

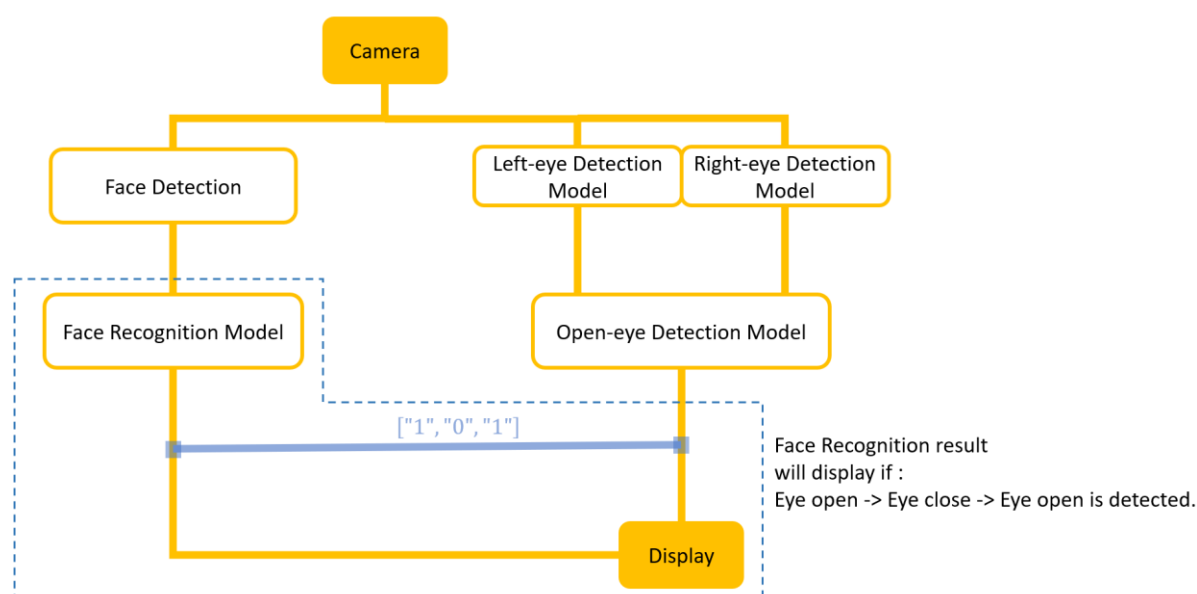
作品名稱：dlib 臉部影像辨識

一、說明

本專題使用 dlib 套件及進行人臉辨識，dlib 是一套包含了機器學習、電腦視覺、圖像處理等的函式庫，使用 C++開發而成，目前廣泛使用於工業界及學術界，也應不用再機器人、嵌入式系統、手機、甚至於大型的運算架構中，而且最重要的是，它不但開源且完全免費，也可以跨平台使用。

二、論文

此專案利用 Pre-train 好的 Dlib model，進行人臉辨識（Face Detection），並且實現僅用一張照片作為 database 就可以作出達到一定效果的人臉識別（Face Recognition）。除此之外，更加入了活體偵測（Liveness Detection）技術，以避免利用靜態圖片通過系統識別的問題。



Dlib 是一套基於 C++ 的機器學習工具包，藉由 Dlib 可以使用這些機器學習工具在任何的專案上，目前無論在機器人、嵌入設備、移動設備甚至是大型高效運算環境中都被廣泛使用。





三、實作

使用 dlib 套件之前請先下載安裝 visual Studio2019 以後版本。安裝時要勾選使用 C++桌面開發、使用 Windows 平台，安裝好 visual Studio2019 後皆可執行安裝 pip install dlib 套件。



Dlib 官方網站提供了許多已經訓練好的模型，我們先將一些常用的模型下載，許多範例需要這些模型檔才能執行。需要的模型依照片為主：(模型檔網頁：

<http://dlib.net/files/>)

	dlib_face_recognition_resnet_model_v...	2020/7/29 下午 07:37	DAT 檔案	21,940 KB
	mmod_human_face_detector.dat	2020/7/29 下午 07:37	DAT 檔案	713 KB
	shape_predictor_5_face_landmarks.dat	2020/7/29 下午 07:37	DAT 檔案	8,937 KB
	shape_predictor_68_face_landmarks.dat	2020/7/29 下午 07:38	DAT 檔案	97,358 KB

