第三十二組小組專題紀錄

107370134 林紀緯

108AB0008 江大衛

108820016 郭梓琳

對兩類別影像進行 SIFT 特徵提取並送入SVM分類

小組專案要求

- 請在Python中導入資料集影像(可自行選擇，人、車、動物等皆可)，及與前者不同類別的資料集影像，分別對兩類別影像進行HOG特徵提取，並送入SVM分類器進行訓練，最後使用小組自行準備的影像進行辨識(兩類別測試影像各準備10張)。
- 根據你的辨識結果，辨識成功or失敗，來思考為何會有這樣的結果，提出改良方式並測試改良方式是否成功。
- 提示：可自行蒐集影像或是下載公開資料集或使用第三方函式庫，如：scikit-image或PIL或其它函式庫提供的影像資料集作為你自己的SVM訓練集。
- 參考資料(scikit-learn的人臉資料集、公開的車輛資料集)：
  - https://scikit-learn.org/stable/modules/generated/sklearn.datasets.fetch_lfw_people.html#sklearn.datasets.fetch_lfw_people
  - https://ai.stanford.edu/~jkrause/cars/car_dataset.html

SIFT 簡介與優點

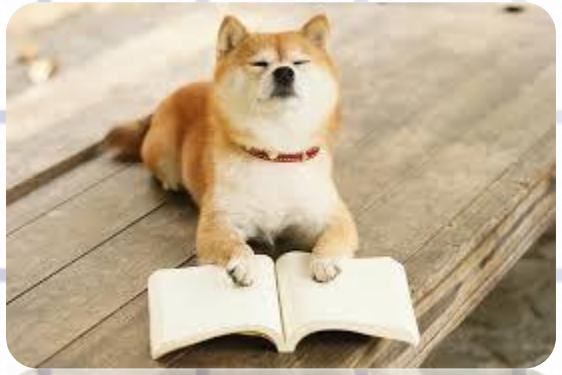
- SIFT(SCALE INVARIANT FEATURE TRANSFORM) 中文名為尺度不變特徵轉換
- 局部特徵、旋轉、尺度縮放、亮度變化，有很好的完整性；對視角變化、仿射變換、噪聲，也有一定的穩定度
- 保留圖片獨特性、訊息量大，適合大量特徵，可進行快速、準確的匹配
- 如果只有少數幾張圖片也可以產生大量的 SIFT 特徵
- 最佳化的 SIFT 不需要花費時間就能完成
- 方便與其他特徵向量結合

資料集的選擇

- 我們選擇兩組兩類別的資料集，一組是貓與狗，另一組是名偵探柯南與哆啦A夢，在HOG邊緣檢測中，我們得知由小貓與小狗的體型差不多，在HOG邊緣檢測中可能會有辨識錯誤的可能性。
- 於是我們這次在用名偵探柯南與哆啦A夢來進行辨識，他們的輪廓並不相同，多拉A夢輪廓較為圓滑、名偵探柯南的輪廓偏瘦，我們想從這兩個資料集來獲得新的啟發。
- 且SIFT能保留圖片獨特性、適合大量特徵，我們想知道SIFT保留的圖片獨特性、加上小狗與小貓的特徵較容易相同，名偵探柯南與哆啦A夢的特徵較不同，且多拉A夢有較多特徵點，想透過SIFT來讓機器辨識圖片，探討辨識是否成功。

資料集-貓與狗

- GOOGLE DRIVE 連結



資料集-貓與狗

- GOOGLE DRIVE 連結

Function 程式碼 Part1

- 由於裡面會有大量的重複程式碼，於是我就將重複的部分寫成FUNCTION，方便除錯也讓程式碼更加整潔。

```
import cv2
import numpy as np
from skimage.feature import hog
from sklearn import svm
from scipy.cluster.vq import kmeans, vq
from sklearn.model_selection import train_test_split
#如果懶惰的小夥伴們可以用 train_test_split 編寫

def read(path, data_amount): #讀取圖片用，path 是圖片路徑、data_amount 是數量
  data = list()
  for i in range(1, data_amount+1):
    image = cv2.imread(path % i) #讀資料
    image = cv2.resize(image, (349,256)) # resize
    data.append(image)
  return data #回傳 list

def sift(data): #SIFT 特徵取出 data 為讀取近 python 的原始圖片
  output = list()
  for image in data:
    sift = cv2.SIFT_create() #初始化特徵點
    kp, des = sift.detectAndCompute(image, None) #對圖片進行 sift 讀取
    output.append(des)
  return output #回傳 list
```

Function 程式碼 Part2

- 將每個 function 拆開貼上，會比較好閱讀。

```python
def sift(data): #SIFT 特徵取出 data 為讀取近 python 的原始圖片
    output = list()
    for image in data:
        sift = cv2.SIFT_create() #初始化特徵點
        kp, des = sift.detectAndCompute(image, None) #對圖片進行 sift 讀取
        output.append(des)
    return output #回傳 list
```

