Final Project Report

Project Topic

Real Time Face Recognition System via WebCam

Group Members (Work Allocation)

106061203 徐浩宇 (Report、整個 Pipeline 設計、Coding)

106061236 朱敏瑋 (PCA、LDA 訓練、影片說明及製作、DataSet 收集加 label)

Project Description

使用現有的 Labeled Faces Dataset 以及自行收集的 Face Images 去實作即時臉孔辨識系統，且使用 WebCam 去做 Demo。

Dataset

Labeled Faces in the Wild (aka LFW) (http://vis-www.cs.umass.edu/lfw/)

Self-collected Face Data (自行收集)

Strategies Used in this Project

Classification model:

(1) SVM

(2) KNN

(3) Random Forest

Face Detection & Face Alignment:

(1) OpenCV

(2) dlib (效果最好)

(3) OpenFace

Data Preprocessing:

(1) PCA

(2) LDA

(3) StandardScaler

(4) OpenCV Image Normalization (最重要的一步)

Parameter Tuning:

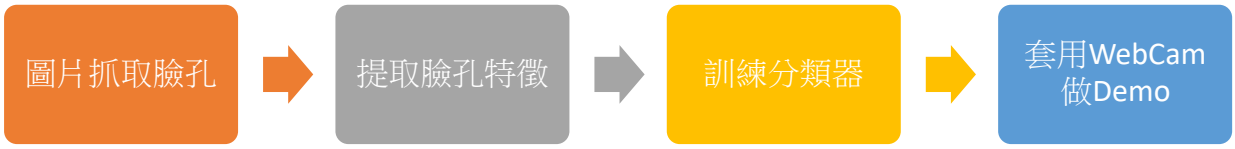
GridSearchCV

Evaluation:

(1) classification_report (包含 Accuracy, Recall, Precision, F1-Score)

(2) confusion_matrix (true label v.s. predicted label)

Pipeline of this Project :

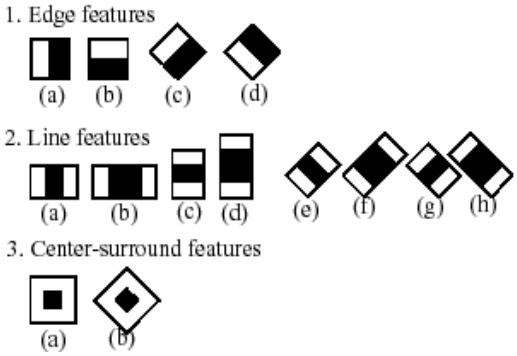


一、 圖片抓取臉孔 (Face Detection) :

(一) 使用 OpenCV 抓取臉孔 :

OpenCV 採用的是 Cascade Classifier，它是類似一種 Boosting 的方法，Boosting 概念是利用多個弱分類器做 Ensemble，做出一個強分類器的結果。

這邊也是類似的概念，Cascade Classifier 就是利用多個弱 classifiers 去做圖片偵測(見下圖)，每個 classifier 都只負責一個簡單的任務(Ex: 偵測邊界)，透過組合多個 Classifier 去抓出人臉的特徵如輪廓、五官等等，進而判斷出人臉位置。



可是實作時發現 Cascade Classifier 會有一些缺點，特別是臉孔無法偵測出來的問題，當臉是側面或是光線強度不一致時，就會抓不到臉孔，所以後來改用 dlib 代替。

(二) 使用 dlib 抓取臉孔 :

