ch13: Convolutional Neural Networks

- **Problems Using Standard NNs(P17)**

像素序列：图像的空间结构被破坏

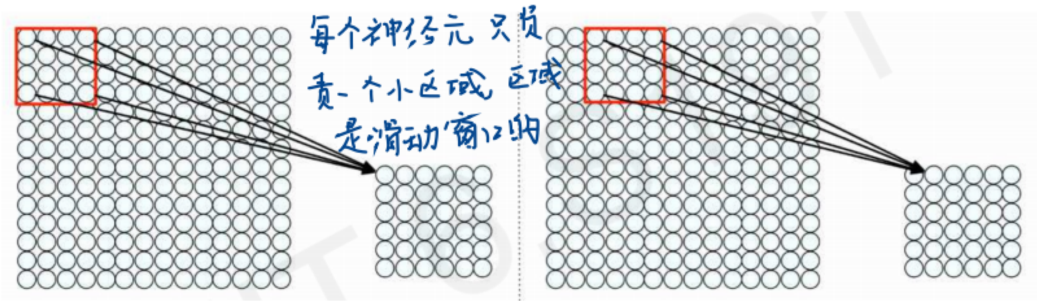
- **Seq-of-Pixels:** The spatial structure of images is destroyed
- **Locality insensitive**: take into account the entire picture instead of local features. 输入整个图片
- **Translation variant:** sensitive to the global position of a feature
- **Too many parameters**: fully connected layers yield a huge volume of parameters. 全连接神经网络参数过多无法学习

- **Modeling Images: Design Criteria(P18)**

- To model images, we need to:
  1. Keep the spatial structure 保留空间结构
  2. Sensitive to locality 对局部信息敏感
  3. Handle variants in input images
  3. Share parameters across the spatial regions

Convolution

The Idea(P22)



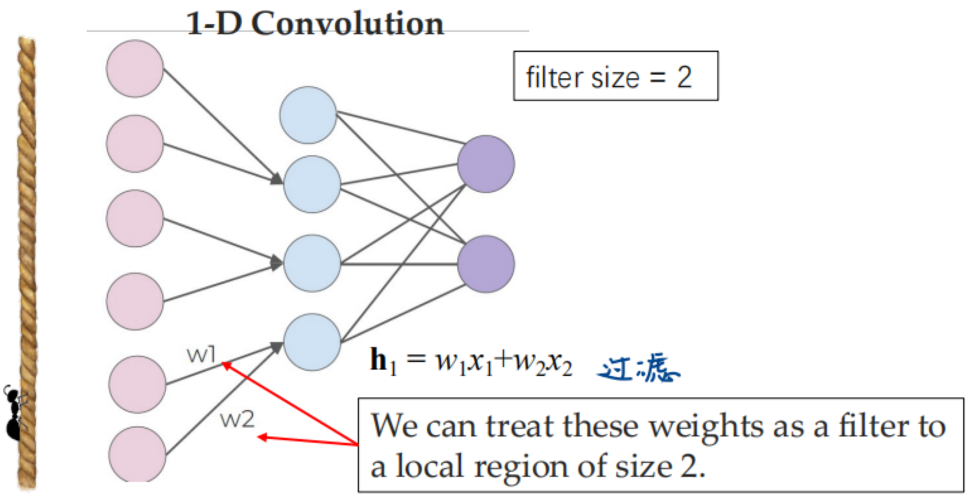
- **Slide** over the image spatially, computing feature transformation (how?)
- During the sliding, the weights for all neuron must be **shared** (why?)

局部连接
只用关心特定的特征

Following are the results of weight sharing:1.It reduces the number of weights that must be learned, which reduces model training time and cost. 2.It makes feature search insensitive to feature location in the image.

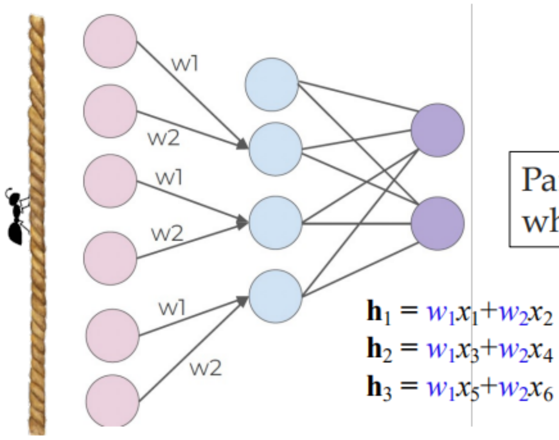
Convolutions and Filters (1-D)(P25-28)

- **Filter** (a.k.a., **convolutional kernel**): weights assigned to the input region by a hidden unit.



- We now apply the **same** set of weights over the **whole picture**.

1-D Convolution



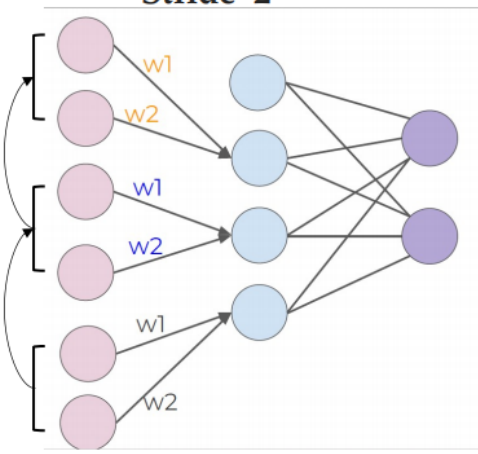
Parameters are **shared** over the whole picture for one filter.

