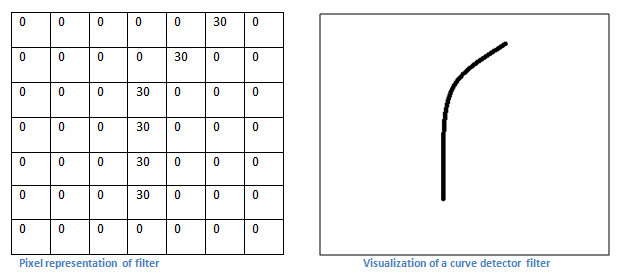
1. Convolution

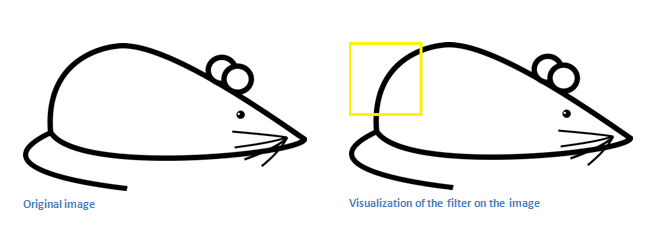
簡單來說，卷積就是每個神經元只會對應到圖片上某個區域，並不像NN是每個神經元都會對應到所有的圖片的pixel，因此卷積可以大幅縮減參數的數量，而卷積的過程大致如下：



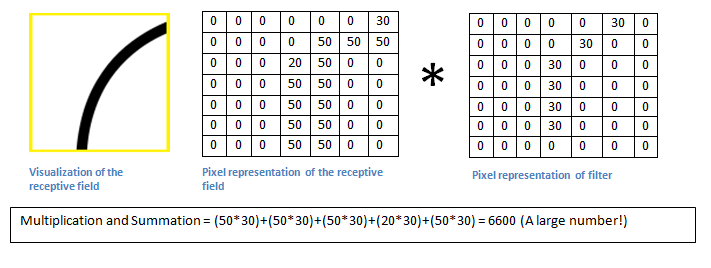
我們利用一個3X3的filter去對整張圖片做convolution，結果就會是右邊的圖具體來說，這些filter可以看成是**feature identifiers，可以用來identify直線、弧線等等，舉例來說**



我們現在有一個filter，是用來identify這種弧線

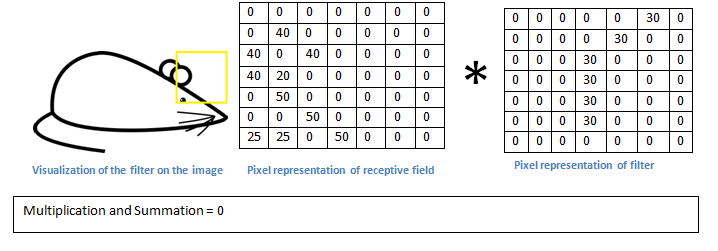


而我們將這個filter掃過這張老鼠的圖片，當掃到黃框的部分時



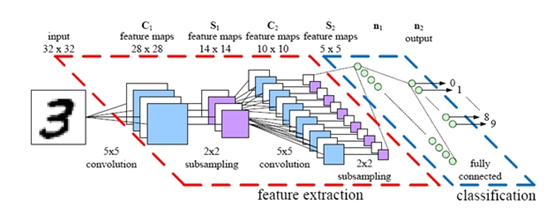
將該區域與filter矩陣相乘會得到一個數字6600

接著我們再將filter往右移



可以看到，當我們這一個filter掃過直線的時候，他的矩陣相乘出來的數值會非常小，因此以此類推，透過不一樣的filter就可以一步步的描繪出圖案的模樣，這也就是**CNN在對於處理圖片訊息的能力上會較強的原因。**