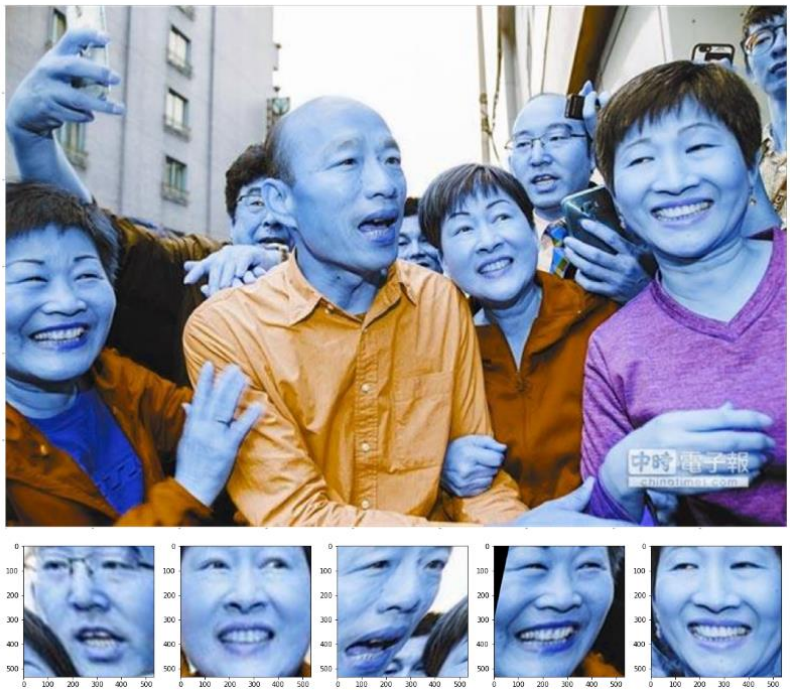
Dlib 使用的是 HOG (Histogram of Oriented Gradients)特徵提取的方法，特別之處在於 HOG 不是針對 pixel 數值大小去做偵測，它反而會去計算 pixel 的 Gradients，透過像素的梯度 flow 去抓出臉孔的輪廓以及五官(見下圖)。

經實作過後發現 Dlib 完美解決了 Cascade Classifier 無法解決的問題。

二、提取臉孔特徵：

(一) 臉孔對齊 (Face Alignment):

使用 OpenFace 的 AlignDlib 去把五官做定位，這步驟其實蠻重要的，因為臉在不同的角度下切出來的五官位置會不固定，不固定的五官會讓特徵提取的困難度增加，來自於 Dataset 的不一致性。故要使用 Face Alignment 去把五官定位在固定位置上。



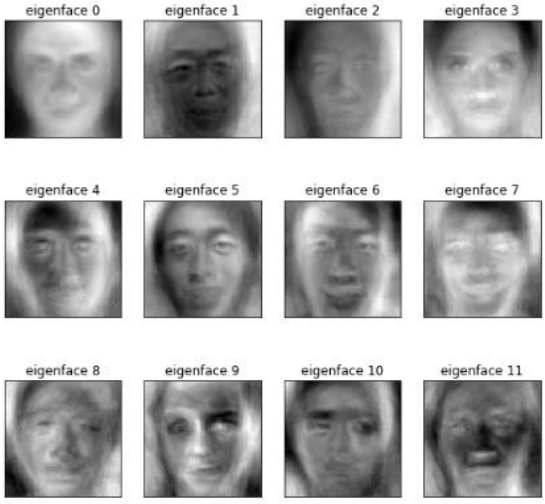
左圖：Face Alignment 效果

(二) 提取特徵 (Feature Extraction) :

1. Dimensionality Reduction Method (PCA, LDA):

直接將 100 x 100 的 cropped images 做 PCA 和 LDA 的降維，簡單明瞭。

將 PCA 的 Principal Components 抽出來做視覺化後，可以發現這些 components 各自代表一些人的臉孔分布(也稱作 Eigenfaces)，而 PCA 就是透過這些 Eigenfaces 對一張照片做內積，透過內積的值去判斷是哪一個人，蠻有趣的發現。



