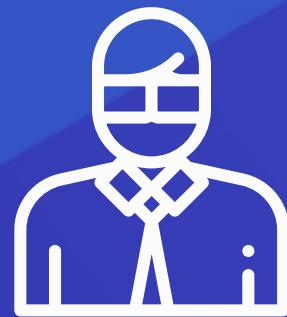




Day 94

深度學習應用卷積神經網路

卷積神經網路 - 卷積(Convolution)層與參數調整



陳宇春

出題教練



知識地圖 卷積網路套件練習

卷積(Convolution)層與參數調整

深度神經網路

Supervised Learning Deep Neural Network (DNN)

簡介 Introduction

套件介紹 Tools: Keras

組成概念 Concept

訓練技巧 Training Skill

應用案例 Application

卷積神經網路

Convolutional Neural Network (CNN)

簡介 introduction

套件練習 Practice with Keras

訓練技巧 Training Skill

電腦視覺 Computer Vision

卷積類神經網路套件練習

Practice CNN with Keras

建立 CNN 模型

Keras 中的 CNN Layers

使用 CNN 完成 CIFAR-10 預測

本日知識點目標

- 了解CNN Flow
- 卷積 (Convolution) 的 超參數(Hyper parameter)設定與應用

卷積 (Convolution) 的 超參數(Hyper parameter)

- 卷積 (Convolution) 的 超參數(Hyper parameter)

- 卷積內核 (kernel)
- Depth (kernels的總數)
- Padding (是否加一圈0值的pixel)
- Stride(選框每次移動的步數)

```
model.add(Convolution2D(25, 3, 3, input_shape=(1, 28, 28)))
```

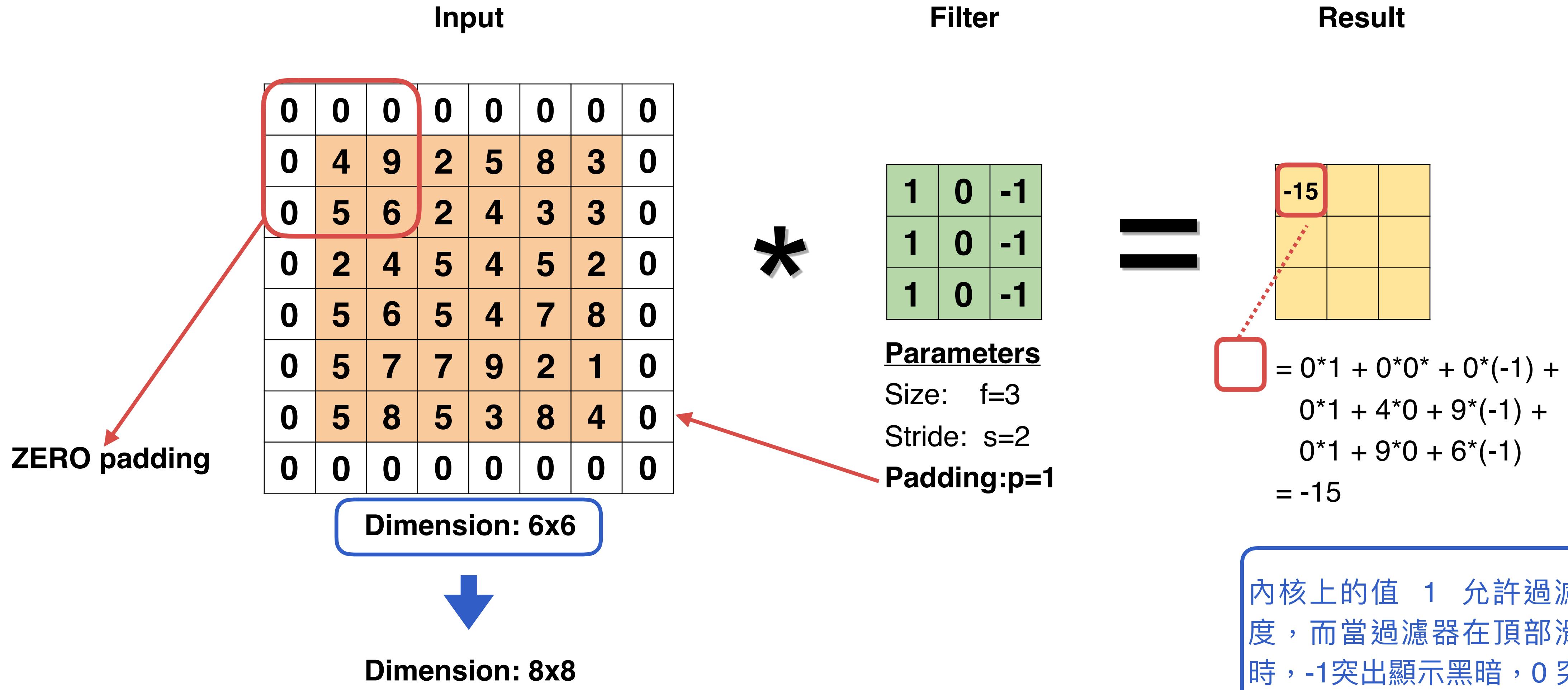
(卷積引數：filter數量，filter長，filter寬，輸入影象的三維 (RGB，長，寬))

給定了 kernels 的 深度與維度

填充或移動步數(Padding/Stride)的用途

- RUN 過 CNN，兩個問題
 - 是不是卷積計算後，卷積後的圖是不是就一定只能變小?
 - 可以選擇維持一樣大
 - 卷積計算是不是一次只能移動一格?
- 控制卷積計算的圖大小 - Valid and Same convolutions
 - **padding = ‘VALID’** 等於最一開始敘述的卷積計算，圖根據 filter 大小和 stride 大小而變小
 - $\text{new_height} = \text{new_width} = (W - F + 1) / S$
 - **padding = ‘Same’**的意思是就是要讓輸入和輸出的大小是一樣的
 - pad=1，表示圖外圈額外加 1 圈 0，假設 pad=2，圖外圈額外加 2 圈 0，以此類推

