是 ReID 模型訓練時所追求的目標。

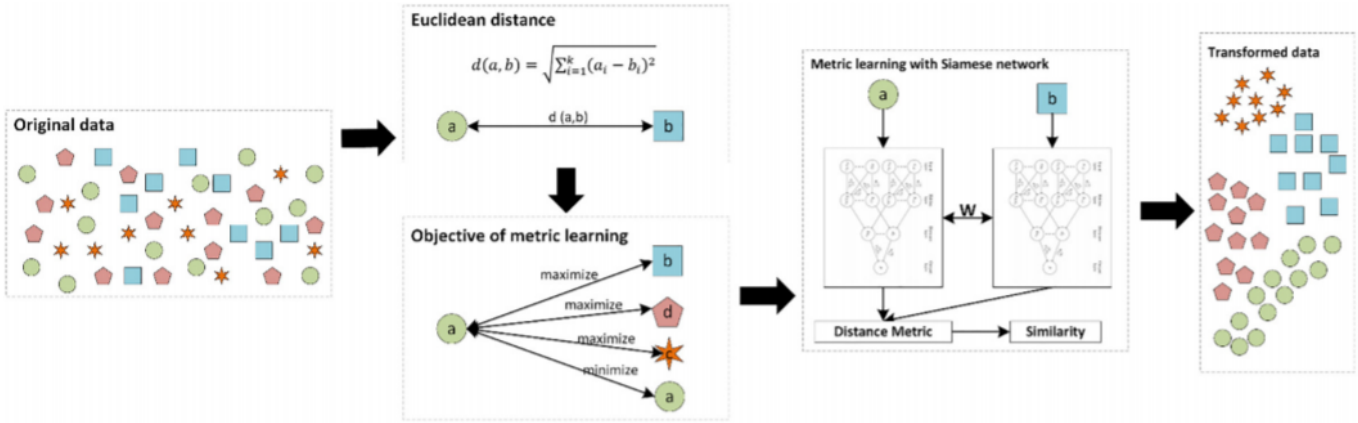


圖 14. ReID - Deep metric learning approach (圖片來源)

5. 相關難題 - 外在環境

在資料本質上，因為捕捉到的行人照片會有遠近的問題，所以捕捉到的畫質層差不齊。在行人的動作與姿態上，因本身的問題設定是針對行人的照片，所以動作缺乏連貫性，導致同一個人在不同的姿態下 (側面、正面、背面) 變化較大。

在外在環境上，像是陽光、背景或是遮擋物都屬於干擾，如何讓模型可以在排除干擾的情況下執行任務也是一大問題。這部分我們使用 fastreid [13] 的資料增強方法去訓練模型 (圖15)。

圖 15. fastreid 資料增強方法示意圖 (圖片來源)

6. 評估方式

在模型目前使用的主流評估結果的方法有 Top1 和 MAP，Top1 表示最終模型取出排名1的照片正確的概率，Top5 表示 1 - 5張圖片面至少有一張命中的比例，以此類推。至於 MAP 的部分其實是從 CMC 方法改善的，關於 MAP 的計算方式其實很多，這邊就使用其中一種作明，由圖 16 可以看到 MAP 的算法，MAP 是多次搜尋的 AP算數平均數。AP 則是單次搜尋的結果，由下圖可以看到第一次搜尋中(橘色)，目標是從 gallery 找出五個已標記和 probe 為同 id 的圖片，方法是分別看這五張的位置，計算自己與前幾名中有多少比例是對的，我們以排名 6 的圖片來看，它的準確率(precision)為 3/6，排名 6 之前有 3 個預測是對的，召回率(recall) 的部分則是 3/5，也就是說總共五個目標中已經找到 3 個的意思。這邊以 top10 為例，也可以依狀況調整。

但有個問題，當圖片 a 有 10 張相似的在 gallery 中，而圖片 b 結果有 100 張相似的在 gallery 中，模型在做預測時，在以 top 100 做計算時，圖a 最多只能從 100 張中答對 10 張，而圖 b 最多可以 100 張，就算使用 top 10，有 100 張相似照片的圖 b 成功預測的機率還是高很多。可以看到圖 17 為 Google Landmark Retrieva 競賽中使用的計算方式，用意是為了調整上述方式的不平衡，可以看到它多除了一個 min (標準答案數量, 100) 。

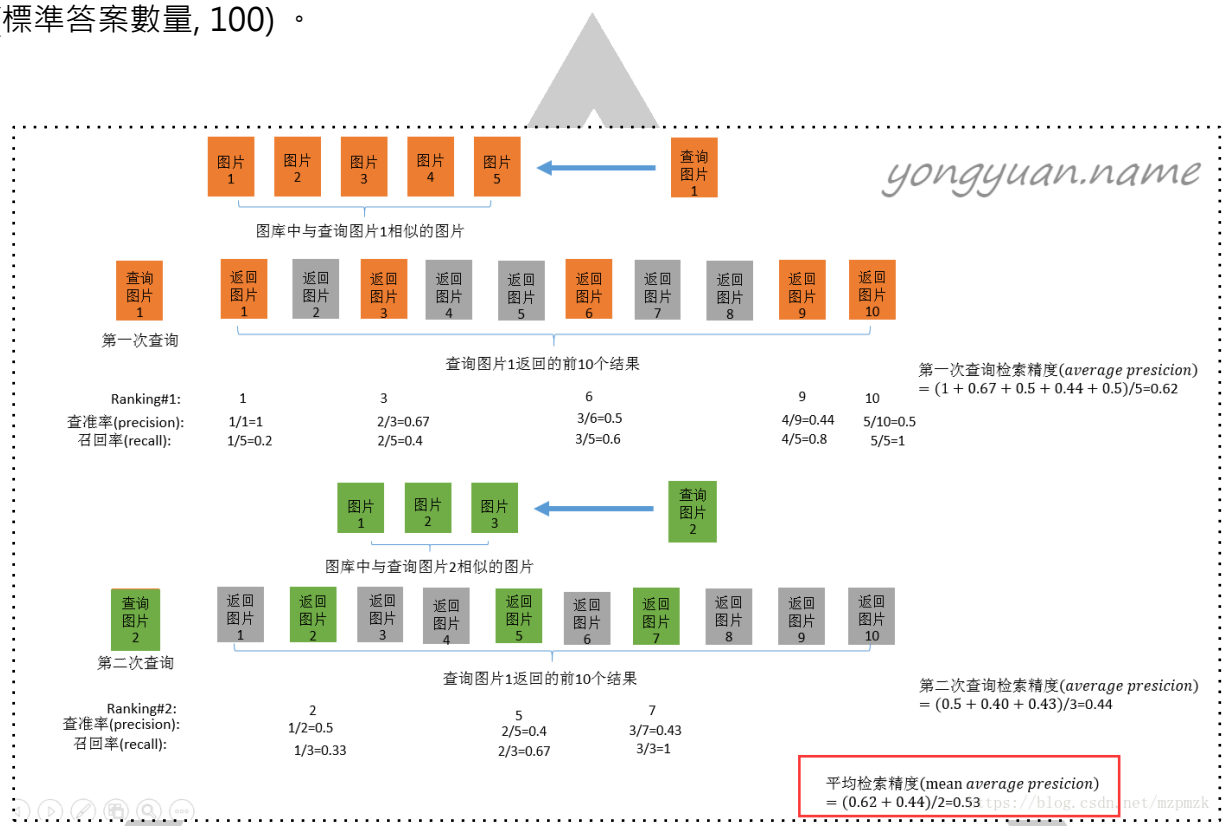


圖 16. mAP 評估結果示意圖 (圖片來源)

Submissions are evaluated according to mean Average Precision @ 100 ($mAP@100$):

$$mAP@100 = \frac{1}{Q}\sum_{q=1}^{Q}\frac{1}{min(m_q, 100)}\sum_{k=1}^{min(n_q,100)}P_q(k)rel_q(k)$$

where:

- Q is the number of query images
- m_q is the number of index images containing a landmark in common with the query image q (note that $m_q > 0$)
- n_q is the number of predictions made by the system for query q
- $P_q(k)$ is the precision at rank k for the q-th query
- $rel_q(k)$ denotes the relevance of prediciton k for the q-th query: it's 1 if the k-th prediction is correct, and 0 otherwise

圖 17. Google Landmark Retrieval 的mAP 評估方式 (圖片來源)

相關參考資料

- 行人重識別簡介（Person ReID）
- 小白入门系列——ReID(一)：什么是ReID？如何做ReID？ReID数据集？ReID评测指标？
- Person Re-Identification（ReID行人重识别）

三、實作方法：

(一) 工具使用

1. OpenCV (github，docs)

OpenCV[5]的全稱是 Open Source Computer Vision Library，是一個跨平台的電腦視覺庫。OpenCV 可用於開發即時的圖像處理、電腦視覺以及圖型識別程式。該程式庫也可以使用英特爾公司的 IPP 進行加速處理。(引用自維基百科)

