The Principle and Application of CNN

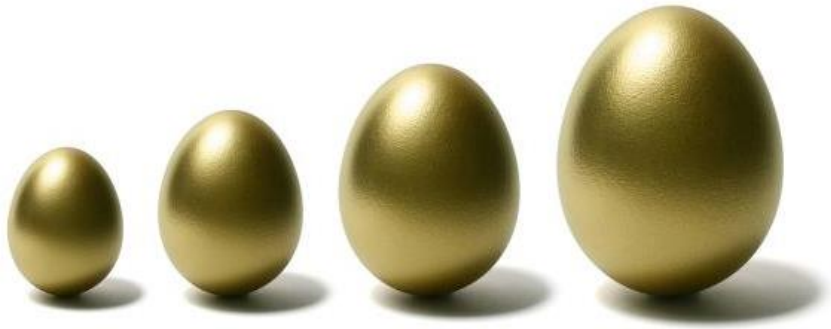
AI赋能培训系列课程

报告人：吴庆甜
指导教师：周翊民

智能仿生中心 2018.11

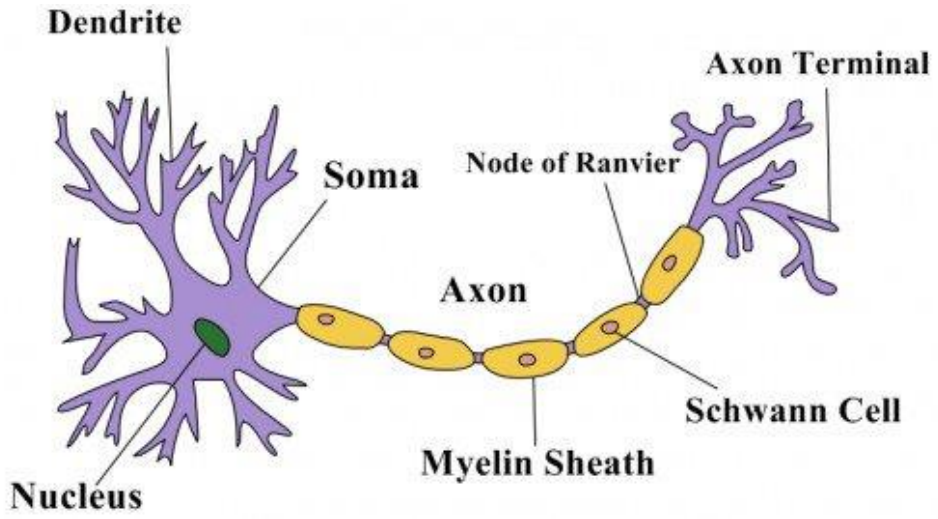
Outline

- McCulloch-Pitts neuron
- Multilayer perceptron model
- **Back-propagation Network**
- **Convolutional Neural Network**



1. what are neurons?

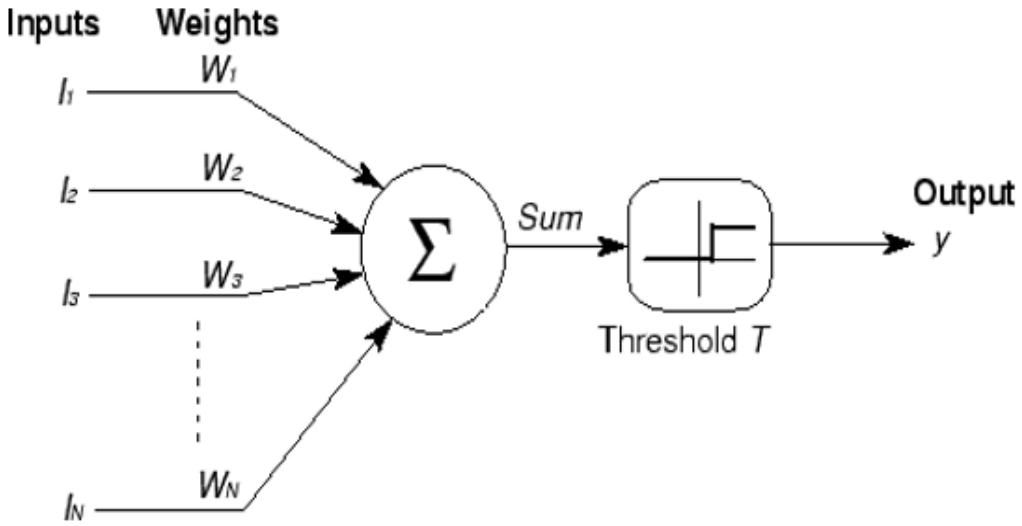
Neurons are nerve cells, composed of 3 basic parts: **cell body, dendrites（树突）** and **axon（轴突）**.



- **Cell body(Soma)** is to keep the neuron alive by performing tasks such as energy production and protein synthesis.
- **Dendrites** is to receive information from other neurons.
- **Axon** is to transmit information from one neuron to another.

2. The McCulloch-Pitts Neuron

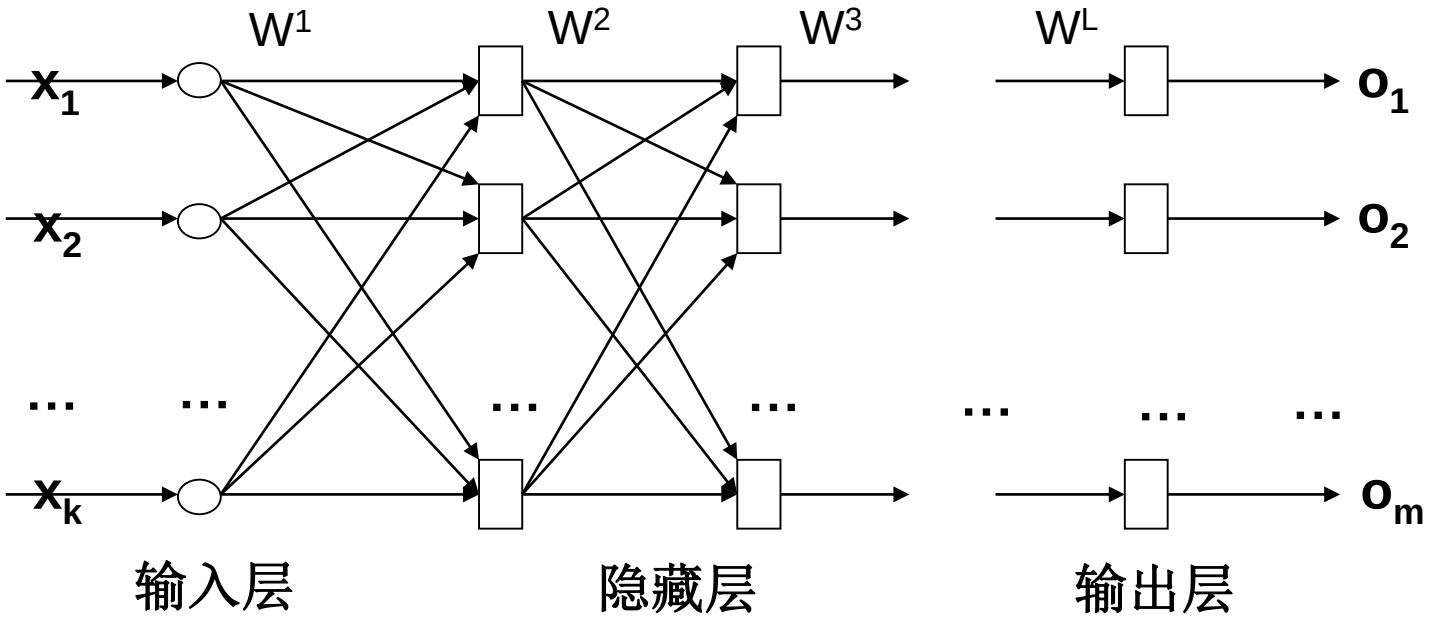
Neuron	M-P
neuron	j
input	l_i
weight	w_i
output	y_j
sum	$\sum$
Membrane po tential	$\sum_{i=1}^{n} w_i l_i$
Threshold	θ_j
Activation Func	**F**



$$y_j = f(\sum_{i=1}^{n} w_i l_i - \theta_j)$$

This vastly simplified model of real neurons is also known as a **Threshold Logic Unit**

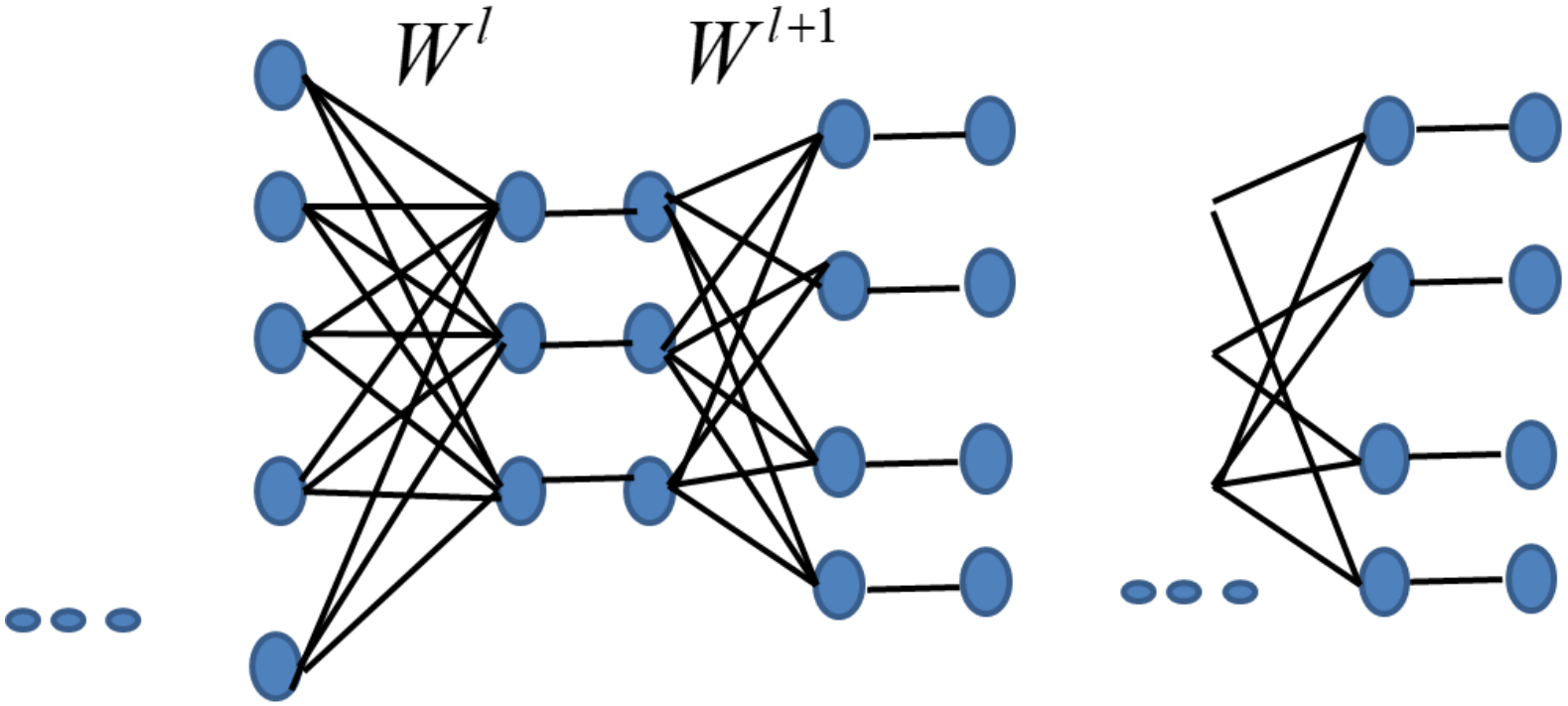
3. Multilayer perceptron（多层感知器）



$$o^{(n)} = f_L(w^L \ldots \cdot f_2(w^2 \cdot f_1(w^1 \cdot x^{(n)})))$$

信息正向传播

4. Back Propagation Network



$$o^{(n)} = f_L(w^L \ldots \cdot f_2(w^2 \cdot f_1(w^1 \cdot x^{(n)})))$$

$$E^{(i)} = \frac{1}{2}\sum_{j=1}^{m}\left(y_j^{(i)} - o_j^{(i)}\right)^2$$ ⟹ 全局误差