加了填充(padding)之後

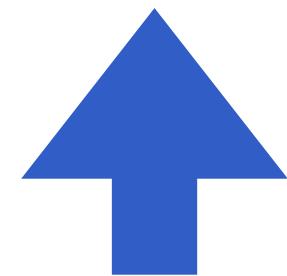


舉例

```
Model.add(Convolution2D(32, 3, 3), input_shape=(1, 28, 28), strides=2,  
padding='valid')
```

- 這代表卷積層 filter 數設定為 32，filter 的 kernel size 是 3，步伐 stride 是 2，pad 是 1。
 - pad = 1，表示圖外圈額外加 1 圈 0，假設 pad = 2，圖外圈額外加 2 圈 0，以此類
(1)kernel size是 3 的時候，卷積後圖的寬高不要變，pad 就要設定為 1
(2)kernel size是 5 的時候，卷積後圖的寬高不要變，pad 就要設定為 2

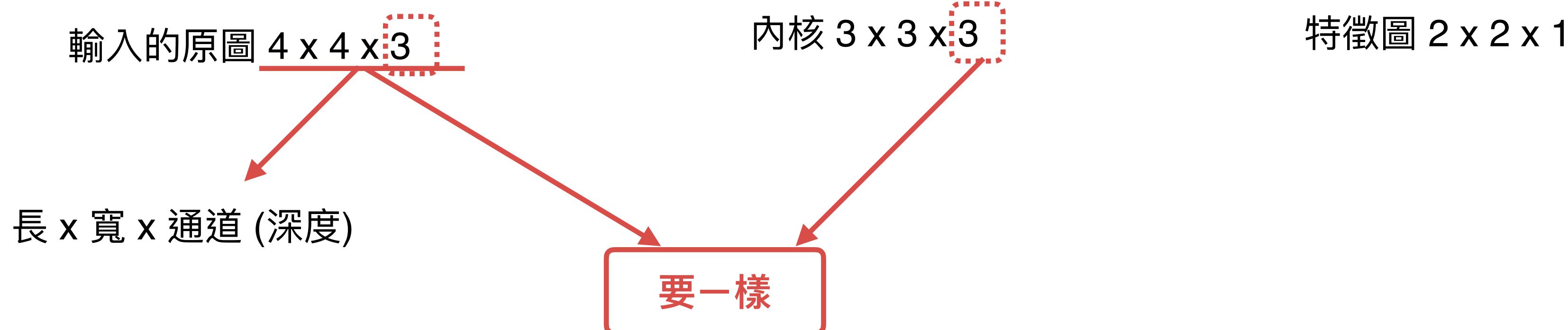
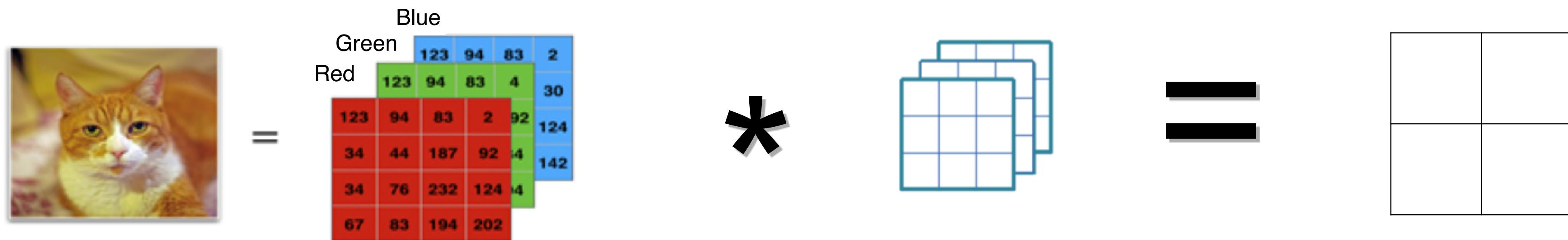
Input = 8x8
Kernel =32x3x3
Pad = p
Output data?



$$(New)^n = \frac{n+2p-f}{S}$$

多個通道(channels)的卷積作法

- 考慮多種顏色- 針對 RGB
- 會有3個對應的 kernel



多個通道(channels)的卷積作法

0	0	0	0	0	0	0	...
0	156	155	156	158	158	158	...
0	153	154	157	159	159	159	...
0	149	151	155	158	159	159	...
0	146	146	149	153	158	158	...
0	145	143	143	148	158	158	...
...

Input Channel #1 (Red)

0	0	0	0	0	0	0	...
0	167	166	167	169	169	169	...
0	164	165	168	170	170	170	...
0	160	162	166	169	170	170	...
0	156	156	159	163	168	168	...
0	155	153	153	158	168	168	...
...

Input Channel #2 (Green)

0	0	0	0	0	0	0	...
0	163	162	163	165	165	165	...
0	160	161	164	166	166	166	...
0	156	158	162	165	166	166	...
0	155	155	158	162	167	167	...
0	154	152	152	157	167	167	...
...

Input Channel #3 (Blue)

-1	-1	1
0	1	-1
0	1	1

Kernel Channel #1



308

+

1	0	0
1	-1	-1
1	0	-1

Kernel Channel #2



-498

0	1	1
0	1	0
1	-1	1

Kernel Channel #3



164

+

-25				...
				...
				...
				...
...