本專案使用 opencv 從影片中切出每一個 frame ，並且對照片做預處理。 其中像是行人照片會因捕捉時的遠近而有大小和畫質差別，這邊可以透過 opencv 的 resize 方法去調整輸入的照片大小，這樣才能順利輸入模型(需要固定的長寬)。

2. JDE (github，parper)

Wang Zhongdao 等人所提出的的方法 - Joint Detection and Embedding (JDE) model [2] ，code 部分以 MIT License 的方式公開在 github。JDE 是一種快速 (接近 22～38 FPS)，高性能的多目標追蹤方法。物件檢測 (object detection) 和外觀嵌入 (appearance embedding) 兩個任務在共享的神經網絡中同時學習。技術細節可以查看 ECCV 2020 的 parper (引用至原 github)。

[5] OpenCV: https://zh.wikipedia.org/zh-tw/OpenCV

本專案使用 JDE 的開源 code 實現多目標追蹤的部分，並將程式碼修改成能儲存每個 frame 所捕捉的行人照片。

3. FastReID (github，parper)

FastReID[13] 是一個專為 ReID 任務設計的 python 工具，整體是使用 Pytroch 開發，並以 Apache 2.0 license 方式公開在 github 上。由圖 18 可以看到，fastreid 的架構完整，且是在 2020 發表的，這也是我們會選用 fastreid 的原因。fastreid 大致上分成訓練 (training) 和推論 (inference) 兩部分，其中包含前處理(pre-processing)、基底模型 (backbone)、集合 (Aggregation)、端口 (head)、損失 (loss) 和訓練策略等部分。

基本上本專案只使用前面訓練的部分，並利用以訓練好的模型實際應用在本專案中。

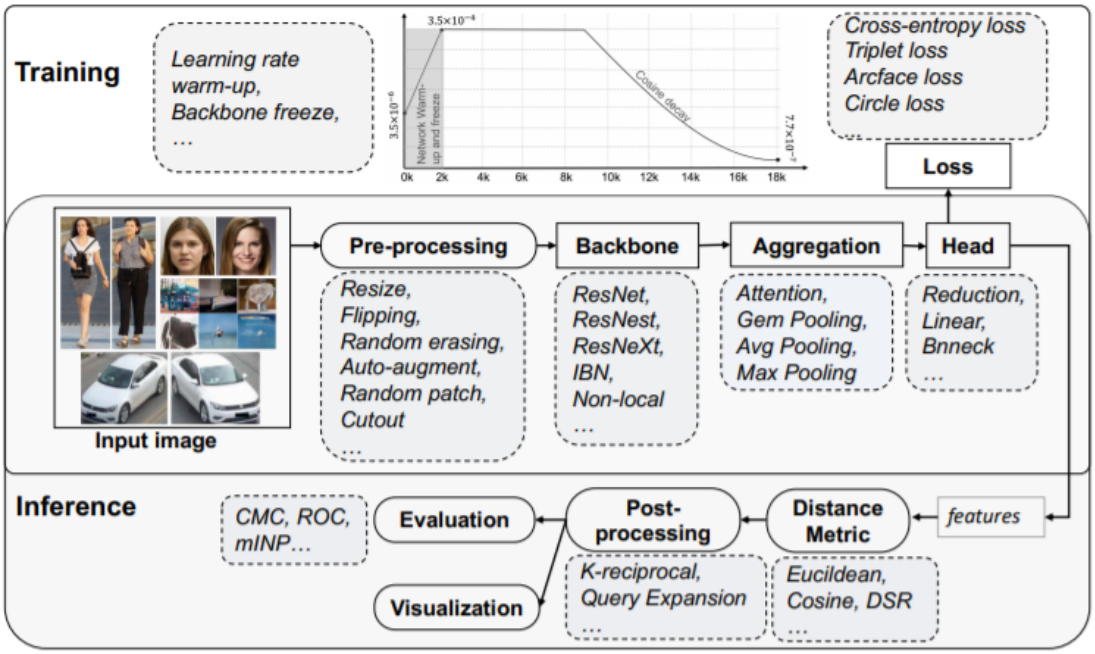


Figure 1. The Pipeline of FastReID library.

圖 18. fastreid 架構圖 (圖片來源)

4. Dlib (github，官網)

Dlib4[6] 是以 C ++ 語言撰寫的跨平台函式庫，開發者為 Davis E. King，其應用領域包括機器學習、機器人學、嵌入式裝置、行動裝置等等，其高效之運算能力使 Dlib 在學術界與業界被廣為使用。在人臉辨識的應用上，其提供的臉部特徵模型，能對臉部的身分識別起到極大之效用。

5. 3DDFA (github，paper)

3DDFA 是一以 MIT License 形式公開在 GitHub 的開源專案，以論文 - TPAMI 2017 - Face Alignment in Full Pose Range: A 3D Total Solution[15] 中的方法為基

[6] Dlib: https://en.wikipedia.org/wiki/Dlib

礎，使用 PyTorch 加以優化的專案，此專案提供的 Face Embedding Model 能將臉部圖片輸出成 62 維之陣列，透過對此陣列的運算，得到圖片中人臉在三維空間的角度。

6. face_recognition (github，API 文件)

face_recognition 宣稱是史上最強大，最簡單上手的人臉辨識專案，簡單之處在於使用者可以直接使用 Python 和命令列工具提取、識別、操作人臉。此專案由軟體工程開發師和諮詢師 Adam Geitgey 開發，其強大之處在於不僅使用了高效能的 C++ 開源庫 Dlib 中的深度學習模型，同時兼容樹莓派系統 (Raspberry Pi OS)。face_recognition 使用的人臉資料集是由美國麻省大學安姆斯特分校製作的 Labeled Faces in the Wild4[7]，此資料集含多達 13,000 張臉部影象，而 face_recognition 的辨識準確率高達 99.38%。

7. Docker (官網，官方文件)

我們使用 Docker 實作本系統之資料庫建置與部屬。Docker 是一個開源軟體，是一個開放平台，用於開發應用、交付 (shipping) 應用、執行應用。 Docker 允許使用者將基礎設施 (Infrastructure) 中的應用單獨分割出來，形成更小的顆粒 (容器)，從而提高交付軟體的速度。

Docker 容器與虛擬機器類似，但二者在原理上不同。容器是將作業系統層虛擬化，虛擬機器則是虛擬化硬體，因此容器更具有可攜式性、高效地利用伺服器。 容器更多的用於表示 軟體的一個標準化單元。由於容器的標準化，因此它可以無視基礎設

[7] Labeled Faces in the Wild: http://vis-www.cs.umass.edu/lfw/

施 (Infrastructure) 的差異，部署到任何一個地方。另外，Docker 也為容器提供更強的業界的隔離相容。

Docker 利用Linux核心中的資源分離機制，例如: cgroups，以及Linux核心命名空間(namespaces)，來建立獨立的容器 (containers)。這可以在單一 Linux 實體下運作，避免啟動一個虛擬機器造成的額外負擔。Linux 核心對命名空間的支援完全隔離了工作環境中應用程式的視野，包括行程樹、網路、使用者 ID 與掛載檔案系統，而核心的 cgroup 提供資源隔離，包括 CPU、記憶體、block I/O 與網路。從 0.9 版本起，Dockers 在使用抽象虛擬是經由 libvirt 的 LXC 與 systemd - nspawn 提供介面的基礎上，開始包括 libcontainer 函式庫做為以自己的方式開始直接使用由 Linux 核心提供的虛擬化的設施，

依據行業分析公司「451 研究」：「 Dockers 是有能力打包應用程式及其虛擬容器，可以在任何 Linux 伺服器上執行的依賴性工具，這有助於實現靈活性和可攜式性，應用程式在任何地方都可以執行，無論是公用雲端伺服器、私有雲端伺服器、單機等。」(引用自維基百科)