卷积不同于MLP的地方：参数是共享的，比如图中都是 w_1, w_2

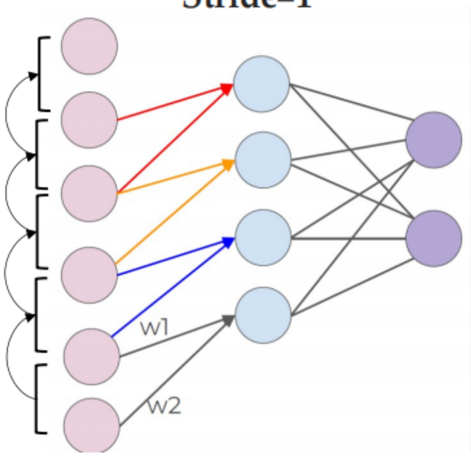
- Note that the filter scans every 2 pixels in this example.
- We can control the filter to scan every N pixels (N=**stride**).

步长：每次跳跃几个神经元

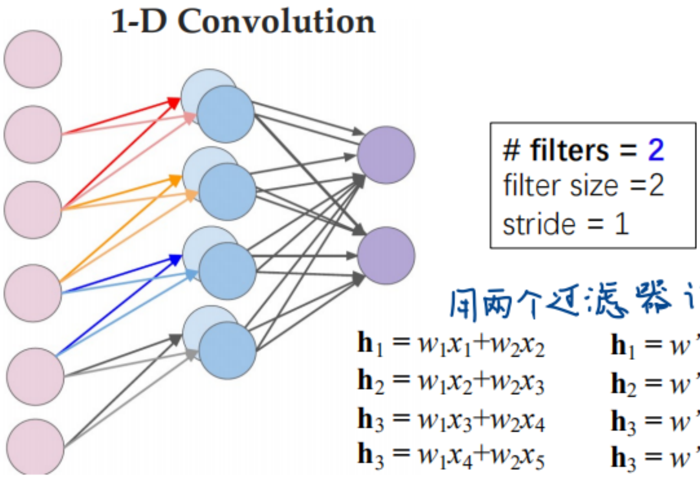
Stride=2



Stride=1



- We can then expand this idea to **multiple filters**.
- Each filter is detecting a different feature



用两个过滤器识别两个特征

$\mathbf{h}_1 = w_1x_1+w_2x_2$ $\qquad$ $\mathbf{h}_1 = w'_1x_1+w'_2x_2$

$\mathbf{h}_2 = w_1x_2+w_2x_3$ $\qquad$ $\mathbf{h}_2 = w'_1x_2+w'_2x_3$

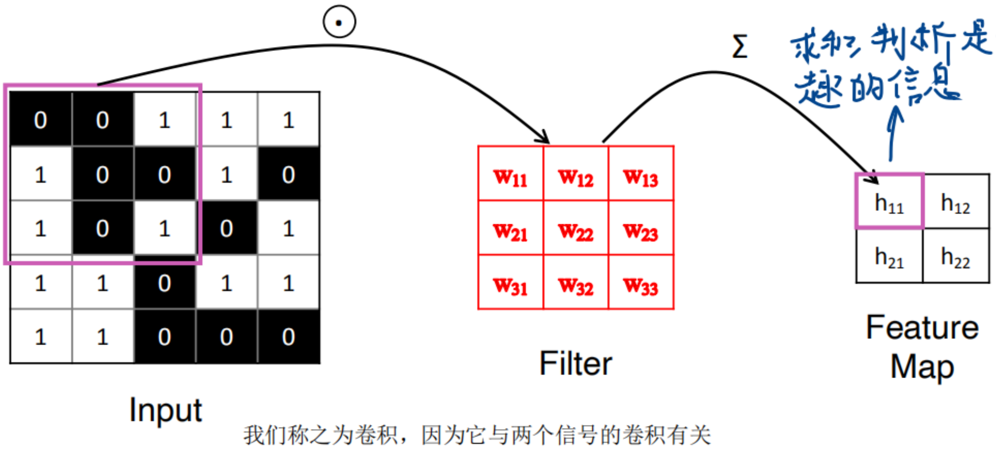
$\mathbf{h}_3 = w_1x_3+w_2x_4$ $\qquad$ $\mathbf{h}_3 = w'_1x_3+w'_2x_4$