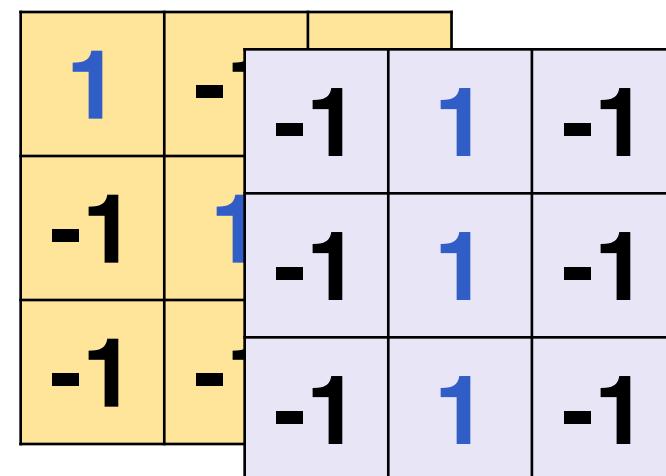
Bias = 1

Output

前述流程 / python程式 對照

Filter number 由少
input side 到多

```
model2.add( Convolution2D( 25, 3, 3,  
                           input_shape=(1, 28, 28) ) )
```

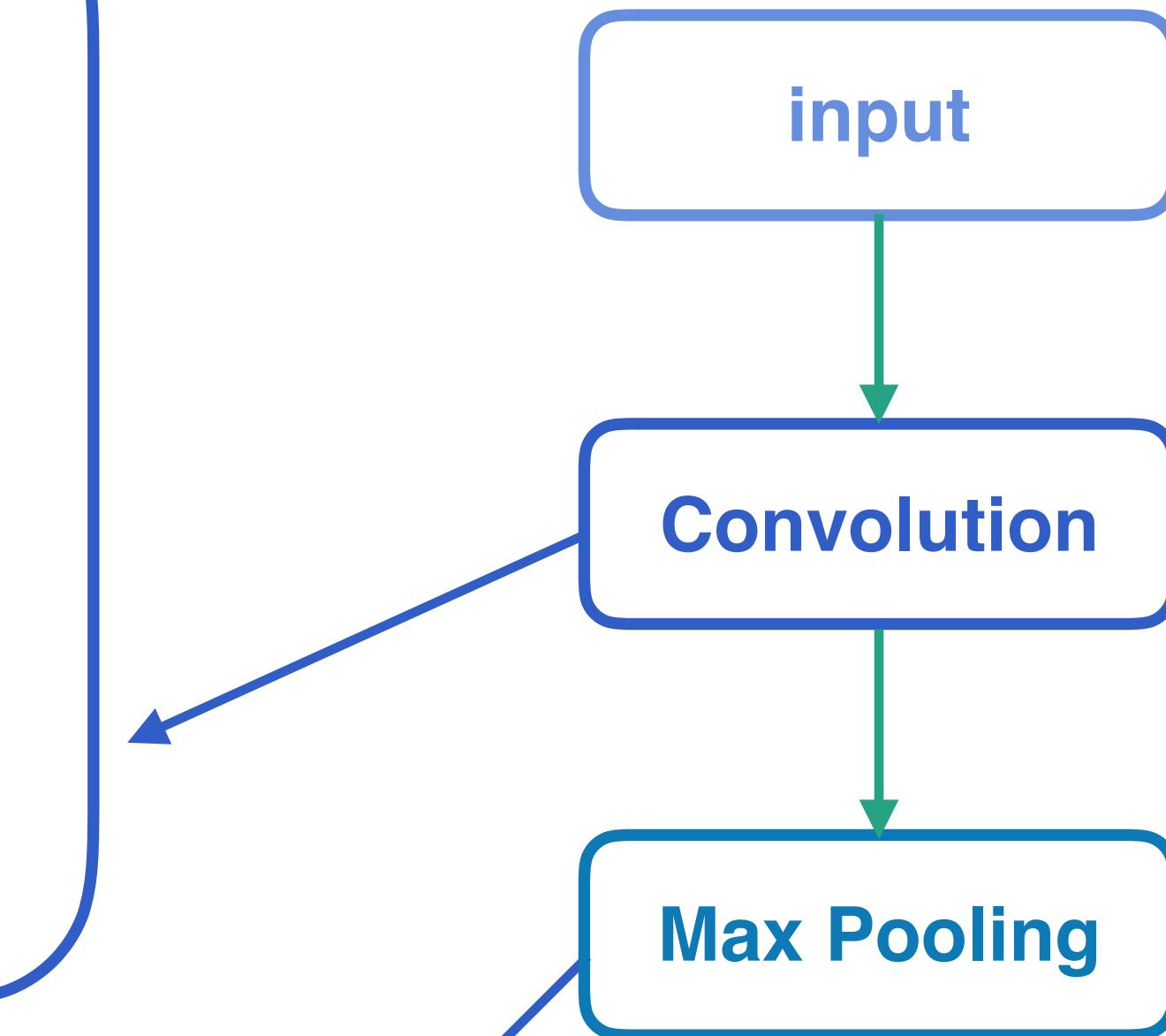


.....There are 25 3x3 filters.