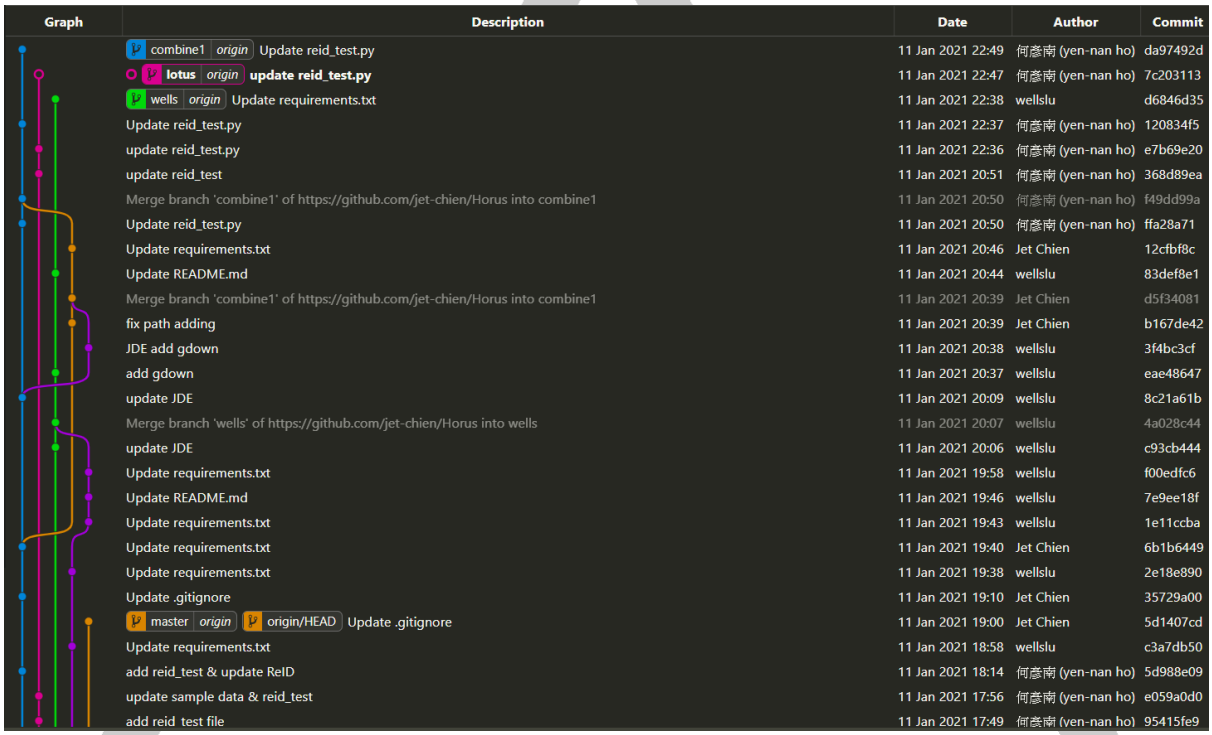
8. GitHub (github)

GitHub[8] 是透過 Git 進行版本控制的軟體原始碼代管服務平台。GitHub 同時提供付費帳戶和免費帳戶。這兩種帳戶都可以建立公開或私有的代碼倉庫，但付費使用者支援更多功能。根據在 2009 年的 Git 使用者調查，GitHub 是最流行的 Git 存取站點。除了允許個人和組織建立和存取保管中的代碼以外，它也提供了一些方便社會化共同軟體開發的功能，即一般人口中的社群功能，包括允許使用者追蹤其他使用者、

[8] GitHub: https://zh.wikipedia.org/zh-tw/GitHub

組織、軟體庫的動態，對軟體代碼的改動和 bug 提出評論等。GitHub 也提供了圖表功能，用於概觀顯示開發者們怎樣在代碼庫上工作以及軟體的開發活躍程度。(引用至維基百科)

本專案因為需要三個人協作，所以選擇使用 github 作為協作平台，三人分別建置一個自己獨立的 branch，開發各自負責的模組，開發完成後再 merge 在一起。圖 19 使用 VS code 的 git graph[9] 繪製出的 git 圖案，可以看到三人協作的紀錄。



Graph	Description	Date	Author	Commit
	combine1 origin Update reid_test.py	11 Jan 2021 22:49	何彥南 (yen-nan ho)	da97492d
	lotus origin **update reid_test.py**	11 Jan 2021 22:47	何彥南 (yen-nan ho)	7c203113
	wells origin Update requirements.txt	11 Jan 2021 22:38	wellslu	d6846d35
	Update reid_test.py	11 Jan 2021 22:37	何彥南 (yen-nan ho)	120834f5
	update reid_test.py	11 Jan 2021 22:36	何彥南 (yen-nan ho)	e7b69e20
	update reid_test	11 Jan 2021 20:51	何彥南 (yen-nan ho)	368d89ea
	Merge branch 'combine1' of https://github.com/jet-chien/Horus into combine1	11 Jan 2021 20:50	何彥南 (yen-nan ho)	f49dd99a
	Update reid_test.py	11 Jan 2021 20:50	何彥南 (yen-nan ho)	ffa28a71
	Update requirements.txt	11 Jan 2021 20:46	Jet Chien	12cfbf8c
	Update README.md	11 Jan 2021 20:44	wellslu	83def8e1
	Merge branch 'combine1' of https://github.com/jet-chien/Horus into combine1	11 Jan 2021 20:39	Jet Chien	d5f34081
	fix path adding	11 Jan 2021 20:39	Jet Chien	b167de42
	JDE add gdown	11 Jan 2021 20:38	wellslu	3f4bc3cf
	add gdown	11 Jan 2021 20:37	wellslu	eae48647
	update JDE	11 Jan 2021 20:09	wellslu	8c21a61b
	Merge branch 'wells' of https://github.com/jet-chien/Horus into wells	11 Jan 2021 20:07	wellslu	4a028c44
	update JDE	11 Jan 2021 20:06	wellslu	c93cb444
	Update requirements.txt	11 Jan 2021 19:58	wellslu	f00edfc6
	Update README.md	11 Jan 2021 19:46	wellslu	7e9ee18f
	Update requirements.txt	11 Jan 2021 19:43	wellslu	1e11ccba
	Update requirements.txt	11 Jan 2021 19:40	Jet Chien	6b1b6449
	Update requirements.txt	11 Jan 2021 19:38	wellslu	2e18e890
	Update .gitignore	11 Jan 2021 19:10	Jet Chien	35729a00
	master origin origin/HEAD Update .gitignore	11 Jan 2021 19:00	Jet Chien	5d1407cd
	Update requirements.txt	11 Jan 2021 18:58	wellslu	c3a7db50
	add reid_test & update ReID	11 Jan 2021 18:14	何彥南 (yen-nan ho)	5d988e09
	update sample data & reid_test	11 Jan 2021 17:56	何彥南 (yen-nan ho)	e059a0d0
	add reid_test file	11 Jan 2021 17:49	何彥南 (yen-nan ho)	95415fe9

圖 19. fastreid 架構圖 (圖片來源)

[9] git graph: https://marketplace.visualstudio.com/items?itemName=mhutchie.git-graph

(二) 測試影片

我們會設定幾個場景並在不同的情境下錄製 20 秒內的短片，表 1 為各影片的詳細資訊，除了主要測試影片 (base)，還另外錄製了 7 部短片，其中共有以下四種情境 (v1~v4)：

1. 兩人擦肩而過
2. 兩人在樹後交叉再往鏡頭走近
3. 被樹遮住後再出現
4. 走出鏡頭後偽裝再回鏡頭

	進出畫面	障礙物	擦肩而過	臉的大小	臉部遮蔽	人數	長度	描述
base		✓		小~大		>2	12s	大爲與彥南混進行人一起走過
v1-1			✓	中		2	7s	大爲在前面擦肩而過
v1-2			✓	中		2	7s	彥南在前面擦肩而過
v2-1		✓		小~中		2	10s	兩人在樹後交叉再往鏡頭走近
v2-2		✓		中~大		2	8s	兩人在樹後交叉再往鏡頭走近
v3		✓		小		1	9s	遠處走過被樹遮住後出現
v4	✓			中	✓	1	18s	走出鏡頭後偽裝再回鏡頭

表1. 測試影片介紹 (影片可以至 google drive 查看)

(三) 系統架構

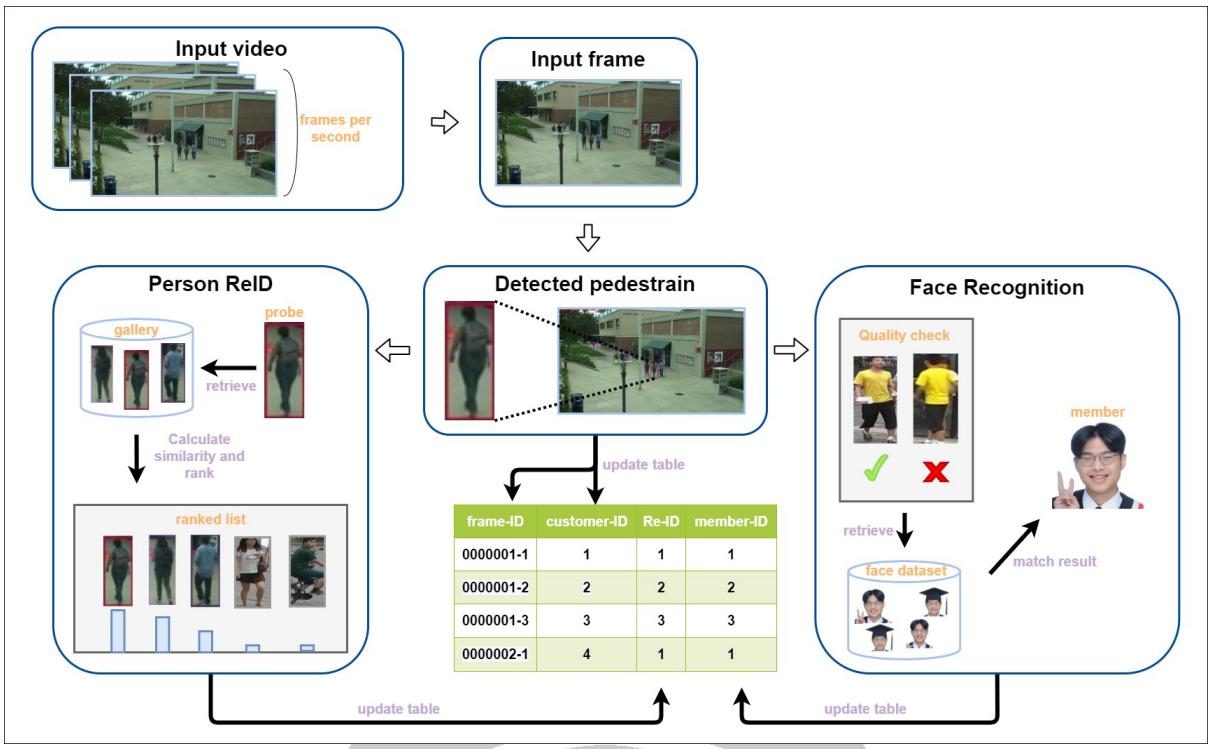


圖 20. 系統架構圖 (使用 draw.io 繪製)

