自然语言处理

Natural Language Processing

Chapter 5

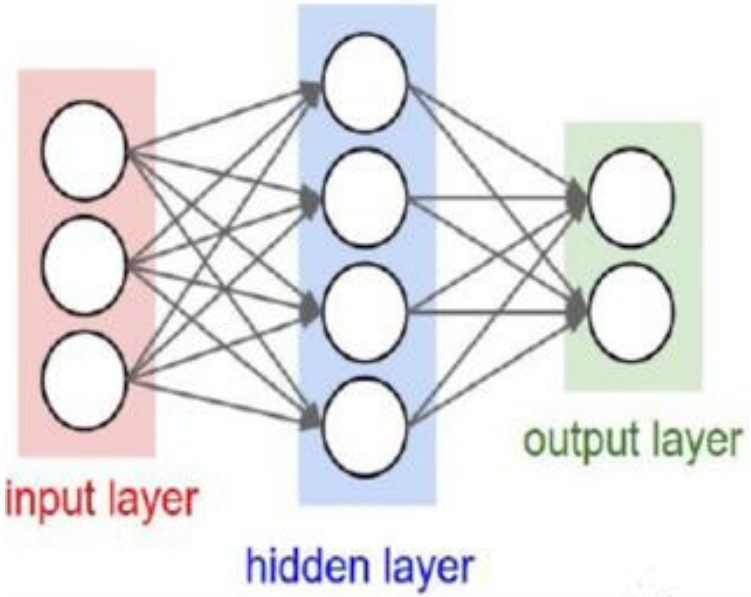
卷积神经网络

Outline

- 卷积神经网络(CNN)
- 文本卷积
- 文本卷积扩展

回顾：前馈神经网络

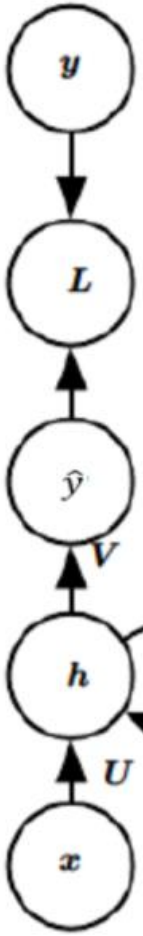
- 全连接
- 参数不共享



Vanilla NN

$$\mathbf{x} \mapsto W_1^T\mathbf{x} \mapsto \mathbf{h}_1 = f(W_1^T\mathbf{x}) \mapsto W_2^T\mathbf{h}_1 \mapsto \mathbf{h}_2 = f(W_2^T\mathbf{h}_1) \mapsto$$
$$W_3^T\mathbf{h}_2 \mapsto \mathbf{h}_3 = f(W_3^T\mathbf{h}_3) \mapsto W_4^T\mathbf{h}_3 = \hat{y}$$

回顾：RNN



记录了神经元最近一次或几次的隐藏状态，具有**短期（short term）**记忆的能力

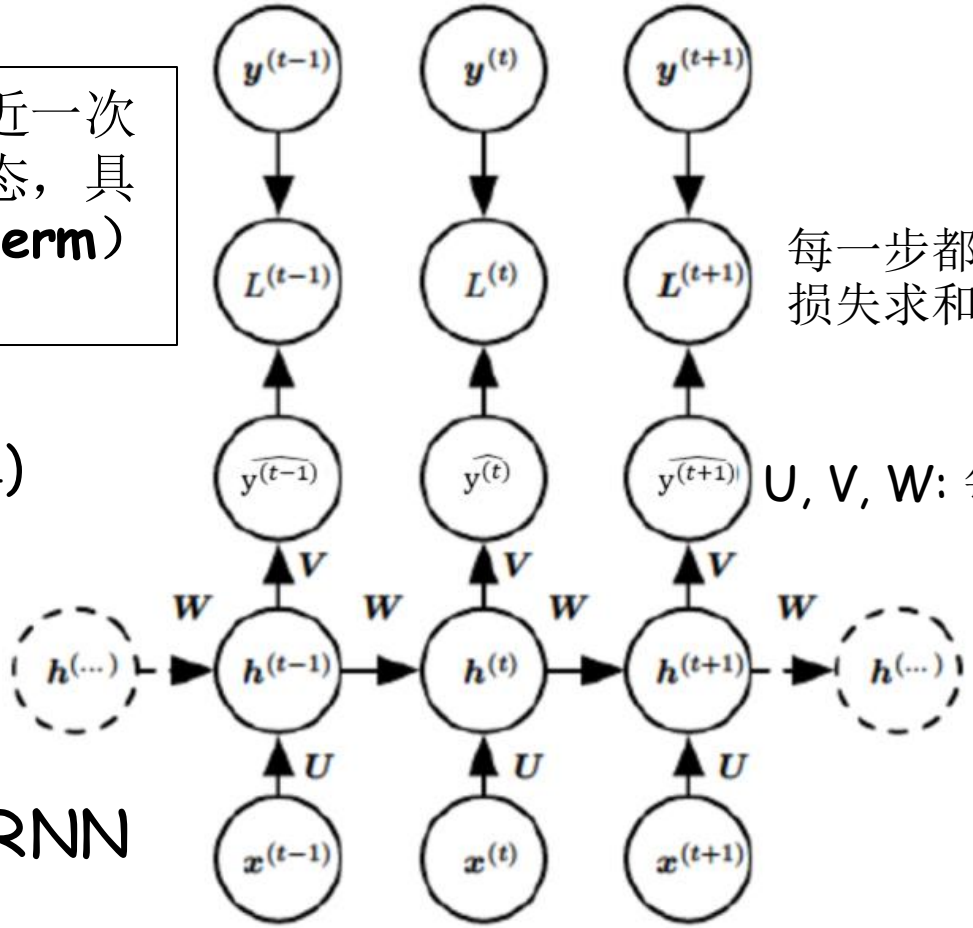
$h_t = f(h_{t-1}, x_t)$

f: 前馈网络

展开

自反馈

Plain vanilla RNN



每一步都有输出：损失求和

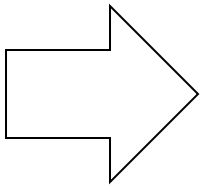
U, V, W: 每步共享

复杂度：O(t)

$x^{(t)}$: 文本中第t个词

思考

- 能否实现对于局部信息的关注？
- E.g. 我 喜欢 学习 自然 语言 处理
- 采用n-gram(n元语法)：我 喜欢 学习，喜欢 学习 自然，学习 自然 语言，自然 语言 处理 → 4个片段向量化
- We went home early yesterday → 3个片段向量化
- 忽略语法、语义合理性
- n-gram向量进行组合

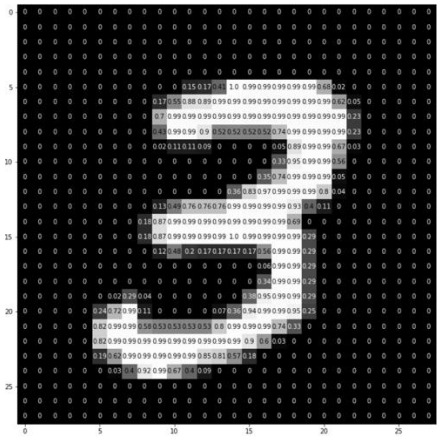


CNN的文本特征提取思路

卷积神经网络

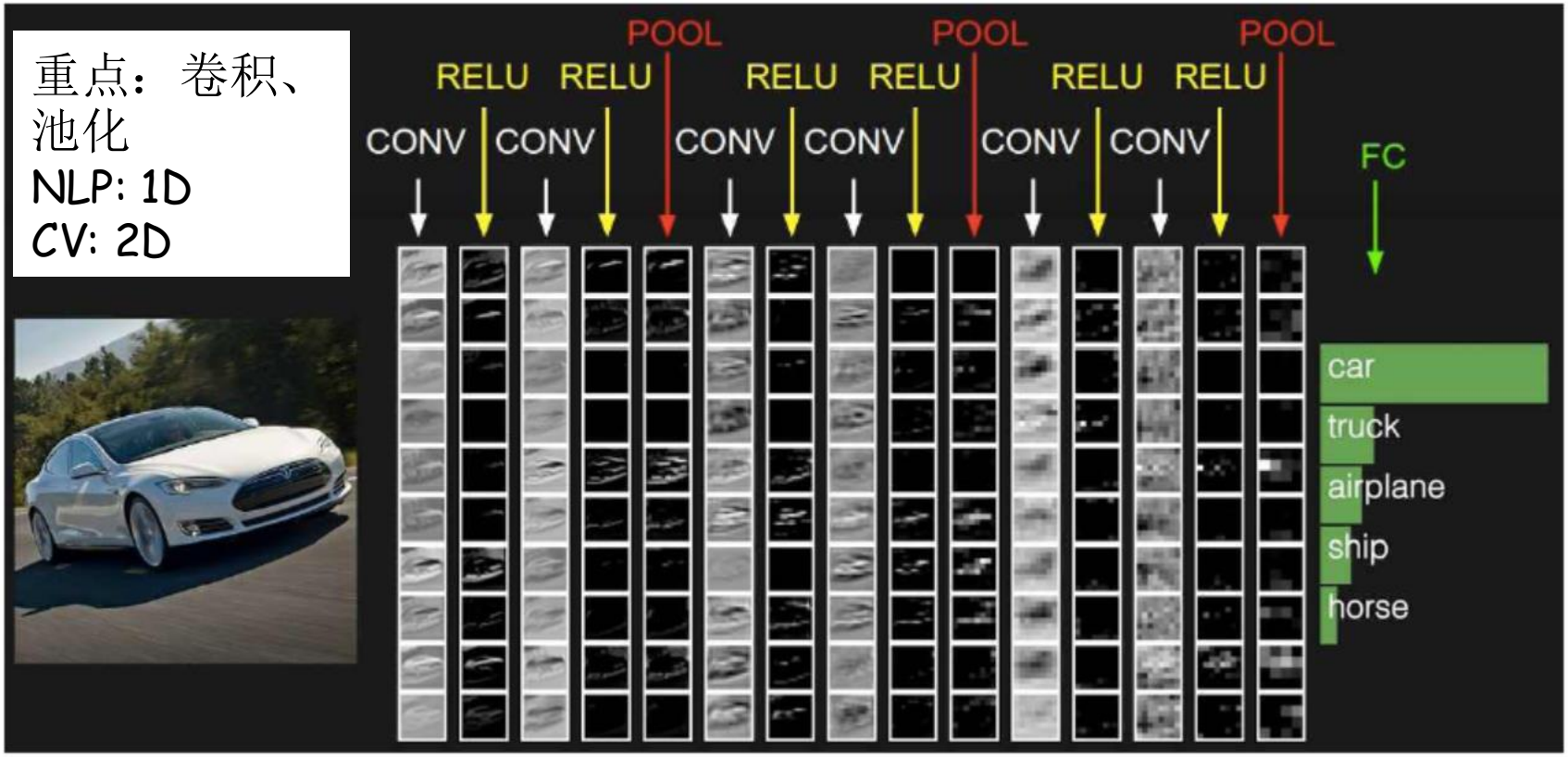
- Convolutional Neural Networks (CNN)
- 提出人：1998, Yann LeCun

- 最初用于识别给定输入图像中的数字。
- CNN中，至少有一层中使用了**卷积**代替一般的矩阵乘法
- 在卷积的操作中，模型中的每个神经元都有**部分**的输入，是一个**局部**连接的网络 **vs** 全连接



