

## Convolutional Neural Network

用图像处理(全连接前馈网络也能做,但会导致参数过多) → 用CNN=减少参数

CNN基本想法: ① 检测图中是否有某种东西, 实际是在检测图中是否有某种 pattern.

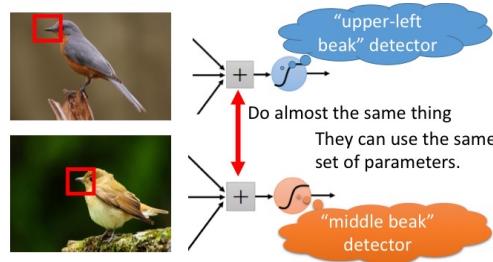
(性质)

Pattern一般在 image N 很多. 一个 neuron 只看一个 pattern 就可.

局部化问题  
不用全连接  
+共享权值  
参数少



② 同一pattern可能在不同区域重复出现, 则用来检测这一pattern的neuron可使用同一组参数. 同/不同images



用 subsampling

图小 → 参数少  
③ subsampling: 去掉一些像素不影响对整个object的判别. 可让整个image更小.  
e.g. 去掉奇数行和奇数列.

CNN整体结构:

The whole CNN



↑: Fully Connected Feedforward N  
需要一个向量作为输入.



↓

Convolution

Max Pooling

Convolution

Max Pooling

性质①②: pattern

性质③: Subsampling

Can repeat many times

## △ Convolution

用一组 filter(矩阵)来进行卷积操作, filter内的值是 NN 行各列的系数.

每个 filter 对一种 pattern 进行检测.

filter 作用于 image: 以步长 stride (步长) 移动. 由 到 到

e.g.

1	0	0	0	0	0	1
0	1	0	0	1	0	
0	0	1	1	0	0	
1	0	0	0	1	0	
0	1	0	0	1	0	
0	0	1	0	1	0	

1	-1	-1
-1	1	-1
-1	-1	1

Filter 1  
Matrix

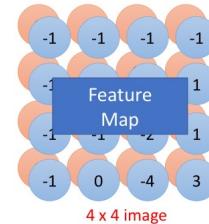
-1	1	-1
-1	1	-1
-1	1	-1

Filter 2  
Matrix

该 filter 检测是否有对角线连续 3 1 的 pattern.  
(因为 image 中只有 0 和 1, 所以只有满足该 pattern 的区域卷积得了 3, 否则卷积一定 < 3).  
(本质①)

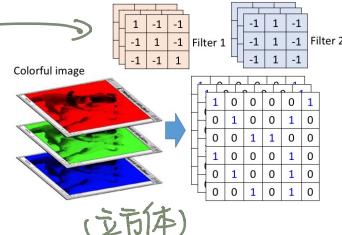


图中检测到有 2 个 filter 检测的 pattern.



用这个 filter 做完 convolution 后会得到多个 4x4 矩阵, 将它们叠起来, 组成一个 feature map.

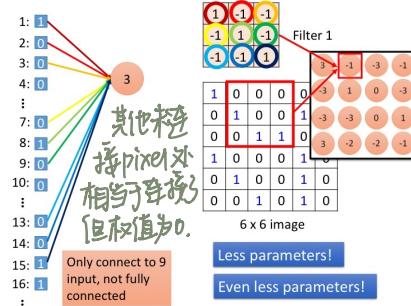
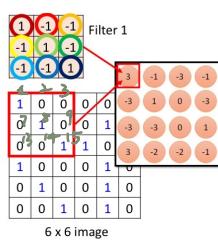
彩色图像: 由 RGB 3 层矩阵叠起来表示, 因此每个 filter 也应该有 3 层.  
(CNN 的 input 是可以有颜色深度的).



## △ Convolution 和 Fully Connected 关系:

在 Fully Connected NN 中, 一个 neuron 看到整个 image.