左圖：Eigenfaces from PC

2. Transform to Embeddings (via Pretrained Model) :

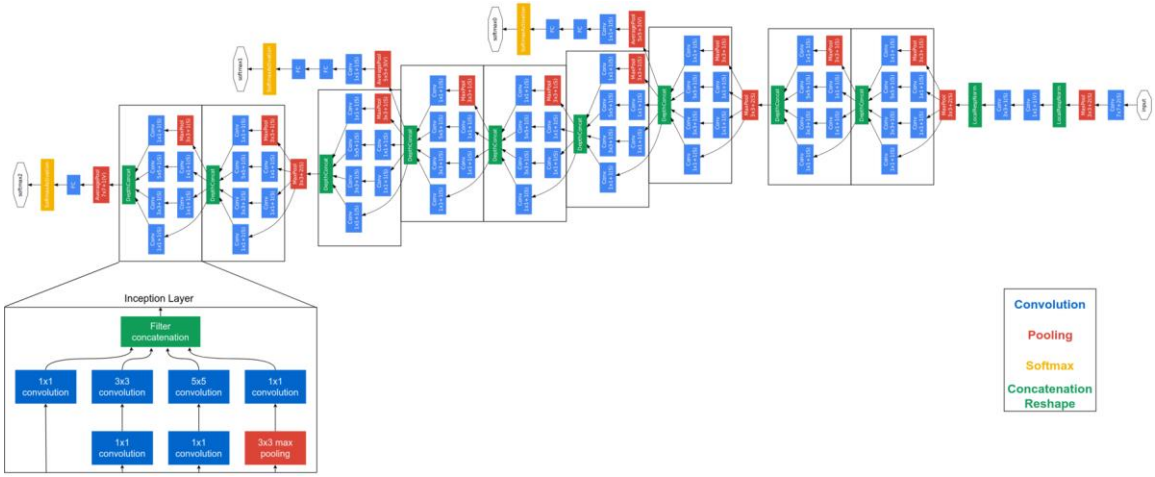
前面的測試發現到簡單的 PCA 或 LDA 降維方法並不足以抓取到臉孔的資訊，雖然 accuracy 算還不錯，但是實作到 WebCam 上會整個爛掉。因此，這邊透過 Pretrained Model 去提取臉孔的特徵。

Pretrained Model 是成功訓練在大型人臉 Dataset 上的 Deep Neural Network，而我們利用其最後那一層的 Output 當作是圖片的特徵向量(embeddings)。

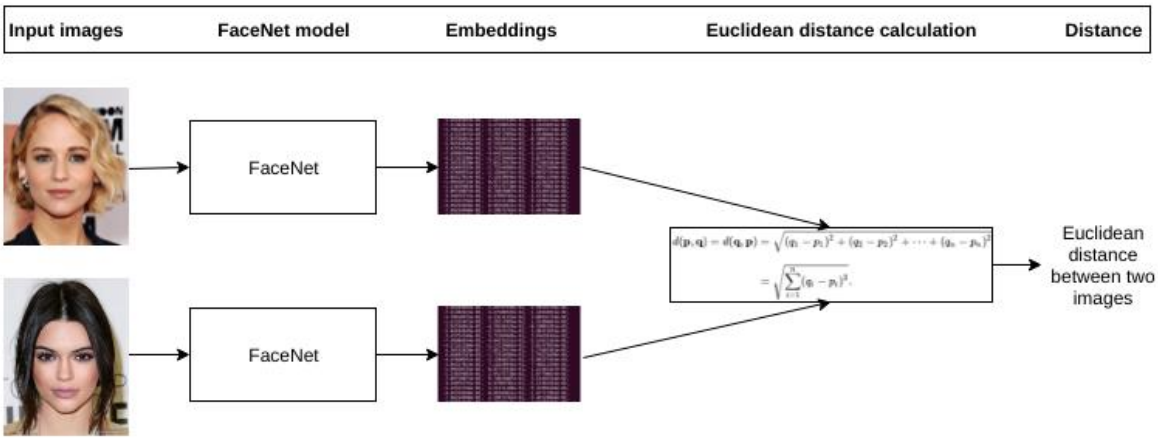
這個想法其實來自很多論文中都有提到一個觀念，就是 CNN 架構的內涵。經過研究發現，CNN 在前幾層的時候多提取小規模的特徵(Ex: 邊界)、而在最後的幾層則是提取大規模的特徵(Ex: 臉型)。因此，一張圖片輸入 Model 後，在最後幾層的數值某種程度上就代表該圖片的特徵啦。