1. 5 點特徵人臉影像辨識：

dlib 套件偵測人臉的矩形區塊，也可以偵測鼻子、左右點及嘴巴等。使用 <shape_predictor_5_face_landmarks.dat> 特徵模型進行人臉矩形區域中偵測 5 點特徵。然後建議請先安裝 `pip install opencv-python` 確保執行時出現缺少套件的問題所在。

程式碼：

```
import dlib

predictor = "shape_predictor_5_face_landmarks.dat" #模型(5 點)
sp = dlib.shape_predictor(predictor) #讀入模型
detector = dlib.get_frontal_face_detector() #偵測臉部

img = dlib.load_rgb_image("media\\LINE_230625.jpg") #讀取圖片
win = dlib.image_window() #建立顯示視窗
win.clear_overlay() #清除圖形
```

```

win.set_image(img) #顯示圖片

dets = detector(img, 1) #臉部偵測,1 為彩色
print("人臉數：{}".format(len(dets)))
#繪製人臉矩形及 5 點特徵
for k, det in enumerate(dets):
    print("偵測人臉 {}: 左：{} 上：{} 右：{} 下：{}".format(k, det.left(),
det.top(), det.right(), det.bottom())) #人臉坐標
    win.add_overlay(det) #顯示矩形
    shape = sp(img, det) #取得 5 點特徵
    win.add_overlay(shape) #顯示 5 點特徵
    dlib.hit_enter_to_continue() #保持影像

```

執行結果：

```

人臉數：8
偵測人臉 0: 左：619 上：350 右：681 下：412
偵測人臉 1: 左：751 上：343 右：803 下：395
偵測人臉 2: 左：923 上：357 右：985 下：419
偵測人臉 3: 左：855 上：435 右：907 下：487
偵測人臉 4: 左：959 上：464 右：1011 下：516
偵測人臉 5: 左：654 上：475 右：705 下：527
偵測人臉 6: 左：536 上：357 右：598 下：419
偵測人臉 7: 左：377 上：370 右：439 下：433

```



2. CNN 訓練模型人臉偵測：

Dlib 官網提供的模型中有一個利用 CNN 訓練模型檔，此模型偵測人臉可得到表情較

佳效果。使用<mmod_human_face_detector.dat>特徵模型進行人臉矩形表情。
程式碼：

```
import dlib
import cv2

cnn_detector =
dlib.cnn_face_detection_model_v1("mmod_human_face_detector.dat") #CNN 模型
img = cv2.imread("media\\LINE_230625.jpg")
dets = cnn_detector(img, 1) #偵測人臉
print("人臉數：{}".format(len(dets)))
for i, det in enumerate(dets):
    #det.rect 是人臉矩形坐標,det.confidence 為信心指數
    face = det.rect
    left = face.left()
    top = face.top()
    right = face.right()
    bottom = face.bottom()
    print("偵測人臉 {}: 左：{} 上：{} 右：{} 下：{} 信心指數：{}".format(i,
left, top, right, bottom, det.confidence))
    cv2.rectangle(img, (left, top), (right, bottom), (0, 255, 0), 1) #畫人臉矩形

cv2.namedWindow("win", cv2.WINDOW_AUTOSIZE)
cv2.imshow("win", img)
k = cv2.waitKey(0)
cv2.destroyAllWindows()
```

執行結果：

```
人臉數：8
偵測人臉 0: 左：923 上：358 右：980 下：415 信心指數：1.0898116827011108
偵測人臉 1: 左：854 上：433 右：911 下：490 信心指數：1.0861622095108032
偵測人臉 2: 左：654 上：472 右：701 下：520 信心指數：1.084273099899292
偵測人臉 3: 左：755 上：343 右：802 下：390 信心指數：1.0825084447860718
偵測人臉 4: 左：611 上：344 右：679 下：412 信心指數：1.0805684328079224
偵測人臉 5: 左：958 上：456 右：1015 下：513 信心指數：1.0772682428359985
偵測人臉 6: 左：543 上：362 右：591 下：409 信心指數：1.0736702680587769
偵測人臉 7: 左：369 上：358 右：437 下：426 信心指數：1.0615956783294678
```