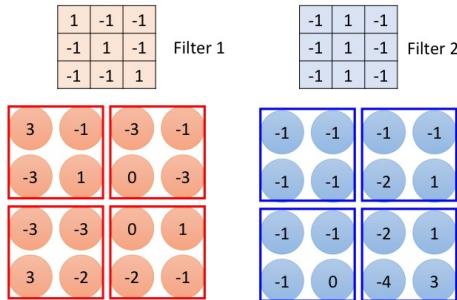
在 Convolution 中, 一个 filter 看一小部分 image, 相当于一个 neuron 大部分 weights 为 0.



同一 filter, 权数相同, 只是连接到的 pixel 不同 (检测不同位置)  
filter 每移动一下相当于有一个 neuron.  
权值共享  
不用全连接 + 权值共享  
= 参数少!

## △ Max Pooling

性质②: subSampling



将卷积得到的矩阵分块，每块取最大值保留，得到更小的矩阵。

每个 filter 的结果都池化，然后叠在一起：

每层为  $-channel$ . #channel = #filter.

深度

还可以再 convolution + max pooling

总结：做完一次 convolution + Max Pooling，会得到更小的 image，但仍有深度。

不同 filter 检测同一区域不同特征，所以每层代表一个特征。

又得到的(有深度的)小 image 再进行 convolution 时，每个 channel 是立体的。

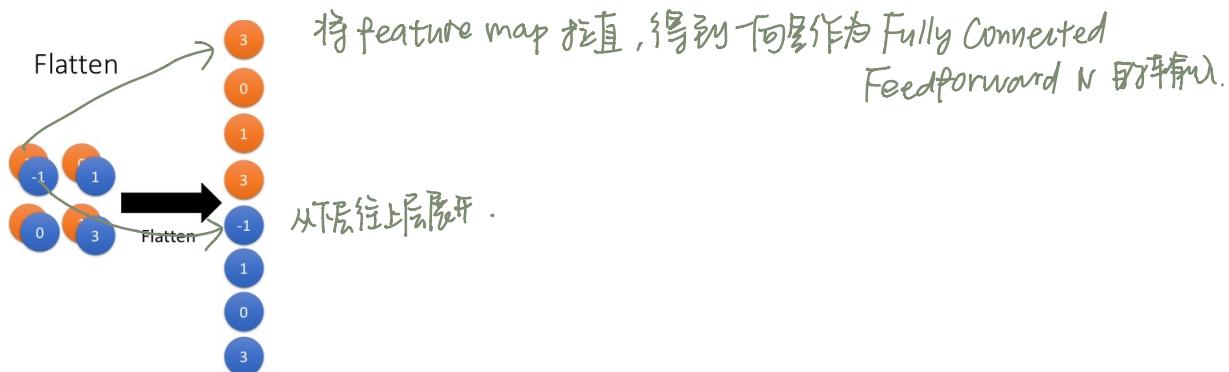
e.g. 第一次有 25 个 filter，得有 25 个 channel 的 image. 第二次还有 25 个 filter，每个 filter

为应对有深度的 image 的深度成 25 层的 filter，卷积得到 25 层的 image，即  $-channel$ ，  
每个 filter

∴ 最后的 feature map 由 25 个 channels 组成。每个 channel 有自身的深度。

多次重复是为了看到更广的范围，因为 max pooling 是提取了  $2 \times 2$  的信息，再次 max pooling 时  $2 \times 2$  的信息实际上为原图中  $2 \times 2 \times 2 \times 2$  的信息。

