

Day 93

深度學習應用卷積神經網路

卷積神經網路架構細節



本日知識點目標



目標
知識點

介紹 CNN



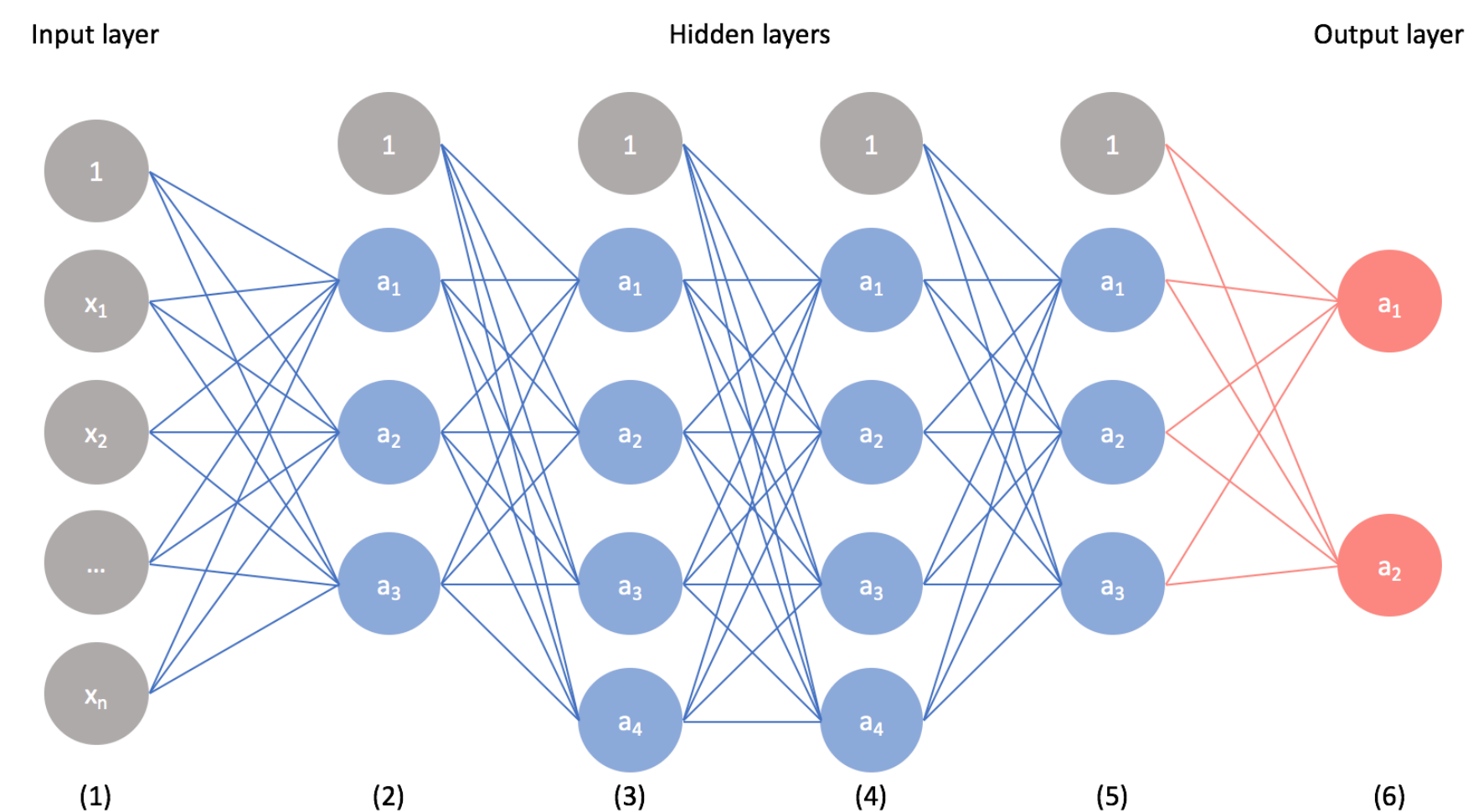
獲得
知識點

完成今日課程後你應該可以了解

- CNN，說明 CNN 為何適用於 Image 處理
- 卷積層中的捲積過程是如何計算的？為什麼卷積核是有效的？

深度神經網路的特例 – CNN（卷積網路）

- 傳統的DNN（即Deep neural network）最大問題在於它會忽略資料的形狀。
 - 例如，輸入影像的資料時，該data通常包含了水平、垂直、color channel等三維資訊，但傳統DNN的輸入處理必須是平面的、也就是須一維的資料。
 - 一些重要的空間資料，只有在三維形狀中才能保留下來。
 - RGB不同的channel之間也可能具有某些關連性、而遠近不同的像素彼此也應具有不同的關聯性