BP-Forward and backward pass

基本思想

- **Forward pass-向前传播阶段：**

（1）从样本集中取一个样本($x^{(i)}$，$y^{(i)}$)，将$x^{(i)}$输入网络；

（2）计算相应的实际输出：

$$o^{(n)} = f_L(w^L \ldots \cdot f_2(w^2 \cdot f_1(w^1 \cdot x^{(n)})))$$

- **Back propagation—误差反向传播阶段：**

（1）计算实际输出与相应的真实输出的差；

$$E^{(i)} = \frac{1}{2}\sum_{j=1}^{m}\left(y_j^{(i)} - o_j^{(i)}\right)^2$$

（2）按极小化误差的方式调整权矩阵。

（3）网络关于第i个样本的误差测度：

$$E = \sum_{i=1}^{N} E^{(i)}$$

梯度下降训练算法

- **梯度下降算法(gradient-descent)**

  输入：训练样本集合：$X=\{X^{(1)},...,X^{(N)}|Y^{(1)},...,Y^{(N)}\}$　学习速率η

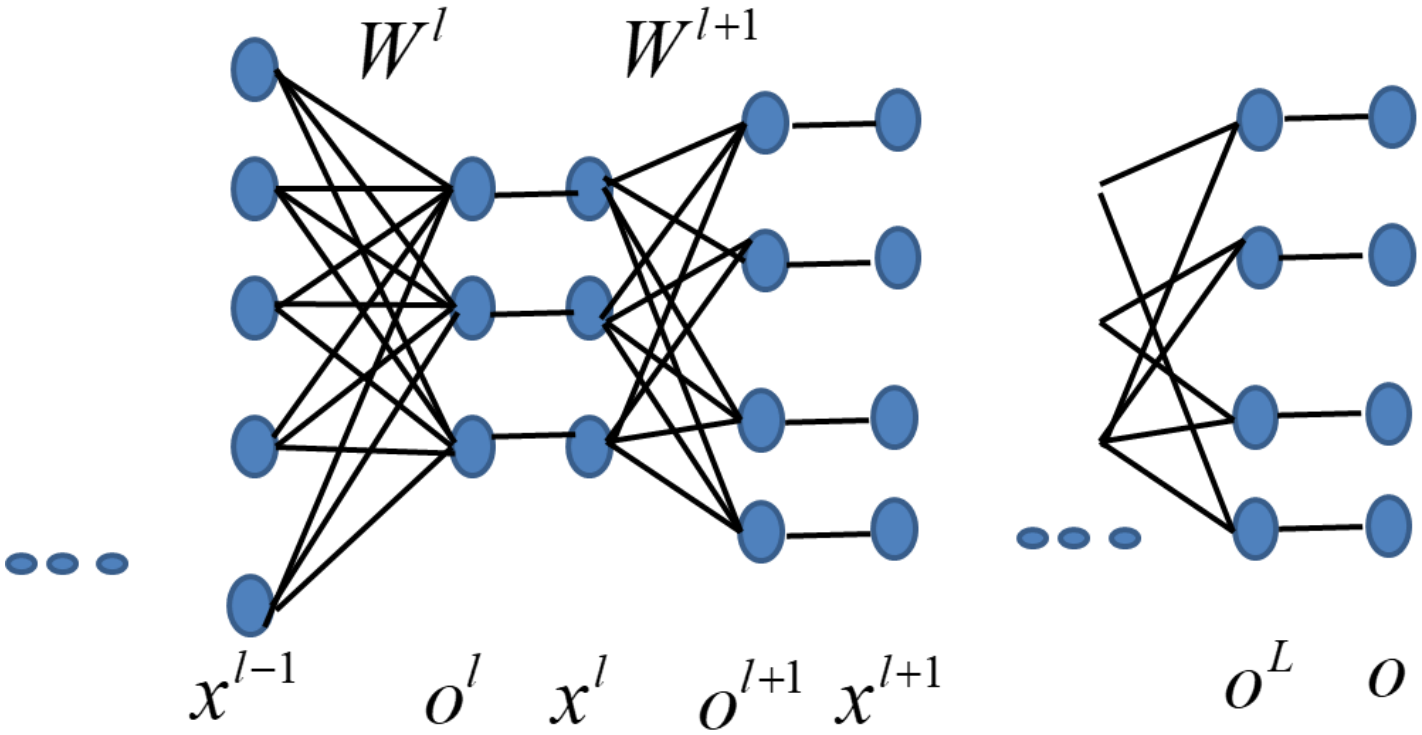
  输出：权值向量：w

  - **初始化**　w 的各个分量 为0-1.0间的随机数
  - 循环更新权值向量：

$$\Delta w_k = \eta \frac{\partial E}{\partial w_k}$$

$$w_k \leftarrow w_k + \Delta w_k$$

反向传播（BP）算法的权值更新



W随机初始化，怎么通过误差反向传播来更新权值W呢？

BP的权值更新与公式推导

$$E = \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{m}\left(y_j^{(i)} - o_j^{(i)}\right)^2 \tag{1}$$

$$x^l = f\left(u^l\right) \qquad o^l = W^l x^{l-1} + w_0 \tag{2}$$

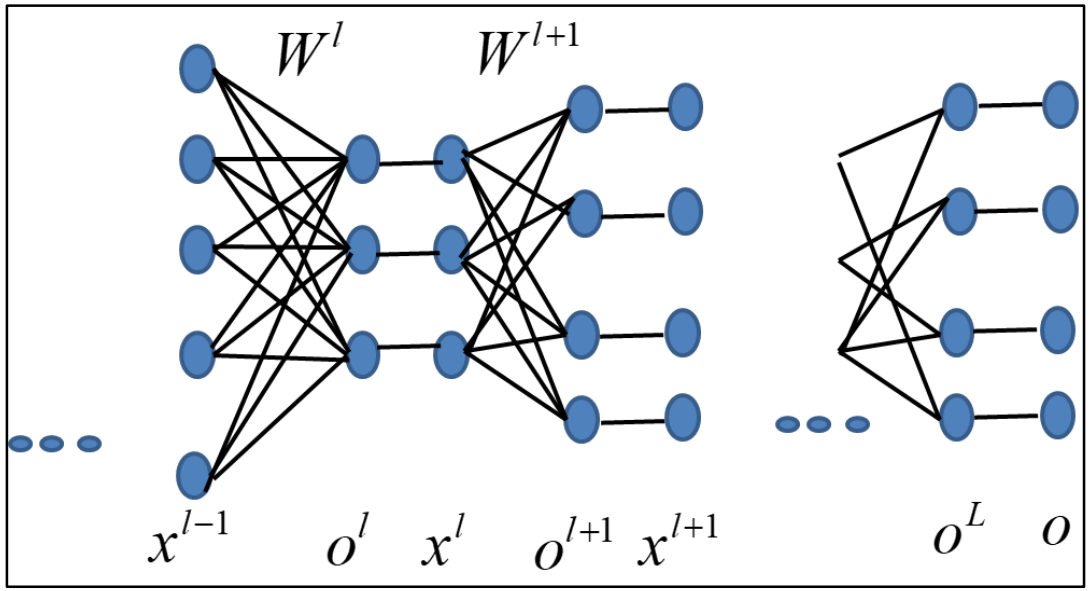
$$\delta^l = (W^{l+1})^T \cdot \delta^{l+1} \circ f'(\mathrm{o}) \tag{3}$$

$$\delta^L = f'(\mathrm{o}^L) \circ \sum_{i=1}^{N}(\mathrm{y}^{(i)} - \mathrm{o}^{(i)}) \tag{4}$$

$$\frac{\partial E}{\partial w^l} = x^{l-1} \circ (\delta^l)^T \tag{5}$$

$$\Delta W = -\eta \frac{\partial E}{\partial W} \tag{6}$$

BP的权值更新与公式推导



$$E=\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{m}\left(y_j^{(i)}-o_j^{(i)}\right)^2 \quad (1)$$

$$x^l=f\left(u^l\right) \qquad o^l=W^l x^{l-1}+w_0 \quad (2)$$

$$\delta^l=(W^{l+1})^T\cdot\delta^{l+1}\circ f'(\mathrm{o}) \quad (3)$$

$$\delta^L=f'(\mathrm{o}^L)\circ\sum_{i=1}^{N}(\mathrm{y}^{(i)}-\mathrm{o}^{(i)}) \quad (4)$$

$$\frac{\partial E}{\partial w^l}=x^{l-1}\circ(\delta^l)^T \quad (5)$$

$$\Delta W=-\eta\frac{\partial E}{\partial W} \quad (6)$$

反向传播（BP）算法的权值更新

$$\delta^l = (W^{l+1})^T \cdot \delta^{l+1} \circ f'(o)$$

(3)的推导

$$\delta^l = \frac{\partial E}{\partial o^l}$$

$$= \frac{\partial E}{\partial o^{l+1}} \circ \frac{\partial o^{l+1}}{\partial o^l}$$

$$= \delta^{l+1} \circ W^{l+1} \cdot f'(o^l)$$

$$o^l = W^l x^{l-1} + w_0$$

$$o^{l+1} = W^{l+1} x^l + w_0$$

反向传播（BP）算法的权值更新

$$\delta^L = f'(\mathrm{o}^L) \circ \sum_{i=1}^{N} (\mathrm{y}^{(i)} - \mathrm{o}^{(i)})$$

(4)的推导

$$\delta^L = \frac{\partial E}{\partial o^L}$$

$$= \frac{\partial E}{\partial o} \circ \frac{\partial o}{\partial o^L}$$

$$= (y^{(i)} - o^{(i)}) \circ f'(o^L)$$

$$o = f(o^L)$$

反向传播（BP）算法的权值更新

$$\frac{\partial E}{\partial W^l} = x^{l-1} \circ (\delta^l)^T$$

(5)的推导

$$\frac{\partial E}{\partial W^l} = \frac{\partial E}{\partial o^l} \circ \frac{\partial o^l}{\partial W^l}$$

$$= \frac{\partial E}{\partial o^l} \circ \frac{\partial o^l}{\partial W^l}$$

$$= x^{l-1}(\delta^l)^T$$

$$o^l = W^l x^{l-1} + W_0$$

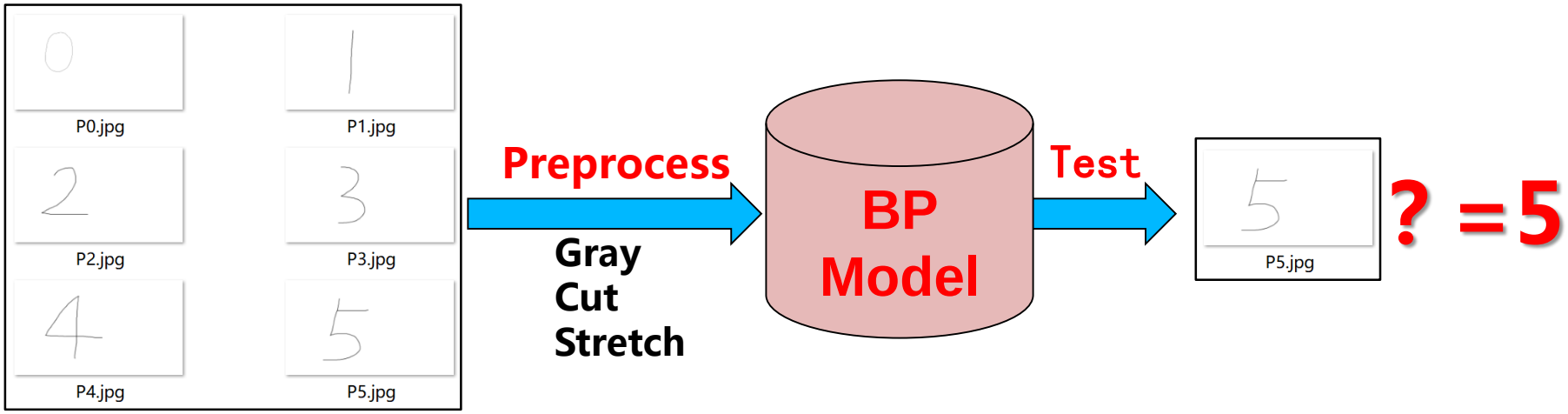
反向传播（BP）算法

- **Learning task of BP-反向传播算法的学习任务**

1. 搜索一个巨大的假设空间，这个空间由网络中所有的单元的所有可能的权值定义，得到误差曲面
2. 在多层网络中，误差曲面可能有多个局部极小值，梯度下降仅能保证收敛到局部极小值
3. 尽管有这个障碍，已经发现对于实践中很多应用，反向传播算法都产生了出色的结果

Application of BP

- **Hand-written digital number recognition**



The overflow of BP application

设计ANN时需要考虑的问题

- 收敛性和局部极小值
- 隐藏层的表示
- 拓扑结构的确定
- 泛化、过拟合和迭代停止

收敛性和局部极小值

- **网络的权越多，误差曲面的维数越高，也就越可能为梯度下降提供更多的局部极小点**
- **迭代过程中网络权值的演化：**
  1. 如果把网络的权值初始化为接近于0的值，那么在早期的梯度下降步骤中，网络将表现为一个非常平滑的函数，近似为输入的线性函数，这是因为Logistic函数本身在权值靠近0时接近线性
  2. 仅当权值增长一定时间后，它们才会到达可以表示较高非线性网络函数的程度；
  3. 在这个能表示更复杂函数的权空间区域存在更多的局部极小值，但是人们认为当权到达这一点时，它们已经靠近全局最小值

收敛性和局部极小值

- **用来缓解局部极小值问题的启发式规则**
  - **为梯度更新法则加一个冲量**，可以带动梯度下降过程，冲过狭窄的局部极小值
  - **使用随机的梯度下降**：对于每个训练样例沿一个不同的误差曲面有效下降，这些不同的误差曲面通常有不同的局部极小值，这使得下降过程不太可能陷入一个局部极小值
  - **使用同样的数据训练多个网络，但用不同的随机权值初始化每个网络**。如果不同的训练产生不同的局部极小值，那么对分离的验证集合性能最好的那个网络将被选中，或者保留所有的网络，输出是所有网络输出的平均值

