影像處理100天（python）

[Day01：OpenCV介紹 1](#_Toc27155809)

[Day02：OpenCV表示 2](#_Toc27155810)

[Day03：操作圖片&直方圖均衡（convertScaleAbs()） 2](#_Toc27155811)

[Day04：圖形矩陣操作&warpAffine()用法 2](#_Toc27155812)

[Day05：在圖上畫圖 3](#_Toc27155813)

[Day06：仿射(affine transformation)&getAffineTransform() 3](#_Toc27155814)

[Day07：透視Perspective transformation，遠近操作 3](#_Toc27155815)

[Day08：取特徵值，Sobel&Gaussian Blur 3](#_Toc27155816)

[Day09-Day10：SIFT，取特徵值 3](#_Toc27155817)

[Day11：CNN 4](#_Toc27155818)

[Day12：步長(strides)&填充(Padding) 4](#_Toc27155819)

[Day13：池化(Pooling) 5](#_Toc27155820)

[Day14：Batch Normalization(BN) 5](#_Toc27155821)

[Day15：總結 5](#_Toc27155822)

[環境安裝 5](#_Toc27155823)

[install pip 5](#_Toc27155824)

[Jupyterlab 6](#_Toc27155825)

[Open CV 6](#_Toc27155826)

[Keras 6](#_Toc27155827)

[Tensorflow 6](#_Toc27155828)

# Day01：OpenCV介紹

1. 影像彩色代表3個channels；灰階則為1個channel
   1. Img影像陣列：

|  |
| --- |
| [[[125 137 226] #B  [125 137 226] #G  [133 137 223] #R  ...  [122 148 230]  [110 130 221]  [ 90 99 200]] |

1. cv2.imread(‘path’)
2. cv2.imshow(‘title’,img)要呈現圖，一定要配合cv2.waitKey(0) & cv2.destroyAllWindows()

# Day02：OpenCV表示

1. OpenCV中的預設圖片表示順序是BGR（Blue,Green,Red）
2. Cv2.COLOR\_BGR2HSL轉換：

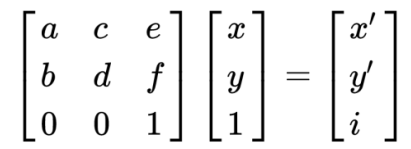
|  |
| --- |
| [[[ 4 176 162]  [ 4 176 162]  [ 1 178 149]  ...  [ 7 176 174]  [ 5 166 158]  [ 2 145 128]]  ...  [178 124 112]  [177 126 111]  [178 130 113]]] |

# Day03：操作圖片&直方圖均衡（convertScaleAbs()）

1. 3個channel轉換成陣列寫法：img[:,:,1]代表取得第1個channel，或用img[…,1]表示之。
2. Img = cv2.convertScaleAbs(img, alpha = alpha, bea=beta)，用來調整圖的對比及明亮度

# Day04：圖形矩陣操作&warpAffine()用法

1. 水平翻轉Flip：可直接對list操作，list[start:end:step]
2. Img[::-1,:,:]代表垂直翻轉，img[ h(y) : w(x) :c]，-1代表倒序
3. cv2.resize()介紹，cv2.INTER\_AREA縮小時使用；cv2.INTER\_CUBIC(速度較慢)/INTER\_LINEAR放大使用，效果較好，比較不會模糊
4. ad：縮放scale；bc：旋轉角度；ef：位移；00遠近操作
5. cv2.warpAffine()用法