$\mathbf{h}_3 = w_1x_4+w_2x_5$ $\qquad$ $\mathbf{h}_3 = w'_1x_4+w'_2x_5$

2-D Convolutions（P29-32）

- **Convolutional Operation**

elementwise multiplication and sum of a filter and the image



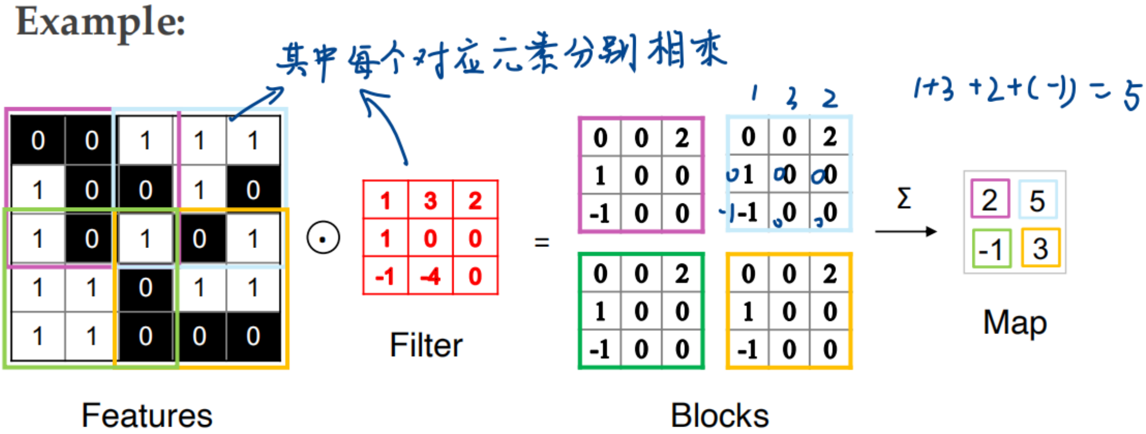
求和，判断是否有感兴趣的信息

我们称之为卷积，因为它与两个信号的卷积有关

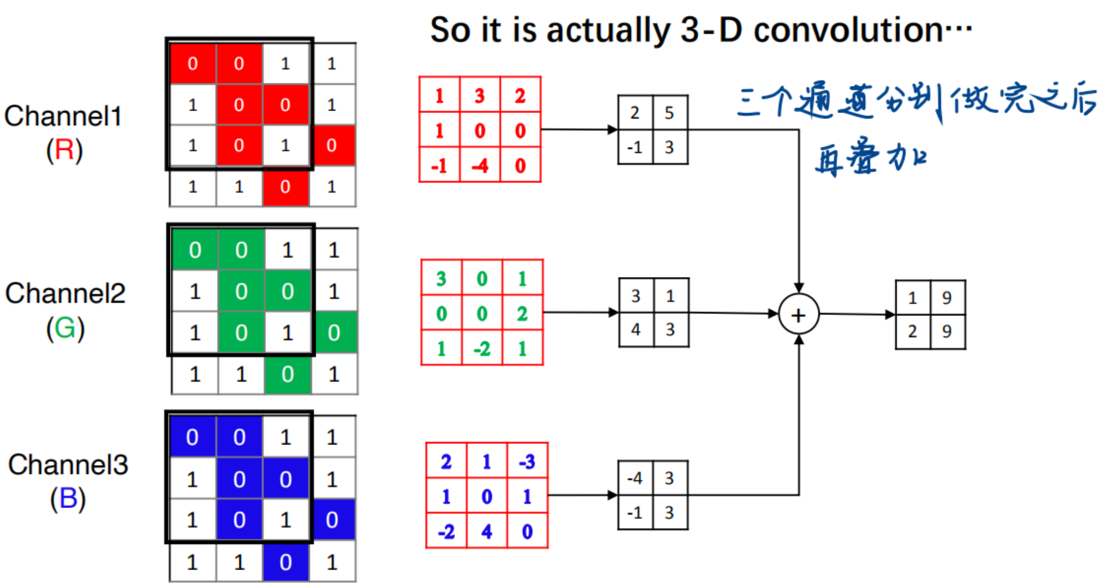
※ We call it convolutional because it is related to convolution of two signals:

$$f[x,y]*g[x,y] = \sum_{n_1=-\infty}^{\infty}\sum_{n_2=-\infty}^{\infty} f[n_1,n_2]\cdot g[x-n_1,y-n_2]$$

- **Example:**

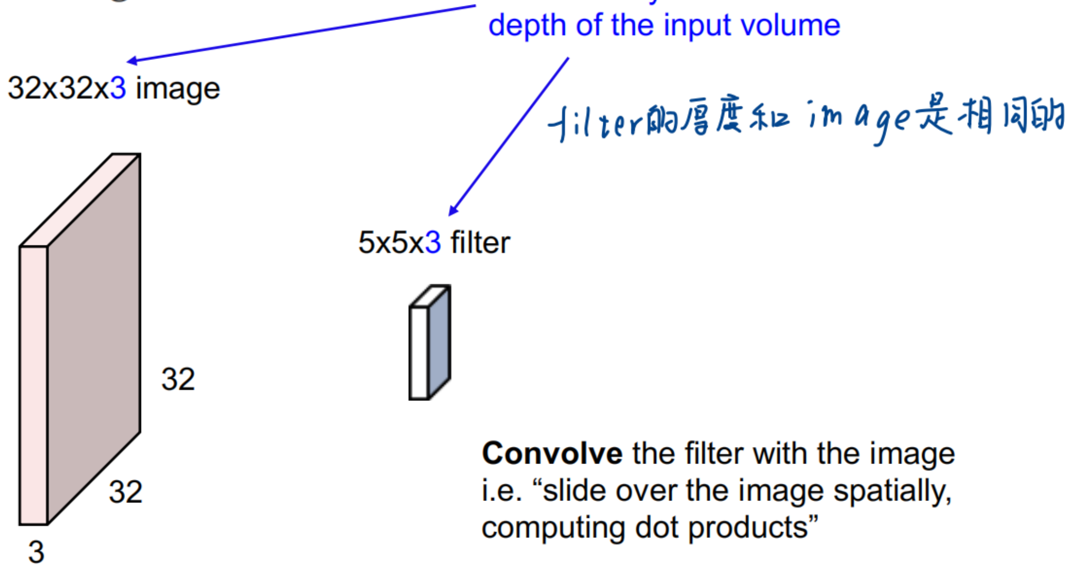


- What if we have color?