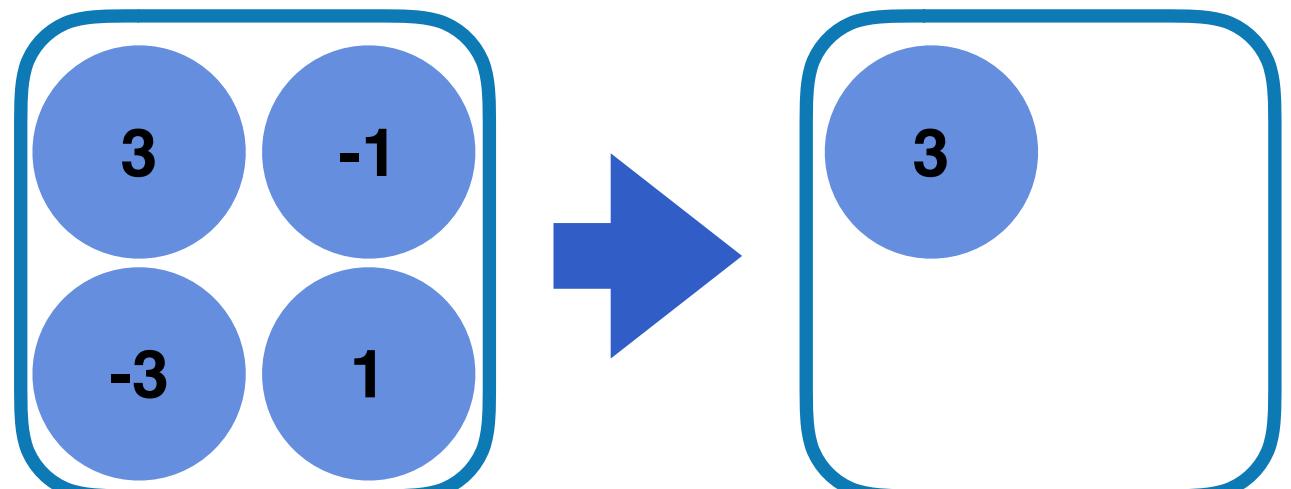
Input_shape = (28 , 28 , 1)