3. 68 點特徵人臉偵測

Dlib 除了可以進行基本及 5 點特徵人臉偵測外，還提供 68 點特徵人臉偵測，包括了雙眼、鼻子、嘴巴及頭部的輪廓詳細資訊，我們可以畫出這些部位的輪廓圖形。使用<shape_predictor_68_face_landmarks.dat>特徵模型進行人臉特徵辨識。程式碼：

```
import numpy as np
import cv2  #影象處理庫 OpenCV
import dlib

predictor = dlib.shape_predictor('shape_predictor_68_face_landmarks.dat')  #構建
68 點特徵
detector = dlib.get_frontal_face_detector()  #偵測臉部正面

img = cv2.imread("media\\LINE_ALBUM_1120609.jpg")  #讀取影象
dets = detector(img, 1)  #偵測人臉
for det in dets:
    #人臉關鍵點識別
    landmarks = []
    for p in predictor(img, det).parts():
        landmarks.append(np.matrix([p.x, p.y]))
    # 取得 68 點座標
    for idx, point in enumerate(landmarks):
```

```

pos = (point[0, 0], point[0, 1])  #[0,0]為 x 坐標,[0,1]為 y 坐標
cv2.circle(img, pos, 5, color=(0, 255, 0))  #畫出 68 個小圓點
font = cv2.FONT_HERSHEY_SIMPLEX  # 利用 cv2.putText 輸出 1-68
#引數依次是：圖片，新增的文字，座標，字型，字型大小，顏色，字型
粗細

cv2.putText(img, str(idx+1), pos, font, 0.4, (0, 0, 255), 1,cv2.LINE_AA)

cv2.namedWindow("img", 2)
cv2.imshow("img", img)          #顯示影像
cv2.waitKey(0)
cv2.destroyAllWindows()

```

執行結果：



4. 攝影機圖像中劃出點輪廓

Imutils 模組是一個圖形處理的模組，它為 dlib 提供臉部 68 點特徵各部位範圍的功能，使用 imutils 模組就可輕鬆取得各部位的特徵點範圍。（請先安裝 pip install imutils）

程式碼：

```
from imutils.video import VideoStream
from imutils import face_utils
import imutils
import time
import dlib
import cv2

detector=dlib.get_frontal_face_detector() #偵測臉部正面
predictor=dlib.shape_predictor("shape_predictor_68_face_landmarks.dat") #構建 68
點特徵

#左右眼特徵點索引
(left_Start,left_End)=face_utils.FACIAL_LANDMARKS_IDXS["left_eye"]
(right_Start,right_End)=face_utils.FACIAL_LANDMARKS_IDXS["right_eye"]
#嘴特徵點索引
(leftMouth,rightMouth)=face_utils.FACIAL_LANDMARKS_IDXS['mouth']
#下巴特徵點索引
(leftJaw,rightJaw)=face_utils.FACIAL_LANDMARKS_IDXS['jaw']
#鼻子特徵點索引
(leftNose,rightNose)=face_utils.FACIAL_LANDMARKS_IDXS['nose']
#左右眉毛特徵點索引
(left_leftEyebrow,left_rightEyebrow)=face_utils.FACIAL_LANDMARKS_IDXS['left_eyebro
w']
(right_leftEyebrow,right_rightEyebrow)=face_utils.FACIAL_LANDMARKS_IDXS['right_ey
ebrow']

vsThread=VideoStream(src=0).start() #開啟攝影機
time.sleep(2.0)

while True:
    frame = vsThread.read() #讀取影格
    frame = imutils.resize(frame, width=720)
    faces = detector(frame, 0) #偵測人臉
    for face in faces:
        shape = predictor(frame, face) #取得人臉
        shape = face_utils.shape_to_np(shape) #轉為 numpy
        #左右眼特徵點
        leftEye = shape[left_Start:left_End]
```