隐藏层的表示

1. 反向传播算法的特性是：它能够在网络内部的隐藏层发现有用的中间表示
2. 多层网络在隐藏层自动发现**中间层特征**的能力是ANN学习的一个关键特性
3. 网络中使用的单元层越多，就可以创造出越复杂的中间层特征

拓扑结构的确定

- **隐层数**

1. 增加隐层数可以降低网络误差，也使网络复杂化，从而增加了网络的训练时间和出现“过拟合”的倾向。
2. 在设计BP网络时可参考这一点，应优先考虑3层BP网络（即有1个隐层）
3. 一般地，靠增加**隐层节点数**来获得较低的误差，其训练效果要比增加**隐层数**更容易实现。

拓扑结构的确定

- **隐层节点数**

1. 是训练时出现“过拟合”的直接原因，但是目前理论上还没有一种科学的和普遍的确定方法。
2. 为尽可能避免训练时出现“过拟合”现象，保证足够高的网络性能和泛化能力，确定隐层节点数的最基本原则是：在满足精度要求的前提下取尽可能紧凑的结构，即取尽可能少的隐层节点数。

拓扑结构的确定

- **隐层节点数**

1. 隐层节点数须小于*N*-1（其中*N*为训练样本数），否则，网络模型的误差与训练样本的特性无关而趋于零，即建立的网络模型没有泛化能力，也没有任何实用价值。同理可推得：输入层的节点数（变量数）必须小于*N*-1。
2. 训练样本数必须多于网络模型的连接权数，一般为2~10倍。

泛化、过拟合和迭代停止

- **权值更新算法的终止条件**
  - 一种策略是，对训练样例的误差降低至某个预先定义的阈值之下
- **泛化精度：网络拟合训练数据外的实例的精度**

过拟合

- **过拟合发生在迭代的后期**

1. 设想网络的权值是被初始化为小随机值的，使用这些几乎一样的权值仅能描述非常平滑的决策面
2. 随着训练的进行，一些权值开始增长，以降低在训练数据上的误差，同时学习到的决策面的复杂度也在增加
3. 如果权值调整迭代次数足够多，反向传播算法可能会产生过度复杂的决策面，拟合了训练数据中的噪声和训练样例中没有代表性的特征

过拟合解决方法

- **权值衰减**
  - 它在每次迭代过程中以某个小因子降低每个权值，这等效于修改E的定义，加入一个与网络权值的总量相应的惩罚项，此方法的动机是保持权值较小，从而使学习过程向着复杂决策面的反方向偏置
- **验证数据**
  - 最成功的方法是在训练数据外再为算法提供一套验证数据，应该使用在验证集合上产生最小误差的迭代次数，不是总能明显地确定验证集合何时达到最小误差

过拟合解决方法

- **k-fold交叉验证方法**

1. 把训练样例分成k份，然后进行k次交叉验证过程，每次使用不同的一份作为验证集合，其余k-1份合并作为训练集合。
2. 每个样例会在一次实验中被用作验证样例，在k-1次实验中被用作训练样例
3. 每次实验中，使用上面讨论的交叉验证过程来决定在验证集合上取得最佳性能的迭代次数，然后计算这些迭代次数的均值
4. 最后，运行一次反向传播算法，训练所有m个实例并迭代

The principle and application of CNN

概述： DNN的背景

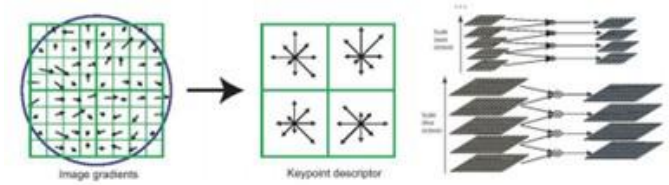
传统的机器学习去解决这些问题的思路如下：

概括起来主要由三部分组成

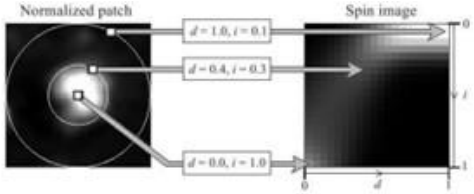
而中间的特征提取部分将很大程度上决定最终的效果，那实际中的特征提取是怎么做的？

概述： DNN的背景

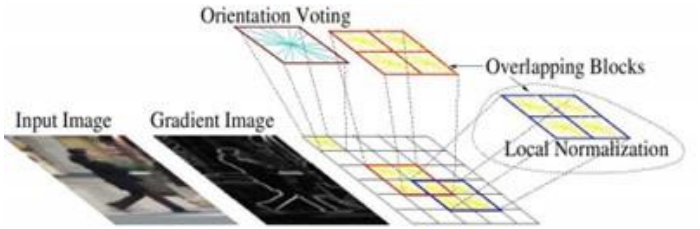
实际中特征提取都是靠手工提取的，截止现在，也出现了不少好特征：



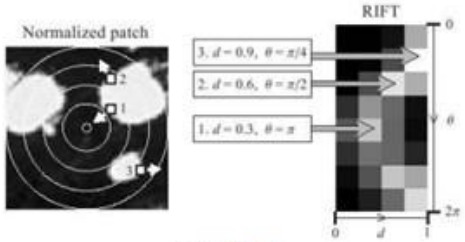
SIFT



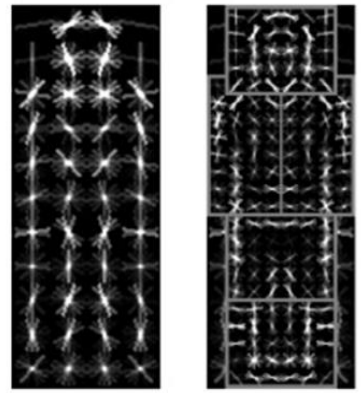
Spin image

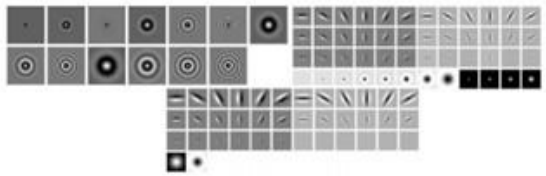


HoG

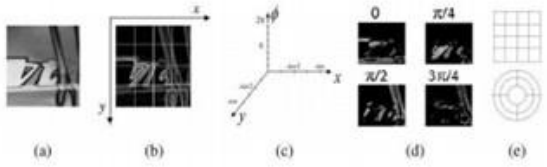


RIFT





Textons



GLOH

手工地选取特征是一件非常费力、启发式（需要专业知识）的方法，能不能选取好很大程度上靠经验和运气！！而且它的调节需要大量的时间。

概述： DNN的背景

这个领域该认识的人

Geoffrey E. Hinton

Andrew N.G

Lecun Yann

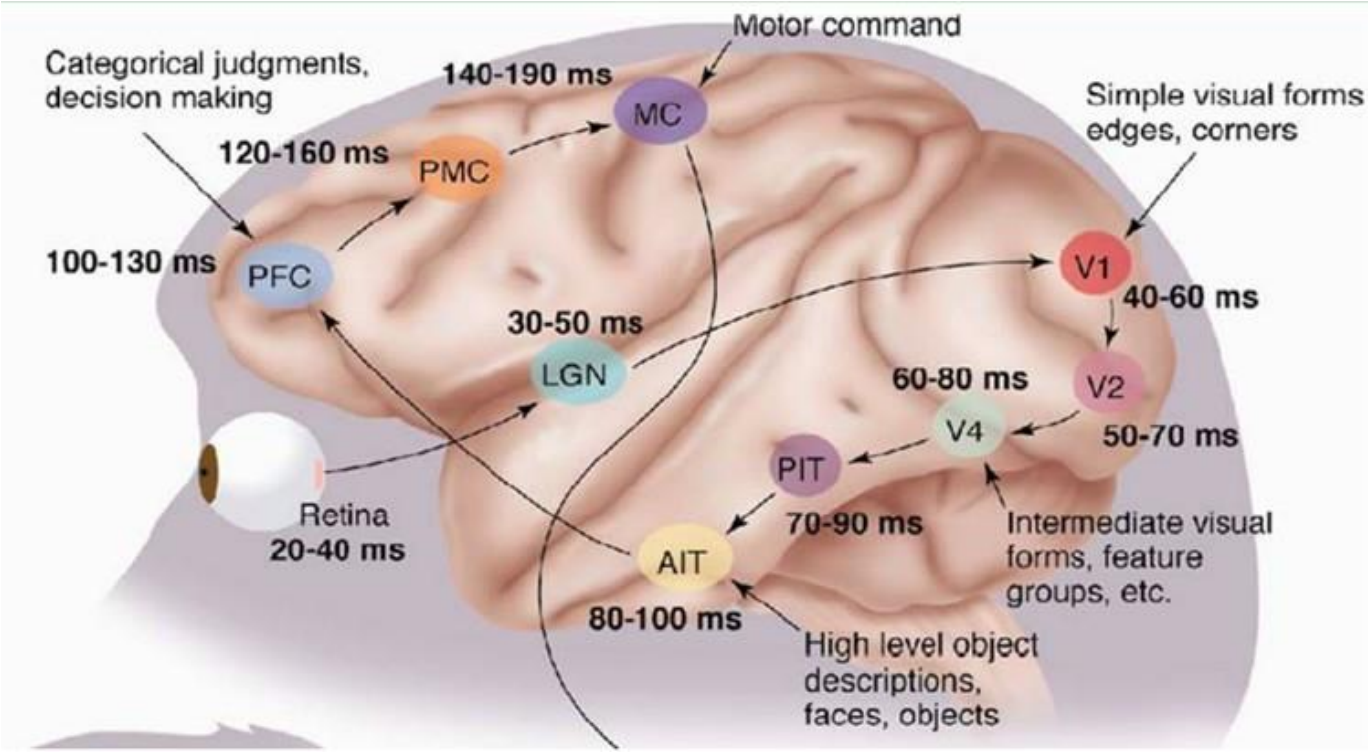
概述： 从视觉认知到DNN

- 大家都注意到关键词了：**分层**。
- 而Deep learning的deep是不是就表示存在多少层，也就是多深呢？
- Deep learning是如何借鉴这个过程的呢？
- 毕竟是归于计算机来处理，面对的一个问题就是怎么对这个过程建模及其程序实现

概述： DNN的背景

- 既然手工选取特征不太好，那么能不能自动地学习一些特征呢？
- Deep Learning的一个重要功能就是**自动学习中层特征**，它的一别名Representative learning。
- 那它是怎么学习的呢？怎么知道哪些特征好哪些不好呢？
  1. **机器学习是一门专门研究计算机怎样模拟或实现人类的学习行为的学科。**
  2. 启发：人的视觉系统是怎么工作的呢？

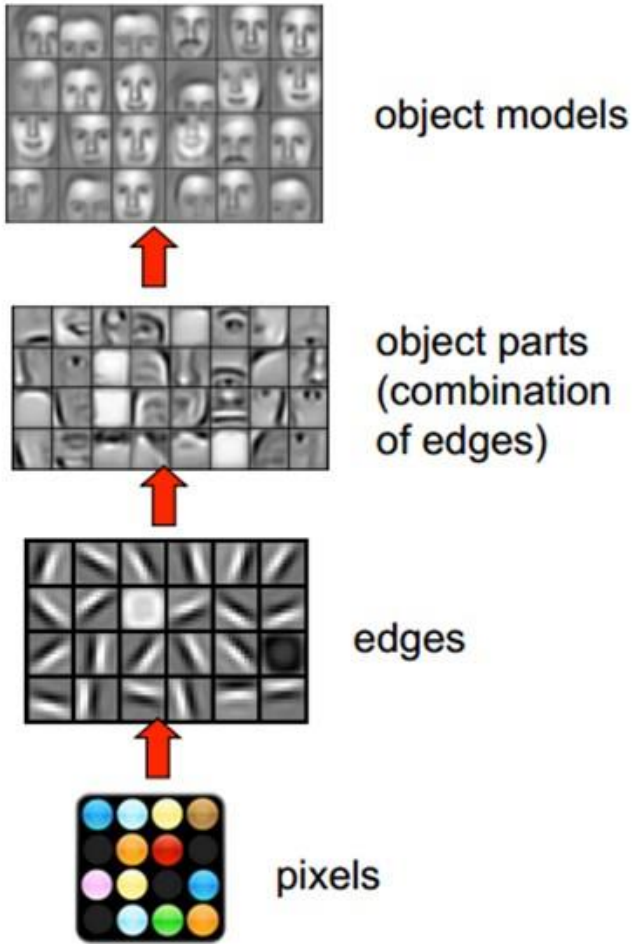
概述：从视觉认知到DNN

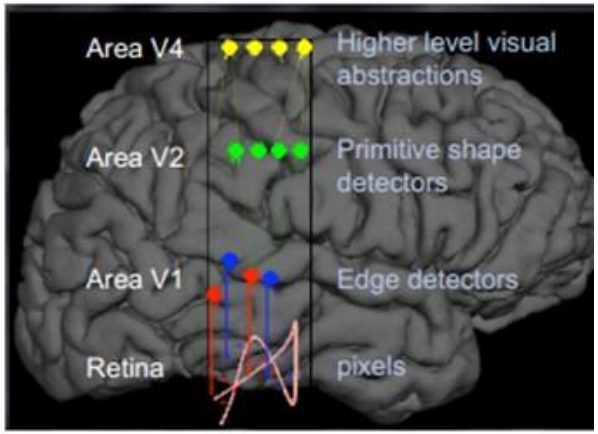


1981 年的诺贝尔医学奖，颁发给了 David Hubel 和TorstenWiesel，以及 Roger Sperry。贡献是“发现视觉系统的过程”：可视皮层是分级的：