28 x 28 pixels

1: black/white, 3 : RGB



```
model2.add(MaxPooling2D( (2, 2)))
```



前述流程 / python 程式 對照

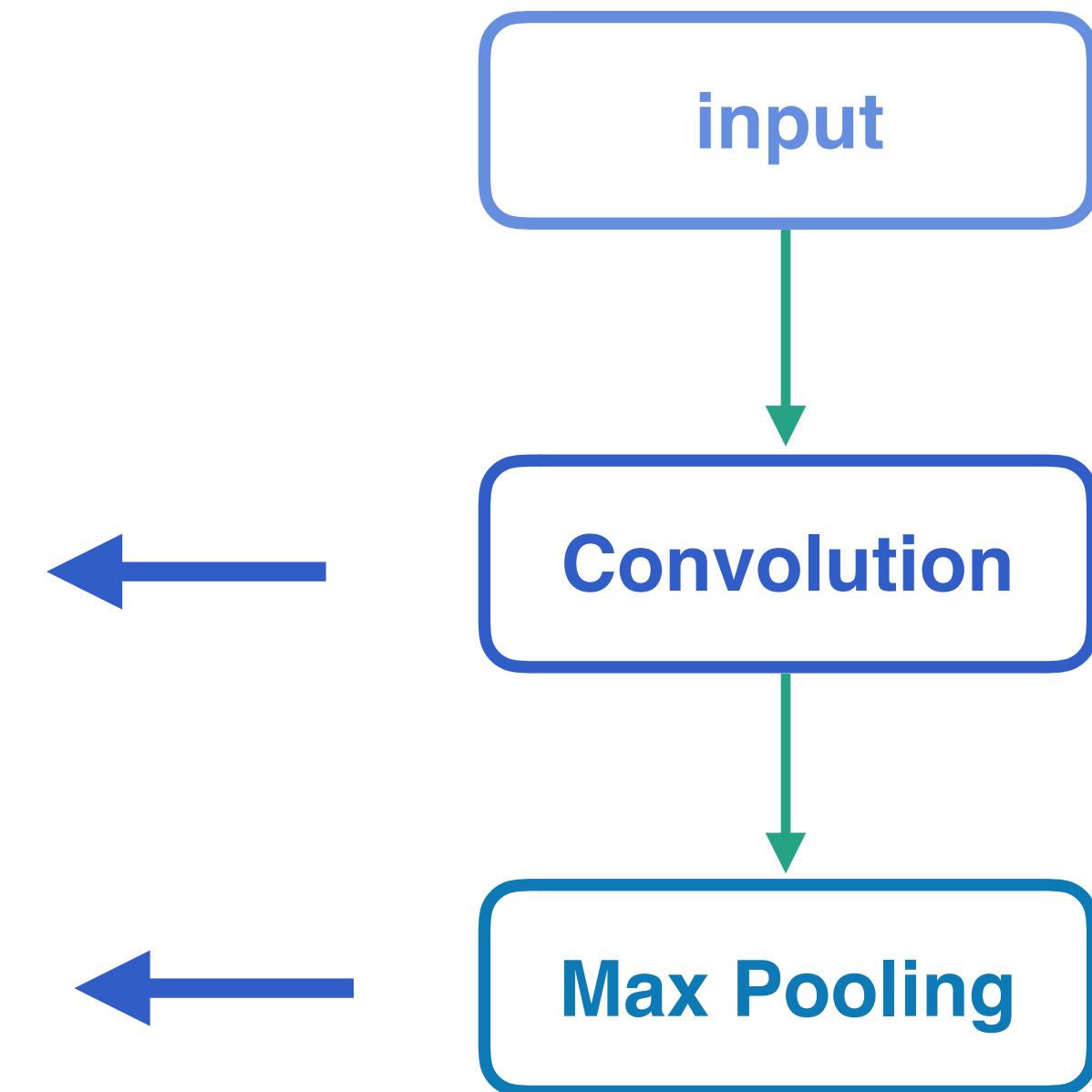
每個 filter 包含多少個參數？

1 x 28 x 28

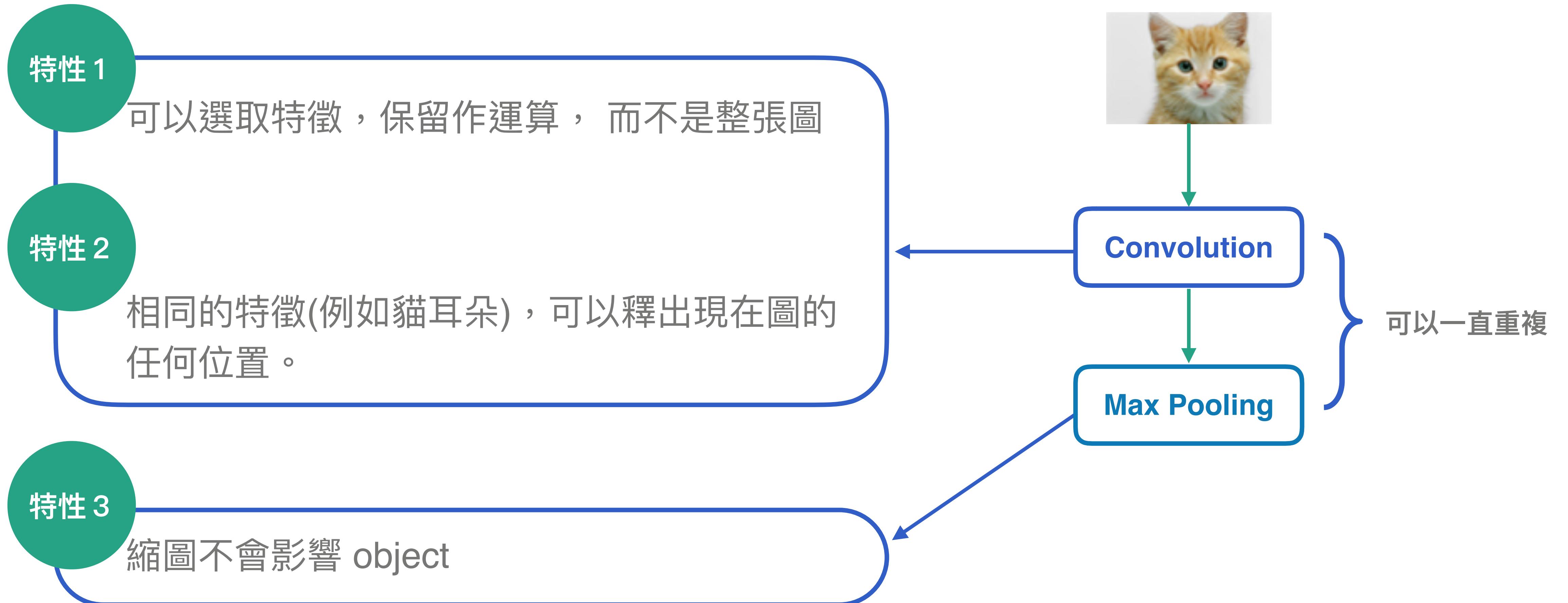
```
model2.add( Convolution2D( 25, 3, 3,  
                           input_shape=(28, 28, 1)) )
```

25 x 26 x 26

```
model2.add(MaxPooling2D( (2, 2)))
```

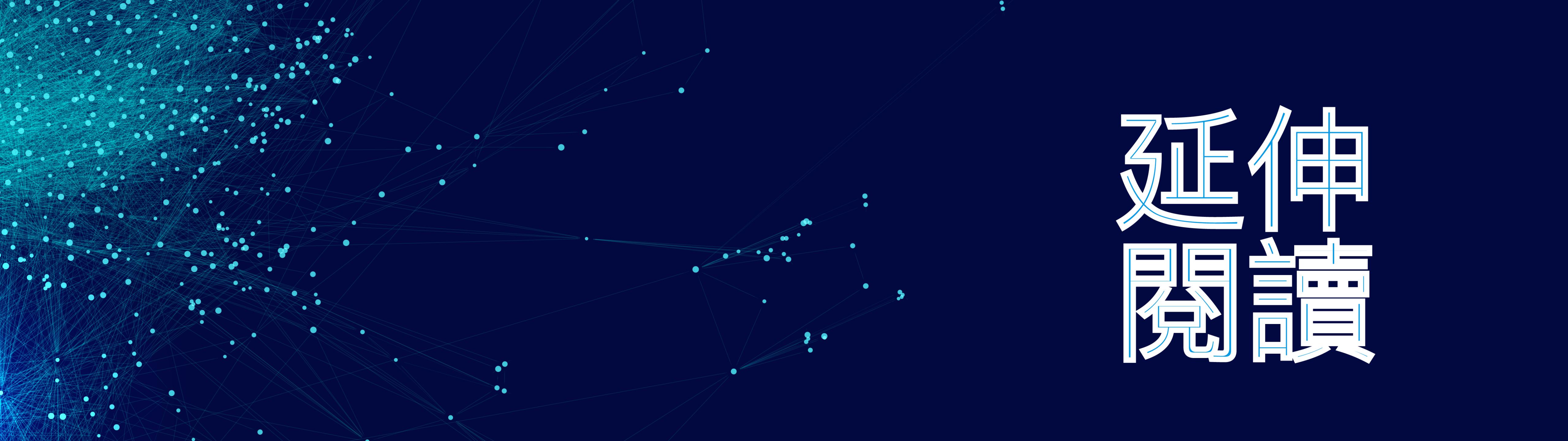


重要知識點複習：卷積(Convolution) 跟池化(Pooling)



卷積神經網路(CNN)特性

- 適合用在影像上
 - 因為 fully-connected networking 如果用在影像辨識上，會導致參數過多 (因為像素很多)，導致 over-fitting
 - CNN 針對影像辨識的特性，特別設計過，來減少參數
 - Convolution：學出 filter 比對原始圖片，產生出 feature map (也當成 image)
 - Max Pooling：將 feature map 縮小
 - Flatten：將每個像素的 channels (有多少個 filters) 展開成 fully connected feedforward network