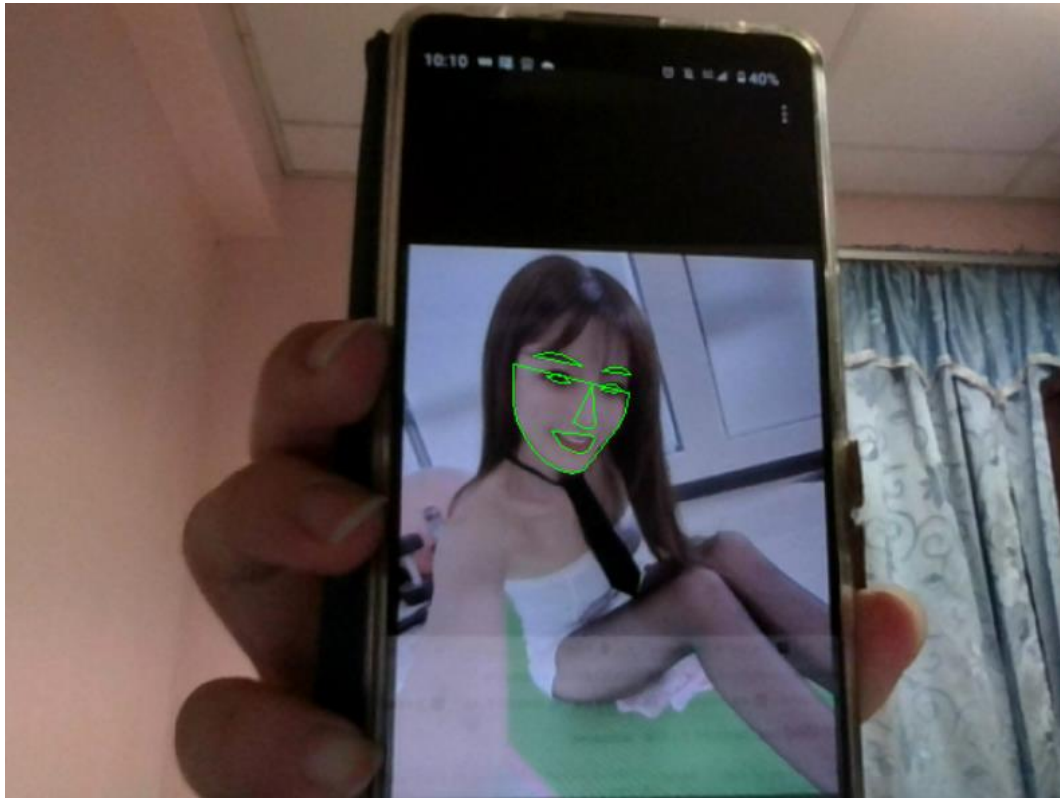
```

rightEye = shape[right_Start:right_End]
#轉為外殼
leftEyeHull = cv2.convexHull(leftEye)
rightEyeHull = cv2.convexHull(rightEye)
#畫出輪廓
cv2.drawContours(frame, [leftEyeHull], -1, (0, 255, 0), 1)
cv2.drawContours(frame, [rightEyeHull], -1, (0, 255, 0), 1)
#嘴
mouth=shape[leftMouth:rightMouth]
mouthHull=cv2.convexHull(mouth)
cv2.drawContours(frame, [mouthHull], -1, (0, 255, 0), 1)
#鼻子
nose=shape[leftNose:rightNose]
noseHull=cv2.convexHull(nose)
cv2.drawContours(frame, [noseHull], -1, (0, 255, 0), 1)
#下巴
jaw=shape[leftJaw:rightJaw]
jawHull=cv2.convexHull(jaw)
cv2.drawContours(frame, [jawHull], -1, (0, 255, 0), 1)
#眉毛
leftEyebrow=shape[left_leftEyebrow:left_rightEyebrow]
rightEyebrow=shape[right_leftEyebrow:right_rightEyebrow]
leftEyebrowHull=cv2.convexHull(leftEyebrow)
rightEyebrowHull=cv2.convexHull(rightEyebrow)
cv2.drawContours(frame, [leftEyebrowHull], -1, (0, 255, 0), 1)
cv2.drawContours(frame, [rightEyebrowHull], -1, (0, 255, 0), 1)

cv2.imshow("Frame", frame)
if cv2.waitKey(1) & 0xFF == ord('q'):
    break
cap.release()
cv2.destroyAllWindows()