二、CNN的架構

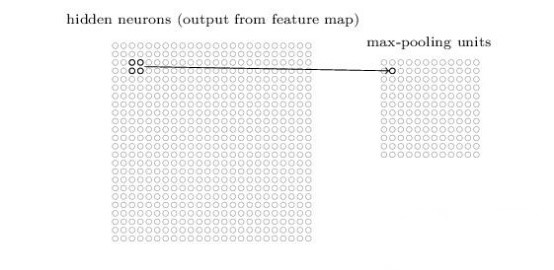


**Convolution + pooling + fully connected**

1. Convolution：

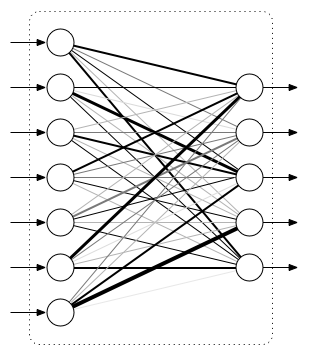


1. pooling：



在經過Convolution之後，pooling所做的是，以一個範圍(舉例來說2\*2)來掃描整張圖片，在每個2\*2範圍內取一個值(平均、最大值…等)

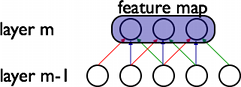
1. fully connected：



最後再經過全連接層進行分類的動作

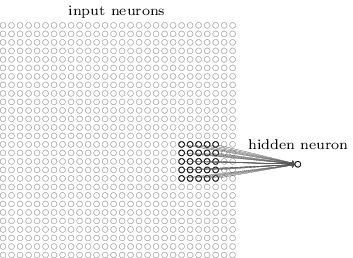
二、CNN的特性

1. input會是一個image(在handwritten digit裡，輸入為28\*28的矩陣)，不同於NN輸入是一行向量
2. weights sharing：在CNN裡，每個filter都會掃過整個圖片，而當次掃過整個圖片的filter裡的weights都是一樣的，而這樣做可以偵測出我們想要的features(無論它在圖片上的哪個位置)，除此之外，weights sharing可以降低參數量，因此可以提升learning effiency

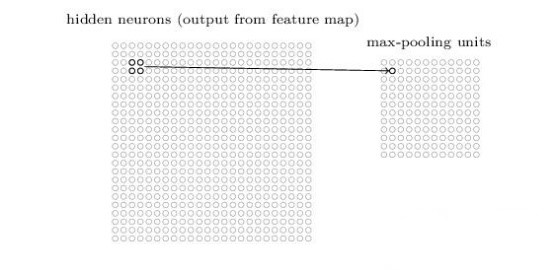


這張圖片裡，隱藏層的三個神經元有同樣的feature map，也就是weights sharing

3.Local connectivity ：每個隱藏層神經元會和部分輸入層神經元相連，舉例來說，我們的輸入層為28\*28個神經元，而在隱藏層每個神經元對應到輸入層5\*5的區域，每次移動一個像素點(移動像素點的數目可以改變)，我們就可以推出隱藏層的結果為24\*24個神經元



1. Pooling layer :  reduce the spatial size of the representation to reduce the amount of parameters and computation in the network. 可以降低神經元數目，但又不會流失過多訊息，在卷積層後面，將卷積層進行壓縮



四、CNN優缺點

優點：

1. weights sharing(權值共享)：透過權值共享，可大量降低所需訓練之參數
2. Convolution(卷積)：

圖片會有很多冗贅的訊息，而對於object recognition來說，edges(邊)常常是我們所需要的訊息，因此我們透過convolution來擷取圖片上重要的訊息，以達到精準的object recognition，這也是為什麼CNN常用來作為圖片辨識的工具

缺點：

1. 訓練時間較長

CNN參數設定：

1. filter：filter size越大，能擷取圖片上的細節越小，size越小，能擷取的細節越多，而filter的數量越多則可以增加complexity of learning。
2. batch normalization：為了避免data經過weights和parameters的相乘後變得過大或過小
3. pooling layer：因為我們的圖片僅28\*28，所以我們用最小的size 2\*2，而max pooling則可以將最重要的訊息擷取出來。