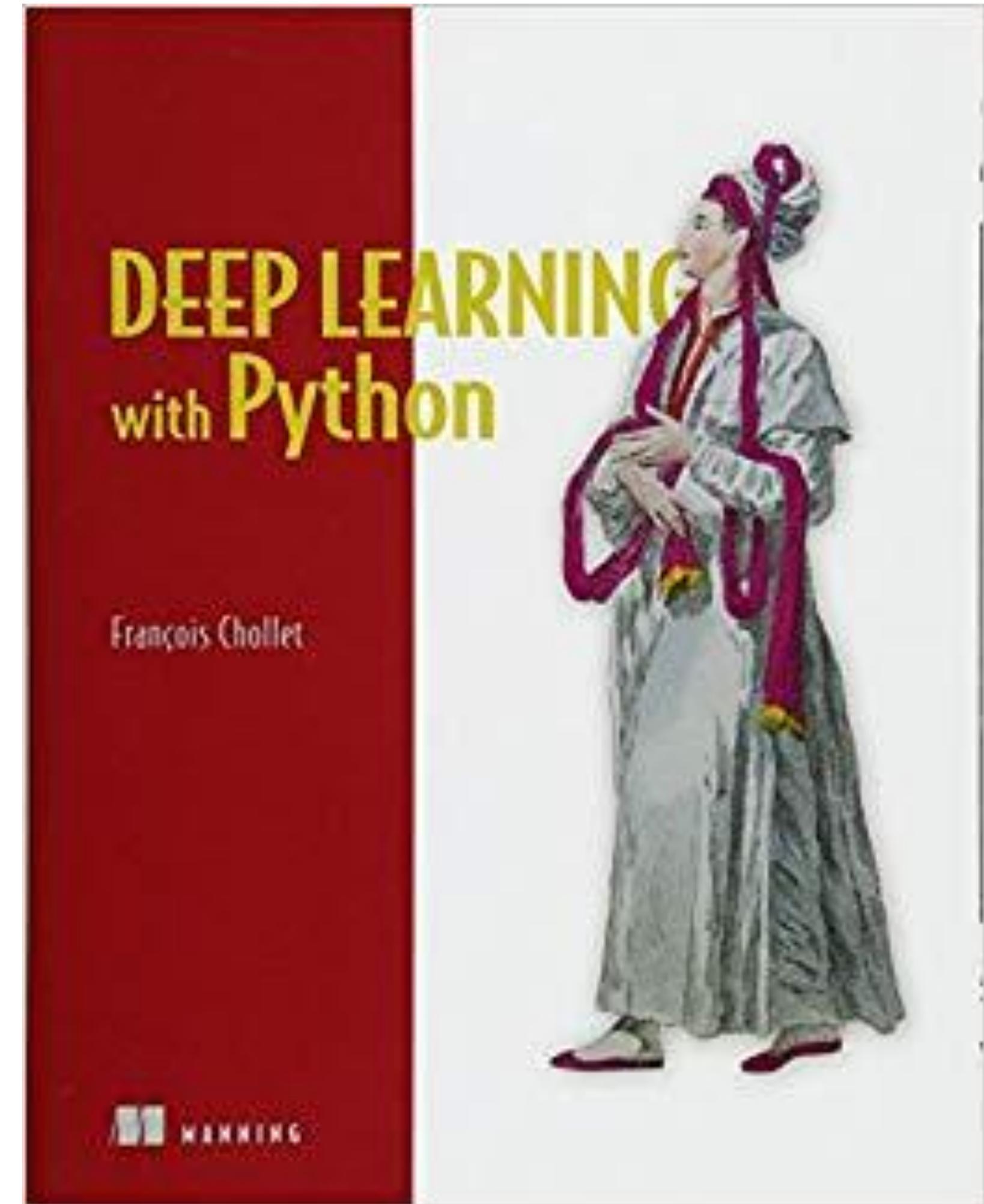


延伸 閱讀

除了每日知識點的基礎之外，推薦的延伸閱讀能補足學員們對該知識點的了解程度，建議您解完每日題目後，若有
多餘時間，可再補充延伸閱讀文章內容。

推薦延伸閱讀

- 介紹三種視覺化方法：
 1. 卷積核輸出的視覺化(Visualizing intermediate convnet outputs (intermediate activations))，即視覺化卷積核經過啟動之後的結果。能夠看到圖像經過卷積之後結果，幫助理解卷積核的作用
 2. 卷積核的視覺化(Visualizing convnets filters)，說明我們理解卷積核是如何感受圖像的
 3. 熱度圖視覺化(Visualizing heatmaps of class activation in an image)，通過熱度圖，瞭解圖像分類問題中圖像哪些部分起到了關鍵作用，同時可以定位圖像中物體的位置。
- 卷積核輸出的視覺化(Visualizing intermediate convnet outputs (intermediate activations))
 - 想法很簡單：向CNN輸入一張圖像，獲得某些卷積層的輸出，視覺化該輸出。
 - 圖片來源：Deep Learning with Python



推薦延伸閱讀

- 處理影像的利器 - 卷積神經網路(Convolutional Neural Network)

每個隱藏層的神經元就只跟Input矩陣(11, 11)作運算，運算負擔就明顯減輕了，另外，還有一個假設，稱為『共享權值』(Shared weights)，就是每一個『感受野』對下一隱藏層均使用相同的一組權重(Weight Matrix)，請參閱下圖，這樣要推估的權重數量減少，又可以減輕運算的負擔，所以，運用卷積層的目的就是針對圖像或語言的特性，簡化計算的過程，進而縮短運算的時間。