卷积神经网络

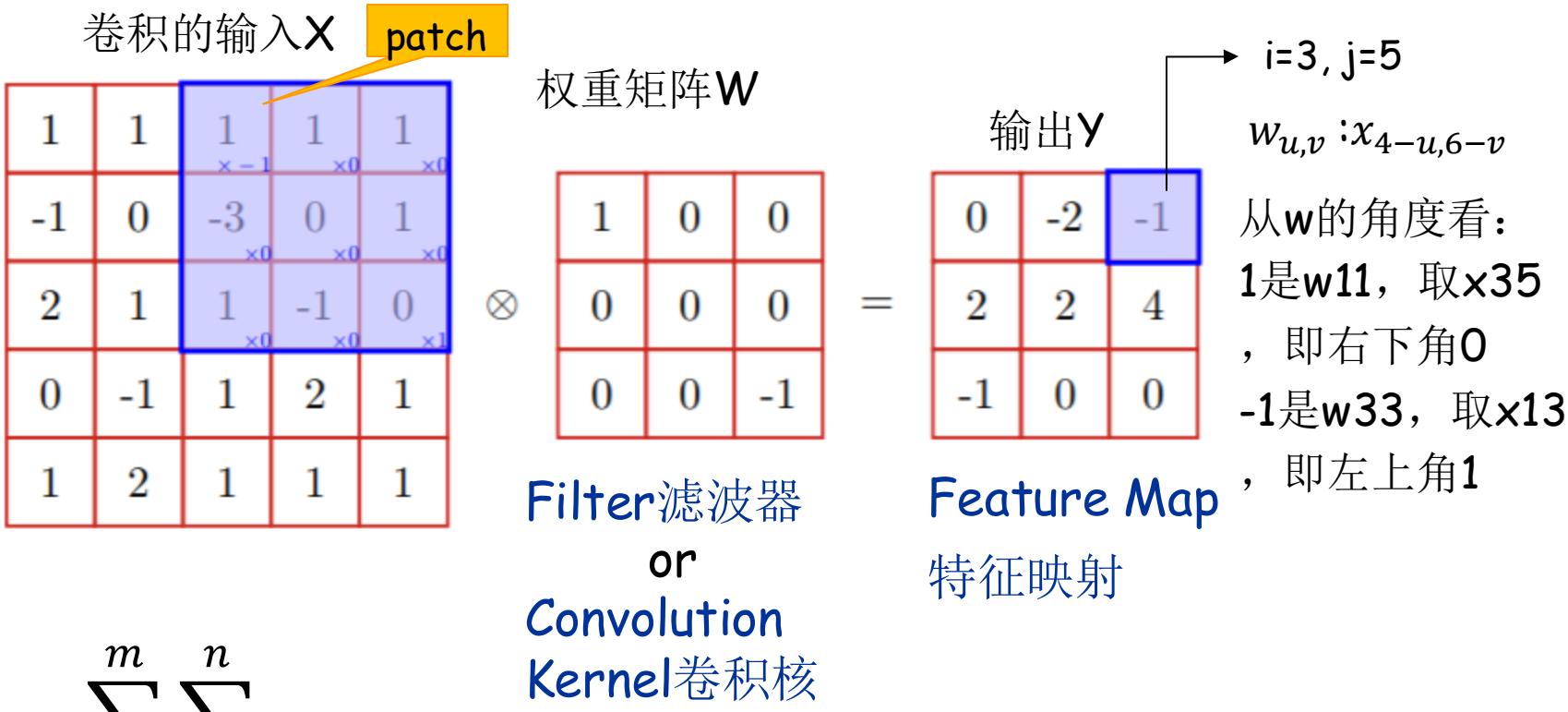
- 框架包含：输入层、卷积层(进行激活)、池化层(pooling, 也叫汇聚层)、输出层(dense)
- CNN一般是由卷积、激活、池化交叉堆叠而成的神经网络
- 特点：局部连接、权重共享以及池化



2D卷积

卷积操作的实质：相乘和相加

在2D卷积中，卷积核**在数据上沿2维滑动**



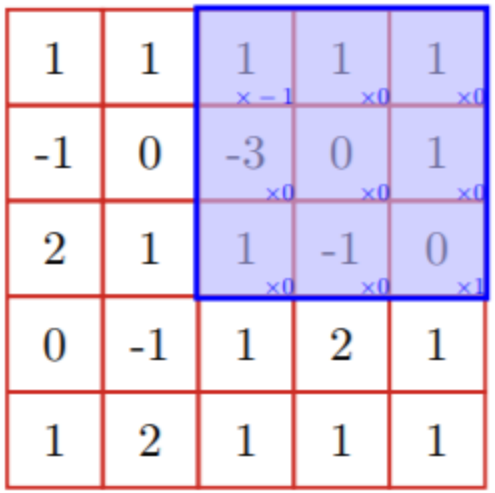
$$y_{ij} = \sum_{u=1}^{m}\sum_{v=1}^{n} w_{uv} \cdot x_{i-u+1,j-v+1}$$

假设(i, j) 从(m, n)开始。例子中，m=3 n=3

2D卷积

我们通常说的“卷积”其实是互相关，唯一区别在于卷积核是否翻转。互相关也称不翻转卷积。

卷积核是否进行翻转和特征抽取的能力无关。



$\otimes$

-1	0	0
0	0	0
0	0	1

Filter滤波器/
Convolutional
Kernel卷积核

$=$

0	-2	-1
2	2	4
-1	0	0

Feature Map
特征映射

$w_{u,v}$:$x_{u,2+v}$

从w的角度看：

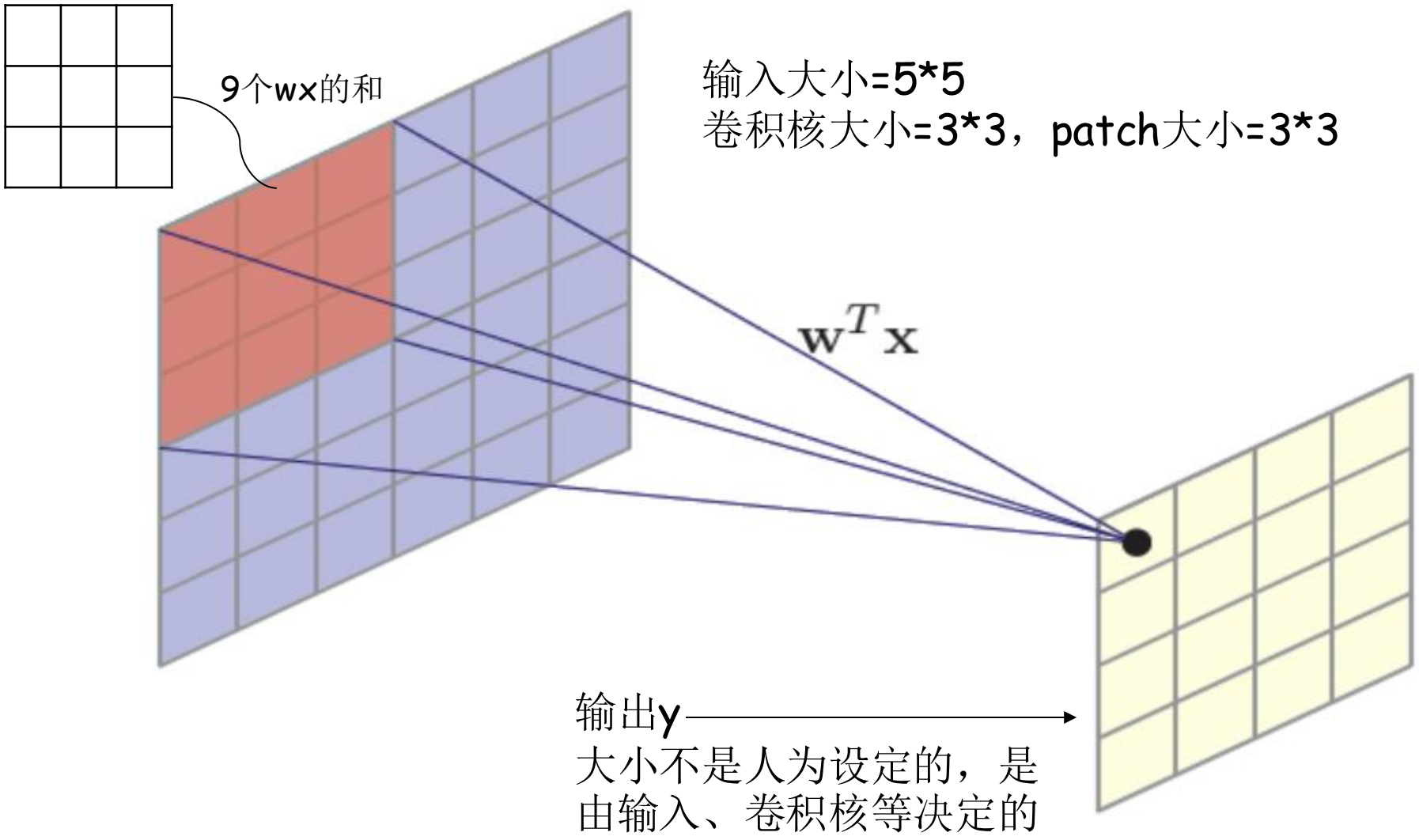
-1是w11，取x13

1是w33，取x35

Cross-Correlation
互相关

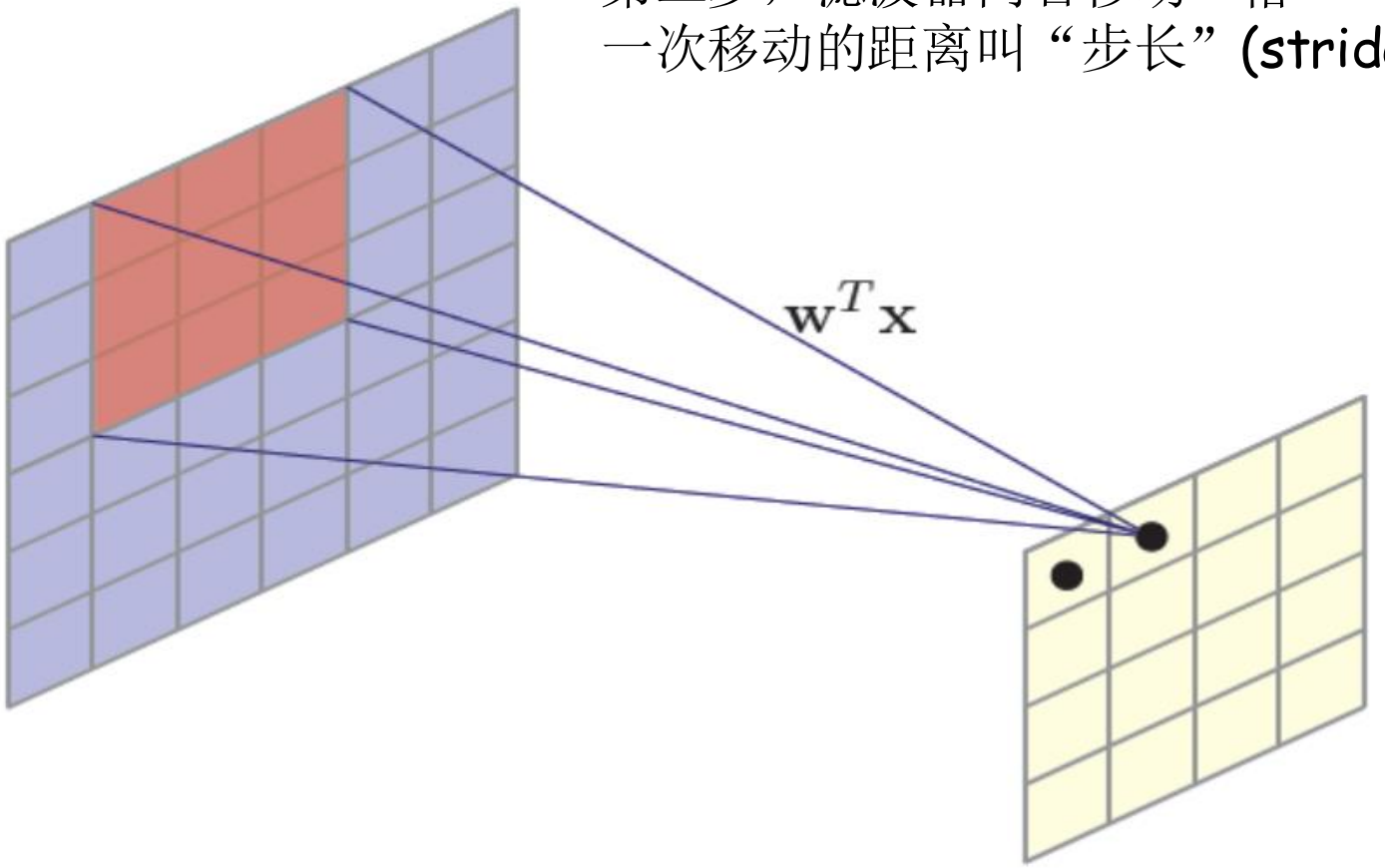
$$y_{ij} = \sum_{u=1}^{m}\sum_{v=1}^{n} w_{uv} \cdot x_{i+u-1,j+v-1}$$