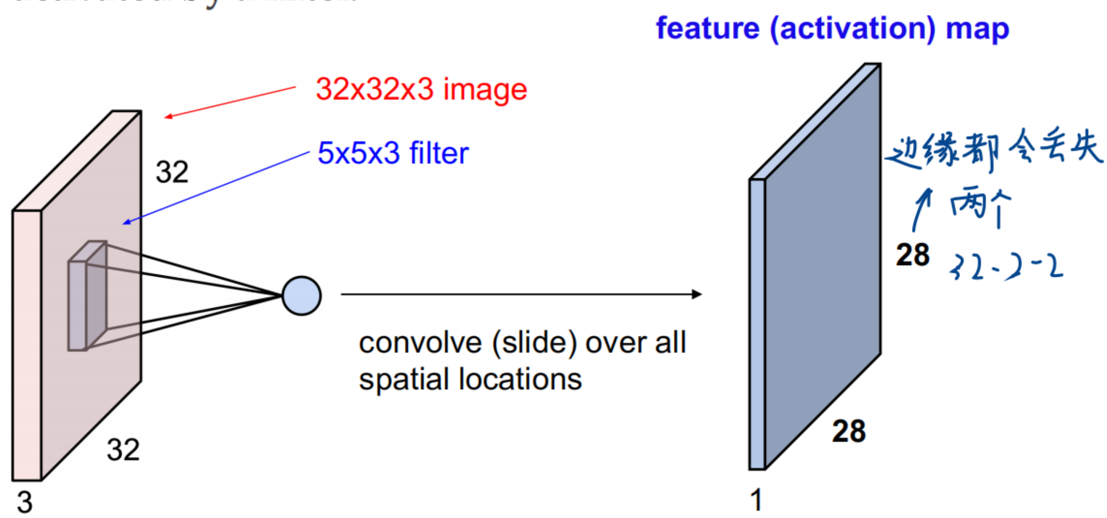
- 拓展到**3-D Convolution**

- **3-D Image and Filter**



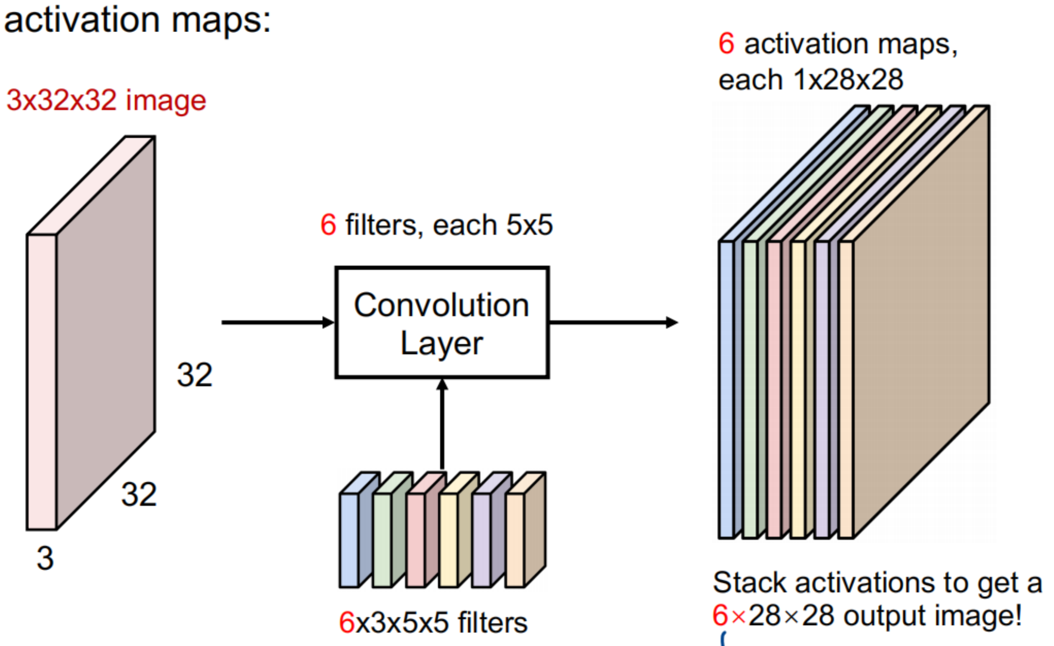
Feature Map（P34-36）

- **A map that stores the locations of a specific feature activated by a filter.**



Using multiple filters to scan multiple features

- For example, if we had 6 5x5 filters, we'll get 6 separate activation maps:



feature 数/信号的通道数

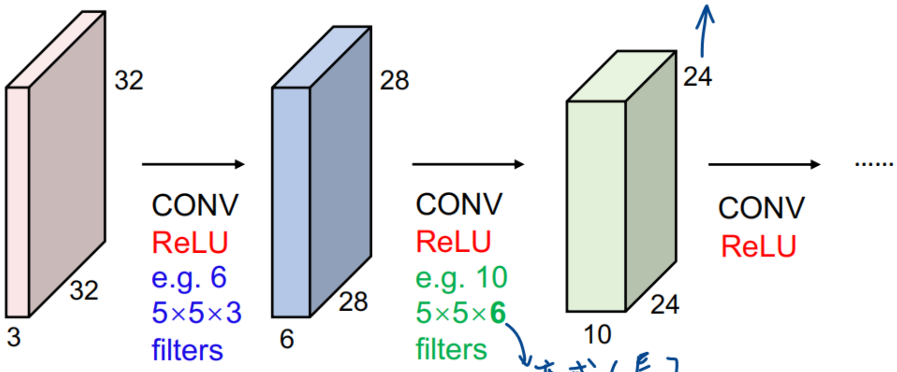
Xiaodong Gu　　Machine Learning: Lecture 13　　36

Multiple Layers(P37-40)

- definition

- A sequence of convolutional layers, interspersed with activation functions (e.g., ReLU).