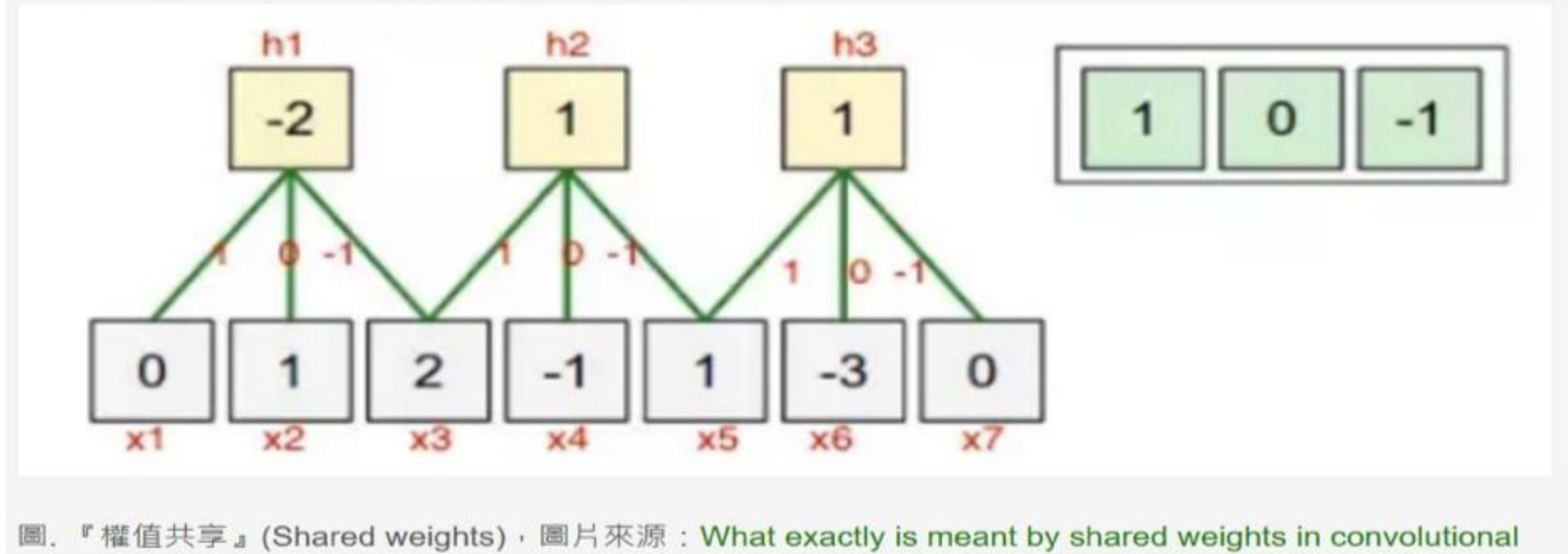
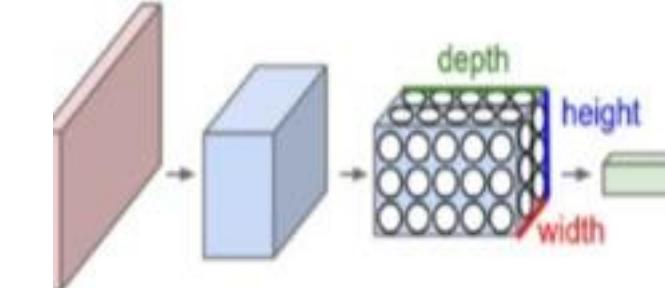


圖.『權值共享』(Shared weights)，圖片來源：[What exactly is meant by shared weights in convolutional neural network?](#)

- 透過多層卷積/池化，萃取特徵當作 Input，再接至一到多個完全連接層，進行分類，這就是CNN的典型作法，下一篇我們就用 CNN 來作阿拉伯數字的辨識，看看有甚麼不同，緊接著，我們再介紹兩個 CNN 應用，說明 Neural Network 不是只能作分類而已。
- 參考來源