2D卷积

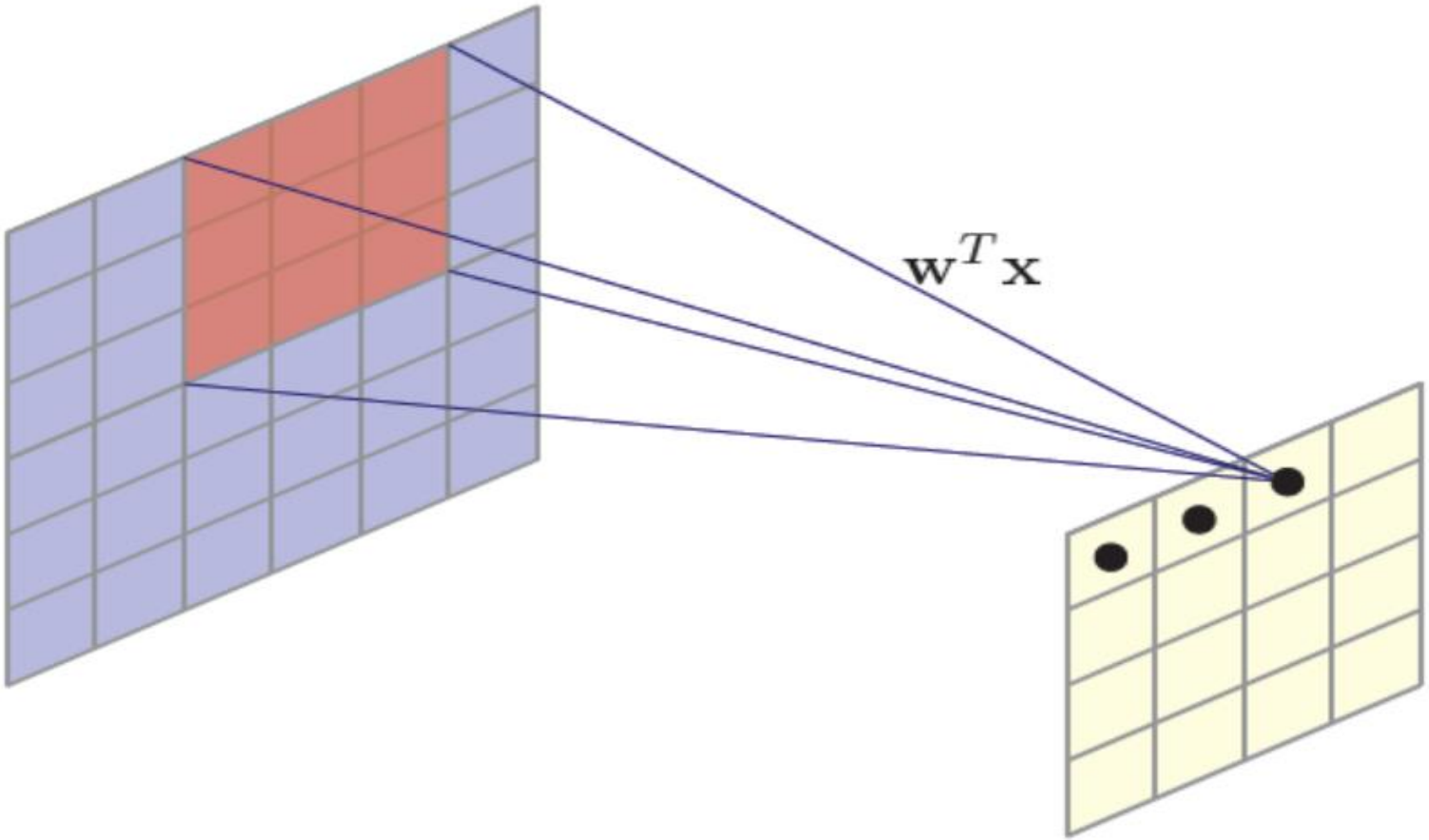

9个wx的和
输入大小=5*5
卷积核大小=3*3，patch大小=3*3
$\mathbf{w}^T\mathbf{x}$
输出y
大小不是人为设定的，是
由输入、卷积核等决定的

2D卷积

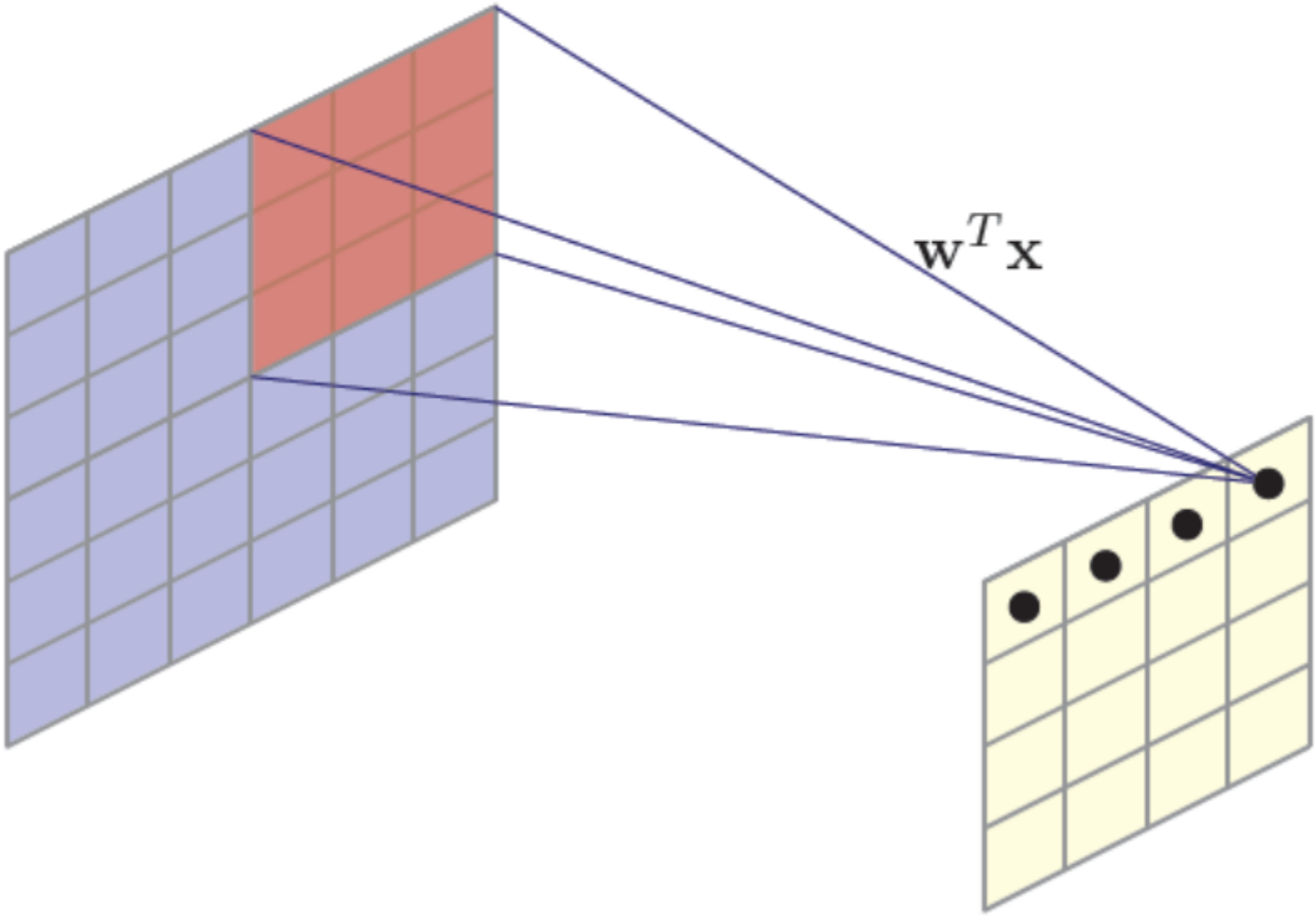
第二步，滤波器向右移动一格
一次移动的距离叫“步长”(stride)