Function 程式碼 Part3

```
def kmeans_return_features(k, sift): #取出特徵，透過 kmeans 分辨
  #k 為要分辨的特徵類別數量，建議是資料類別乘以 10，sift 為擁有圖片特徵的 list
  descriptors = sift[0] #先給予一個值，以避免不能合併
  for it in sift[1:]:
    descriptors = np.vstack((descriptors, it))  #採用水平方式將陣列堆疊起來

  voc, variance = kmeans(descriptors,k,1)
  #透過 kmeans 將特徵進行分配，相似的放在一起，並分成 k 倍，只進行 1 次 iteratior
  #voc 回傳陣列，長度為 k，第 i 類有著相同的特徵，
  #variance 回傳觀察值與中心點，可能會有失真問題，如果不要失真或許可以使用 kmeans2 方法

  #features histogram
  im_features = np.zeros((len(sift), k), "float32") #生成一個全為 0 的陣列
  for i in range(len(sift)): #
    words, distance = vq(sift[i], voc)
    #將 voc 收集到的特徵，與圖片 sift[i] 與中心點進行比較，將最相似的特徵分配給適合的圖片
    #words 回傳陣列，接受此圖片最適合的所有特徵
    #distance 觀察值與中心點的距離，可能會有失真的可能性

    for j in words:  #將特徵傳給 im_features
      im_features[i][j] += 1 #表示此圖片的 j 特徵加一

  return im_features  #回傳 list
```

Main 程式碼 Part1

```python
import cv2
import numpy as np
from skimage.feature import hog
from sklearn.datasets import fetch_lfw_people
from sklearn import svm
from scipy.cluster.vq import kmeans, vq
from sklearn.model_selection import train_test_split
import hw06_fn #這是我自己寫的 functional progamming，請自行將上面的 function python code
#與此檔案放在同個資料夾底下

data_amount = 100 #每一個資料集總數 100
train_amount = 80 #訓練資料集筆數
test_amount = 20 #訓練資料集筆數

#如果想要讓電腦辨識貓跟狗，請使用這些程式碼
dogs = hw06_fn.read("../resize_dog/dog_%.3d.jpg", data_amount)
cats = hw06_fn.read("../resize_cat/cat_%.3d.jpg", data_amount)

#如果想要讓電腦辨識多拉A夢跟名偵探柯南，請使用這些程式碼
# dogs = hw06_fn.read("../Doraemon/images (%d).jpg", data_amount)
# cats = hw06_fn.read("../conan/images (%d).jpg", data_amount)

sift_dogs = hw06_fn.sift(dogs) #將資料送去 sift 特徵
sift_cats = hw06_fn.sift(cats)
```

Main 程式碼 Part2

```
train_sift = sift_dogs[:train_amount] + sift_cats[:train_amount] #產生訓練集
test_sift = sift_dogs[-test_amount:] + sift_cats[-test_amount:] #產生測試集

train_features = hw06_fn.kmeans_return_features(20, train_sift) #找出訓練集每張圖片特
徵
test_features = hw06_fn.kmeans_return_features(20, test_sift) #找出測試集每張圖片特徵
train_target = [0] * train_amount + [1] * train_amount #給予其正確的類別
test_target = [0] * test_amount + [1] * test_amount #給予其正確的類別

clf = svm.SVC(kernel="linear", C=1, gamma="auto") #透過 SVC 訓練
clf.fit(train_features, train_target) #開始進行訓練

print("accuracy") #準確率
print("train:", clf.score(train_features, train_target)) #訓練集分數
print("test:", clf.score(test_features, test_target)) #測試集分數
```

辨識結果-貓與狗

- 由於貓跟小狗的身材較為相像且亮度變化在這裡或許會容易誤導 SVM 進行分別，導致相同的光亮度的貓與狗不好進行分配，才會使其在訓練集的準確率就不高，在此情況下，相對測試集就不高。
- 可能需要白底 PNG 的大量貓狗照片，就能使得此機器學習辨識率更高。

```
In [95]: runfile('D:/NTUT/大二下/多媒體技術與應用/hw06/david/G02.py',
wdir='D:/NTUT/大二下/多媒體技術與應用/hw06/david')
Reloaded modules: hw06_fn
accuracy
train: 0.925
test: 0.725
```

辨識結果-多拉A夢與柯南

- 這組是比較正確的一組，其中我認為哆啦A夢與名偵探柯南本身的光影明亮度稍微不同、邊緣也有大不同，因此在交由 SVM 進行分辨時，較能夠分辨得出差異性。因此在訓練集時準確率可以達到 92%，在測試集也可以達到 72%，算是不錯的辨識率。
- 但認為可能還是需要白底的 PNG 照片，會讓他更為準確，我認為可能有些辨識失敗的原因是圖片有大量不相干的背景使得，那不相干的背景如果與另一類型的背景相似時，就有分辨失誤的可能性。

```
In [95]: runfile('D:/NTUT/大二下/多媒體技術與應用/hw06/david/G02.py',
wdir='D:/NTUT/大二下/多媒體技術與應用/hw06/david')
Reloaded modules: hw06_fn
accuracy
train: 0.925
test: 0.725
```