这个发现激发了人们对于神经系统的进一步思考。神经-中枢-大脑的工作过程，或许是一个**不断迭代、不断抽象**的过程。

概述：从视觉认知到DNN





这个过程关键在于抽象和迭代。从原始信号开始，做低级抽象，逐渐向高级**抽象迭代**。

如图，从原始信号摄入开始（瞳孔摄入像素Pixels），接着做初步处理（大脑皮层某些细胞发现边缘和方向），然后抽象（大脑判定，眼前的物体的形状），然后进一步抽象（大脑进一步判定该物体是谁的脸）

概述： DNN的基本思想

- 假设有一个系统S，它有n层（$S_1, \cdots S_n$），它的输入是I，输出是O，形象地表示为： $I \Rightarrow S_1 \Rightarrow S_2 \Rightarrow \cdots\cdots \Rightarrow S_n \Rightarrow O$，如果输出O等于输入I，即输入I经过这个系统变化之后没有任何的信息损失，保持了不变，这意味着输入I经过每一层S_i都没有任何的信息损失，即在任何一层S_i，它都是原有信息（即输入I）的另外一种表示。

$i = o$

概述： DNN的基本思想

- 对于深度学习来说，其思想就是堆叠多个层，也就是说这一层的输出作为下一层的输入。通过这种方式，就可以实现对输入信息进行分级表达了。

- 问题：CNN是如何学习特征和训练的？

The layer structures of CNN

- 卷积神经网络的层级结构

① 数据输入层:Input layer

② 卷积计算层:CONV layer

③ ReLU激励层:ReLU layer

④ 池化层:Pooling layer

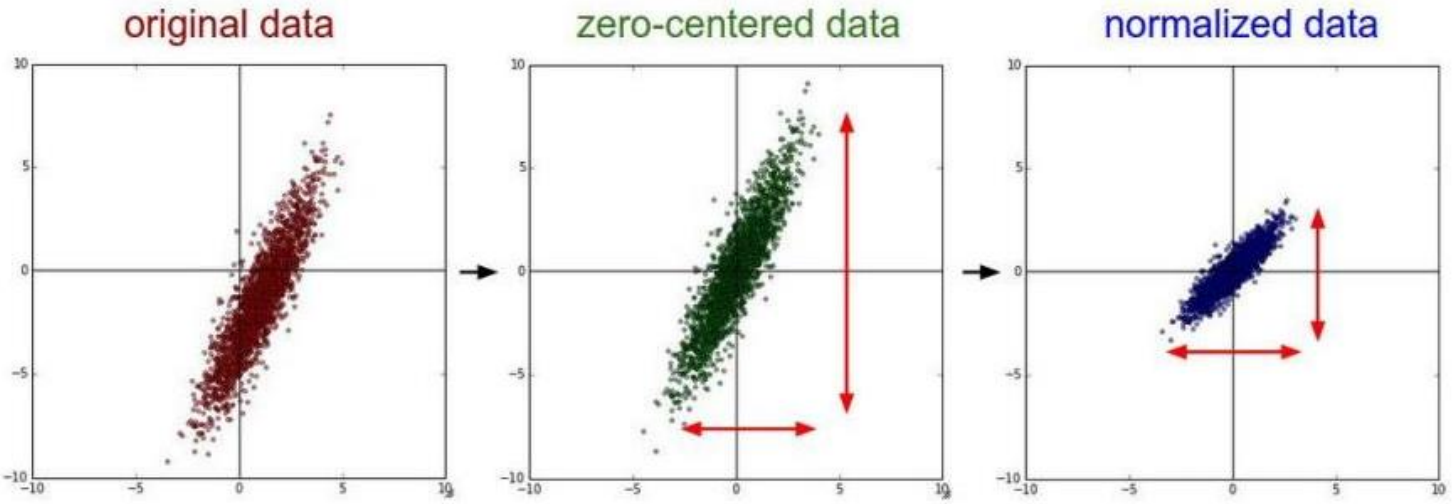
⑤ 全连接层:FC layer

CNN依旧是层级网络，只是层的功能和形式做了变化，是传统神经网络的一个改进

1、数据输入层

该层要做的处理主要是对原始图像数据进行**预处理**，其中包括：

1. 去均值：把输入数据各个维度都中心化为0，目的就是把样本的中心拉回到坐标系原点上。

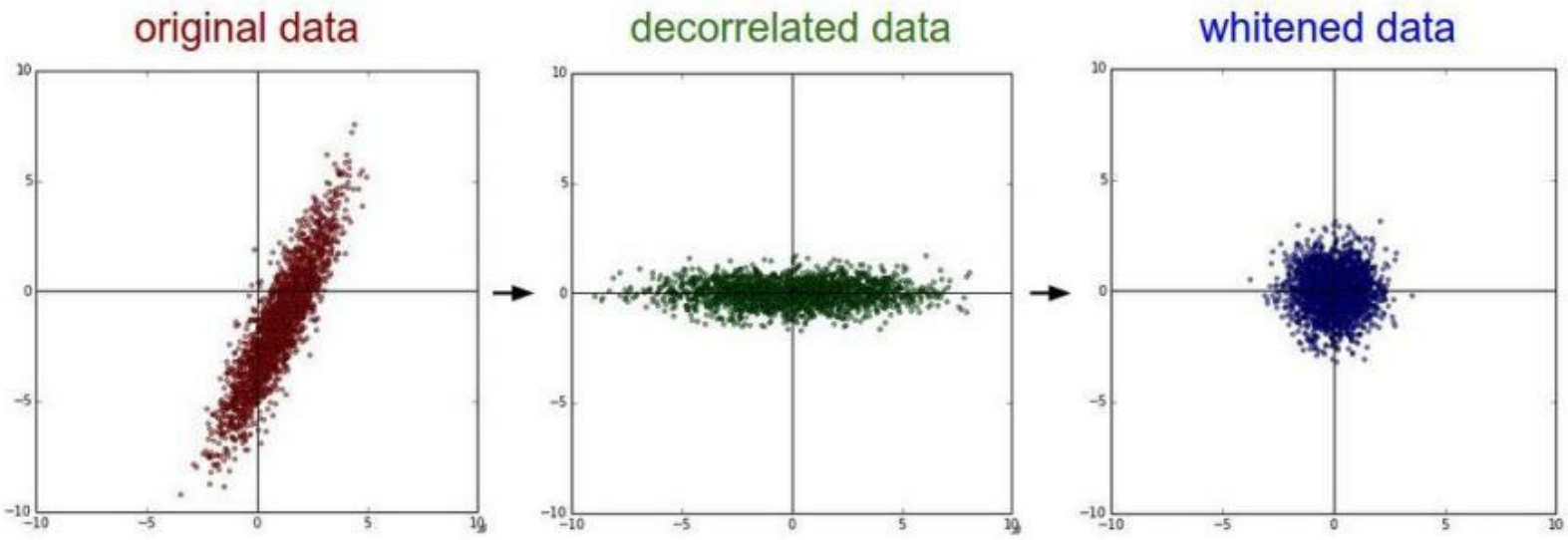


去均值与归一化效果图：

2. 归一化：幅度归一化到同样的范围，即减少各维度数据取值范围的差异而带来的干扰，比如，我们有两个维度的特征A和B，A范围是0到10，而B范围是0到10000，如果直接使用这两个特征是有问题的，好的做法就是归一化，即A和B的数据都变为0到1的范围。

1、数据输入层

3. PCA/白化：用PCA降维；白化是对数据各个特征轴上的幅度归一化

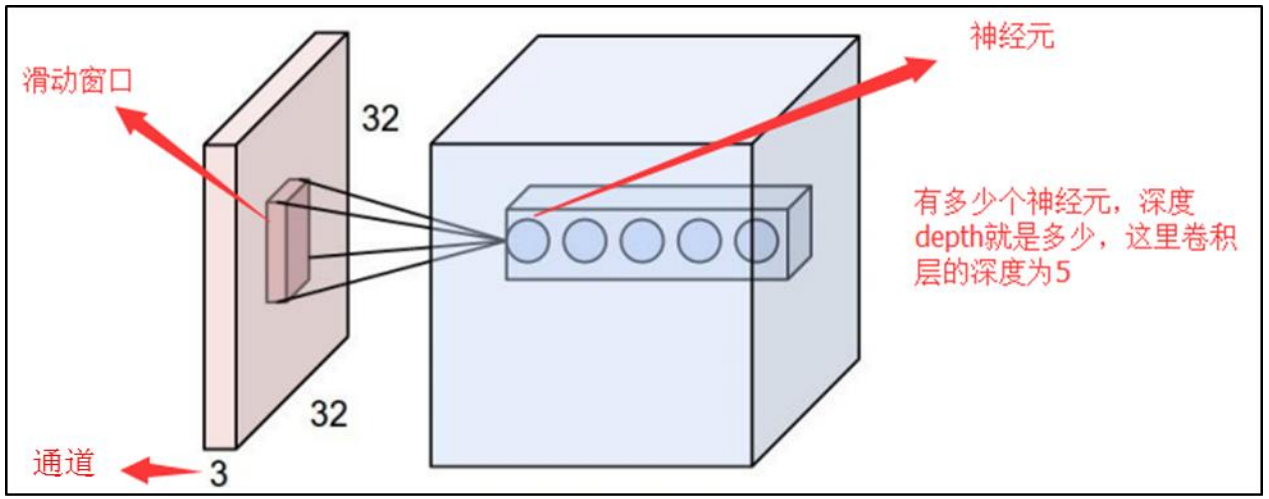


去相关与白化效果图

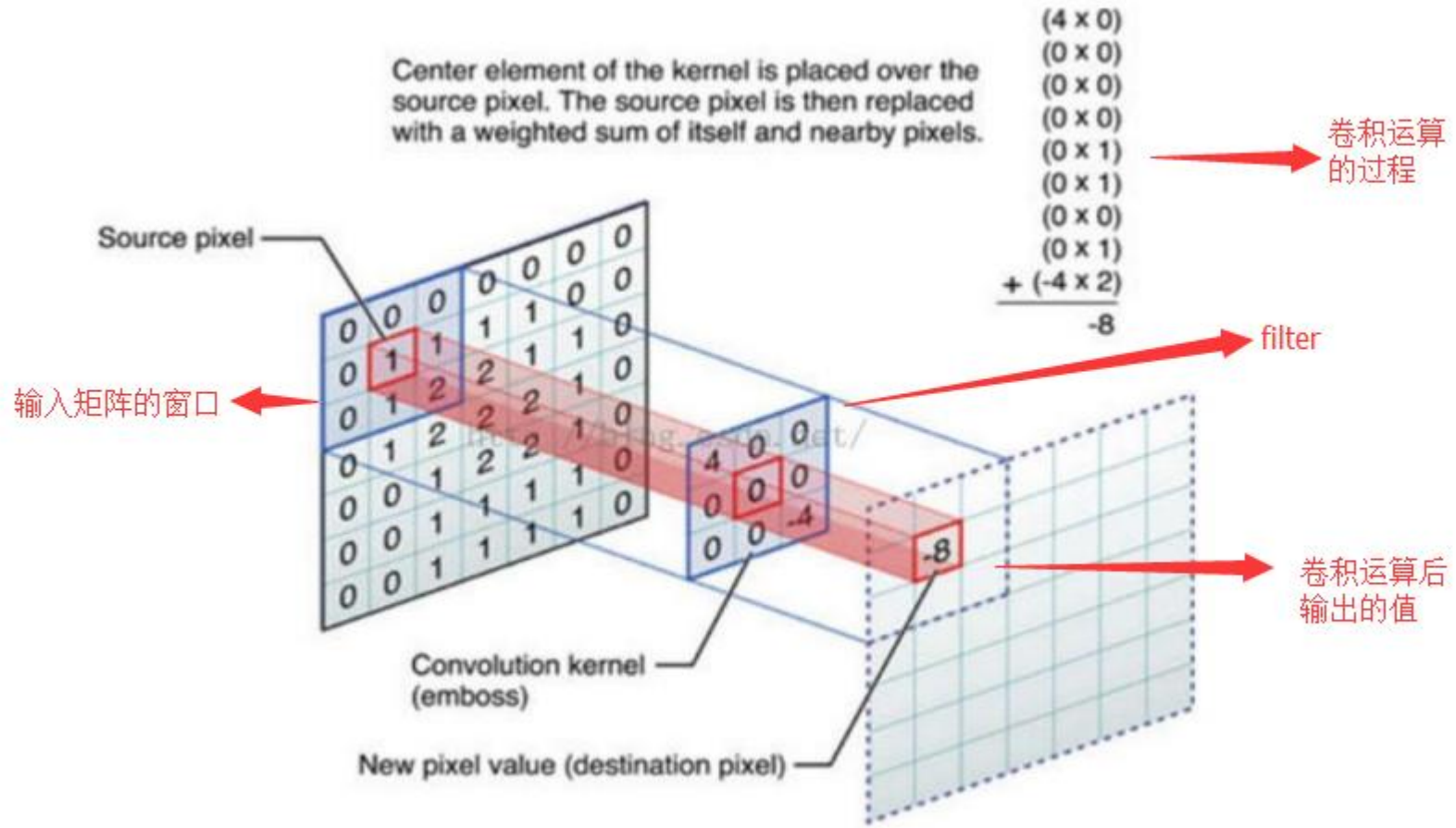
2、卷积计算层 Convolutional layer(CL)