2D卷积

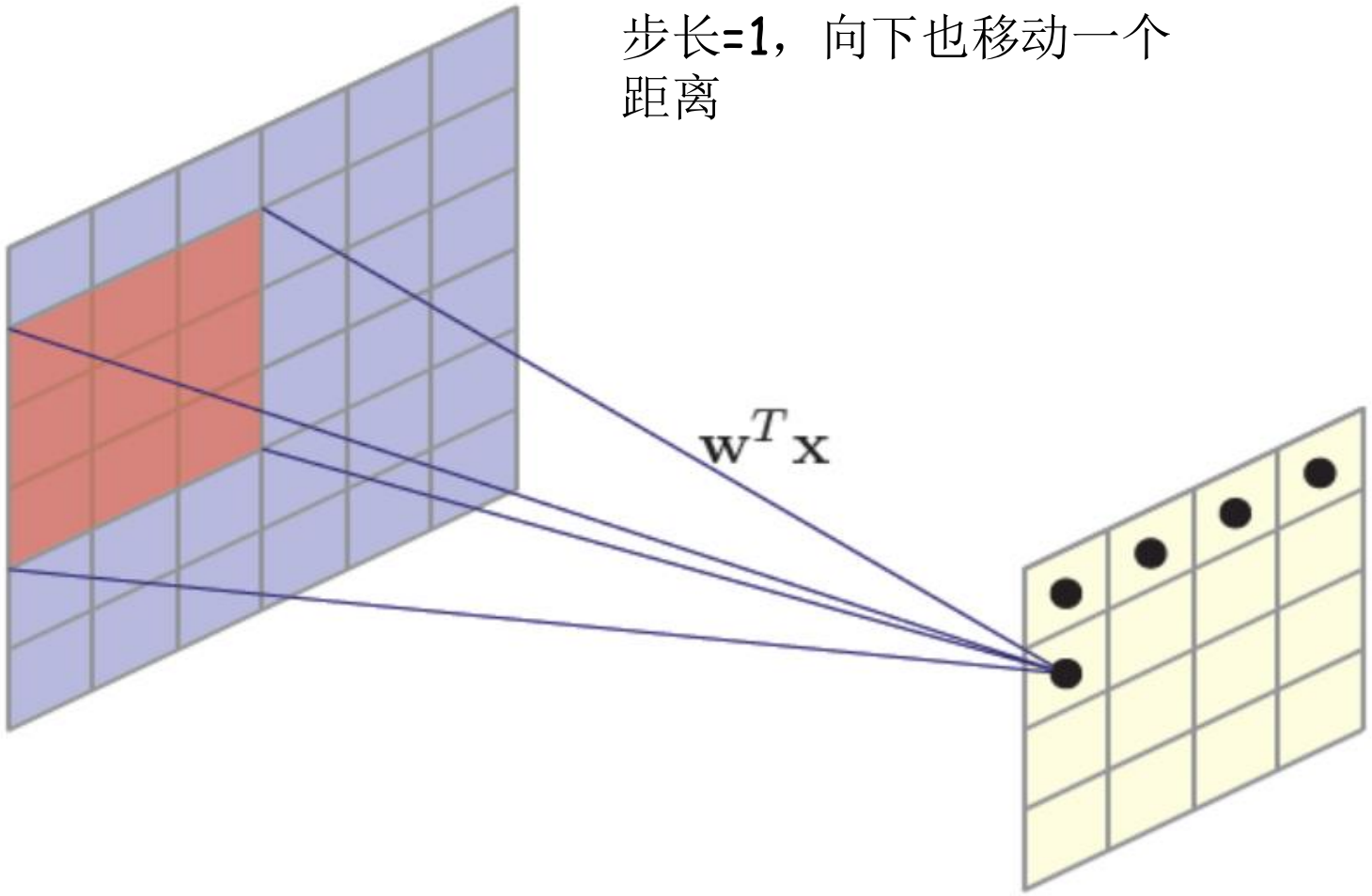


2D卷积

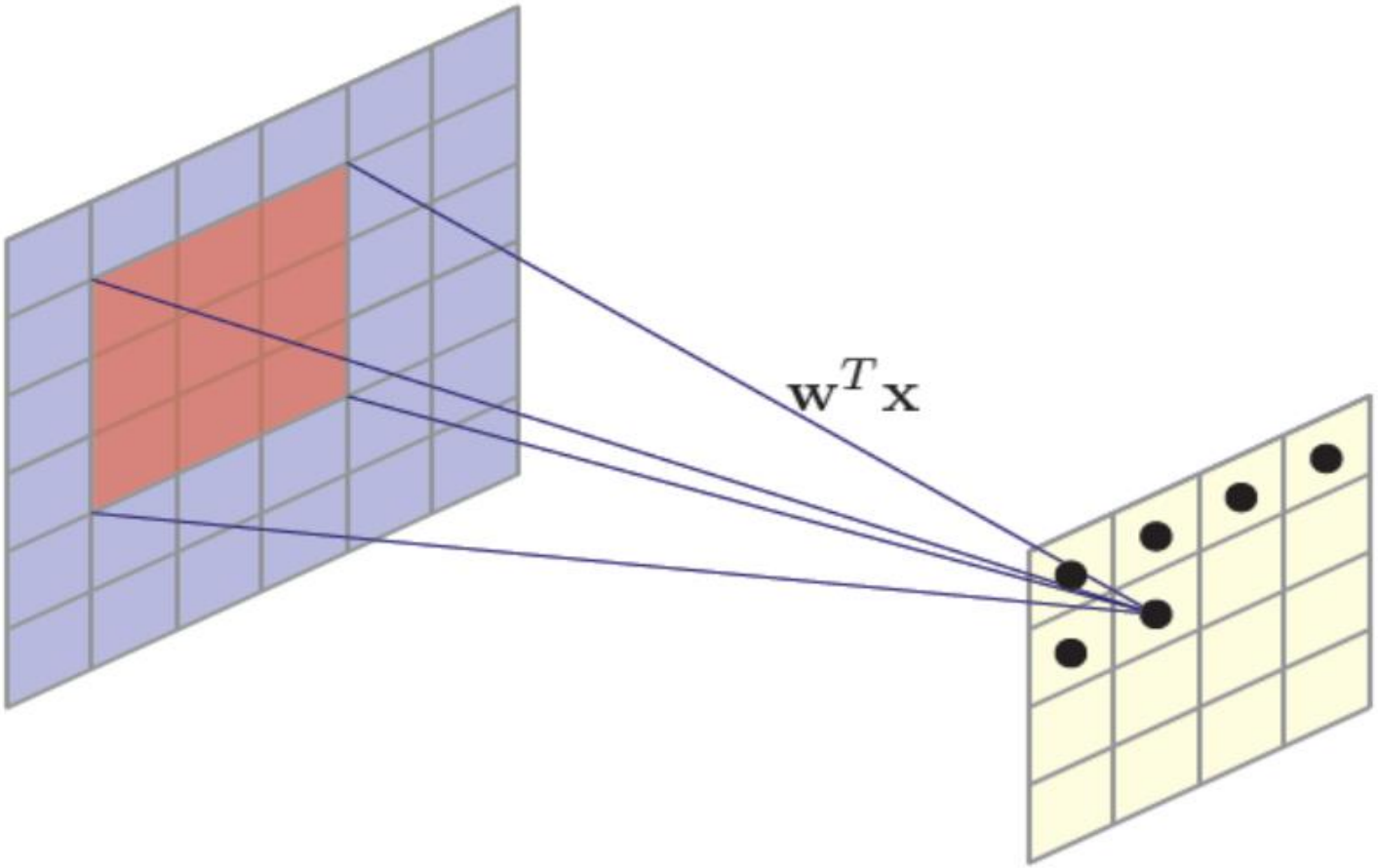


2D卷积

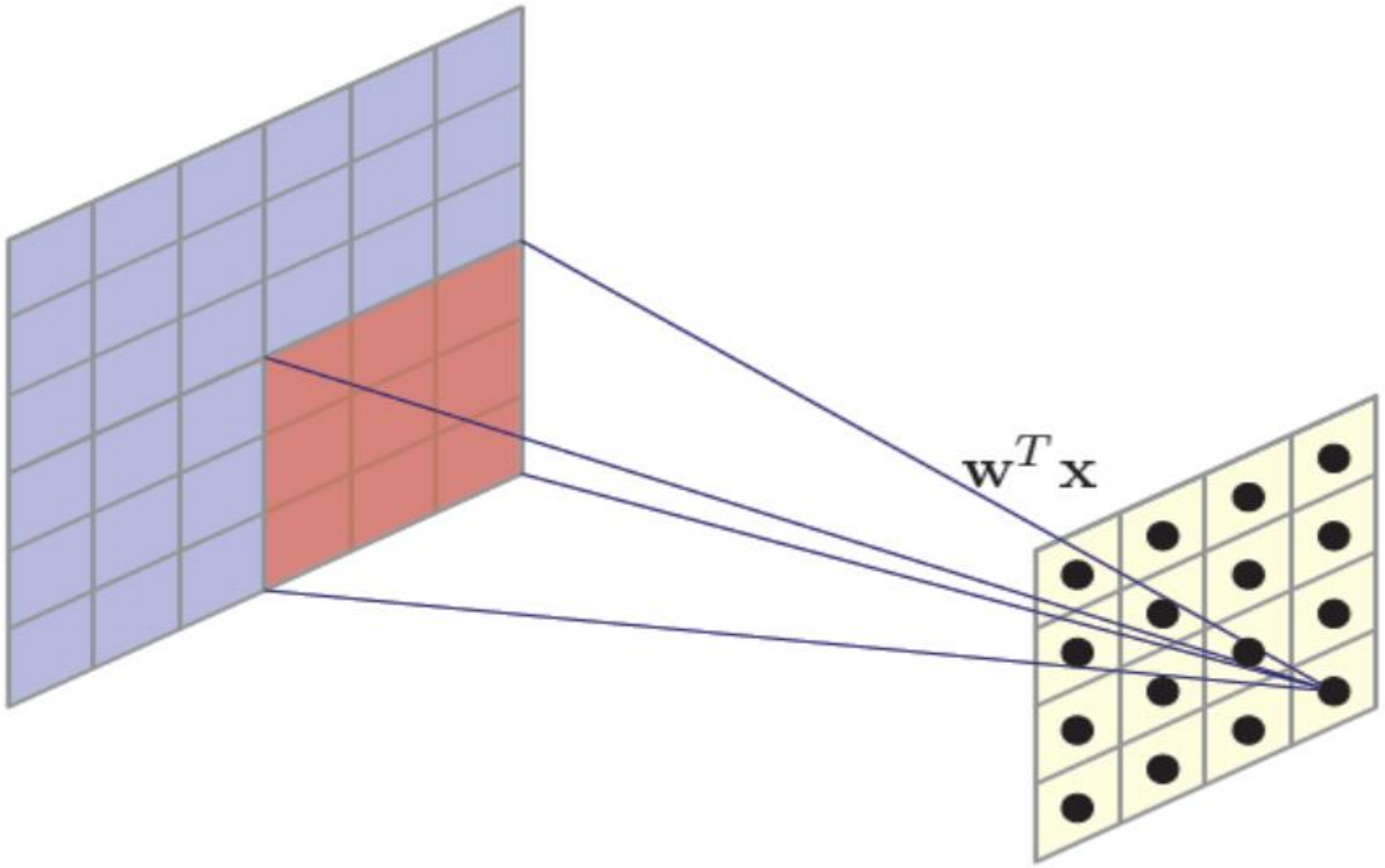
步长=1，向下也移动一个距离



2D卷积

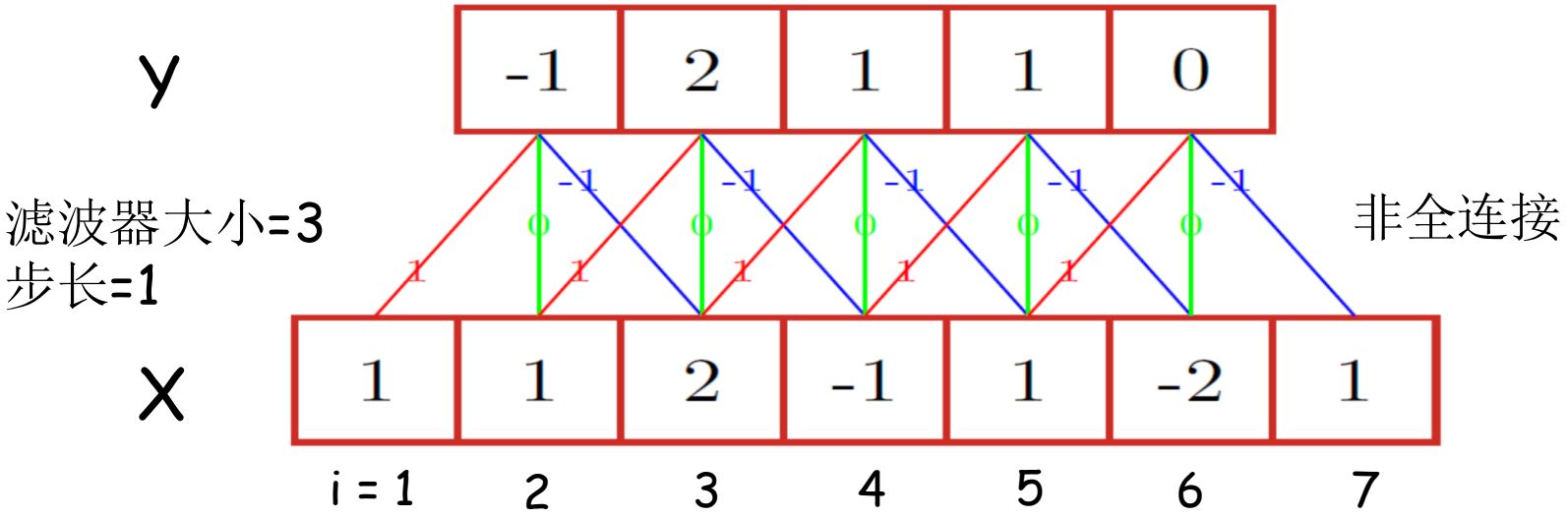


2D卷积



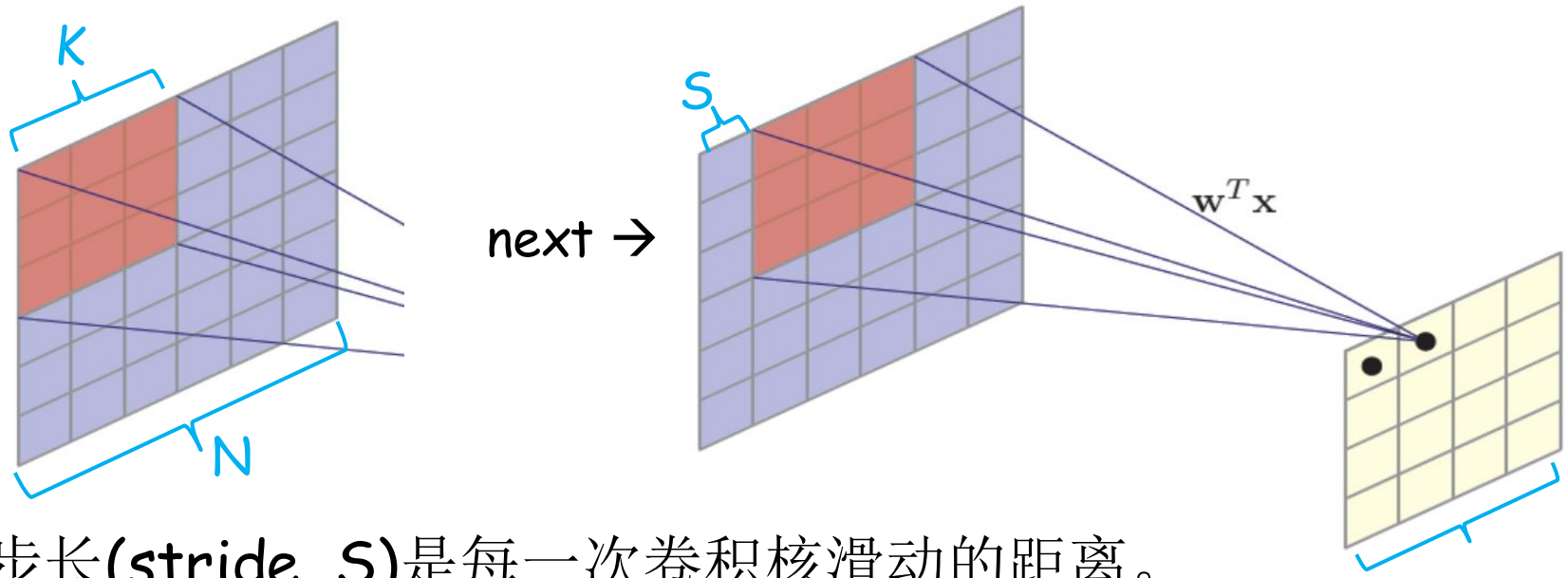
1D卷积

- 在1D卷积中，卷积核**在数据上沿1维滑动**
- 1维、2维是卷积核滑动方式的不同，而非输入形状的不同。1D卷积中X和滤波器也可以是矩阵形式。



$$y_t = \sum_{k=1}^{m} f_k \cdot x_{t-k+1}$$

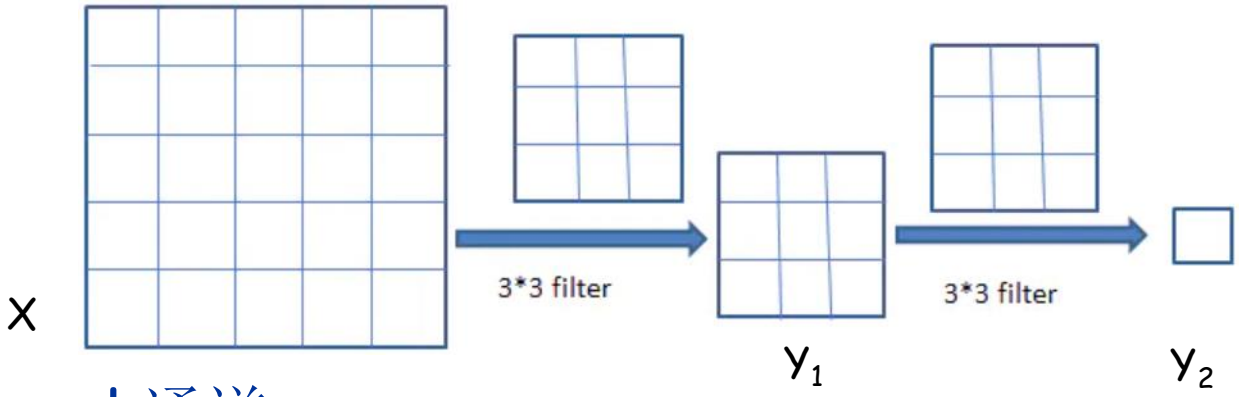
卷积:输出大小



- 步长(stride, S)是每一次卷积核滑动的距离。
- 假设使用的步长= 1, 卷积核大小 = 3 * 3
- 输出大小(此时看列数): $\frac{N-K}{S}+1$
- 本例子中: N = 6, K = 3, S = 1, 特征映射的列数= 4

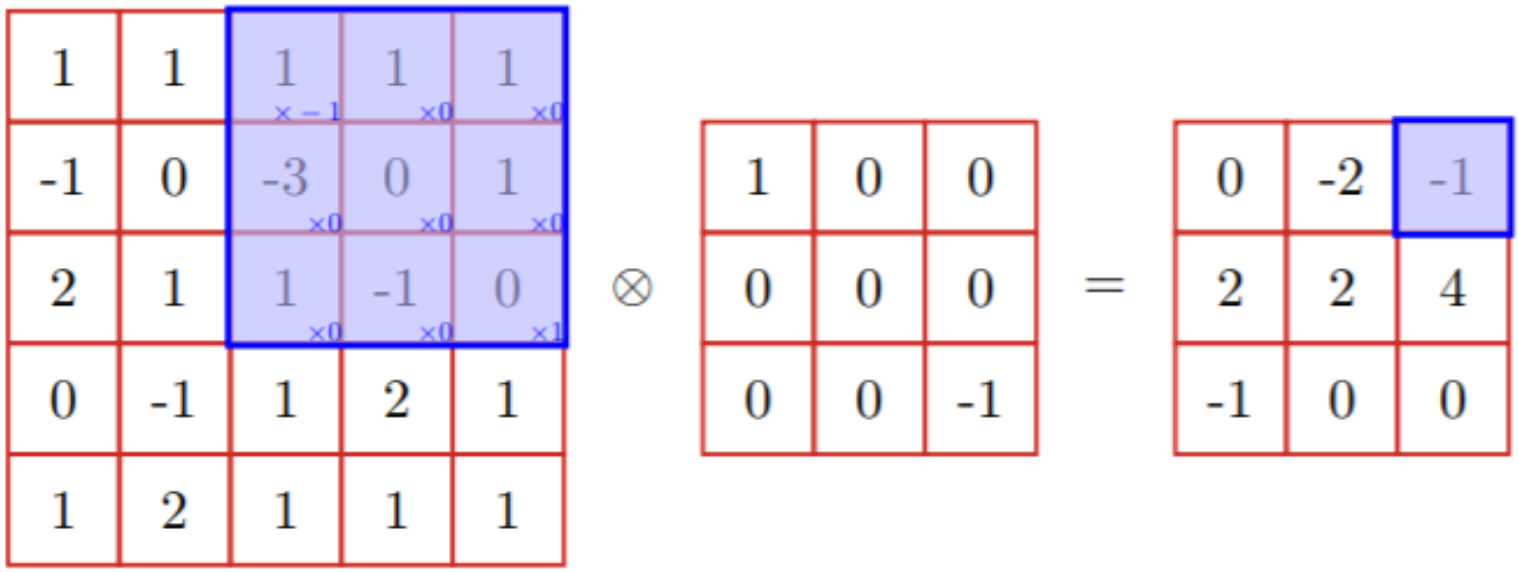
卷积中的概念

- Filter 滤波器 在NLP中基本等同
- (Convolutional) Kernel 卷积核
- Feature map 特征映射
- Receptive field 感受野 = 神经元对原始输入的感受范围



- Channel 通道
- 输入通道个数= 卷积核通道个数(NLP中一般是词向量维度)