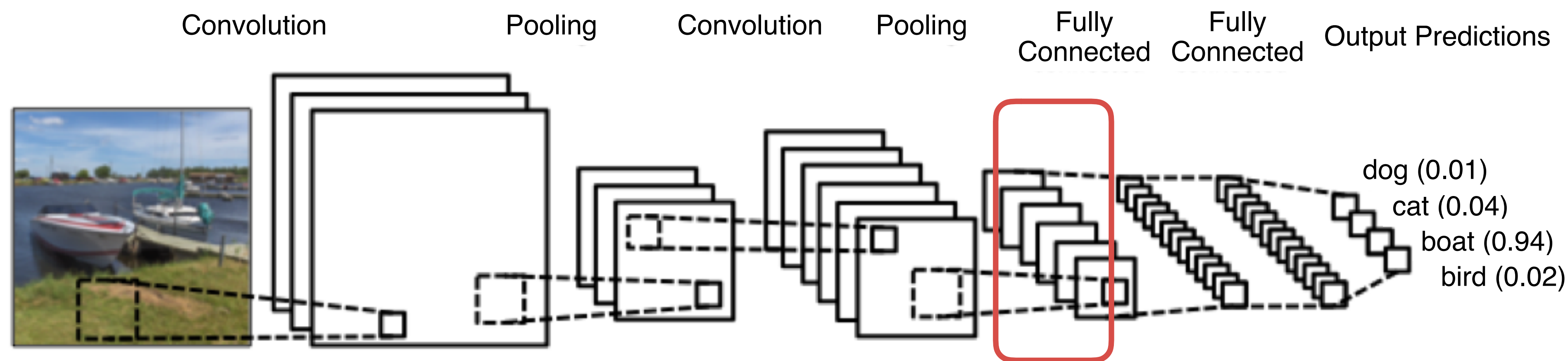


深度神經網路的特例 – CNN (卷積網路)

- Deep learning中的CNN較傳統的DNN多了Convolutional（卷積）及池化（Pooling）兩層layer，用以維持形狀資訊並且避免參數大幅增加。
- Convolution原理是透過一個指定尺寸的window，由上而下依序滑動取得圖像中各局部特徵作為下一層的輸入，這個sliding window在CNN中稱為稱為Convolution kernel (卷積內核)
- 利用此方式來取得圖像中各局部的區域加總計算後，透過 ReLU activation function輸出為特徵值再提供給下一層使用