CL就是CNN最重要的层次，也是“卷积神经网络”的名字来源

- **两个关键操作：**
  - 局部关联。每个神经元看做一个滤波器(filter)
  - 滑窗(receptive field)。filter对局部数据计算
- **三个名词：**
  - 深度/depth
  - 步长/stride （窗口一次滑动的长度）
  - 填充值/zero-padding



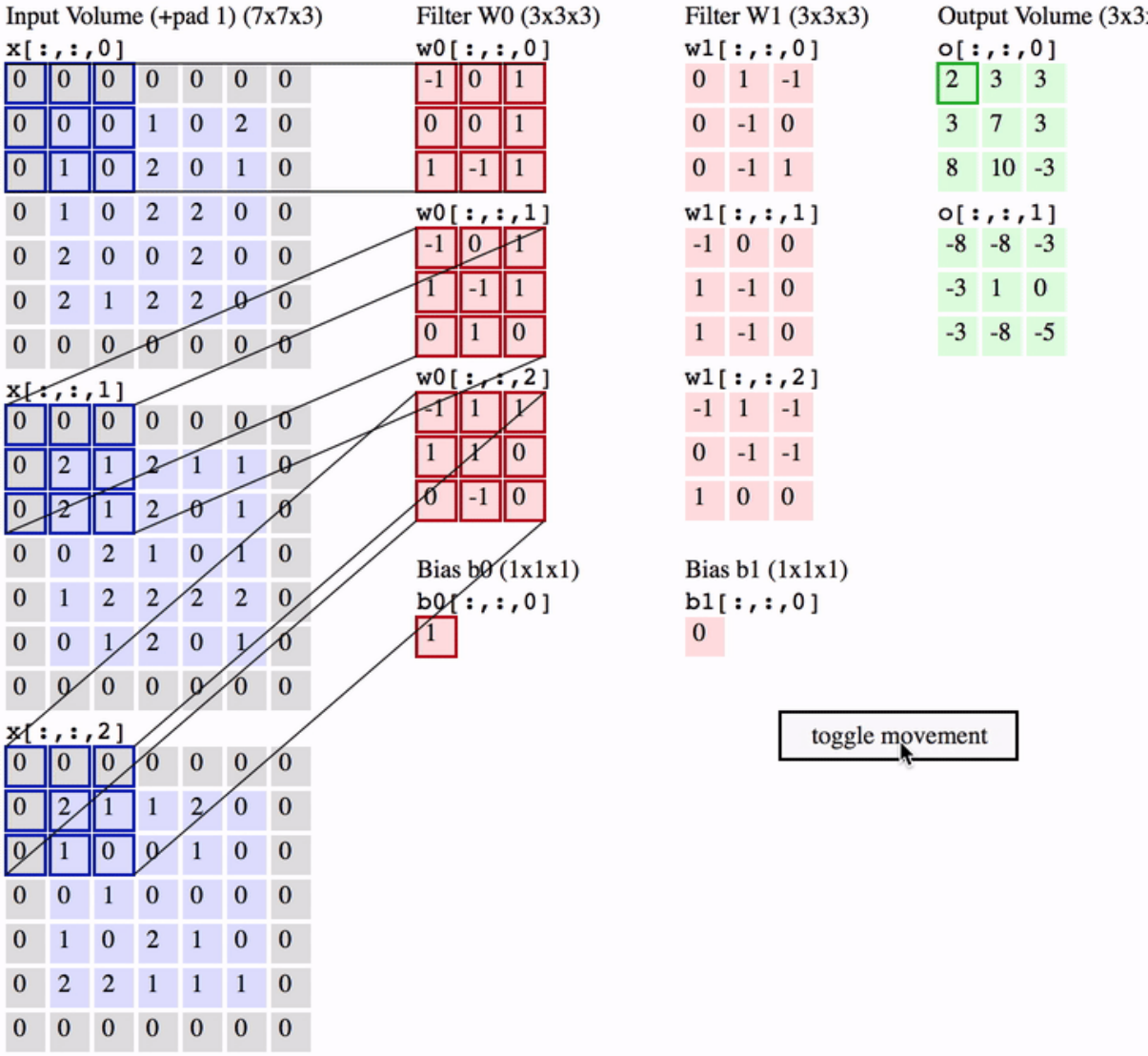
2、Convolution Layer



- The process is a 2D convolution on the inputs.
- The “dot products” between weights and inputs are “integrated” across “channels”.

3D卷积过程

- Filter weights are shared across receptive fields.
- The filter has same number of layers as input volume channels, and output volume has same “depth” as the number of filters.



- 蓝色的矩阵(输入图像)对粉色的矩阵（filter）进行矩阵内积计算并加上偏置值b，如 输出层中的第一个数 ‘2’ 是怎么得到的？

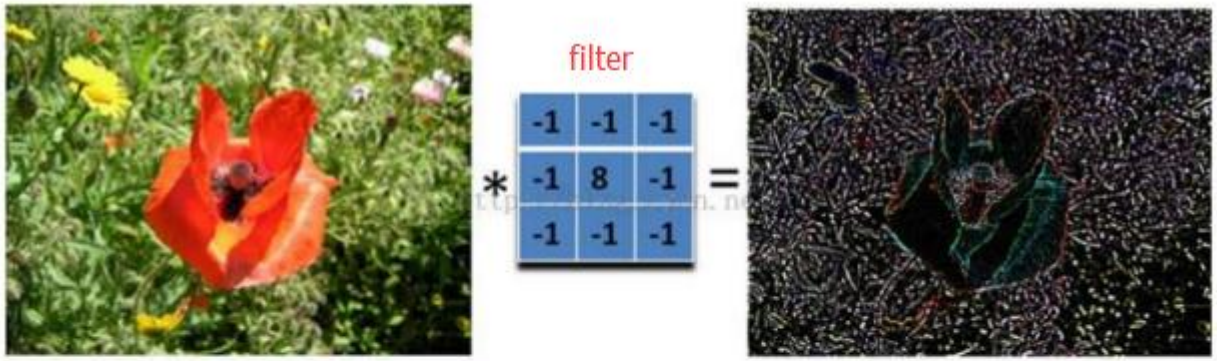
$$-1+(-2+1+2)+(2-1)+1=-1+1+1+1=2$$

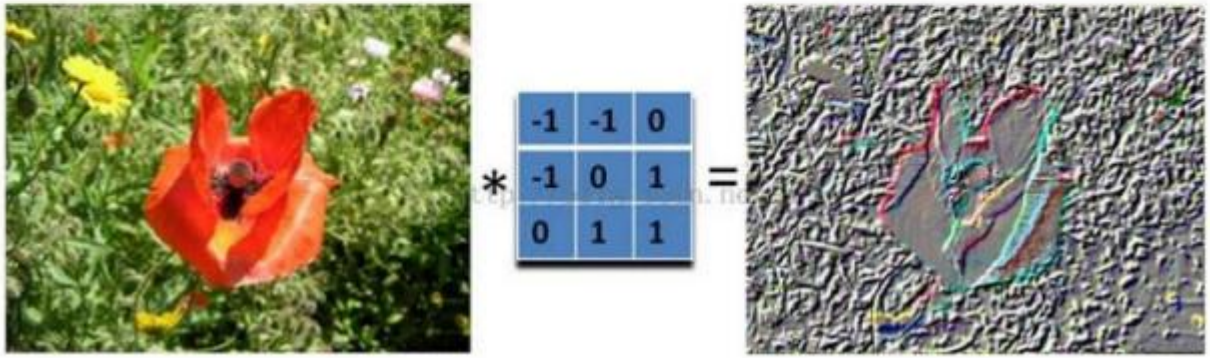
参数共享机制

a) 在卷积层中每个神经元连接数据窗的权重是固定的，每个神经元只关注一个特性。神经元就是图像处理中的滤波器，比如边缘检测专用的Sobel滤波器，即卷积层的每个滤波器都会有自己所关注一个图像特征，比如垂直边缘，水平边缘，颜色，纹理等等，这些所有神经元加起来就好比就是整张图像的特征提取器集合。

b) 需要估算的权重个数减少：AlexNet 1亿 => 3.5w

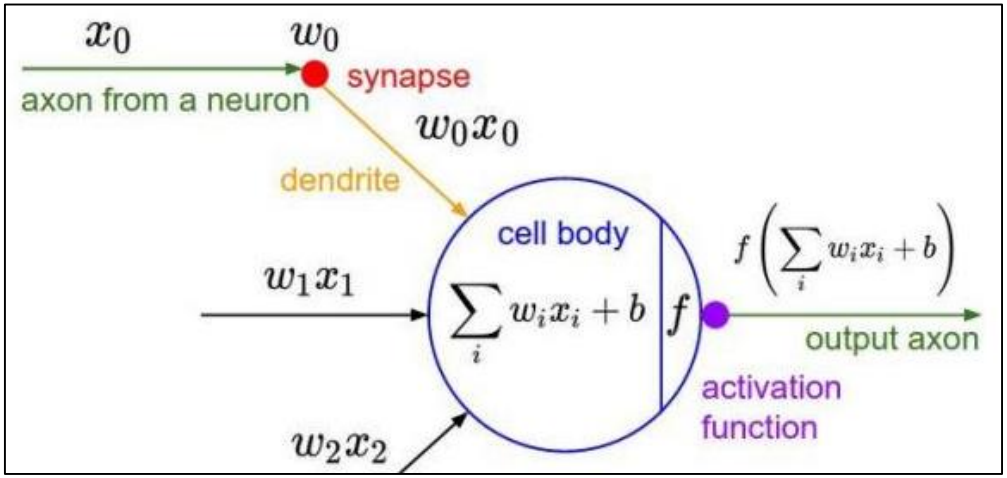
c) 一组固定的权重和不同窗口内数据做内积：卷积

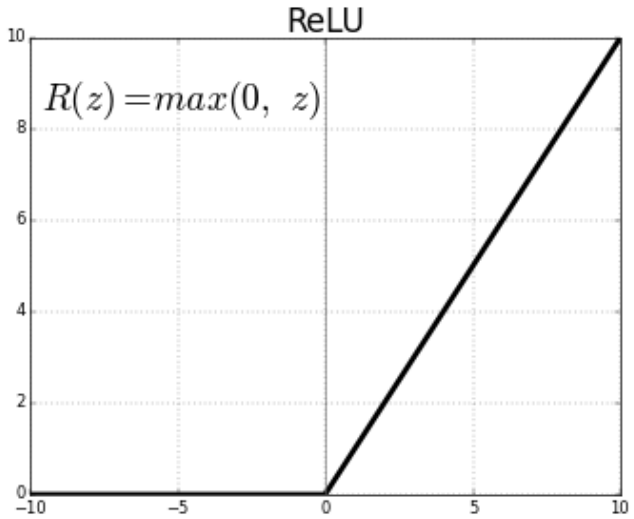




3、Activation Layer

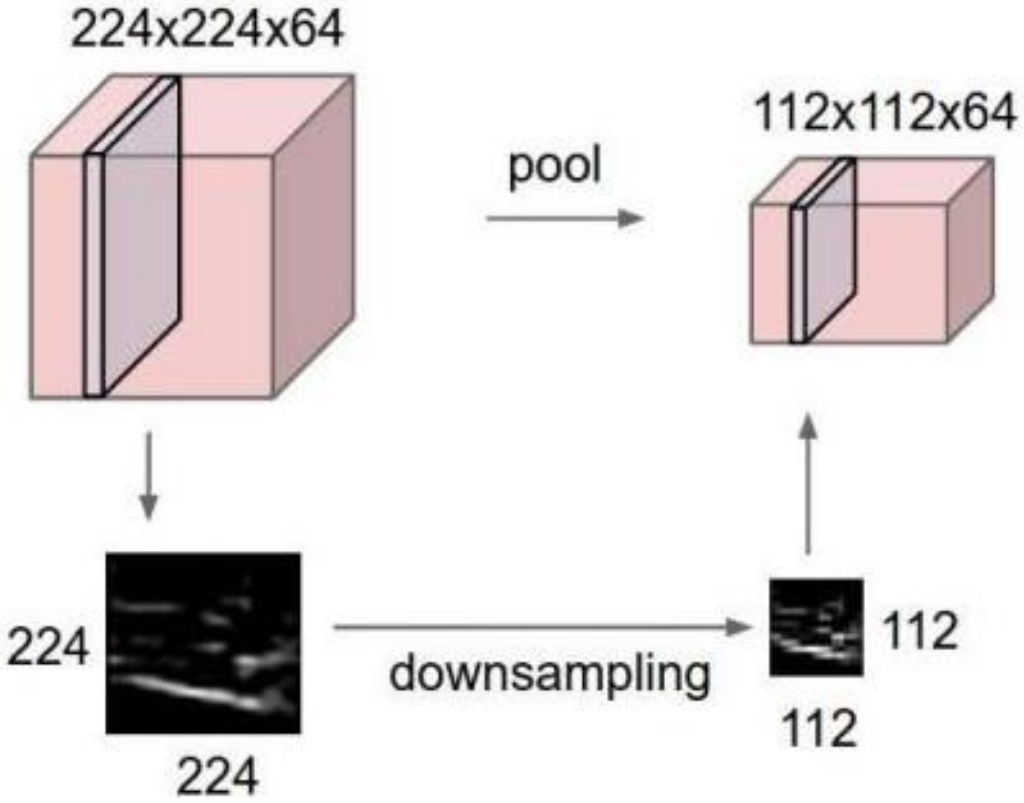
- Used to increase non-linearity of the network without affecting receptive fields of convolution layers
- 在深度学习中，CNN一般采用的激励函数为ReLU(The Rectified Linear Unit/修正线性单元)，它的特点是收敛快，求梯度简单，但较脆弱
- **Other types:** Leaky ReLU, Randomized Leaky ReLU, Parameterized ReLU　Exponential Linear Units (ELU), Scaled Exponential Linear Units Tanh, hardtanh, softtanh, softsign