- 输出通道个数= 每个卷积层中卷积核数量

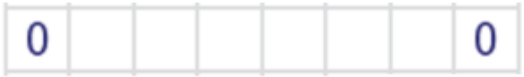
卷积的问题

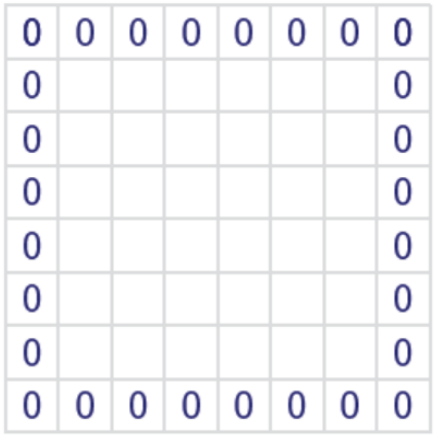


- 输入经过卷积之后，得到的矩阵大小比输入小很多，信息“缩水”非常严重，滤波器越大，输出尺寸越小
- 为了让更深层的layer的输入依旧保持有足够大的信息量，输出和输入最好能有相同大小的尺寸，如何做到？
- 此时，一般边缘的数值只会参与一次运算，而中间的数值更可能参与多次计算，如何平等利用边缘位置信息？

卷积的padding

- padding? 填充/补全。zero padding?用来补全的值是0
- 通常会把输入的两端都进行补全



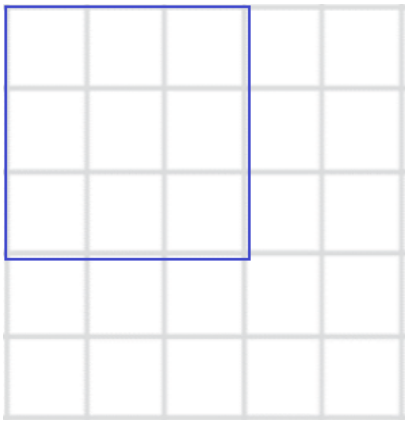


- stride = S, filters size = K:
- 不填充的输出大小M= $\frac{N-K}{S}+1$
- 如果想要输入输出等长，则一边补0个数=(N-M)/2
- 当S=1时，补(K-1)/2

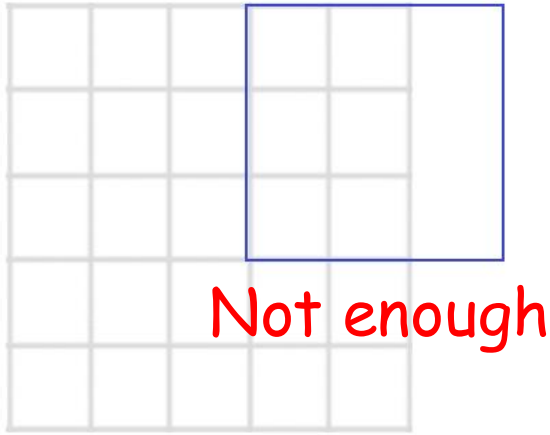
卷积的padding

- 在前面例子中步长= 1，所以在输入上滑动的时候，能够很自然地滑到最后
- 如果步长>1, 可能无法滑到最后。不填充：信息丢失。

卷积核大小=3



步长=1， ok

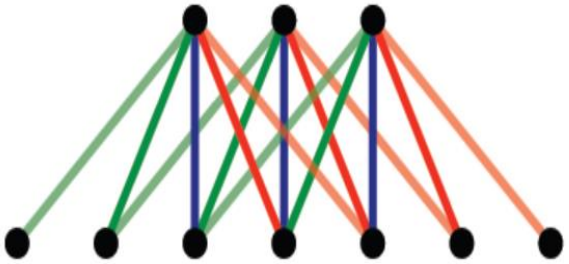


步长=3

卷积：根据padding分类

- 以一维卷积为例，假设卷积的输入长度=m, 输出长度=n, 滤波器大小=k, 步长=s. 输出要连接k个输入神经元
- 1) 窄卷积 (narrow) = 不做padding