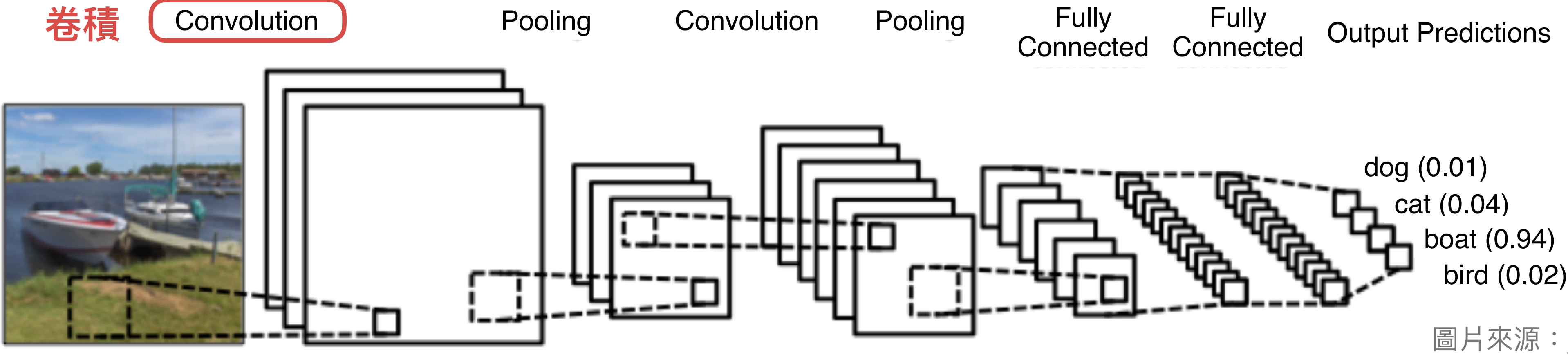
卷積網路的組成

- Convolution Layer 卷積層
- Pooling Layer 池化層
- Flatten Layer 平坦層
- Fully connection Layer 全連接層