4、Pooling Layer 池化层

- 池化层夹在连续的卷积层中间，用于压缩数据和参数的量，减小过拟合；简而言之，如果输入是图像的话，那么池化层的最主要作用就是压缩图像。



4、Pooling Layer 池化层

叙述池化层的**具体作用**:

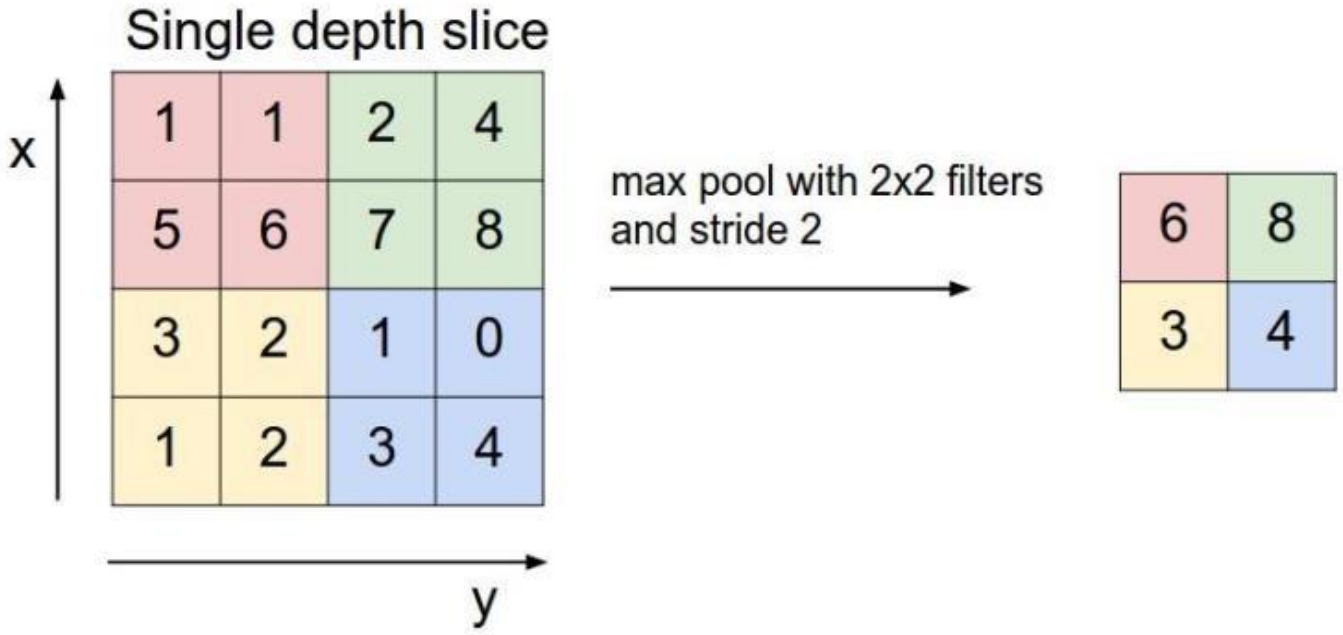
1. 特征不变性。也就是在Image Processing中经常提到的特征尺度不变性，池化操作就是图像的resize，图像压缩时去掉的信息只是一些无关紧要的信息，而留下的信息则是具有尺度不变性的特征，是最能表达图像的特征。

2. 特征降维。我们知道一幅图像含有的信息是很大的，特征也很多，但是有些信息对于我们做图像任务时没有太多用途或者有重复，我们可以把这类冗余信息去除，把最重要的特征抽取出来，这也是池化操作的一大作用。

3. 在一定程度上防止过拟合，更方便优化。

4、Pooling Layer 池化层

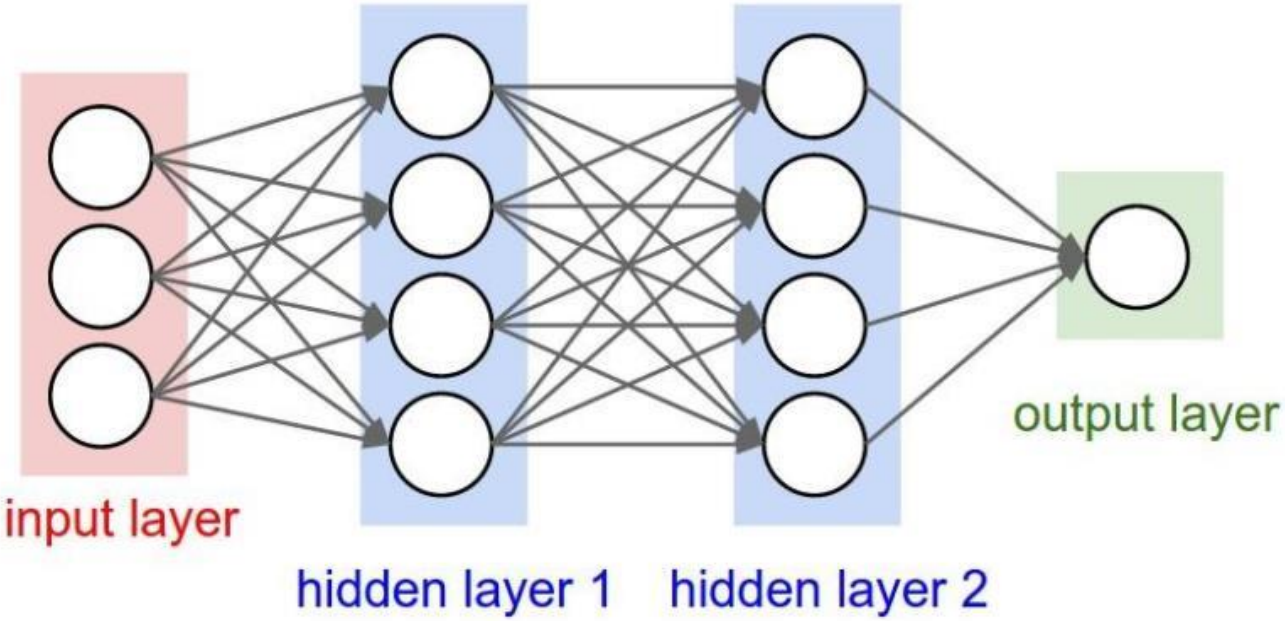
- 池化层用的方法有**Max pooling 和 average pooling**，而实际用的较多的是Max pooling



Max pooling

5、Fully Connect Layer全连接层

两层之间所有神经元都有权重连接，通常全连接层在卷积神经网络尾部。
也就是跟传统的神经网络连接方式是一样的，所以叫全链接层。



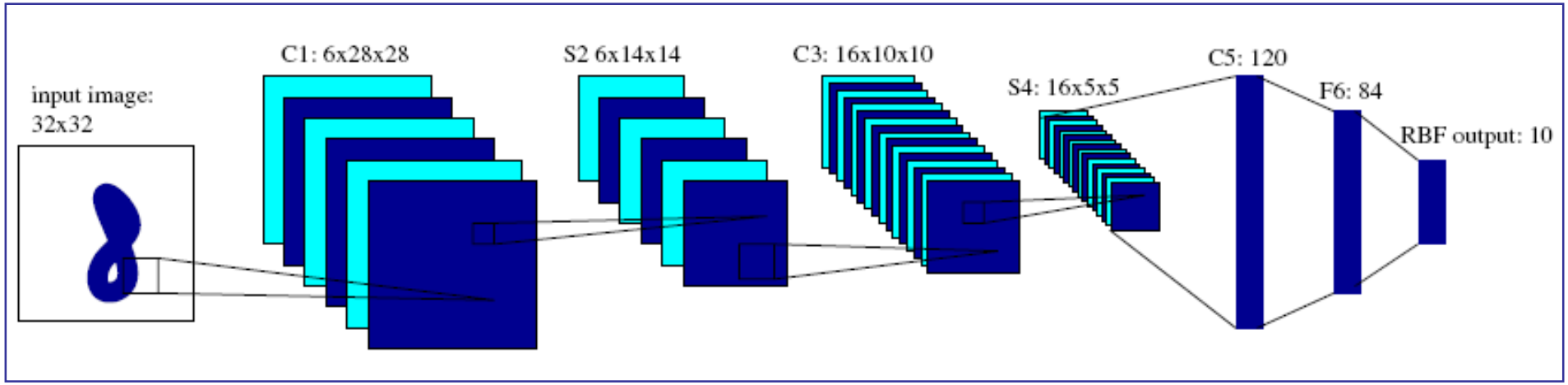
LeNet-5：Typical CNN Structure

① raw image of 32 × 32 pixels as input.

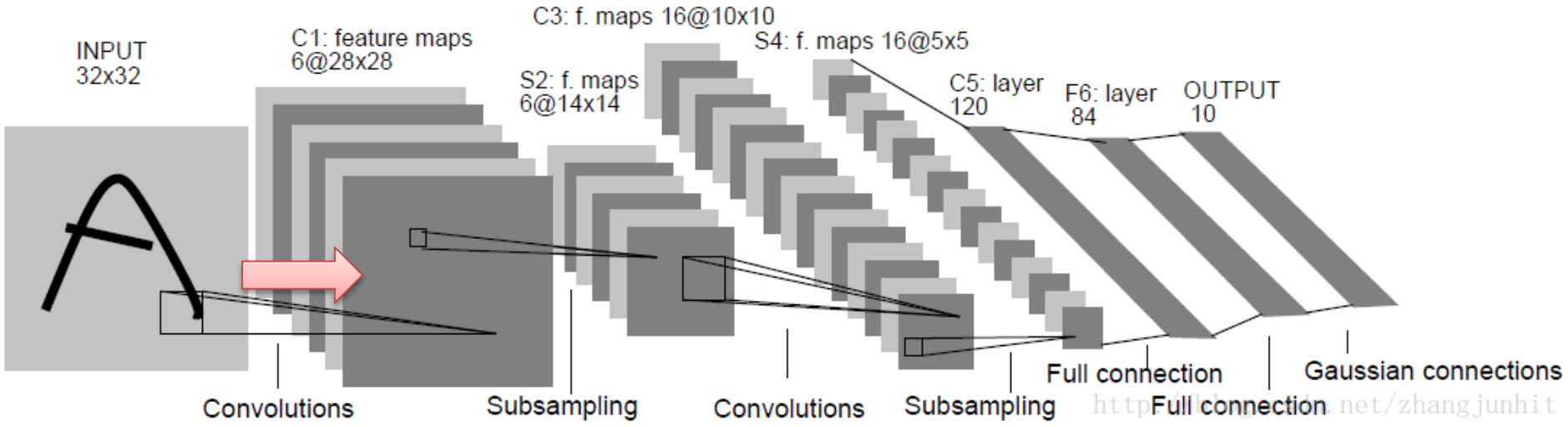
② C1,C3,C5 : Convolutional layer.（5 × 5 卷积核）.

③ S2 , S4 : Subsampling layer.（Mean-Pooling：2）.

④ F6 : Fully connected layer.