卷积后



卷积前

$n= (m-k)/s+1$

m=7, k=5, s=1

- 2) 宽卷积：两端padding. 全填充：每个输入神经元要与k个输出神经元连接(s=1) 分别补k-1个0 $n= (m+k-2)/s+1$

滤波器长度相对输入较大时，一般用宽卷积

确保滤波器中的所有权重都到达整个句子

保证滤波器应用于输入句总是产生有效的非空结果

卷积：根据padding分类

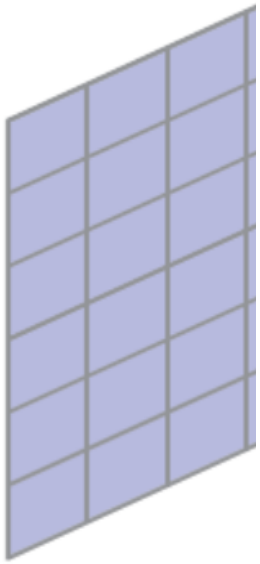
- 假设卷积的输入长度=m, 输出长度=n, 滤波器大小=k, 步长=s. 以一维卷积为例
- 3) 等宽卷积 (Equal-Width Convolution) 默认情况

步长=1时：
两端补(k-1)/2

n=m

多个滤波器

- 在一层卷积中，可以设置多个滤波器，这些滤波器的大小可以不一样，用于捕获不同的特征，这样得到多个特征映射



高斯滤波器，平滑去噪

$\frac{1}{16}$	$\frac{1}{8}$	$\frac{1}{16}$
$\frac{1}{8}$	$\frac{1}{4}$	$\frac{1}{8}$
$\frac{1}{16}$	$\frac{1}{8}$	$\frac{1}{16}$

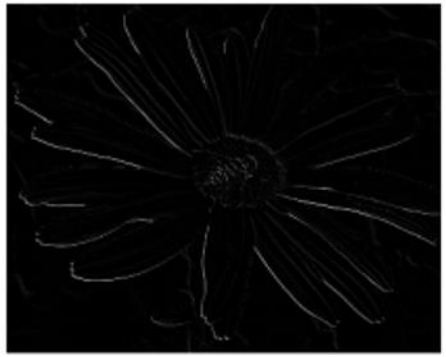
=

*

0	1	1
−1	0	1
−1	−1	0

滤波器

=

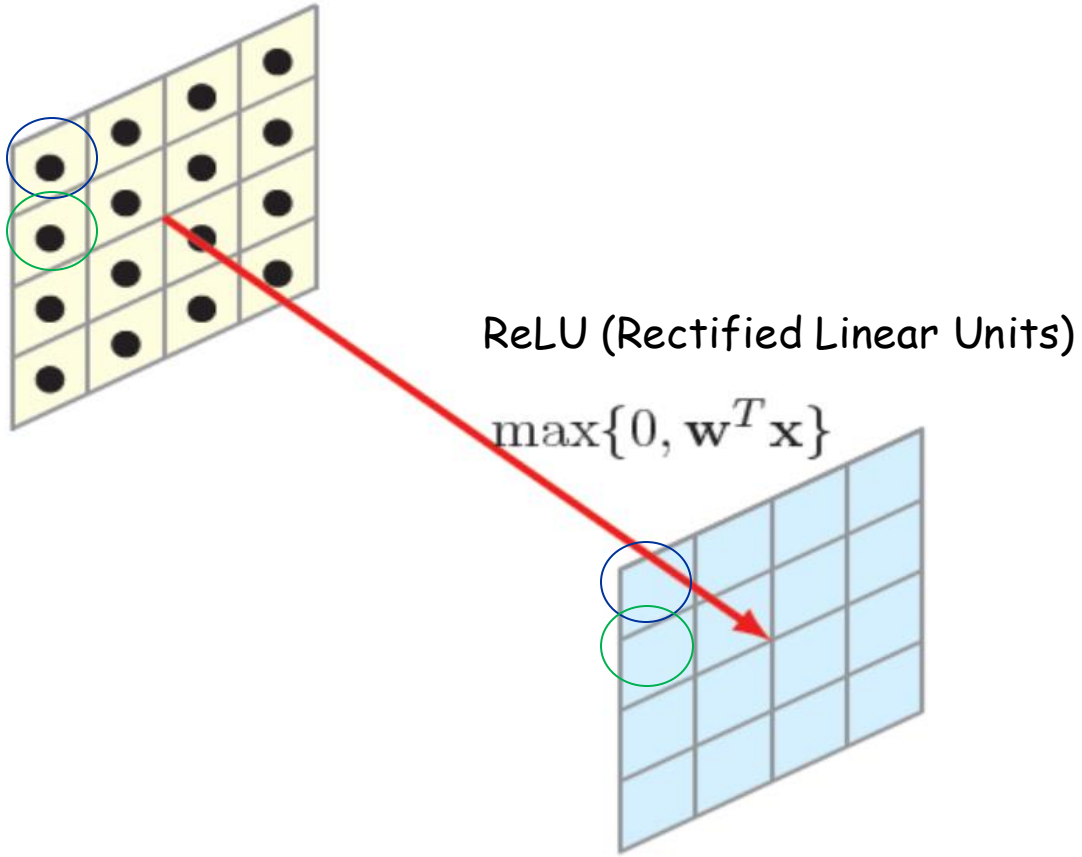


输出特征映射

提取边缘特征

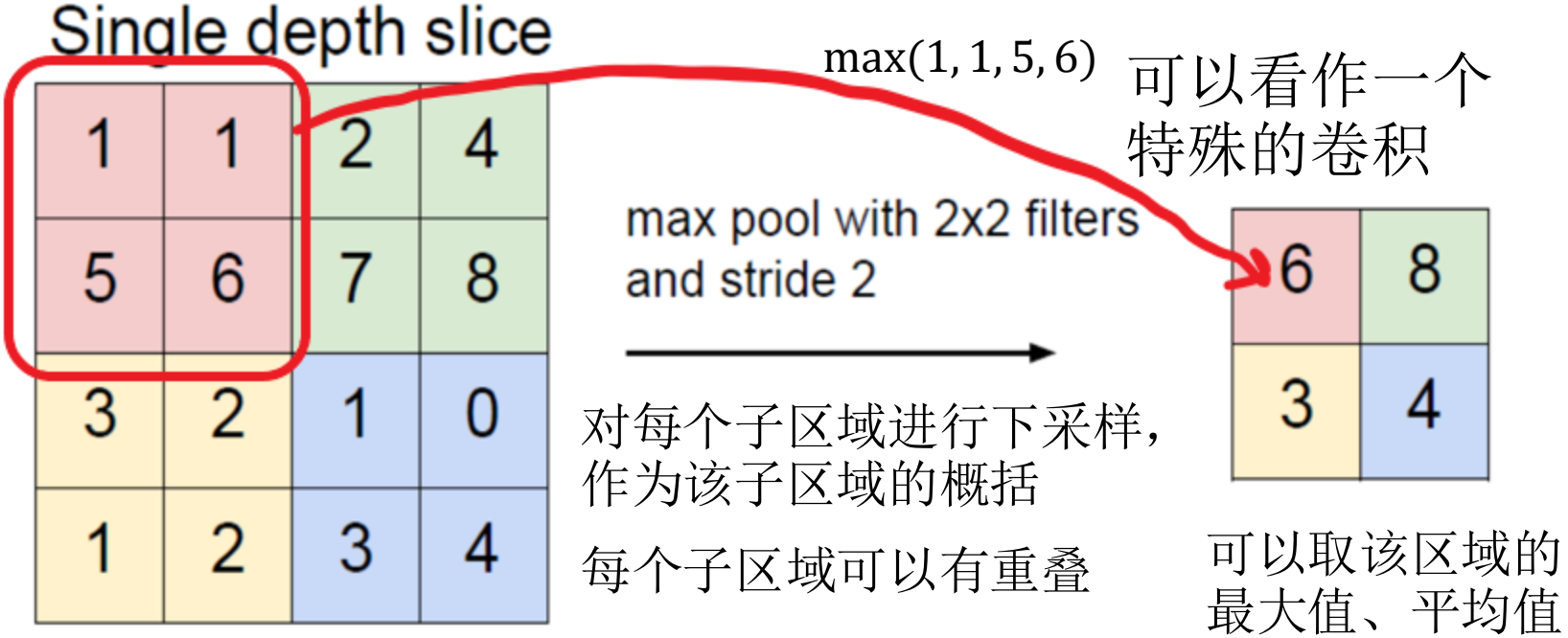
激活

- 卷积操作是乘法和加法，本质上是线性的
- 通过卷积操作获得特征映射后，采用point-wise/ element-wise 非线性操作，大小不变



池化

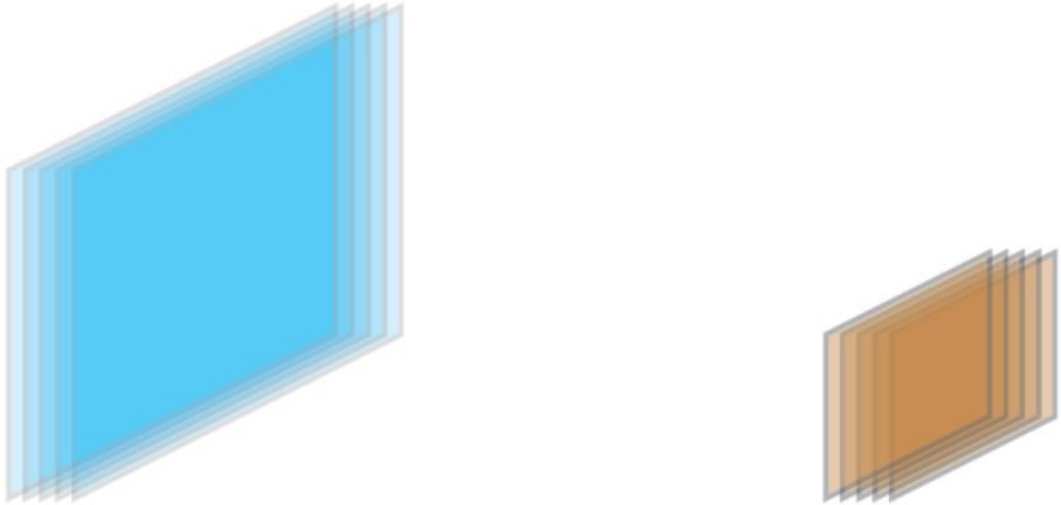
- 卷积层可以显著减少网络中连接的数量， 但特征映射中的神经元个数并没有显著减少 → 直接后接分类器容易过拟合
- pooling 池化层又称下采样层(Subsampling)、汇聚层，可以降低特征维数。实际上是一个提取重点、降维的过程。