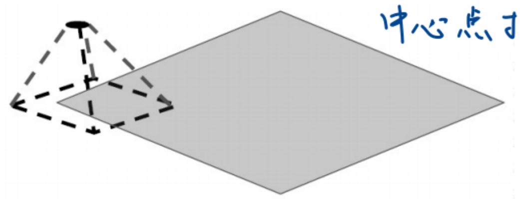
边缘会有损失



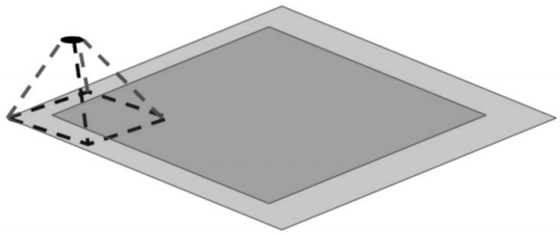
变成6层了

Padding(P41-42)

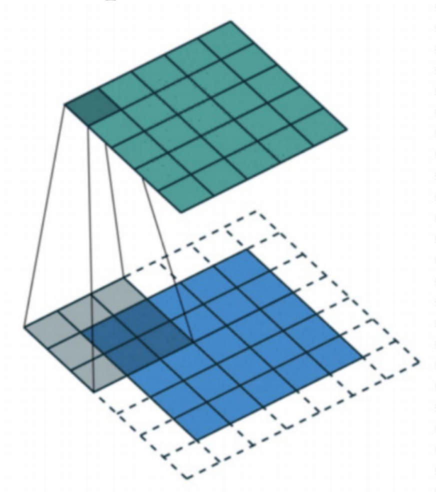
- We run into a possible issue for edge neurons! There may not be an input there for them. 为了让图像的边缘也可以作为中心点扫描到，在图片周围补0

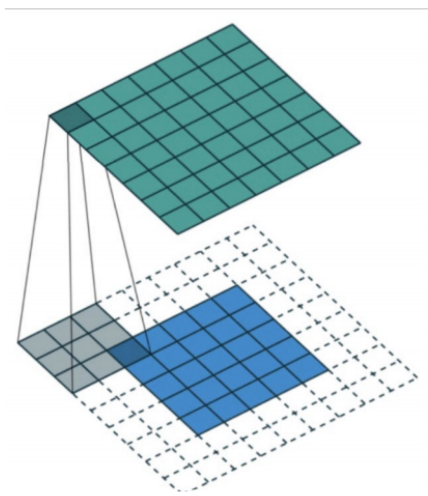


- We can fix this by adding a "**padding**" of zeros around the image.



- **(Zero-)Padding** refers to the process of symmetrically adding zeroes to the input matrix. It's a commonly used modification that allows the size of the input to be adjusted to our requirement.

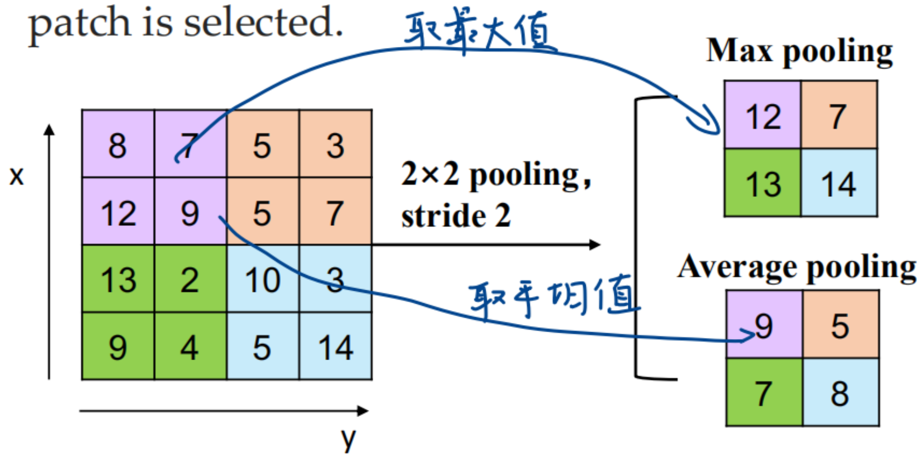




Pooling(P42-45)

- **Downsample** the input image to reduces the memory use.
- Will not change the object 像素降低，把图片缩小

- **Max pooling**: The maximum pixel value of the patch is selected.
- **Average pooling**: The average value of all the pixels in the patch is selected.