Flatten Layer 平坦層

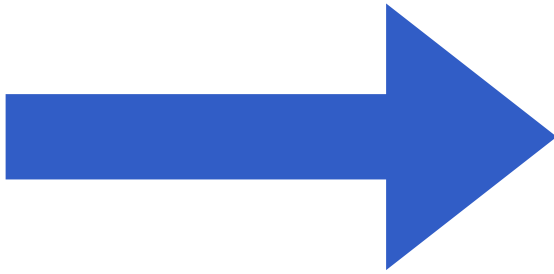
卷積如何做



人眼
取一個數字「0」



取出pixel 值：
範圍：0~255



0~ 沒有
255 ~ 全黑

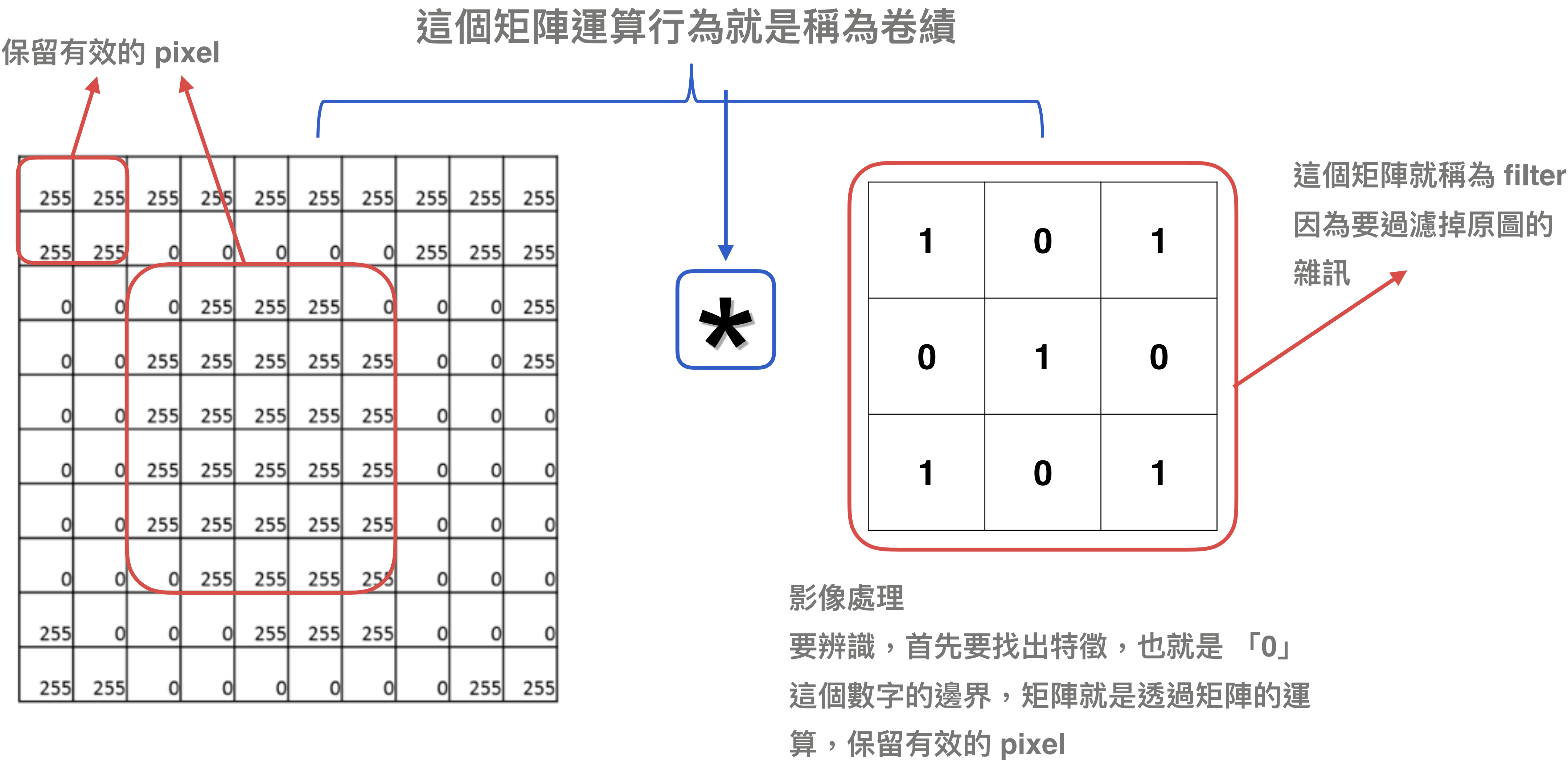
取每點pixel 值的「矩陣」

255	255	255	255	255	255	255	255	255	255
255	255	0	0	0	0	0	255	255	255
0	0	0	255	255	255	0	0	0	255
0	0	255	255	255	255	255	0	0	255
0	0	255	255	255	255	255	0	0	0
0	0	255	255	255	255	255	0	0	0
0	0	255	255	255	255	255	0	0	0
0	0	0	255	255	255	255	0	0	0
255	0	0	0	255	255	255	0	0	0
255	255	0	0	0	0	0	0	255	255