# Day05：在圖上畫圖

# Day06：仿射(affine transformation)&getAffineTransform()

1. 一張圖片只需要1組（即2個）個3個點（共6個點），就可以建構出矩陣，透過矩陣來操作圖片

# Day07：透視Perspective transformation，遠近操作

1. getPerspectiveTransform()& warpPerspective()轉換

# Day08：取特徵值，Sobel&Gaussian Blur

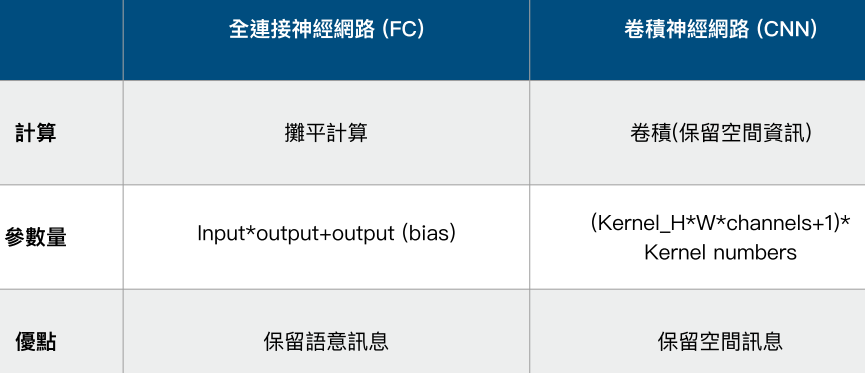
1. Filter=kernel，用來取特徵值
2. 操作方式，使用corss-correlation & convolution，convolution比較常用
3. Cv2.GaussianBlur()將圖模糊化；cv2.Sobel()邊緣檢測
4. 邊緣檢測（sobel）完畢後需要先將圖型做正規化，正整數，不得為負數

# Day09-Day10：SIFT，取特徵值

1. 使用邊緣檢測描取特徵，不是使用Sobel
2. 分為LoG & DoG(Difference of Gaussian)
   1. LoG，模糊化Gaussian Blur→二階導數取得邊緣
   2. DoG，先縮放→Gaussian Blur→合併取差值，無二階導數
3. 需將圖片轉成灰階
4. Feature Matching特徵配對(ratio test)最好設定在0.7-0.8之間

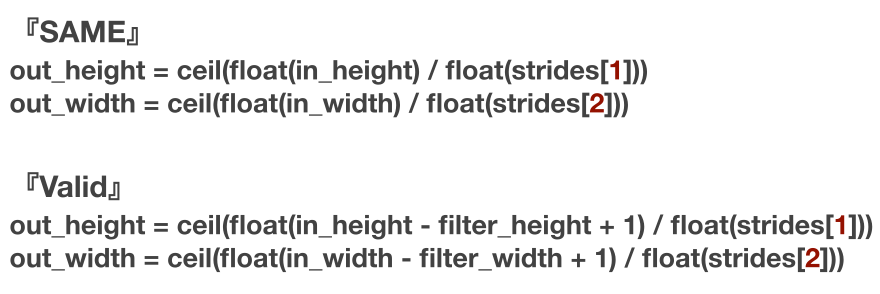
# Day11：CNN

1. 卷積核：Filter/Kernel/特徵偵測
2. 常見的kernel\_size=3\*3、5\*5
3. 特徵接受域(Receptive field)，換言之，看到特徵值大小
4. 張數通常在16、32、64



# Day12：步長(strides)&填充(Padding)

1. Strides影響Feature map
2. Padding週邊補0值，same指取完特徵值後，Feature map一樣大；valid則會縮小



1. Feature map的尺寸：



# Day13：池化(Pooling)

1. Max pooling(MaxPooling2D)，最常使用此方法，降低運算及加快收斂
2. Average pooling：強調特徵平滑性，所以容易把重要的特徵移除
3. CNN最後在output分類時，會用FC

# Day14：Batch Normalization(BN)

1. Activation的公式中，sigmoid方式最容易造成梯度消失，所以會讓程式跑不下去
2. Activation的公式盡量使用relu
3. 若使用sigmoid，Batch Normalization的function必須放在Activation(sigmoid)之前，效果最好
4. 若使用Relu，BN放在前後都無所謂了
5. Mini batch：一般都用32，所以很多範例都會看見batch\_size=32

# Day15：總結

1. 資料取特徵(卷積層)：Convolution + (Max Pooling/Agv Pooling)+ Activation + BN
2. 最後將特徵展開與Fully Connection連結，使用方法：Flatten or Global Average Pooling(GAP)
   1. Fully Connection的Layer層愈少愈好，一般用於最後output分類時使用 (softmax分類機率)
   2. Softmax，將所有分類的機率總合為1，例如：結果有3種，貓：0.3；狗：0.4；豬：0.3，其總合為1，代表成為狗的機率較高。
3. 影像：彩色為3 Channels，灰色為1 Channel

# 環境安裝

## install pip

download:get-pip.py

#>python get-pip.py

## Jupyterlab

pip install jupyterlab

## Open CV

pip install –user opencv-python==3.4.2.16

pip install –user opencv-contrib-python==3.4.2.16

## Keras

pip install Keras

## Tensorflow

pip install tensorflow

注意：乃森建議安裝1.8.0版，但python 3.7無1.8.0版，故可直接安裝新版即可

pip install scikit-learn (day15)