由圖 20 可以看到，本系統模型的部分主要由以下三個部分構成:

1. 物件追蹤技術偵測並儲存行人照片

此部分使用的是多物件追蹤技術 (MOT)，負責偵測並、擷取和儲存行人圖片，每一偵在此階段會新增一列資料在圖片紀錄表，並給予 frame-ID 與 track-ID。

- **frame-ID:** 由影片擷取的畫面編號。
- **customer-ID (cid):** 物件追蹤所捕捉到的同一個連續物件。

2. 臉部辨識判定是否為會員

臉部辨識結果為會員資料表中的 member-ID 資料。臉部辨識模組將讀取儲存的 frame 並畫面中的人臉進行辨識，擷取未知之臉部特徵與預先建立好的會員臉部特徵資料庫進行比對，從超過認證門檻值中相似度最高的 member-ID 作為辨識結果 。

- **member-ID(mid)**: 對應的員工編號，此欄位會有三種狀態:
    - ○ NA: 尚未檢索的照片。
    - ○ -1 : 檢索後無達到門檻值的會員。
    - ○ member-ID: 所配對到的 member-ID。

3. 使用 ReID 模型計算並找出相同外觀的行人照片

對行人重新識別的編號。每天的 ID 和 gallery 都會重製，因為基本上 ReID 是捕捉整體特徵，所以衣服影響很大。大多人隔天就會換衣服的。方法是將新進來的人圖片作為 probe (搜尋的照片)，將其 gallery 裡的每個人的圖片資料夾中抽取部分照片算相似度，並排名，如果第一名的值超過門檻，則取當第一名的照片與 probe 為新的 Re-ID。

- **Re-ID (reid) :** 辨識同一天中身體特徵相似的照片，以此作為串接多個 customer-ID 的依據。
    - ○ NA: 尚未檢索的照片。
    - ○ -1 : 檢索後無達到門檻值的會員。

○ Re-ID: 所配對到的 customer-ID。

(四) 資料庫架構

1. customer table

此表為本系統主要的資料表，也是三個模組主要更新的地方，是以物件追蹤的每一段追蹤為一筆資料，只要畫面中出現新的人，就會新增一筆追蹤。下表為欄位資訊:

欄位名稱	類型	資料描述
id	int	
cid	int	追蹤編號，而每段追蹤都會有一個獨立的追蹤編號。
last_cid	int	上次追蹤編號，若遇到離開畫面或是被障礙物遮擋後被強行更換cid，將進行ReID並填上過去的cid。
mid	int	會員ID，與資料表member串連的key值，偵測出畫面中的人物是否為會員，若是則填上該member的ID。
customer_img	str	影像存放的位置，存取該cid擷下範圍中的影像到

		該相對位置。
enter_time	int	出現在畫面中的時間點，測試時為影片因此以第幾幀代替。
leave_time	int	離開畫面時的時間點，測試時為影片因此以第幾幀代替。
created_at	timestamp	此筆資料創建的時間。
updated_at	timestamp	此筆資料上次更動時間。

表2. customer table 欄位介紹

2. member table

會員資料表是儲存已經登入會員的資料表，我可以從此表知道會員id與此會員的大頭貼儲存位置。此資料表主要是提供給人臉辨識模組使用。下表為相關的欄位資訊:

欄位名稱	類型	資料描述
id	int	

mid	int	會員ID，與資料表member串連的key值，偵測出畫面中的人物是否為會員，若是則填上該member的ID。
face_img	str	會員照片存放相對位置
created_at	timestamp	此筆資料創建的時間
updated_at	timestamp	此筆資料上次更動時間

表3. member table 欄位介紹

3. 資料表更新邏輯與流程控制

horus 為一個龐大的系統，而且是以 real-time 為目標，所以為確保三個模組之間可以順利互動、避免衝突，進而造成不必要的成本消耗。本系統有預先設計控制流程，這部分包含三個模組的啟動、暫停和更新邏輯。架構如下圖所示:

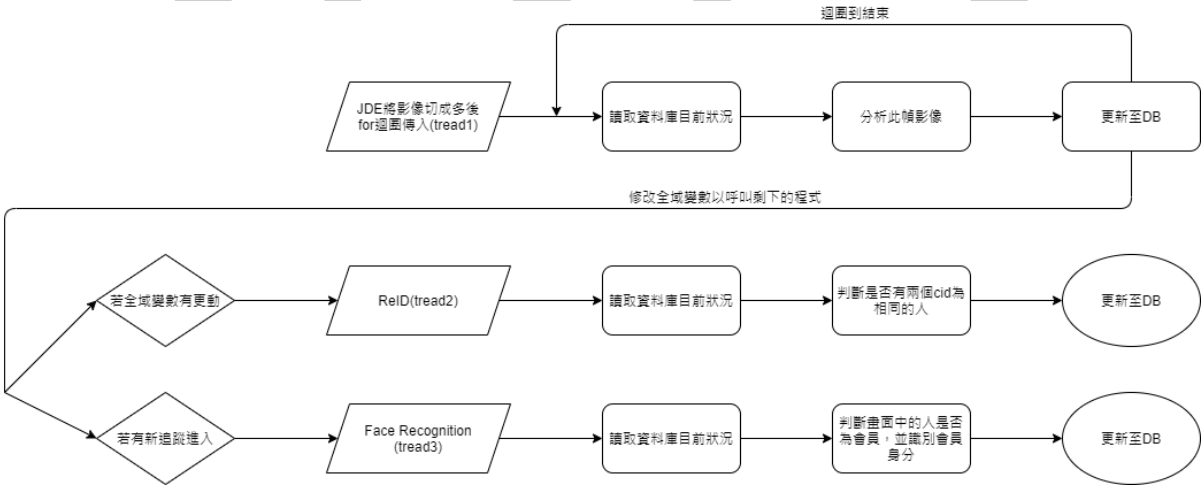


圖 21. 流程控制圖 (使用 draw.io 繪製)

(五) 使用 JDE 實現行人追蹤，並捕捉行人圖片

多目標追蹤是我們這次專題最主要的核心，物件追蹤所捕捉的行人圖片為其他機制所輸入的資料來源，而物件追蹤的 **track-ID** 為我們最基礎的行人辨識層級。所以在這部分物件追蹤最主要的任務是追蹤畫面中出現的人物，並且定時去擷取行人圖片，為確保為同個人的第一道門檻。

而在這一部分實作方面我們所採用的是在 2019 年推出的 MOT 算法 JDE (Jointly Detector and Embedding model)，實際實現後的範例如下圖，這是某一部範例影片的某兩幀 (如圖 22 和圖 23)，從圖中我們可以發現在不同幀的圖片上同一人給予的框框顏色與 track-ID 仍是相符的，這就表式當影片在進行時程式仍然能自動辨別多個物體。要做到能辨識行人且持續追蹤各個物體的第一步是得先找出這一幀圖片中所有的人並框出，再來第二步就是我們得分辨得出前後幾幀各個相同的行人進行track-ID的配對，這樣才算是達成我們多目標追蹤的目的，不是只是找出這張圖片上有多少人。針對必須連續追蹤相同類型但不同個體的問題，這邊是基於從前幾幀連續圖片去判斷此框中人物移動的方向，進而預測此幀中哪一個框會是屬於此人移動到的位置。

如上一段所陳述的，此類型 SDE 的實現大多都需要進行兩階段的辨識，造成速度變慢的情況，然而 JDE 採用的是一階段辨識，將辨識行人、框的位置、辨識 track-ID 等工作一次性地完成，也因此 JDE 比以往的 SDE 解法都來的快速，但經過我們實測後這可能也引申了幾個無法單獨改善的問題，一、有些行人在畫面中沒有離開過，但

可能因為過小或被擋住太久等問題給予新的track-ID；二、當兩人要交錯而過時，可能出現 track-ID 被對調或跟錯人等問題，由於 JDE 沒有像以往的 SDE 都有單獨做 ReID 的步驟，導致會有些微的錯誤。而此次我們的計劃就是為了達成使速度不減的情況下提升精度這一目的，而構思出 Horus 系統，在事後加上 ReID 與 Face Recognition 的交叉判斷，讓原本出錯的地方得以改正回來。