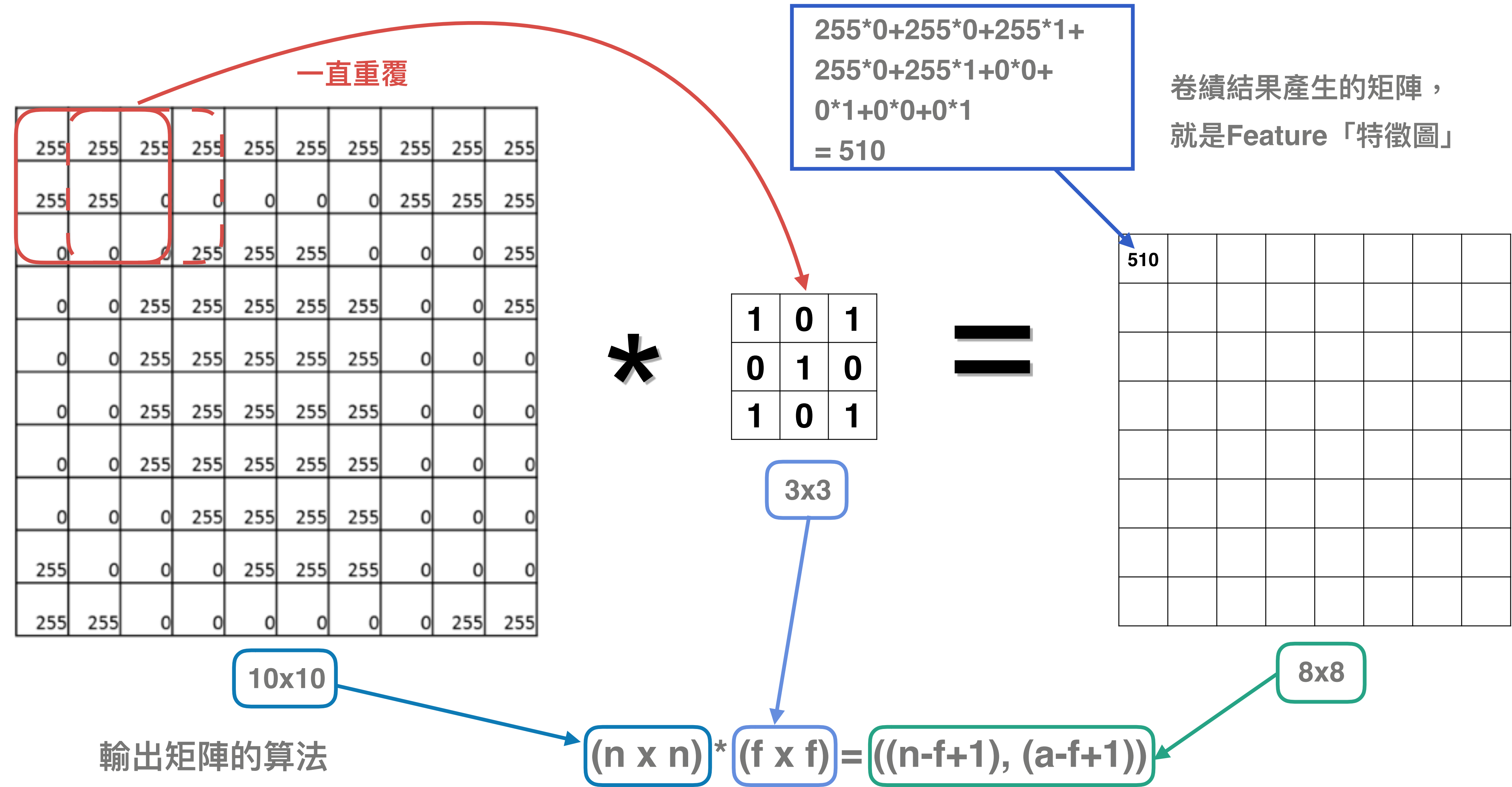
數位

- 丟一張圖給 Neural Network (NN) 做辨識，NN 看到的跟人眼看到是不一樣的

卷積如何做

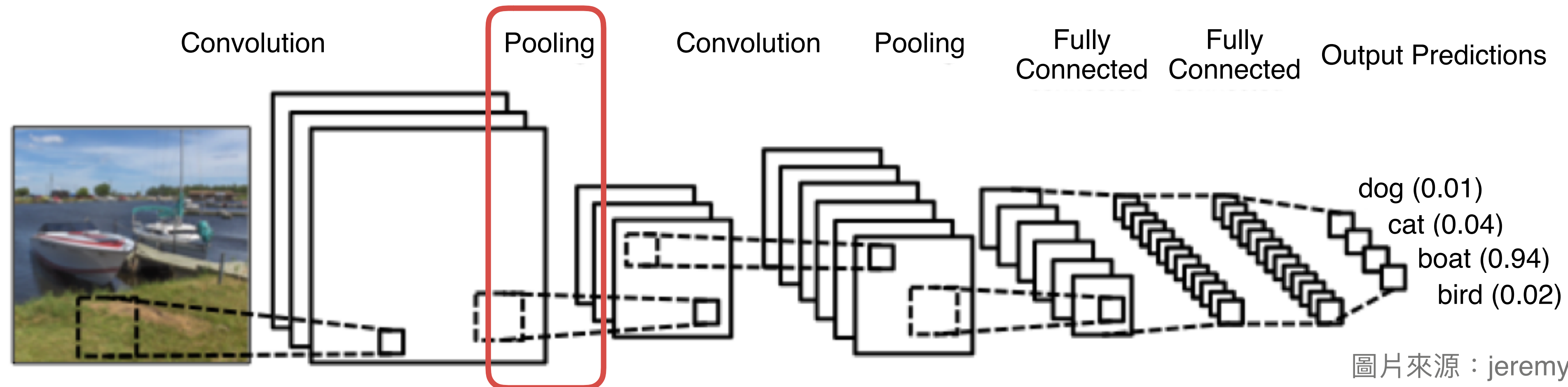


卷積如何做



Pooling Layer

- Pooling layer 稱為池化層，它的功能很單純，就是將輸入的圖片尺寸縮小（大部份為縮小一半）以減少每張 feature map 維度並保留重要的特徵，其好處有：
 - 特徵降維，減少後續 layer 需要參數。
 - 具有抗干擾的作用：圖像中某些像素在鄰近區域有微小偏移或差異時，對 Pooling layer 的輸出影響不大，結果仍是不變的。
 - 減少過度擬合 over-fitting 的情況。



Flatten – 平坦層

- Flatten：將特徵資訊丟到 **Full connected layer** 來進行分類，其神經元只與上一層 kernel 的像素連結，而且各連結的權重在同層中是相同且共享的