```

執行結果：



5. 攝影機臉部偵測

與上述的使用模型套件相同，此範例需要使用 openCV 套件框選人臉的數值比數之特點範圍。

程式碼：

```
import dlib
import cv2
import imutils
# 開啟影片檔案
cap = cv2.VideoCapture(0)
# Dlib 的人臉偵測器
detector = dlib.get_frontal_face_detector()
# 以迴圈從影片檔案讀取影格，並顯示出來
while(cap.isOpened()):
    ret, frame = cap.read()
    # 偵測人臉
    face_rects, scores, idx = detector.run(frame, 0)
    # 取出所有偵測的結果
    for i, d in enumerate(face_rects):
        x1 = d.left()
        y1 = d.top()
        x2 = d.right()
```

```

y2 = d.bottom()
text = "%2.2f(%d)" % (scores[i], idx[i])
# 以方框標示偵測的人臉
cv2.rectangle(frame, (x1, y1), (x2, y2), (0, 255, 0), 4, cv2.LINE_AA)
# 標示分數
cv2.putText(frame, text, (x1, y1), cv2.FONT_HERSHEY_DUPLEX,
            0.7, (255, 255, 255), 1, cv2.LINE_AA)
# 顯示結果
cv2.imshow("Face Detection", frame)
if cv2.waitKey(1) & 0xFF == ord('q'):
    break
cap.release()
cv2.destroyAllWindows()

```

執行結果：



6. 辨識不同的臉：

人臉偵測只能得知圖片中是否有人臉存在，並測得人臉的位置；人臉辨識則會進一步指出人臉是屬於何人，或者比對兩個人臉圖形是否為同一人。使用 `dlib_face_recognition_resnet_model_v1.dat` 模型做預測。

程式碼：

```

import dlib, numpy

```

```

predictor = "shape_predictor_68_face_landmarks.dat"  #人臉 68 特徵點模型
recogmodel = "dlib_face_recognition_resnet_model_v1.dat"  #人臉辨識模型

detector = dlib.get_frontal_face_detector()  #偵測臉部正面
sp = dlib.shape_predictor(predictor)  #讀入人臉特徵點模型
facerec = dlib.face_recognition_model_v1(recogmodel)  #讀入人臉辨識模型

#取得人臉特徵點向量
def getFeature(imgfile):
    img = dlib.load_rgb_image(imgfile)
    dets = detector(img, 1)
    for det in dets:
        shape = sp(img, det)  #特徵點偵測
        feature = facerec.compute_face_descriptor(img, shape)  #取得 128 維特徵
向量
        return numpy.array(feature)  #轉換 numpy array 格式

#判斷是否同一人
def samePerson(pic1, pic2):
    feature1 = getFeature(pic1)
    feature2 = getFeature(pic2)
    dist = numpy.linalg.norm(feature1-feature2)  # 計算歐式距離,越小越像
    print("歐式距離={}".format(dist))
    if dist < 0.3:
        print("{} 和 {} 為同一個人!".format(pic1, pic2))
    else:
        print("{} 和 {} 不是同一個人!".format(pic1, pic2))
    print()

samePerson("media\\LINE_ALBUM_111.jpg", "media\\LINE_ALBUM_11106.jpg")  #不
同人
samePerson("media\\LINE_ALBUM_1125.jpg", "media\\LINE_ALBUM_1120610.jpg")
#同一人

```

執行結果：

歐式距離=0.2952158570099693

media\LINE_ALBUM_111.jpg 和 media\LINE_ALBUM_11106.jpg 為同一個人！

歐式距離=0.38403234459641017

media\LINE_ALBUM_1125.jpg 和 media\LINE_ALBUM_1120610.jpg 不是同一個人！

四、結論

使用了以上的四個模型做人臉辨識後對於論文所寫出來或者是所提出了的議題與 FaceNet 其概念幾乎完全一樣，但是用了更快的方法來達成而已，或許，這個專案未來可以試著用 FaceNet 來替代 Dlib 的方式進行辨識。對於人工智慧與大數據來說是個專題實作上的作品，對求職的人來說是不可或缺的作品，請於嘗試者做出來。

五、參考

巨匠電腦 python 深度學習開發

巨匠電腦 python 人工智慧整合開發