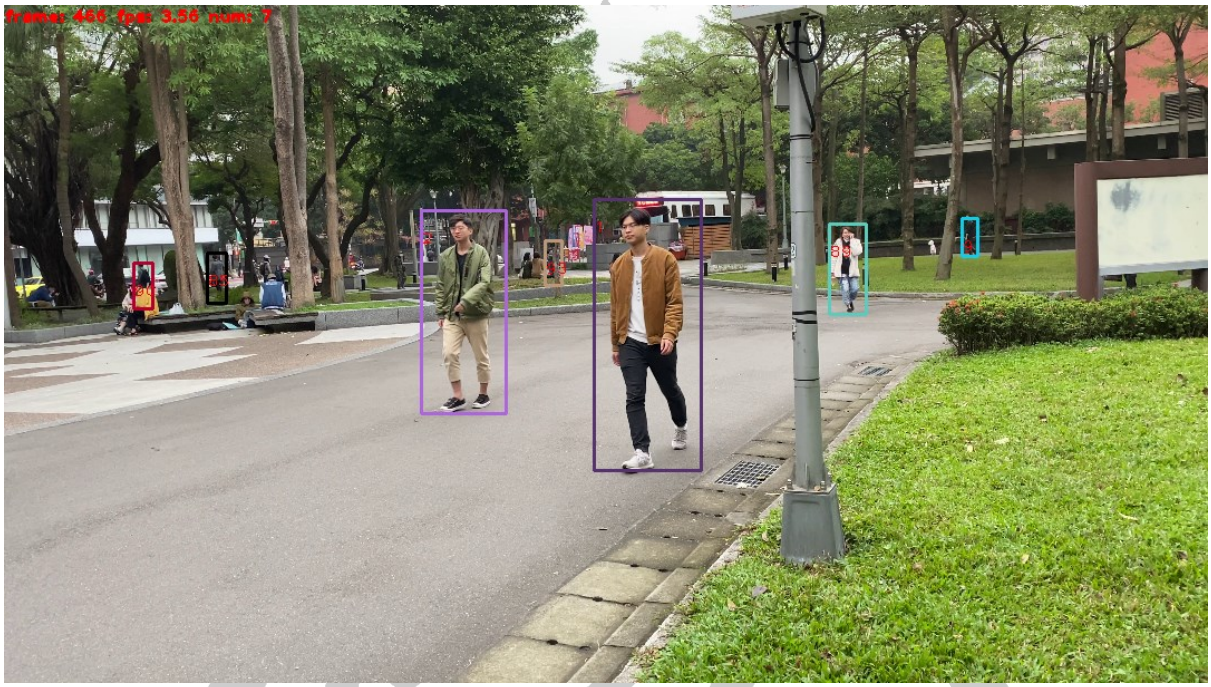


圖 22. 使用 JDE 方法行人追蹤結果 1

圖 23. 使用 JDE 方法行人追蹤結果 2

相關參考資料

- 【MOT】对JDE的深度解析
- 阅读心得：JDE：Towards Real-Time Multi-Object Tracking
- Towards Real-Time Multi-Object Tracking

(六) 使用行人重新識別模型辨識兩段物件追蹤的行人片段

行人重新識別 (ReID) 技術在 Horus 系統中負責可能之追蹤片段串接，藉此可以增進臉部辨識的價值，也就是說當我們人臉辨識成功辨識會員後，可以依造 ReID 所以串接的追蹤片段，蒐集更廣泛的資料，如此就可以有效利用臉部辨識無法確認身分之資料。

此部分使用的是 Lingxiao He 等人所開發的 fastreid 套件[13]。使用以 resnet 50 為基底的 BOT (Bag of Tricks)[14] 模型訓練，距離演算法是使用餘弦相似性 (Cosine similarity)[10]，資料部分使用 market1501[11] 以標記的開源資料作為訓練資料，該資料集包含 32,668 已標記的 bounding boxes 還有 1,501 個id，其中 train set 有 751，test set 有 750 個。訓練的部分只使用 fastreid 。

我們將訓練好的模型抽取出來使用（圖24），並使用訓練是的距離演算法-餘弦相似性 作為我們判定方式。

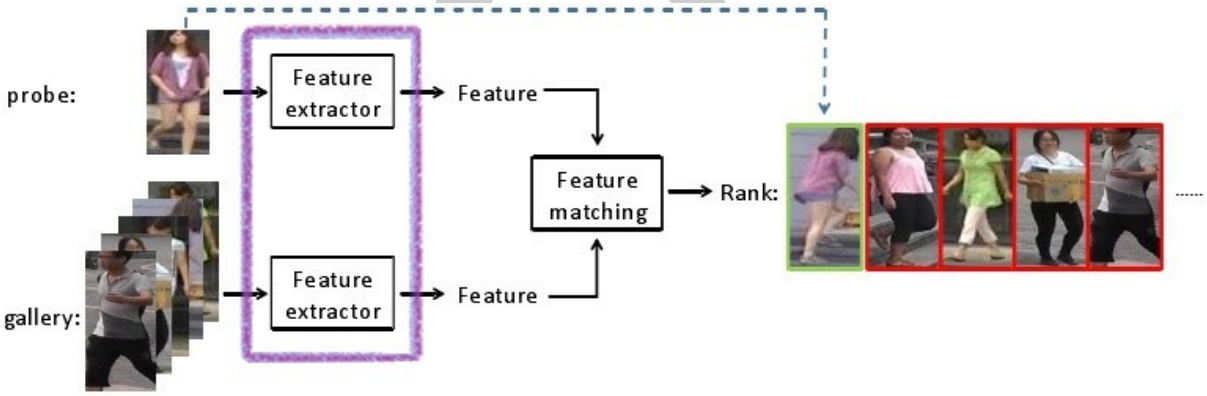


圖 24. ReID訓練架構圖 (圖片來源)

對 ReID 來說，我需要的樣本不用太多，所以我會對每一段追蹤做抽樣的動作，只取 10 張照片(不同片段做抽取)作為該追蹤的比對照片轉為比對特徵。當有新的追蹤產生，會以首張行人圖片作為比較，和當天的追蹤片段(已經結束)的比對特徵做配對，並回傳配對結果。