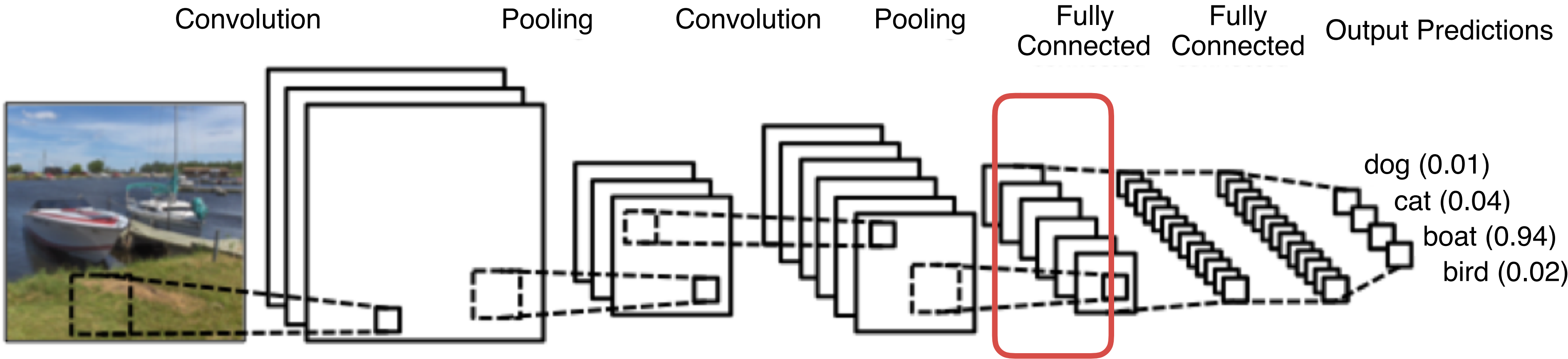
1	0	1
0	1	0
1	0	1

Featruue Map

Flattening



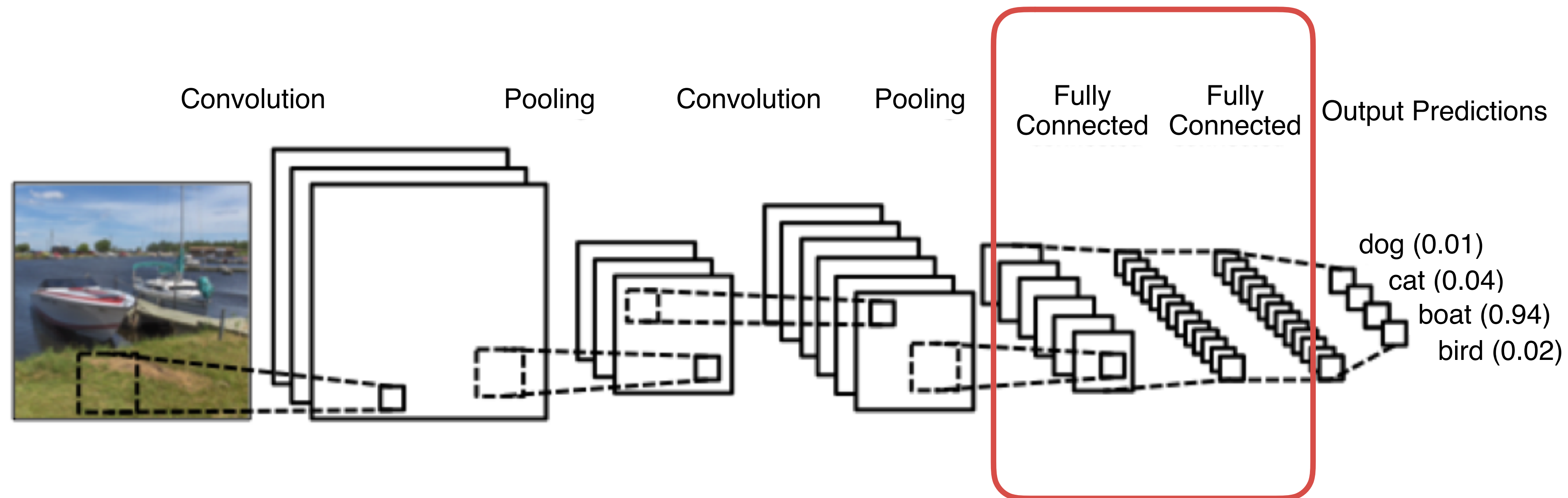
1
1
0
4
2
1
0
2
1



Flatten Layer 平坦層

全連接層 - Fully connected layers

- 卷積和池化層，其最主要的目的分別是提取特徵及減少圖像參數，然後將特徵資訊丟到 **Full connected layer** 來進行分類，其神經元只與上一層kernel的像素連結，而且各連結的權重在同層中是相同且共享的