经过了卷积、激活的特征映射

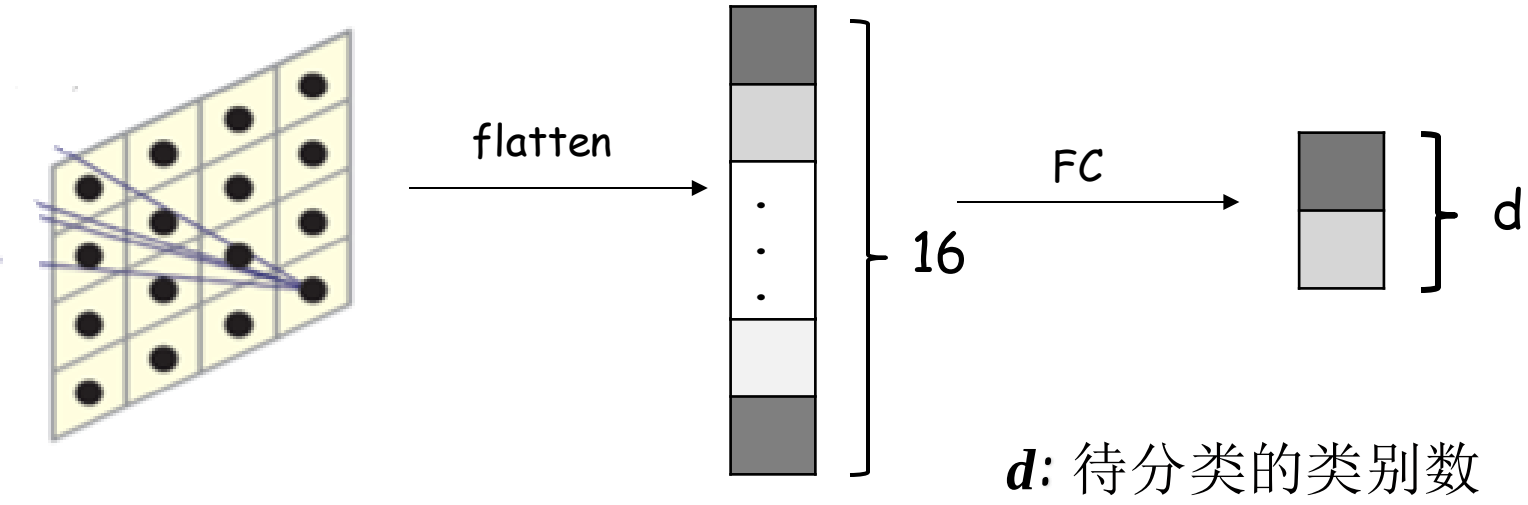
池化

- 如果使用了n个滤波器，得到n个特征映射，则对它们分别进行池化，又可以得到n个降维后的特征映射
- 池化层不但可以有效地减少神经元的数量、减少参数量，还可以使得网络对一些小的局部形态改变保持不变性(e.g. 最大池化取最大值)，并拥有更大的感受野。
- 过大的采样区域会急剧减少神经元的数量，造成过多的信息损失，因此要有一个合适的选择



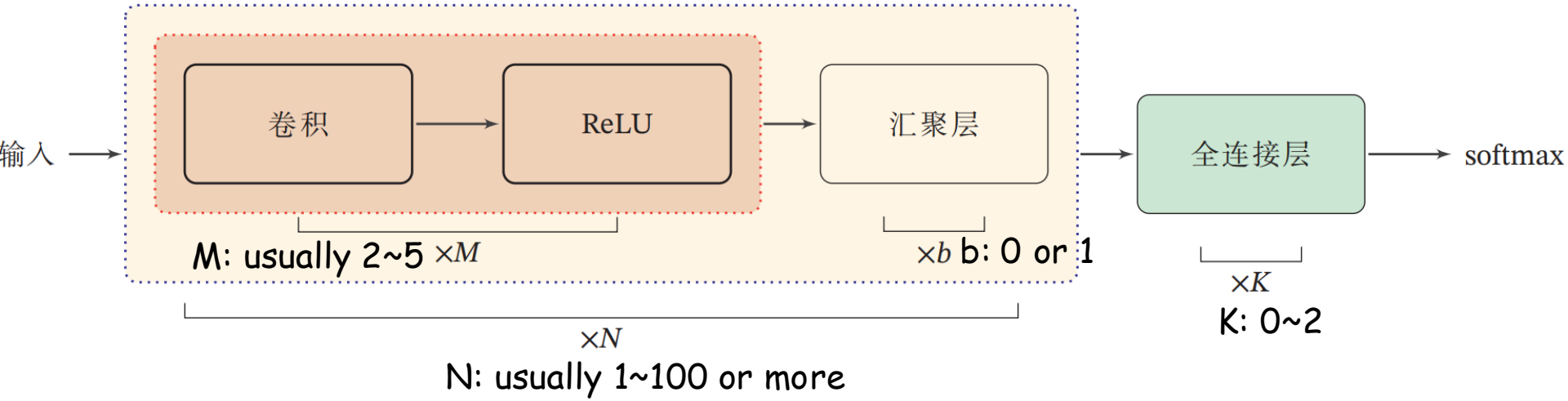
全连接层

- 将池化后的特征映射flatten (压平), e.g. 4*4的二维向量压平成长度为16的一维向量
- 全连接层出现在整个网络的最后部分，接受压平后的向量，充当分类器的角色



- 卷积、池化等：将原始数据映射到隐层特征空间
- 全连接层：将学到的特征表示映射到样本的标记空间

CNN整体结构



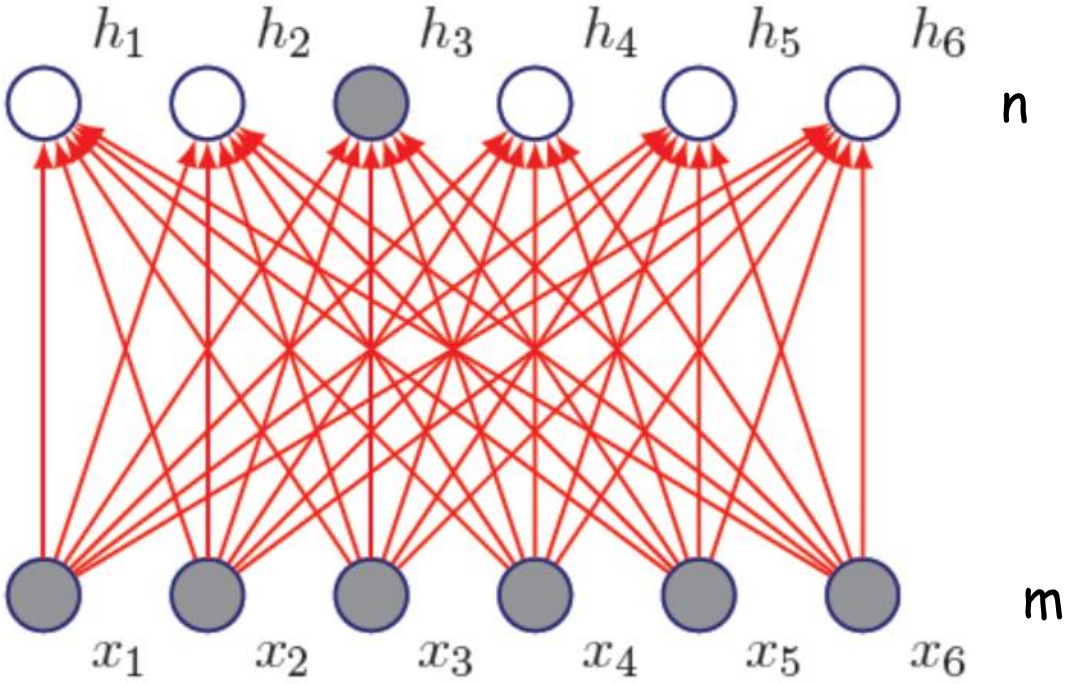
- 卷积网络的整体结构趋势:
  - 使用更小的卷积核(e.g. 1 × 1 和 3 × 3)
  - 更深的网络结构 (e.g. 50 layers...)
  - 越来越灵活的卷积操作(e.g. 不同的步长)
  - pooling的作用越来越小 (比例逐渐降低，趋于全卷积)

CNN特点

- 卷积的几个重要特征:
  - 稀疏连接
  - 参数共享
  - 接受变长数据
  - 等变表示 (Equivariant representations)

(1) 稀疏连接

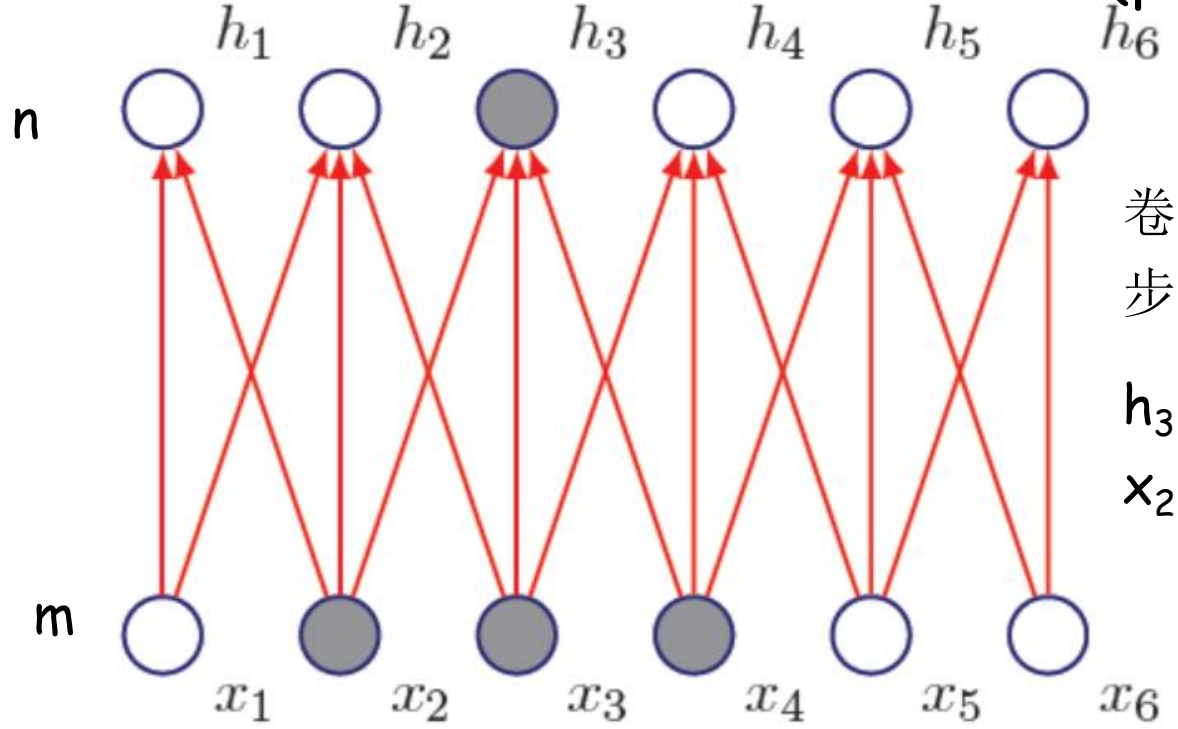
- 前馈神经网络中($y \in R^n$, $x \in R^m$): 每一层都需要进行矩阵乘法$y = Wx$. e.g. h_3需要利用所有输入神经元计算得到



- 全连接的稠密网络结构, 需要m*n个权重项表示输入与输出之间的关系 → O(m×n)

(1) 稀疏连接

- 卷积网络通过尺寸小于输入的卷积核实现稀疏连接
- 输出神经元的计算需要用到k个输入神经元(padding)