[10] 餘弦相似性 (Cosine similarity): https://en.wikipedia.org/wiki/Cosine_similarity

[11] market1501: https://deepai.org/dataset/market-1501

相關參考資料

- 行人重识别02-00：fast-reid(BoT)-目录-史上最新无死角讲解
- fast-reid复现（Market1501）

(七) 使用臉部辨識技術對會員身分之確認

1. 辨識架構

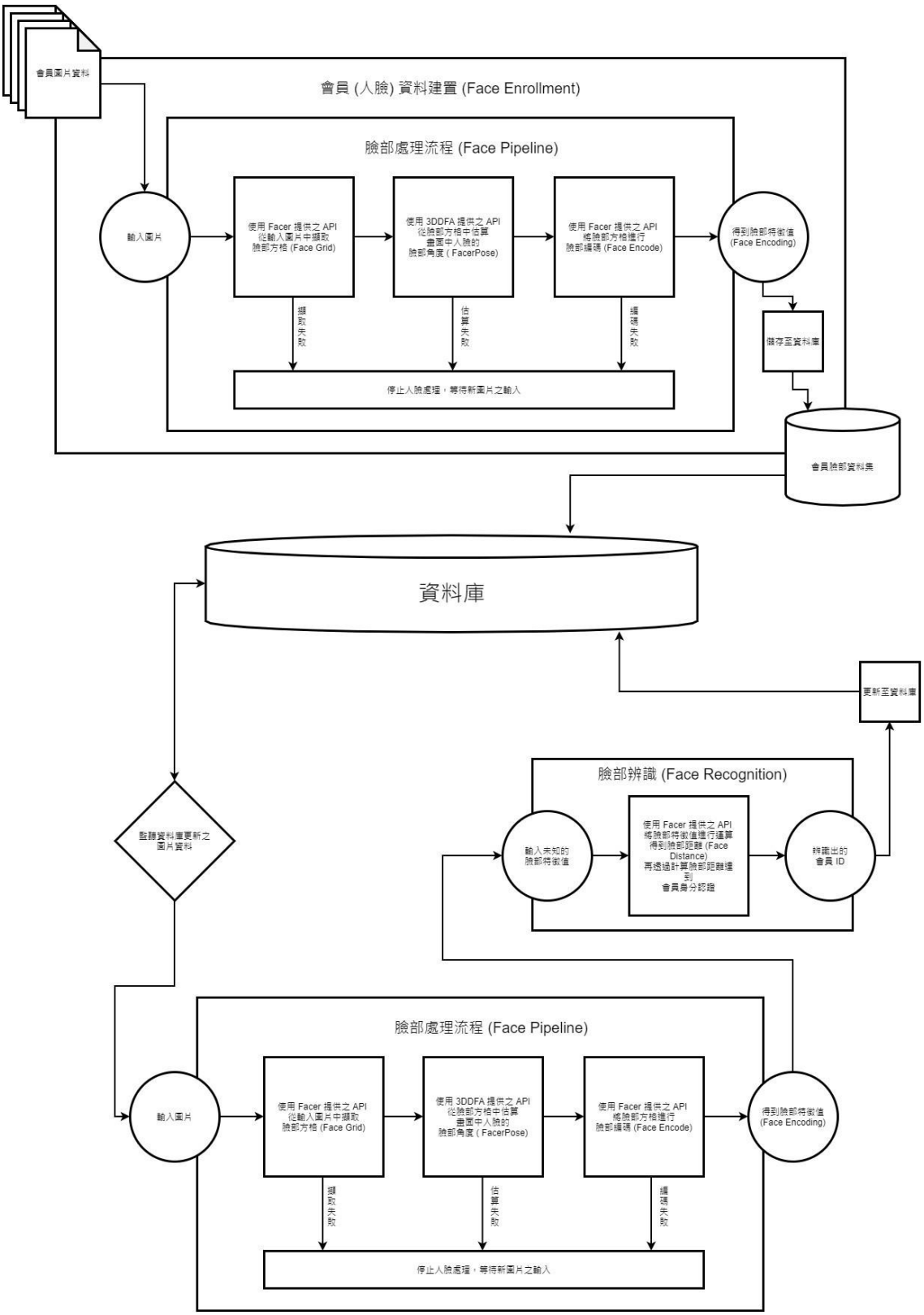

會員圖片資料
會員 (人臉) 資料建置 (Face Enrollment)
臉部處理流程 (Face Pipeline)
輸入圖片
使用 Facer 提供之 API
從輸入圖片中擷取
臉部方格 (Face Grid)
使用 3DDFA 提供之 API
從臉部方格中估算
畫面中人臉的
臉部角度 (FacerPose)
使用 Facer 提供之 API
將臉部方格進行
臉部編碼 (Face Encode)
得到臉部特徵值
(Face Encoding)
擷取失敗
估算失敗
編碼失敗
儲存至資料庫
停止人臉處理，等待新圖片之輸入
會員臉部資料集
資料庫
更新至資料庫
監聽資料庫更新之
圖片資料
臉部辨識 (Face Recognition)
輸入未知的
臉部特徵值
使用 Facer 提供之 API
將臉部特徵值進行運算
得到臉部距離 (Face
Distance)
再透過計算臉部距離遠
近
會員身分認證
辨識出的
會員 ID
臉部處理流程 (Face Pipeline)
輸入圖片
使用 Facer 提供之 API
從輸入圖片中擷取
臉部方格 (Face Grid)
使用 3DDFA 提供之 API
從臉部方格中估算
畫面中人臉的
臉部角度 (FacerPose)
使用 Facer 提供之 API
將臉部方格進行
臉部編碼 (Face Encode)
得到臉部特徵值
(Face Encoding)
擷取失敗
估算失敗
編碼失敗
停止人臉處理，等待新圖片之輸入

圖 25. 人臉辨識架構圖

2. 實作說明

一開始會先進行人臉資料建置 (Face Enrollment)，即會員臉部資料建置，要完成這項流程，必須將我們預先整理好的會員臉部圖片資料 (組員簡大爲生活照、組員何彥南生活照) 進行臉部處理流程 (Face Pipeline)。

臉部處理流程包含以下三項任務(圖25):

1. 臉部方格擷取
2. 臉部角度估算
3. 臉部編碼

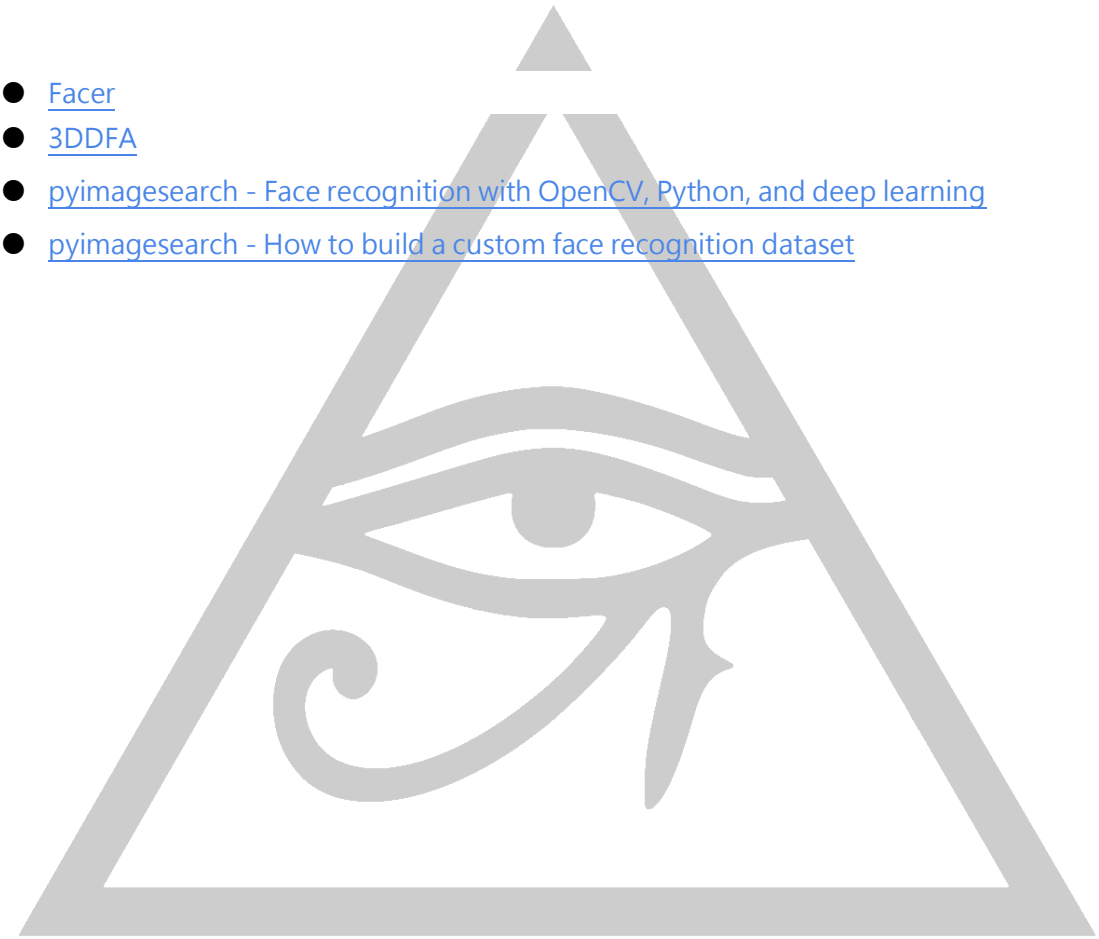
臉部方格擷取與臉部編碼之實作，將使用組員簡大爲開發之獨立開源專案 - Facer (**github**)，此專案利用 OpenCV, Dlib, face_recognition 等開源套件，提供針對人臉處理之函式庫與 API 供使用者使用，讓人臉處理工作更為方便。而臉部角度估算的工作將使用開源專案 3DDFA (**github**) 內提供的預訓練嵌入式模型與 API，將人臉圖片轉化成內含 62 維臉部特徵之陣列，並透過計算此陣列估算出圖片中的人臉在三維空間中的偏航軸 (Yaw, Pitch, Roll) 之數值，以過濾臉部角度偏轉過大的圖片，來增加後續人臉辨識之正確率。

在人臉辨識階段，將持續監聽資料庫的更新情形，當有新圖片寫入資料庫，人臉辨識模組將讀取該圖片，並進入臉部處理流程最後得到一組未經過身分認證的臉部特徵資料，隨後再將此特徵資料和資料庫的會員臉部特徵資料一併傳入 Facer 提供的

臉部身分識別 API 進行會員身分認證，若認證結果與超過認證門檻值，模組將更新最佳符合之會員 ID 置資料庫內。

相關參考資料

- Facer
- 3DDFA
- pyimagesearch - Face recognition with OpenCV, Python, and deep learning
- pyimagesearch - How to build a custom face recognition dataset



四、影片測試結果

(一) 情境介紹

1. 彥南與大爲與混進路人一起走 (base)

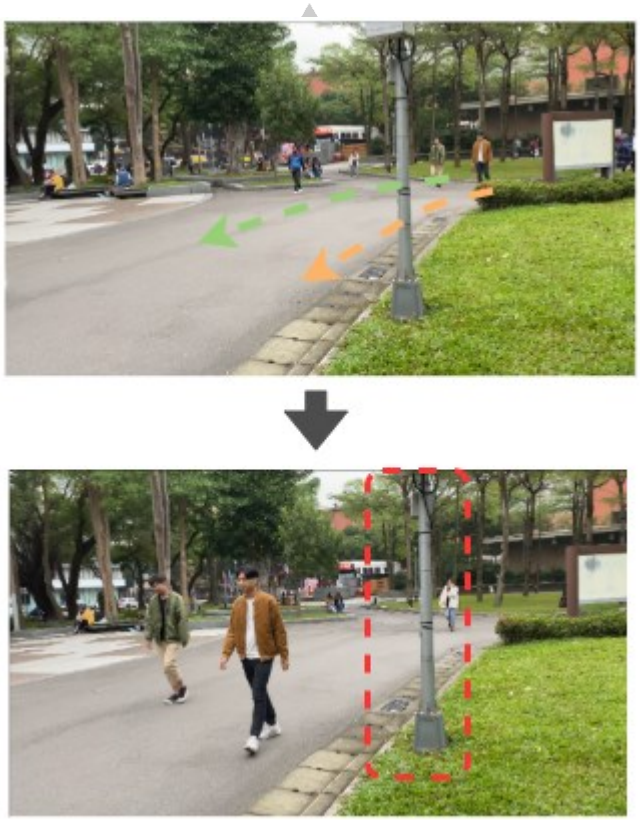


圖 26. 彥南與大爲與混進路人一起走 (base)