圖片來源：jeremyjordan.me

前述流程 / python程式 對照

用 Keras 程式碼簡單模擬 CNN 在 MNIST 資料集上的工作流程，便於直觀理解

● 卷積層

```
#建立一個序列模型
model = models.Sequential()
#建立一個卷積層，32 個內核，內核大小 3x3，
#輸入影像大小 28x28x1
model.add(layers.Conv2D(32, (3, 3), input_shape=(28, 28, 1)))
```

● 池化層

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

(池化引數：劃分的尺寸)

```
#建立第二個卷積層，池化層，
#請注意，不需要再輸入 input_shape
model.add(layers.Conv2D(25, (3, 3)))
```

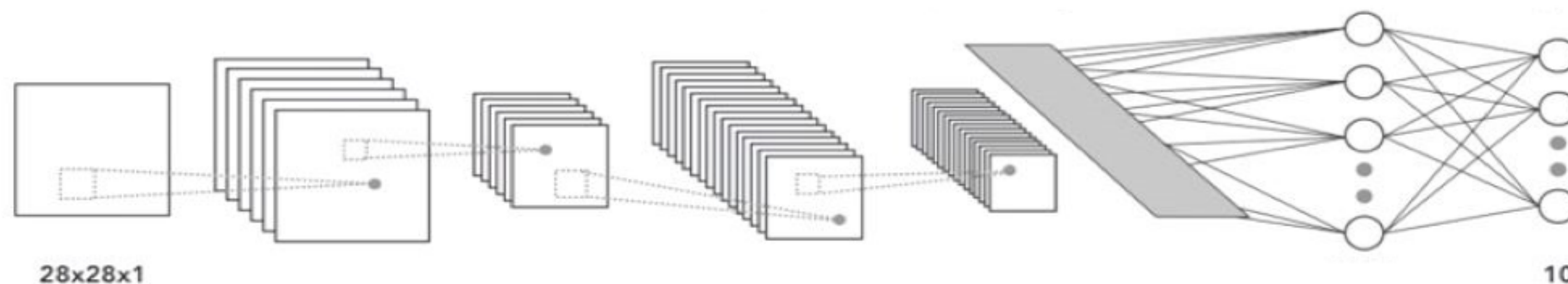

前述流程 / python程式 對照

- 平坦層

```
model.add(Flatten())
```

- 投入全連線網路與輸出

```
model.add(Dense(output_dim=100))  
model.add(Activation('relu'))  
model.add(Dense(output_dim=10))  
model.add(Activation('softmax'))
```



重要知識點複習：What is Convolution

- 卷積是圖像的通用濾鏡效果。
 - 將矩陣應用於圖像和數學運算，由整數組成
 - 卷積是通過相乘來完成的
 - 像素及其相鄰像素 矩陣的顏色值
 - 輸出是新修改的過濾圖像

$$\text{Image} * \text{Kernel} = \text{New Image}$$

- Kernel 內核 (or 過濾器 filter) :
 - 內核（通常）很小 用於的數字矩陣 圖像卷積。
 - 「不同大小的內核」包含不同的模式 數字產生不同的結果
 - 在卷積下。 「內核的大小是任意的」但經常使用 3x3

Convolutional over volume

- **input 上的變化**

- 單色圖片的 input，是 2D，Width x Height
- 彩色圖片的 input，是 3D，Width x Height x Channels

- **filter 上的變化**

- 單色圖片的 filter，是 2D, Width x Height
- 彩色圖片的 filter，是 3D, Width x Height x Channels
但2個 filter 的數值是一樣的

- **feature map 上的變化**

- 單色圖片，一個 filter，是 2D, Width x Height
多個 filters，Width x Height x filter 數量
彩色圖片，也是如此

解題時間 It's Your Turn

請跳出PDF至官網Sample Code & 作業
開始解題