## △ Flatten

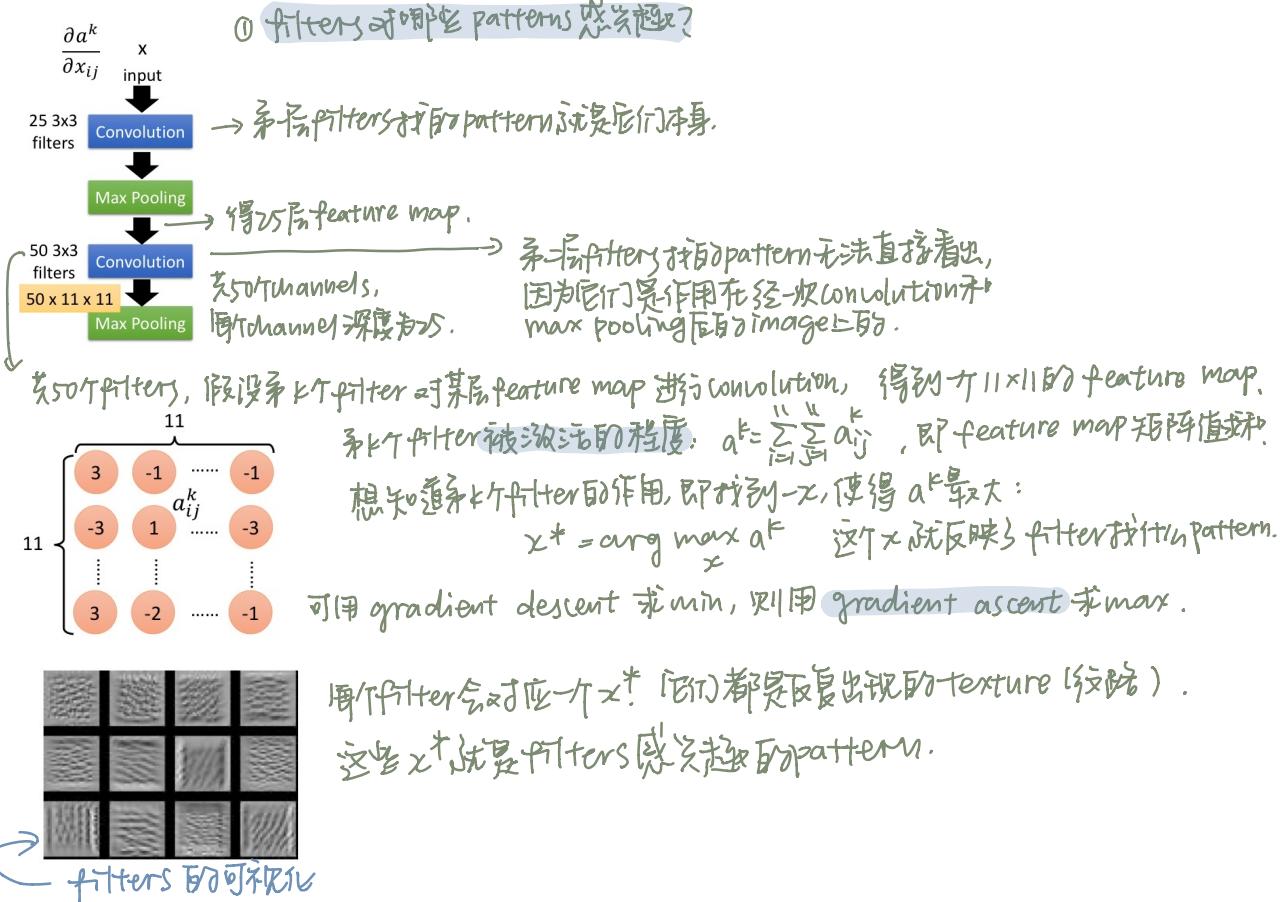


## △ CNN 如何 train?

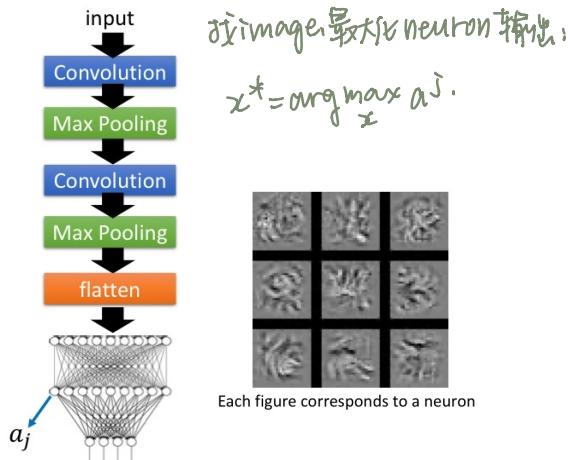
和之前的 Backpropagation 相同，但有一些 weight 被固定为 0。