2. 兩人擦肩而過(v1)

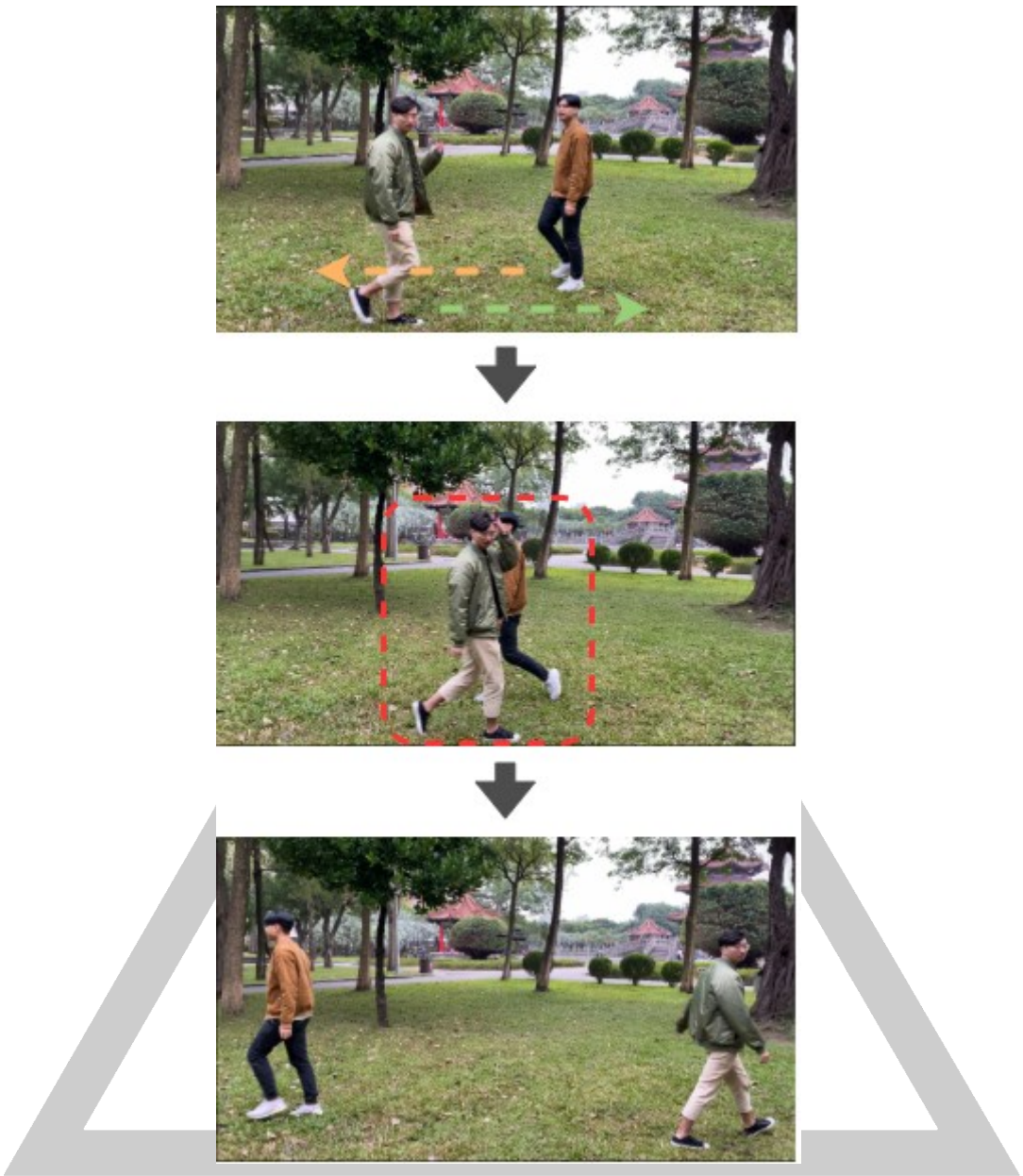


圖 27 . 兩人擦肩而過 (v1)

3. 兩人在樹後交叉再往鏡頭走近(v2)

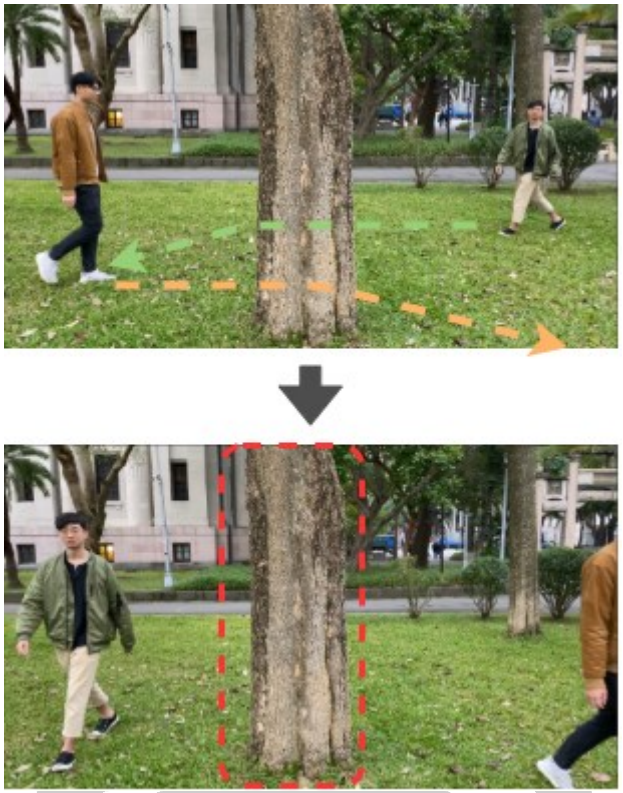


圖 28. 兩人在樹後交叉再往鏡頭走近 (v2)

4. 被樹遮住後再出現 (v3)

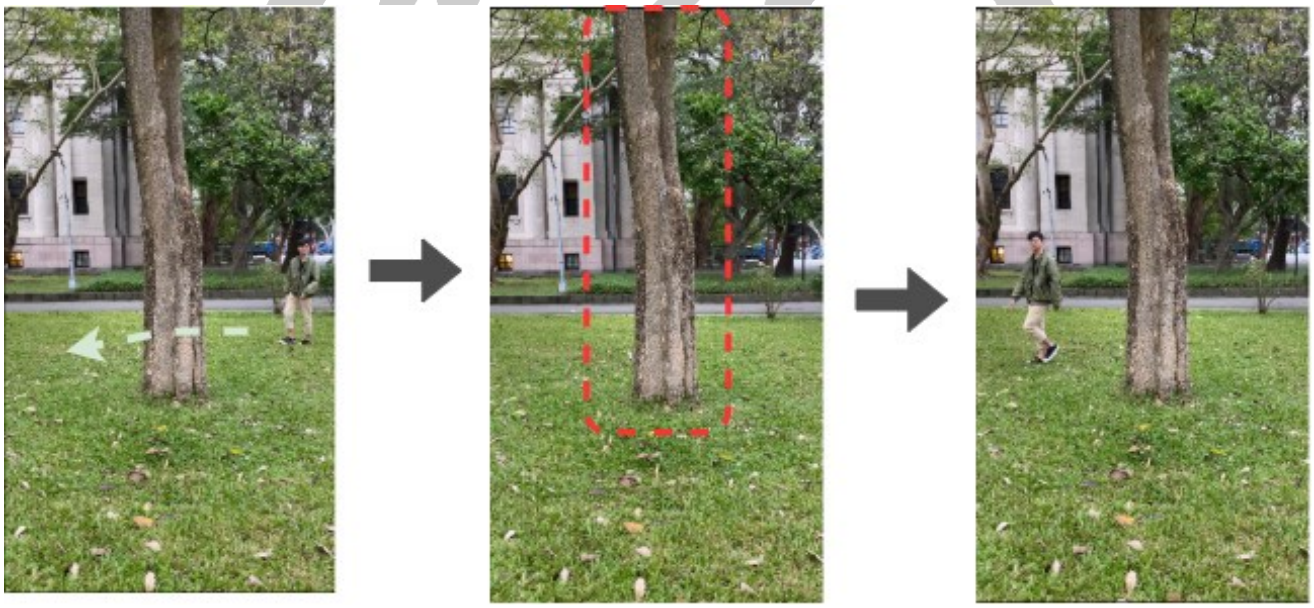


圖 29. 被樹遮住後再出現 (v3)