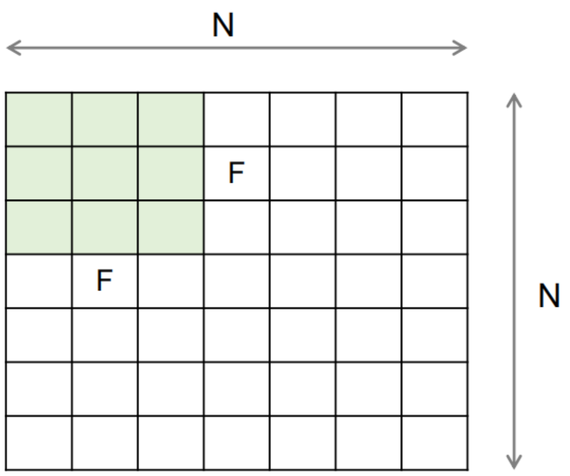


No learnable parameters
Introduces spatial invariance

没有任何可学习的参数，这就引入了空间不变性

> Max pooling selects the **brighter** pixels from the image.(相当 于 变 亮)
>
> Average pooling method **smooths** out the image and hence the sharp features
>
> may not be identified.(相当 于 变 模糊)

outsize的计算（P46-48）



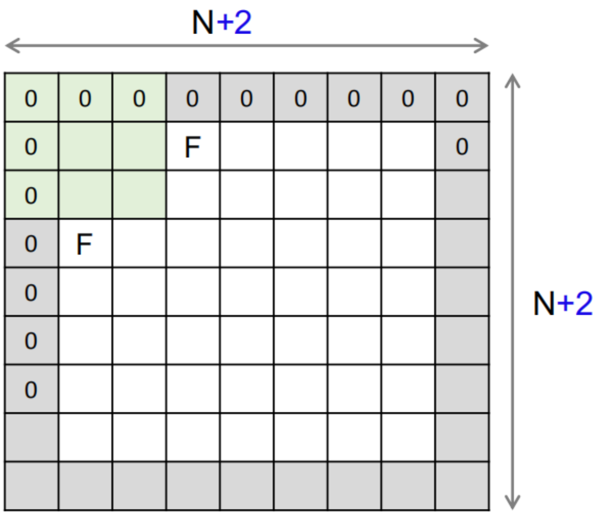
Input size: **N×N**
Filter size **F×F**

Output size:
(N - F) / stride + 1

e.g. N = 7, F = 3:
stride 1 => (7 - 3)/1 + 1 = 5
stride 2 => (7 - 3)/2 + 1 = 3
stride 3 => (7 - 3)/3 + 1 = 2.33 :\

向下取整，应该是2

In practice: Common to zero **pad** the border



Pad with P pixel border

Output size:
(N +2P - F) / stride + 1

e.g. N = 7, F = 3:
stride 1 => (7 +2 - 3)/1 + 1 = 7
stride 2 => (7 +2 - 3)/2 + 1 = 4
stride 3 => (7 +2 - 3)/3 + 1 = 3

The number of parameters for a convolutional layer is

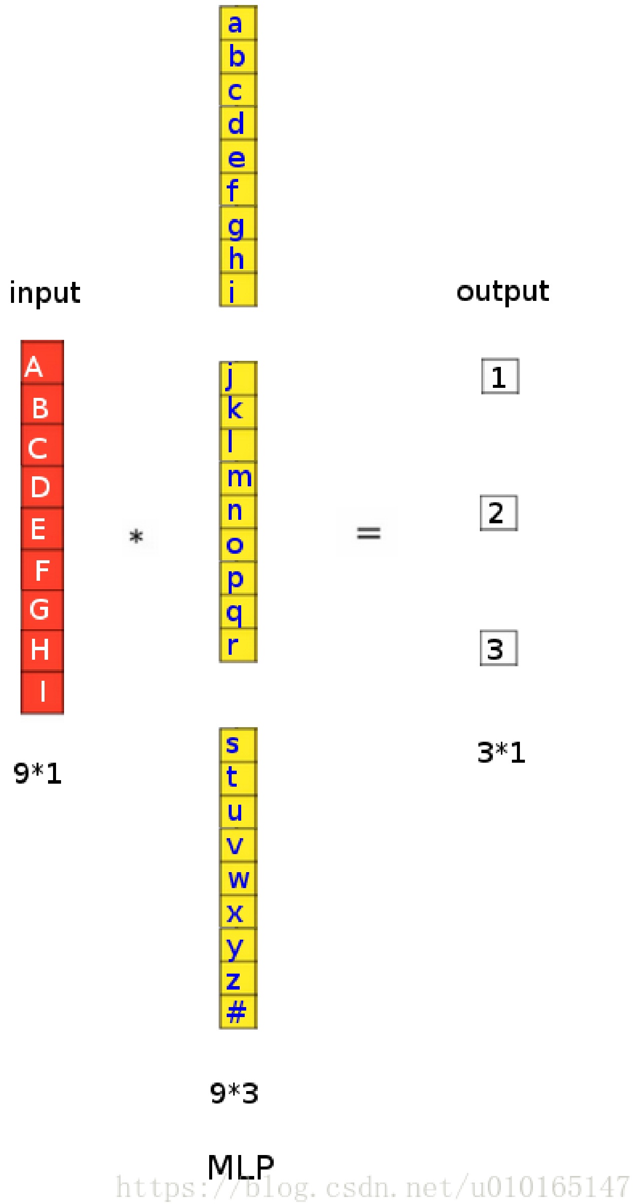
```
(filter_height * filter_width * in_channels * out_channels) + out_channels
```

> two questions:

1. **Can we let CNN act as MLP? How?**

MLP实际是1*1的卷积，n个卷积核就将原来的d维变为n维.