因为有权值共享，所以将相同的 weights 一起更新：将它们的 gradient 求均值后一起更新。

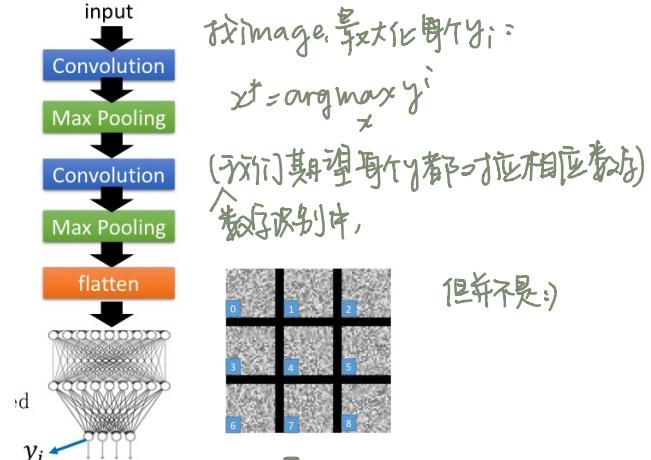
## △ CNN 是什么？



② 前馈网络中的 hidden layers 是干什么的？



③ output layers 是干什么的？

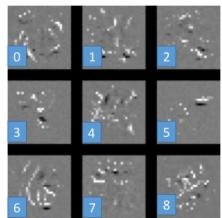


Deep Neural Networks are Easily Fooled!

和人类认知还存在差异。

一张图像不嘛数字，会有一些基本假设：只有部分有墨水（为1），其他部分为0。

∴ 加上该约束： $x^* = \arg \max_x (y_i - \sum_j |x_{ij}|)$

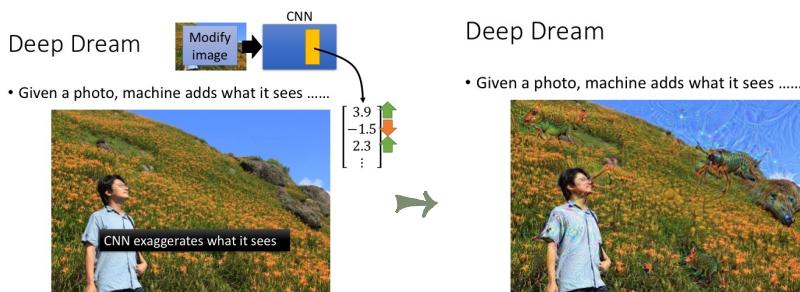


$\sum_j |x_{ij}|$   
正则项 (regularization)  
pixel values 求和.

再加更多约束，可能得到更好结果。

## △ 应用

① Deep Dream：给 machine 一张图，machine 会往上看它看到的东西。

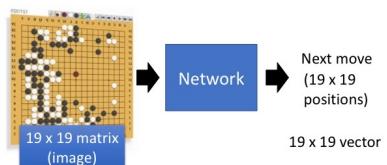


将 convolution layer 里的 filter 或  
hidden layer 的 output 拿出来，正的  
调大，负的调小，作为 target，用 gradient descent  
找到一个  $x$  使通过该 hidden layer 的 output  
等于这个 target. (夸大 CNN 看到的东西)

② Deep Style：将照片改成名画风格。



### ③ Playing Go



Fully-connected Feedforward Network

但 CNN 表现更好。(因为满足性质①、②)

因为性质③不满足, 所以 AlphaGo 的 CNN 中无 Max Pooling.