5. 走出鏡頭後偽裝再回鏡頭 (v4)

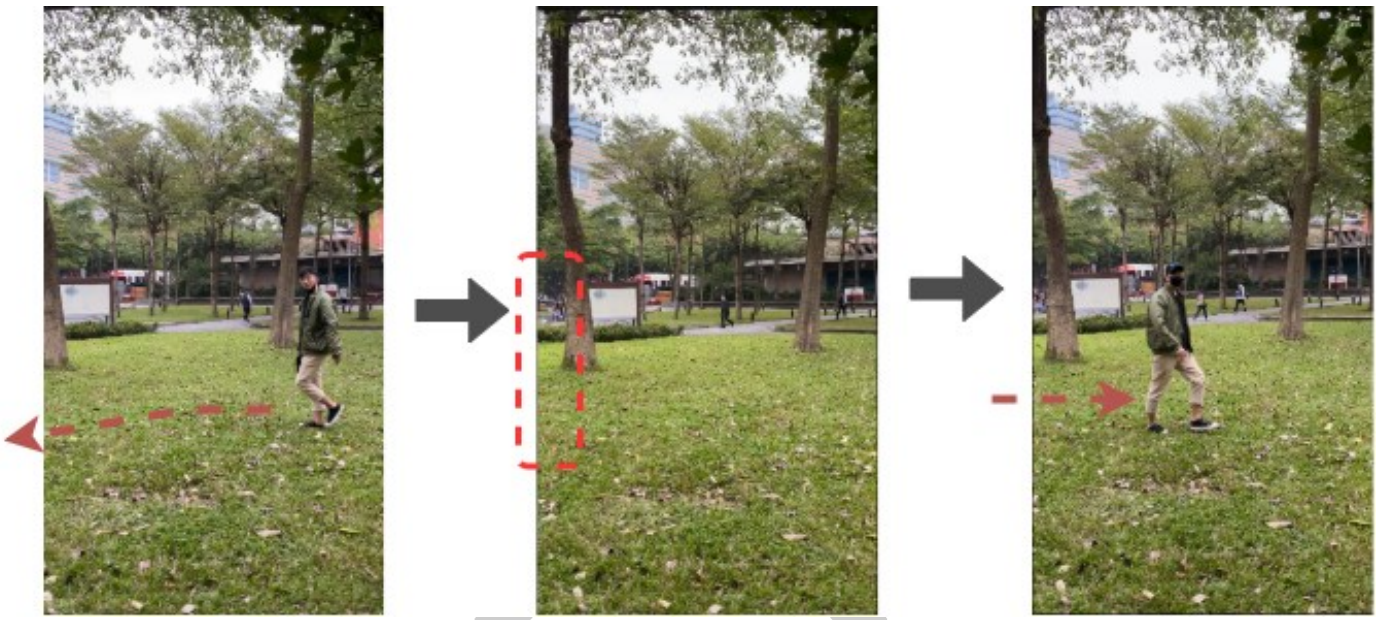


圖 30. 走出鏡頭後偽裝再回鏡頭 (v4)

(二) 實際運作結果比較

下表為三個模組分別對 8 部測試影片執行的結果:

	情境	臉的大小	人數	JDE	ReID	Face Recognition
base	障礙物(小)	小~大	>2	✓	-	✓
v1-1	擦肩而過	中	2	✓	-	✓
v1-2	擦肩而過	中	2	✓	-	✓
v2-1	障礙物(大)	小~中	2	X	✓	✓
v2-2	障礙物(大)	中~大	2	X	✓	✓
v3	障礙物(大)	小	1	X	✓	✓

v4	臉部遮蔽	中	1	X	✓	X

表 4. 測試影片結果比較

五、結論與感想

透過這次實作專案，我們學習到如何結合不同電腦視覺技術，建立系統。並利用不同技術的優點和特性，去彌補實際場景時會遇到的問題。這次專案比較困難的地方是如何結合三種技術的環境，還有速度上的控制，目前要達到 real-time 還是偏困難。

感謝寫文獻檢閱的作者，讓第一次踏入電腦視覺領域的我們可以很快地進入狀況並找到方向。再來要感謝一下開源的貢獻者，不僅讓本專案可以全部使用開源的程式碼完成，也讓這世界更美好。另外還有們辛苦的測試影片演員 - 簡大爲和何彥南，感謝他們友情贊助。最後要感謝我們的指導老師 - 黃福明教授，用心的指導我們，給了我們許多建議，並且在我們方向錯的時候指點我們。

六、參考文獻

Multiple Object Tracking

[1]. Wenhan Luo, Junliang Xing, Anton Milan, Xiaoqin Zhang, Wei Liu, Xiaowei Zhao, Tae-Kyun Kim. "Multiple object tracking: A literature review," *Artificial Intelligence*, 2020, 103448, ISSN 0004-3702, https://doi.org/10.1016/j.artint.2020.103448.

(Link | arxiv)

[2]. Wang, Zhongdao & Zheng, Liang & Liu, Yixuan & Wang, Shengjin. " Towards Real-Time Multi-Object Tracking," 2019.

(Link | ECCV 2020 | arxiv | video)

[3]. Peng J. et al. "Chained-Tracker: Chaining Paired Attentive Regression Results for End-to-End Joint Multiple-Object Detection and Tracking," *In: Vedaldi A., Bischof H., Brox T., Frahm JM. (eds) Computer Vision – ECCV 2020. ECCV 2020*. Lecture Notes in Computer Science, vol 12349. Springer, Cham.

(Link | ECCV2020 | video)

Face Recognition

[4]. Mei, Wang & Deng, Weihong. (2018). "Deep Face Recognition: A Survey. *Neurocomputing,*" 10.1016/j.neucom.2020.10.081.

(Link | arxiv)

[5]. delong, Chen & Liu, Fan & Li, Zewen. " Deep Learning Based Single Sample Per Person Face Recognition: A Survey," 2020.

(Link | arxiv)

[6]. Guodong Guo, Na Zhang, "A survey on deep learning based face recognition," *Computer Vision and Image Understanding*, Volume 189, 2019, 102805, ISSN 1077-3142, https://doi.org/10.1016/j.cviu.2019.102805.

(Link | arxiv)

[7]. A. Boka and B. Morris, " Person Recognition for Access Logging," *2019 IEEE 9th Annual Computing and Communication Workshop and Conference (CCWC)*, Las Vegas, NV, USA, 2019, pp. 0933-0936, doi: 10.1109/CCWC.2019.8666483.

(Link)

[8]. J. Celine and S. A. A, " Face Recognition in CCTV Systems," *2019 International Conference on Smart Systems and Inventive Technology (ICSSIT)*, Tirunelveli, India, 2019, pp. 111-116, doi: 10.1109/ICSSIT46314.2019.8987961.

(Link)

[9]. X. Zhu, X. Liu, Z. Lei and S. Li, "Face alignment in full pose range: A 3D total solution", IEEE Trans. Pattern Anal. Mach. Intell., vol. 41, no. 1, pp. 78-91, Jan. 2019.

(arxiv)

Persion ReID

[9]. N. Gheissari, T. B. Sebastian and R. Hartley, "Person Reidentification Using Spatiotemporal Appearance," 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06), New York, NY, USA, 2006, pp. 1528-1535, doi: 10.1109/CVPR.2006.223.

(Link)

[10]. Zheng, L., Y. Yang and A. Hauptmann. (2016). "Person Re-identification: Past, Present and Future," ArXiv abs/1610.02984 : n. pag.

(Link)

[11]. Ye, M., Shen, J., Lin, G., Xiang, T., Shao, L., & Hoi, S. (2020). "Deep Learning for Person Re-identification: A Survey and Outlook," ArXiv, abs/2001.04193.

(Link)

[12]. Khawar Islam, "Person search: New paradigm of person re-identification: A survey and outlook of recent works," *Image and Vision Computing*, Volume 101, 2020, 103970, ISSN 0262-8856, https://doi.org/10.1016/j.imavis.2020.103970.

(Link)

[13]. He, Lingxiao, Xingyu Liao, Wu Liu, Xinchen Liu, Peng Cheng and Tao Mei. "FastReID: A Pytorch Toolbox for General Instance Re-identification," ArXiv abs/2006.02631 (2020): n. pag.

(Link | arxiv)

[14]. H. Luo, Y. Gu, X. Liao, S. Lai and W. Jiang, "Bag of Tricks and a Strong Baselin e for Deep Person Re-Identification," *2019 IEEE/CVF Conference on Computer Vi sion and Pattern Recognition Workshops (CVPRW),* Long Beach, CA, USA, 2019, pp. 1487-1495, doi: 10.1109/CVPRW.2019.00190.

(Link | arxiv)

[15]. Liang Zheng, Liyue Shen, Lu Tian, Shengjin Wang, Jingdong Wang, Qi Tian, "Scalable Person Re-identification: A Benchmark," IEEE International Conference on Computer Vision (ICCV), 2015.

(Link | ICCV2015)