下图为MLP的计算过程（为了方便MLP的计算过程图权重W被拆开了实际为9x3的矩阵，而输入计算时应该先转置，输出也是需要转置，即 1x9 dot 9x3 = 1x3）



上面我们可以看到，CNN和MLP计算过程实际对应数值标号是完全一致的，也就是说上两图MLP和CNN计算过程完全等价，可以互相转换。显然可以推导出，当CNN卷积核大小与输入大小相同时其计算过程等价于MLP，也就是说MLP等价于卷积核大小与每层输入大小相同的CNN（如输入图片为100x100，卷积核大小为100x100），所以MLP是CNN的一个特例。而卷积核大小与每层输入大小相同会直接丢失非常多的输入空间信息，所以MLP这种运行模式不适合图像这种空间信息丰富的数据。

参考：(76条消息) CNN与MLP之间的关系，优缺点_Pengsen Ma的博客-CSDN博客_mlp和cnn的区别

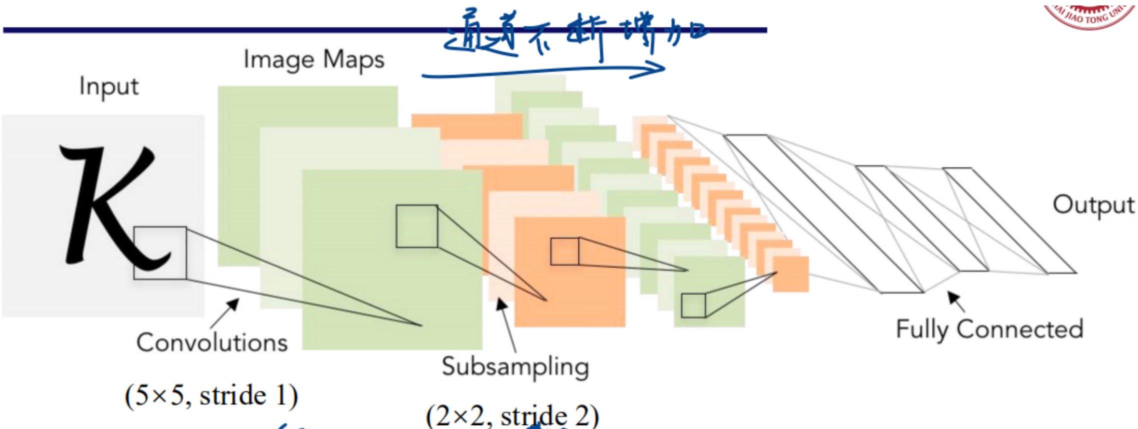
2. **Can we apply CNN for texts**

可以，在文本中，每个词都可以用一个行向量表示，一句话就可以用一个矩阵来表示，那么处理文本就与处理图像是类似的了

参考：(76条消息) NLP-使用CNN进行文本分类_spring_willow的博客-CSDN博客_基于cnn的文本分类

Some CNNs

- **LeNet-5（P51）**



卷积　pool　卷积

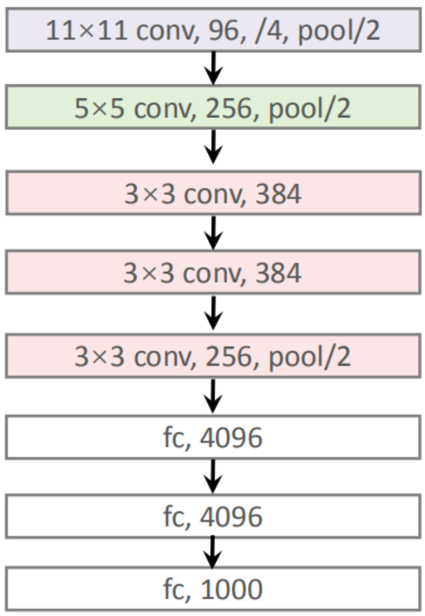
Architecture: [CONV1-POOL1-CONV2-POOL2-FC1-FC2]

Conv filters were 5×5, applied at stride 1
Subsampling (Pooling) layers were 2×2 applied at stride 2