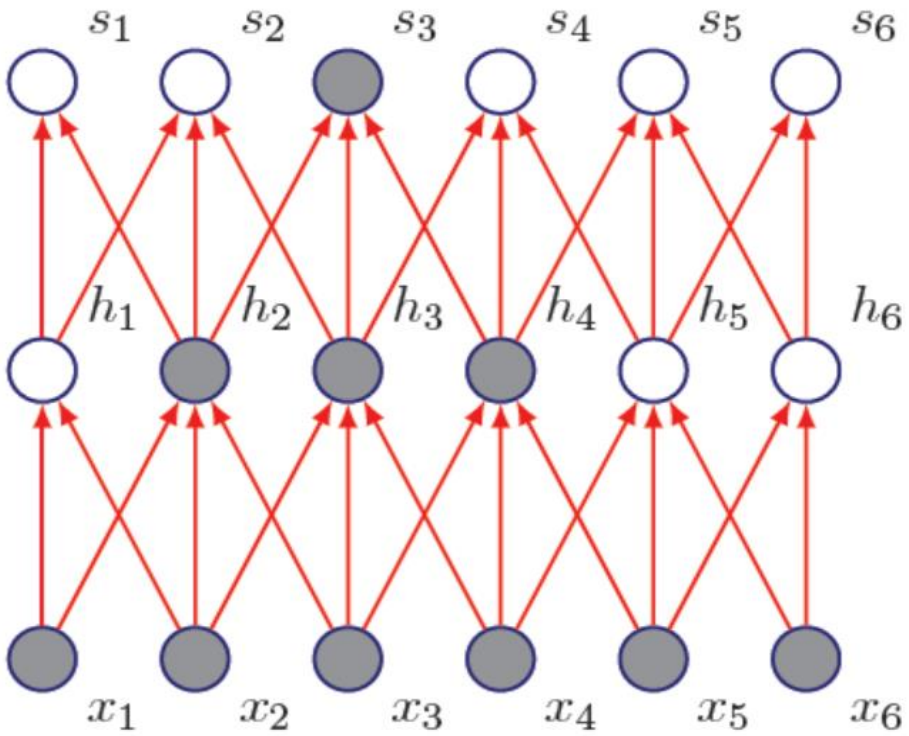
卷积核大小k= 3,
步长 = 1

h_3 的计算依赖于
x_2, x_3, x_4

- CNN参数量更少，需要n*k个权重项表示输入与输出之间的关系. O(m×n) vs O(k×n)

(1) 稀疏连接

- 深层中的神经元可以间接连接到输入中的所有神经元
- 连接到更多的输入神经元意味着更大的感受野



s_3: 感受野范围
$=x_1$-x_5

h_3: 感受野范围
$=x_2$-x_4

(2) 参数共享

- 前馈神经网络中: 权重矩阵W中的某一列和某一个特定输入相乘，W的每一列都是不共享的
- 在卷积网络中, 参数是绑定的: 卷积核在输入上面滑动，对每一个位置，参与计算的都是同样的滤波器参数，即对于一个卷积核来说参数共享
- CNN中的权重共享可以理解为一个卷积核只捕捉输入数据中的一种特定的局部特征
- 前向传播的复杂度为$O(k \times n)$
- 由于 $k << m, n$，CNN相对于前馈网络需要更小的存储空间

(3) 处理变长数据

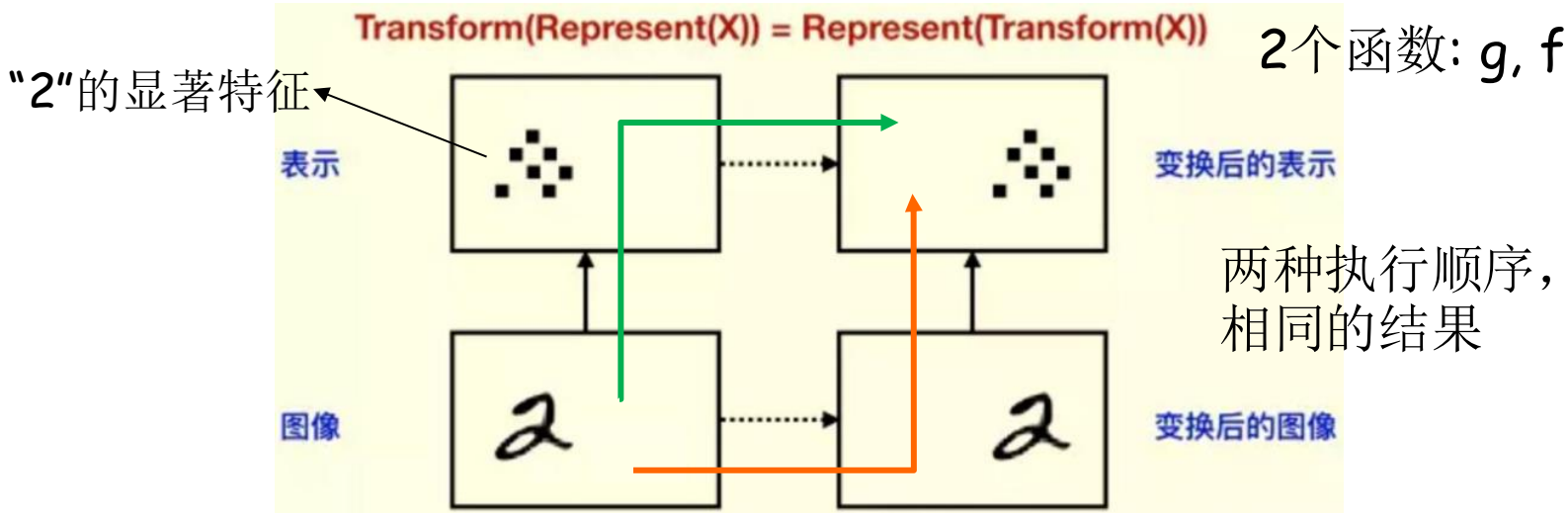
- 卷积层中滤波器大小与输入长度无关，在输入数据上任意位置的操作均使用固定的卷积核，**卷积操作本身与输入数据尺寸无关**。如果后接全连接层，则可能对输入尺寸有要求。
- 池化层进行了下采样，**全局**采样可以实现与数据长度的无关

(4) 等变性(Equivariance)

- 当f(g(x)) = g(f(x))，即函数f等同于函数g
- CNN中，每一个区域(patch)和一个固定的滤波器进行计算。滤波器就是一个特征检测器，所以无论某个成分出现在输入中的哪个位置，都会检测到同样的这些特征，输出同样的响应
- 针对滤波器的**参数共享**让卷积就像是实现了输入的平移。如果一个函数g可以把输入进行平移，那么g和卷积就是等同的

(4) 等变性(Equivariance)

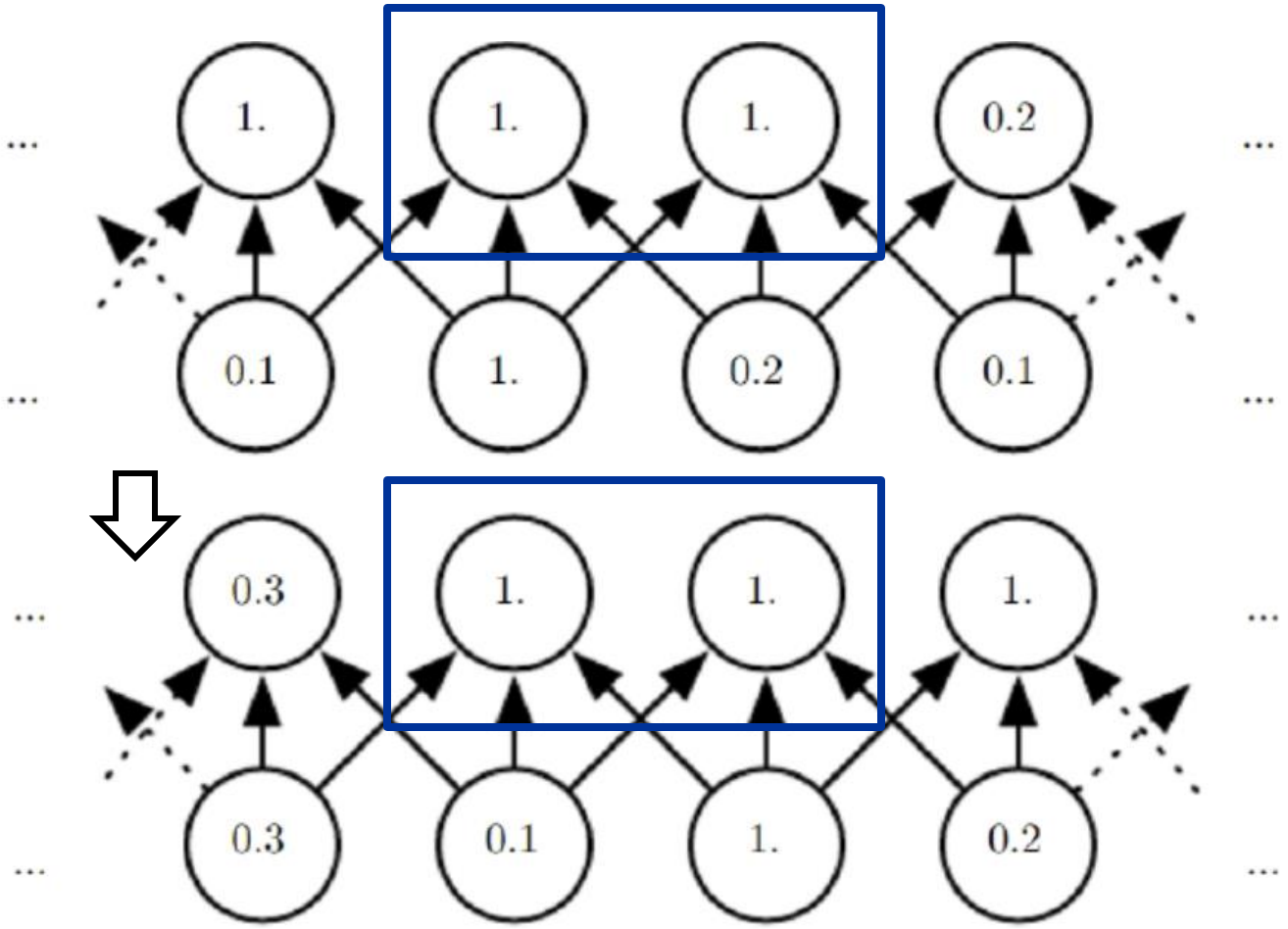
- 在处理时间序列数据时，卷积产生一个时间表，显示不同的特征出现的时间。如果输入中的某一个事件发生了时间偏移，那么输出中将出现相同的表示形式
- 例子: 输入图片里的物体移动，对应的特征位置也会跟着转移
- 滤波器对特定的特征才会有较大激活值，所以不论某一特征平移到何处，滤波器都会找到该特征并在此处呈现较大的激活值。先移动，再表示=先表示，再移动。这就是CNN的“平移等变性”



平移不变性 (Invariance)

- 和“平移等变性” 相反的是平移不变性(Invariance)
- 平移不变性意味着不管它的输入如何平移，系统会产生完全相同的输出。即对输入x进行改动，输出不变:
  $f(g(x)) = f(x)$
- 池化实现了小范围的平移不变性。E.g. 最大池化
- CNN既具有不变性，又具有等变性。
- 举例：如果输出是给出图中猫的位置，则将猫从左移到右，平移也会体现在输出上，检测猫特征的位置也从左到了右，说明CNN等变性；如果输出图中是否有猫，那么无论猫怎么移动，都输出"有猫"，说明CNN不变性。

平移不变性(Invariance)



最大池化

如果更关心某个特征是否存在，而不是它的确切位置，那么局部平移不变性会很有意义。