LeNet-5中的参数变化



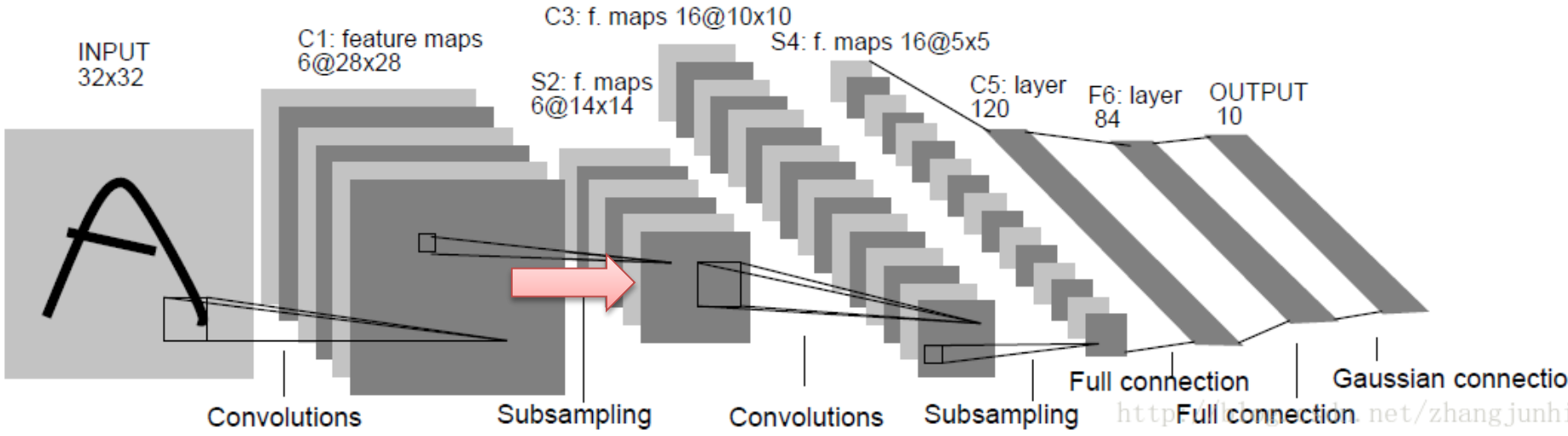
卷积C1：使用 6 个5*5 的Kernel（32-5+1=28)

参数： 156个。6*（5*5+1）个参数，其中+1是表示一个核有一个bias。

连接：122304个。156*28*28=122304个连接（connection），对于卷积层C1每个像素都与输入图像中的5*5个像素和1个bias有连接。

有122304个连接，但只需要学习156个参数，主要是通过**权值共享**实现的。

LeNet-5中的参数变化

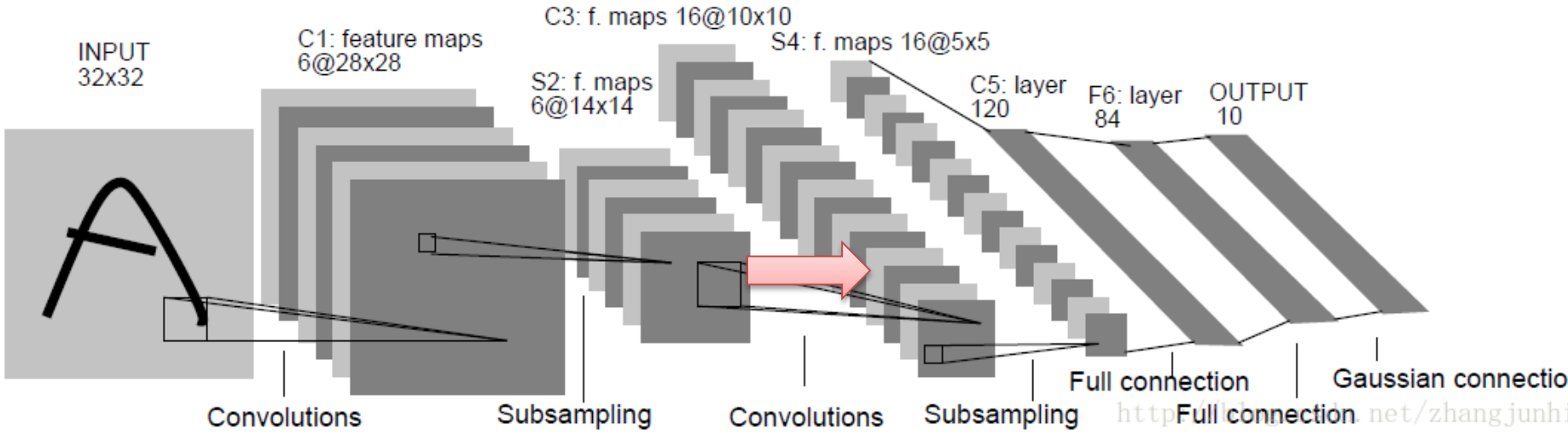


pooling层是对C1中的2*2区域内的像素求和，乘以一个权值系数再加上一个偏置

参数：12个。每个池化核有两个训练参数，所以共有2x6=12个训练参数。

连接：5880个。 5x14x14x6 =5880个连接（connection），对于池化层S2每个像素都与输入图像中的2*2个像素和1个bias有连接。

LeNet-5中的参数变化



卷积：第二次卷积的输出是C3，16个10x10的特征图，kernel size是 5*5.

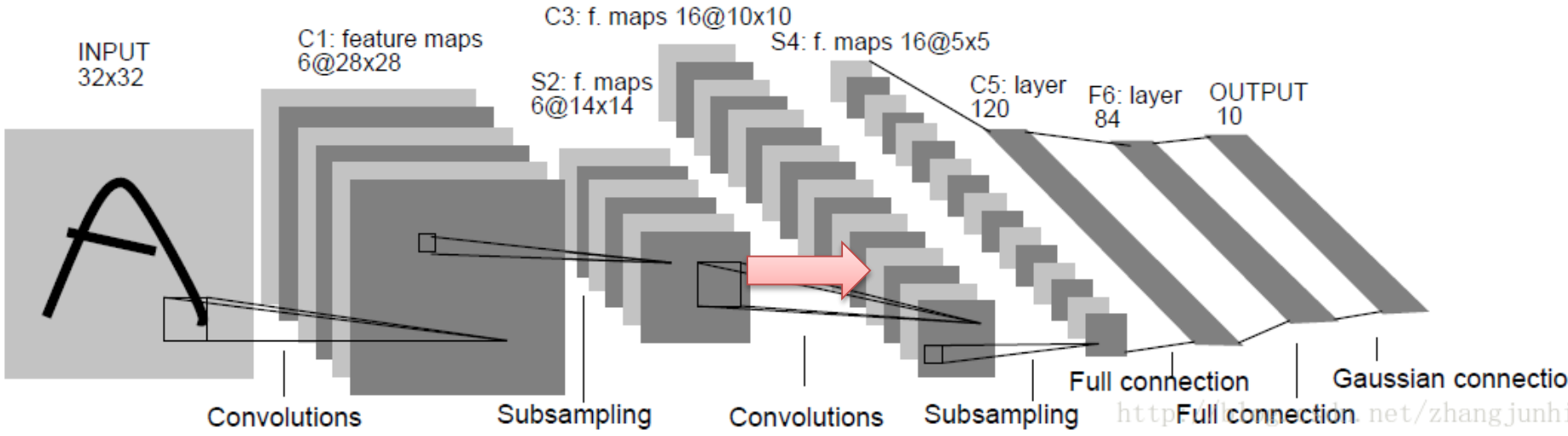
怎么从6 个feature map到 16个呢？

参数：156个。6*（5*5+1）个参数，其中+1是表示一个核有一个bias。

连接：122304个。156*28*28=122304个连接（connection），对于卷积层C1每个像素都与输入图像中的5*5个像素和1个bias有连接。

有122304个连接，但只需要学习156个参数，主要是通过权值共享实现的。

LeNet-5中的参数变化

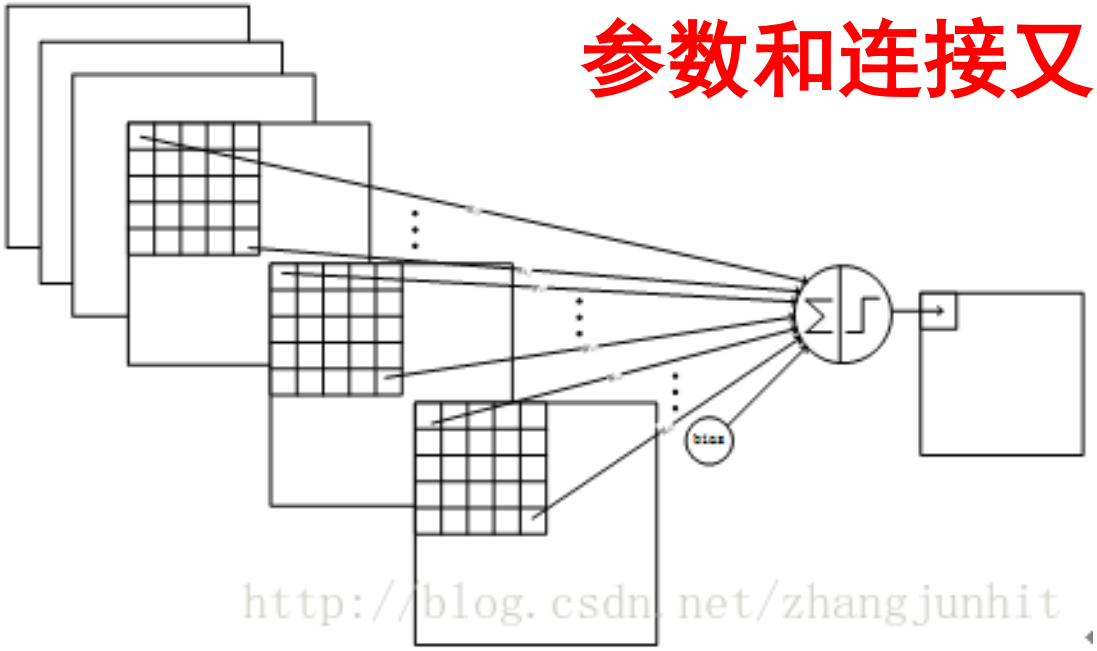


卷积：第二次卷积的输出是C3，16个10x10的特征图，kernel size是 5*5.

怎么从6 个feature map到 16个呢？

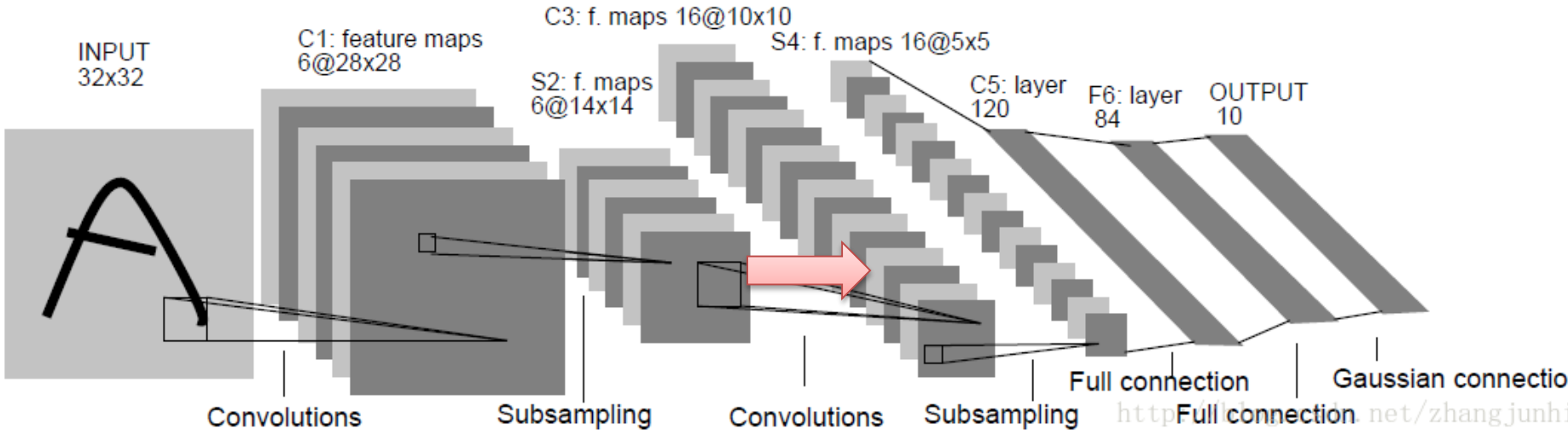
feature map从6 个到 16个

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	X				X	X	X			X	X	X	X		X	X
1	X	X				X	X	X			X	X	X	X		X
2	X	X	X				X	X	X			X		X	X	X
3		X	X	X			X	X	X	X			X		X	X
4			X	X	X			X	X	X	X		X	X		X
5				X	X	X			X	X	X	X		X	X	X