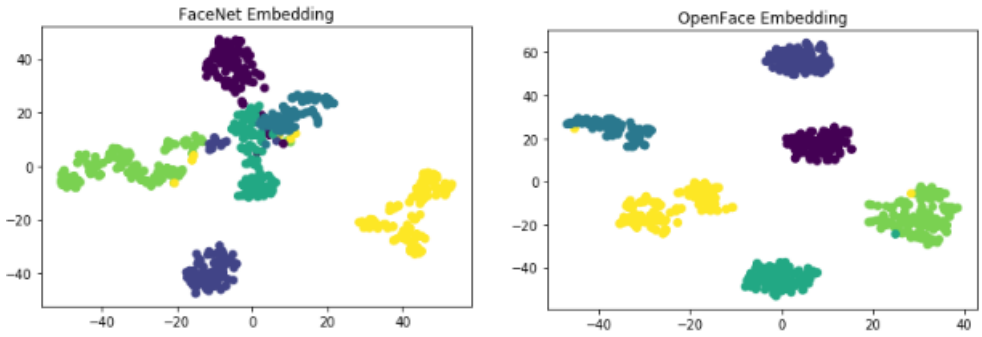
至於 Pretrained Model 如何挑選 ？ 我選擇了兩個 Model，分別是 Openface 和 Facenet，兩個都是使用 Inception Network 的架構。Inception Network 的特色在於其 Inception block，每個 Inception block 裡面含有四種 Convolution Layer，一個輸入會有四個輸出，並將輸出組合起來，這樣就可以避免只有單一 Convolution Layer 造成資訊抓取不完全的問題。

關鍵一步: 資料輸入到 Model 前要先做 Channel-Wise Normalization，這樣才可以避免掉光線等等環境因素干擾造成辨識效果的降低。



最後的作法就是透過比較 embedding 的差異去辨識臉孔，流程如下圖所示。





結果發現，OpenFace 的降維效果略比 FaceNet 降維效果好。

三、訓練分類器 ：

(一) 一般的 Classifier :

使用 SVM, KNN, Random Forest 共三種 classifier，套用不同的 Data 去做 Training，包括: (1)什麼都不做 (2) PCA 降維 (3) LDA 降維 (4) Facenet 壓縮 (5) OpenFace 壓縮

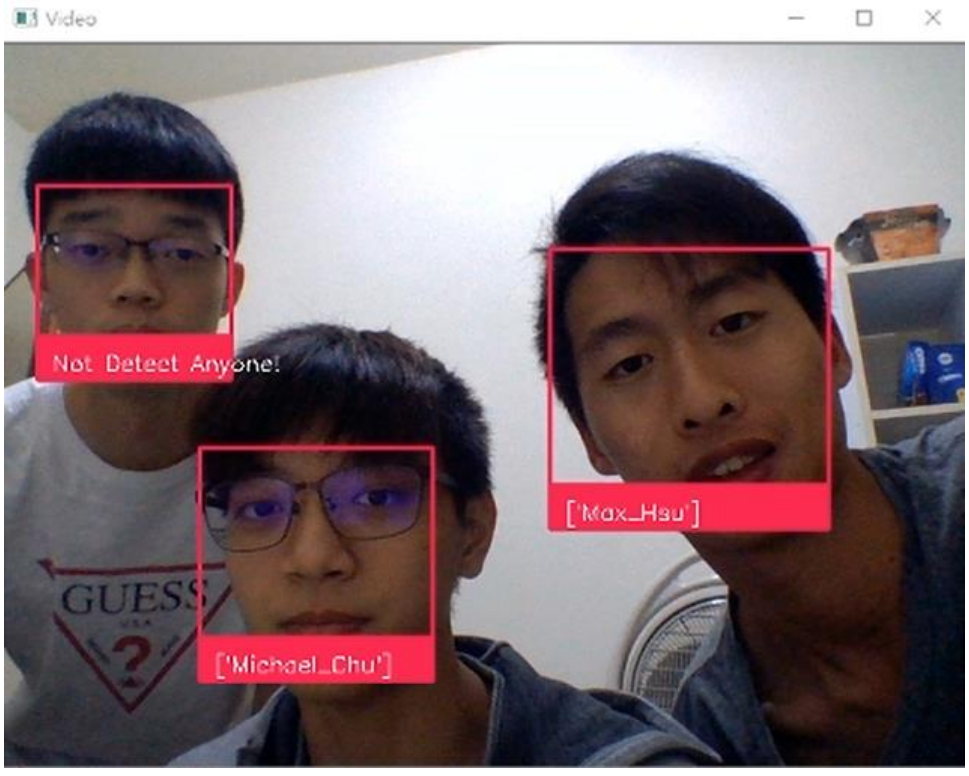
測試結果放在後面喔!!!