推薦延伸閱讀

- An Intuitive Explanation of Convolutional Neural Networks

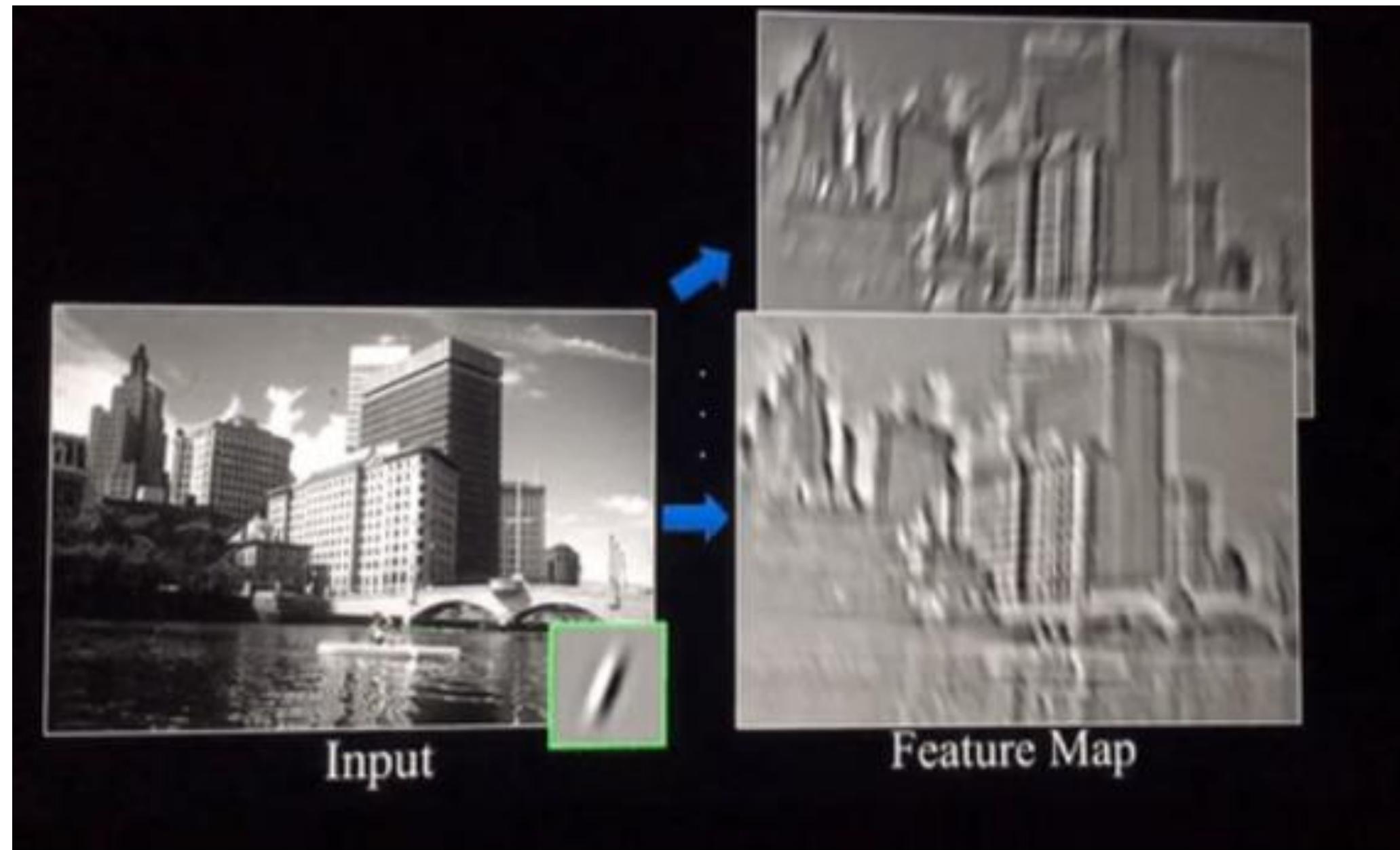


Figure 6: The Convolution Operation. Source [9]

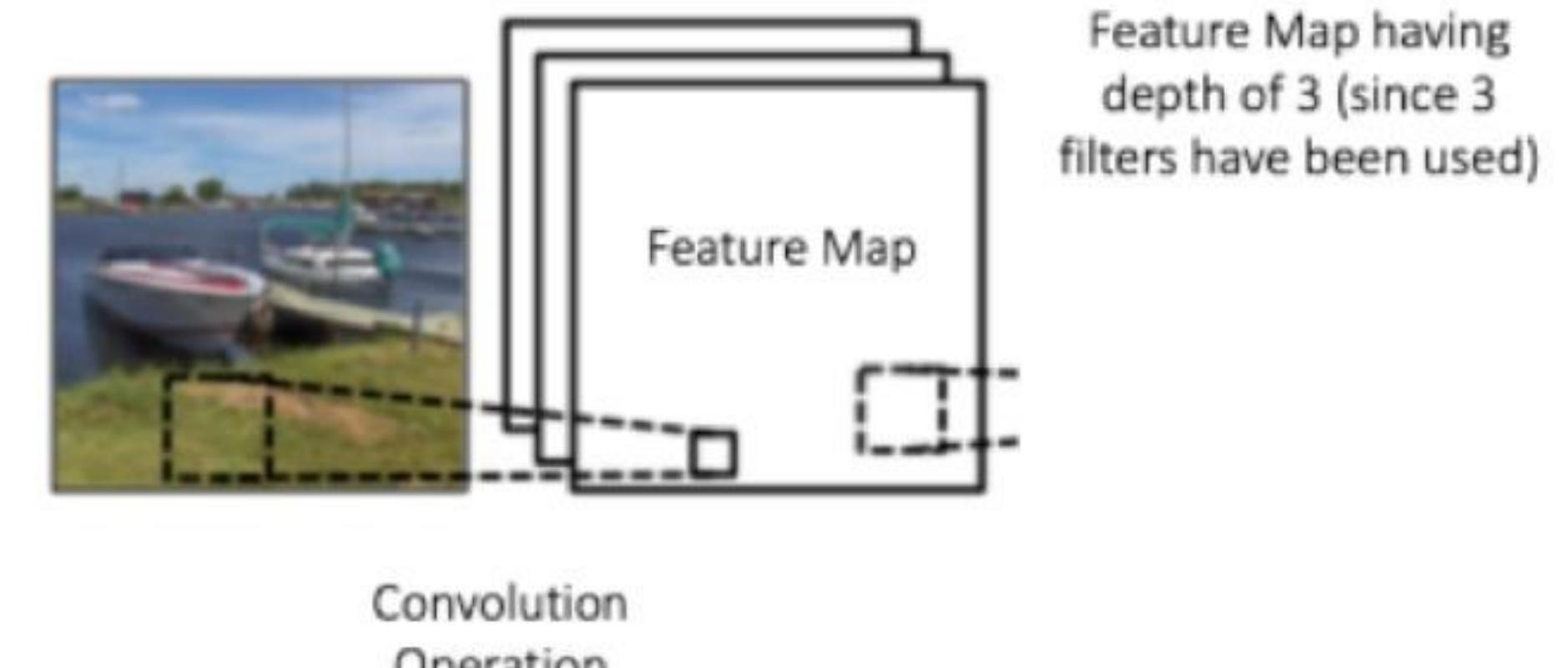


Figure 7



解題時間

It's Your Turn

請跳出PDF至官網Sample Code & 作業
開始解題