- **AlexNet（P52-53）**

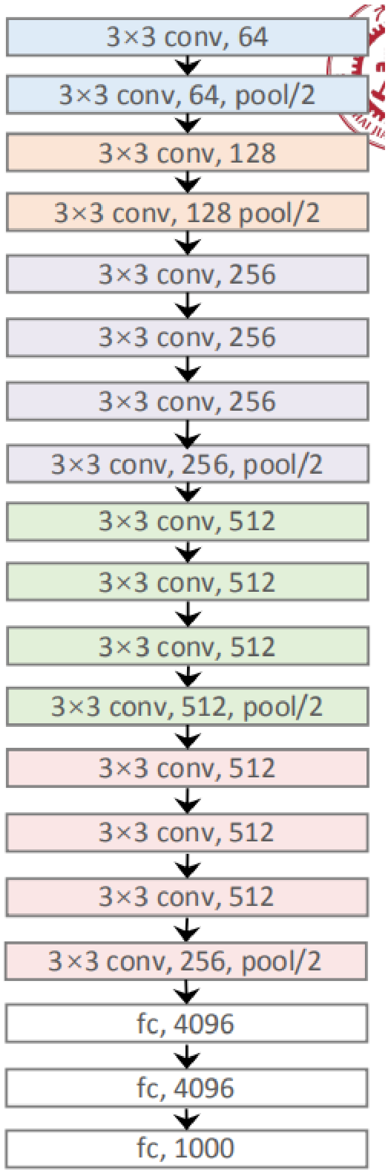
- **Architecture:**

CONV1: 96 11×11 filters at stride 4, pad 0
MAX POOL1 : 3×3 filters at stride 2
NORM1 : Normalization layer
CONV2 : 256 5×5 filters at stride 1, pad 2
MAX POOL2 : 3×3 filters at stride 2
NORM2 : Normalization layer
CONV3 : 384 3×3 filters at stride 1, pad 1
CONV4 : 384 3×3 filters at stride 1, pad 1
CONV5 : 256 3×3 filters at stride 1, pad 1
MAX POOL3 : 3×3 filters at stride 2
FC6 : 4096 neurons
FC7 : 4096 neurons
FC8 : 1000 neurons (class scores)



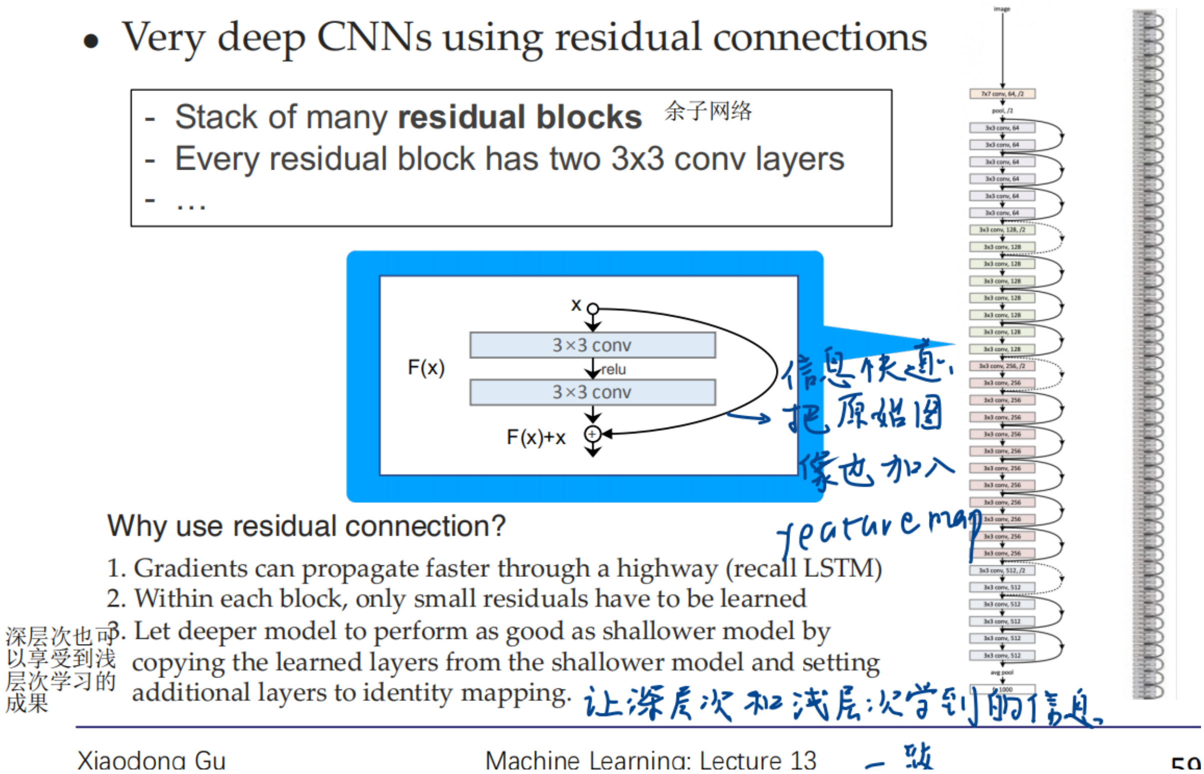
- **VGGNet（P54-56）**

- **Small filters:** 参数更少 ← 更小的卷积
  - ⊳ only 3×3, stride 1 and 2×2 max pool stride 2 可以有更多 ← 更多的层数
- **Deeper networks** 抽象
  - ⊳ 8 → 16/19 layers



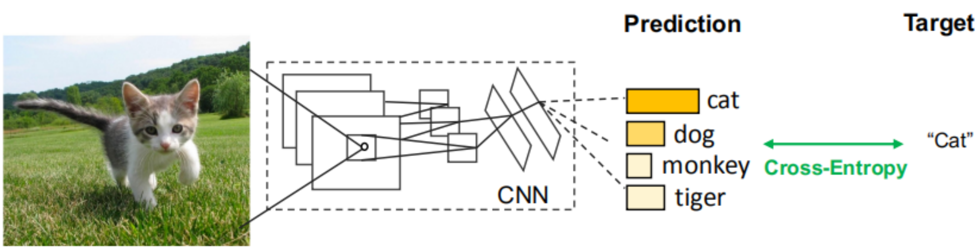
> convolutional neural network 并不是越深越好：**deeper models are harder to optimize， 模型越深, 误差的反向传播更多**

- **ResNet（P59-60）**



a big picture

- **Convolution + MLP + Softmax**



questions:

1. CNN和MLP的优缺点对比？