(二) Minimal L2-Distance :

用一般的 classifier 會產生一個 issue，就是 classifier 沒有辦法區別出不在 Dataset 裡面的臉孔，它反而會給出一個辨識結果，這樣是不對的。

假設現在使用人臉辨識的保全系統，當一個非資料庫內的臉孔被偵測時，系統該做的是判斷出該臉孔不在資料庫內，而不是給出一個辨識結果並放他通行。

因此，最直覺的方法就是將該臉孔的 Embedding 跟資料庫內部所有臉孔的 Embedding 做比對，選出差距最小的人名當作「潛在輸出」。該差距會再跟設定好的閾值做比較，若差距大於閾值，則代表該臉孔跟所有資料庫臉孔不相符，回報找不到人(見下圖)。



四、套用 WebCam 做 Demo :

這邊主要是利用 OpenCV 裡面的 Video Capture 模組去抓取 WebCam 的影像，並將影像送入前面設計好的 Pipeline 裡面，產生結果。

Demo 展示時，會把偵測到臉孔的 Bounding Box 畫出來，並將預測的臉孔名稱顯示再 Bounding Box 下方。

Some Training Results :

	Grid Score	Test Acc	Precision	Recall	F1-Score
SVM	0.874	0.93	0.93	0.93	0.93
SVM_pca	0.874	0.94	0.95	0.94	0.94
SVM_lda	0.874	0.86	0.86	0.86	0.86
SVM_facenet	0.988	0.97	0.97	0.97	0.97
SVM_openface	0.996	0.985	0.99	0.99	0.99

Figure 1: SVM classifier with different processing approaches

	Grid Score	Test Acc	Precision	Recall	F1-Score
RF	0.87	0.87	0.89	0.86	0.87
RF_pca	0.892	0.93	0.94	0.93	0.93
RF_lda	0.857	0.82	0.82	0.82	0.82
RF_facenet	0.971	0.97	0.97	0.97	0.97
RF_openface	0.998	0.99	0.99	0.99	0.99

Figure 2: Random Forest classifier with different processing approaches

	Grid Score	Test Acc	Precision	Recall	F1-Score
KNN	0.775	0.85	0.88	0.85	0.86
KNN_pca	0.788	0.82	0.85	0.83	0.83
KNN_lda	0.87	0.87	0.87	0.87	0.87
KNN_facenet	0.956	0.955	0.95	0.95	0.95
KNN_openface	0.996	0.985	0.99	0.99	0.99

Figure 3: KNN classifier with different processing approaches

Some Discussion :

(1) 可以試著減小 Pretrained Model Size，這樣就不會花太多時間在產生 Image Embedding，可以加快 Video Capture 運行的速度 (增加 FPS)，比較會有 Real-Time 的效果。

(2) 推測目前這個方法可以運用在更多類別且每個類別有更少照片的 Dataset，因為 Pretrained Model 的特徵提取效果真的蠻不錯的。

(3) 實作時許多額外套件安裝會比較麻煩，彼此都有版本相容性的問題，所以要跑之前都要再三確認 Python 或 numpy 版本等等以免發生衝突。