参数和连接又是多少呢？

LeNet-5中的参数变化

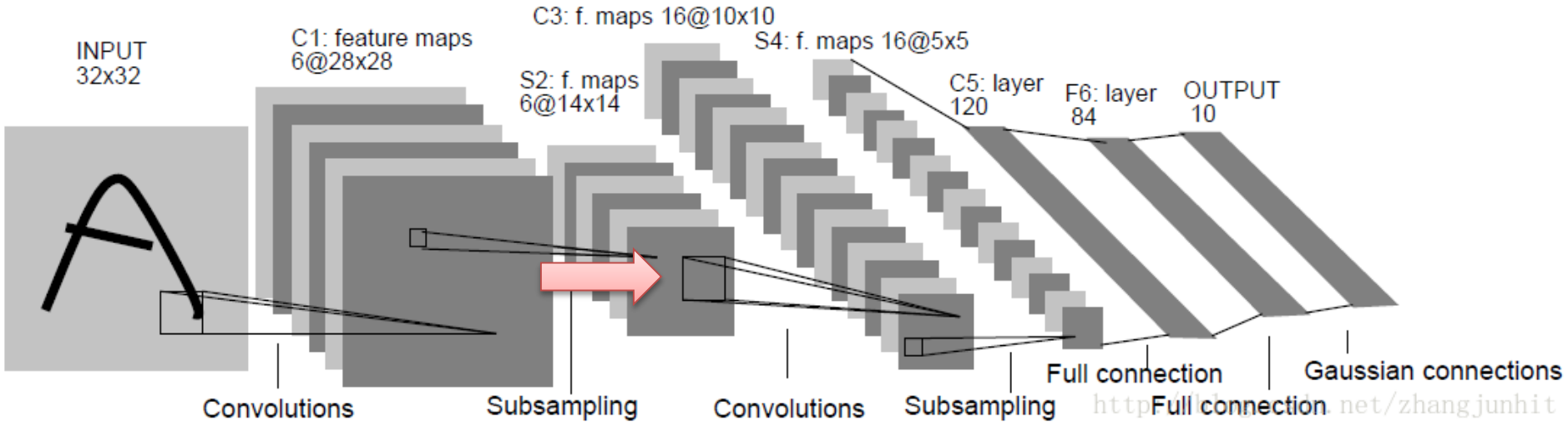


	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	X				X	X	X			X	X	X	X		X	X
1	X	X				X	X	X			X	X	X	X		X
2	X	X	X				X	X	X			X		X	X	X
3		X	X	X			X	X	X	X			X		X	X
4			X	X	X			X	X	X	X		X	X		X
5				X	X	X			X	X	X	X		X	X	X

参数：卷积核大小依然为5*5，所以总共有6*（3*5*5+1）+6*（4*5*5+1）+3*（4*5*5+1）+1*（6*5*5+1）=1516个参数。

连接：而图像大小为10*10，所以共有151600个连接。

LeNet-5中的参数变化

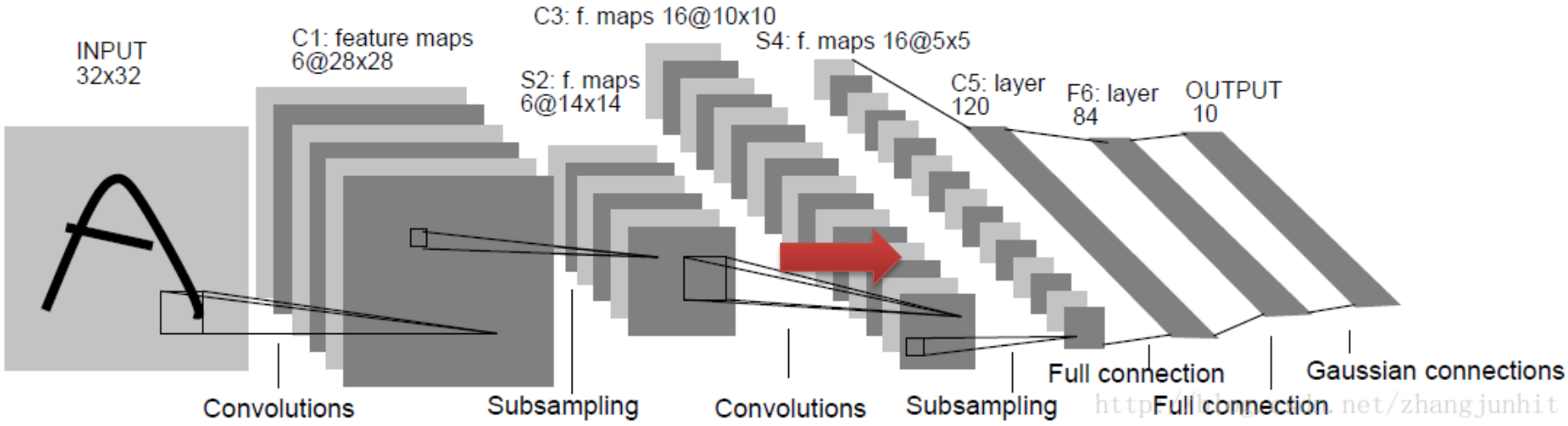


pooling层S4是对C3中的2*2区域内的像素求和，乘以一个权值系数再加上一个偏置

参数：32个。每个池化核有两个训练参数，所以共有2x16=32个训练参数。

连接：2000个。 5x16x5x5 =2000个连接（connection），对于池化层S4每个像素都与输入图像中的2*2个像素和1个bias有连接。

LeNet-5中的参数变化



卷积：第三次卷积的输出是C5，120个1x1的特征图，kernel size是 5*5.

怎么从16 个feature map到 120呢？

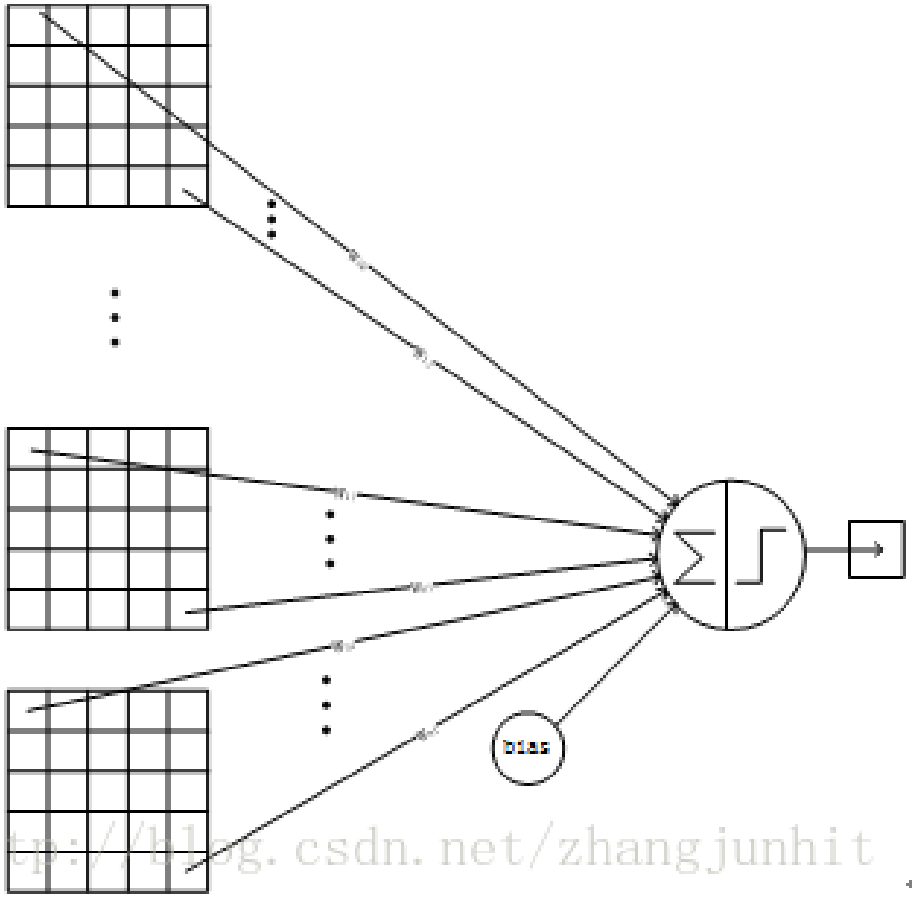
参数：156个。6*（5*5+1）个参数，其中+1是表示一个核有一个bias。

连接：122304个。156*28*28=122304个连接（connection），对于卷积层C1每个像素都与输入图像中的5*5个像素和1个bias有连接。

有122304个连接，但只需要学习156个参数，主要是通过权值共享实现的。

LeNet-5中的参数变化

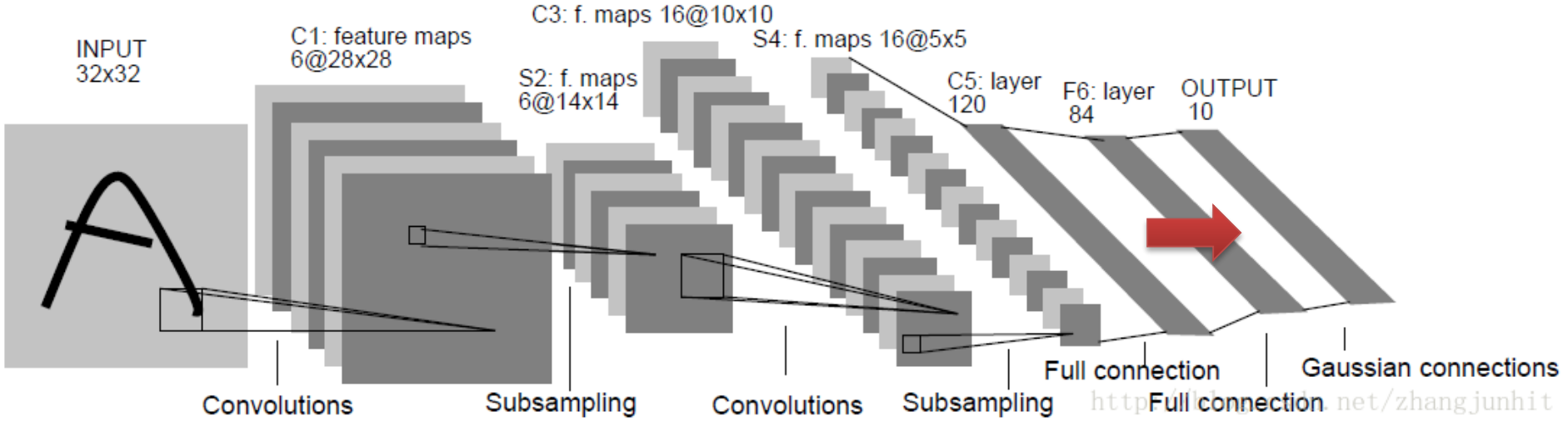
卷积：由于S4层的16个图的大小为5x5，与卷积核的大小相同，所以卷积后形成的图的大小为1x1。这里形成120个卷积结果。



参数：每个都与上一层的16个图相连。所以共有(5x5x16+1)x120 = 48120个参数

连接：同样有48120个连接

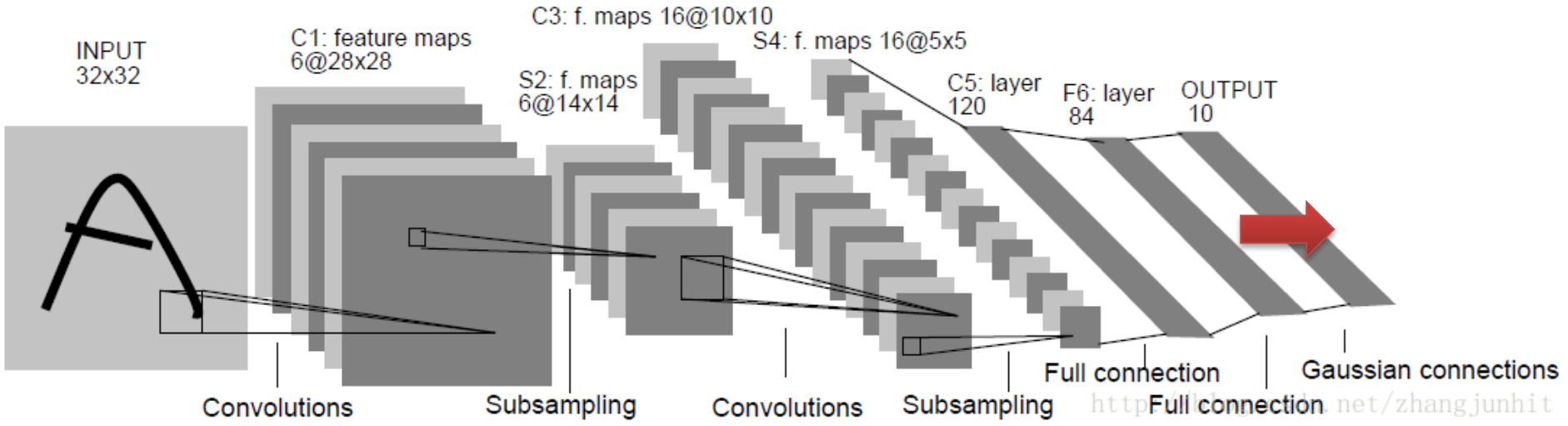
LeNet-5中的参数变化



全链接层F6：有84个节点，对应于一个7x12的比特图，-1表示白色，1表示黑色，这样每个符号的比特图的黑白色就对应于一个编码。

参数和连接数：（120 + 1）x84=10164

LeNet-5中的参数变化



输出层：有10个节点，对应阿拉伯数字0-9，采用的是径向基函数（RBF）的网络连接方式。

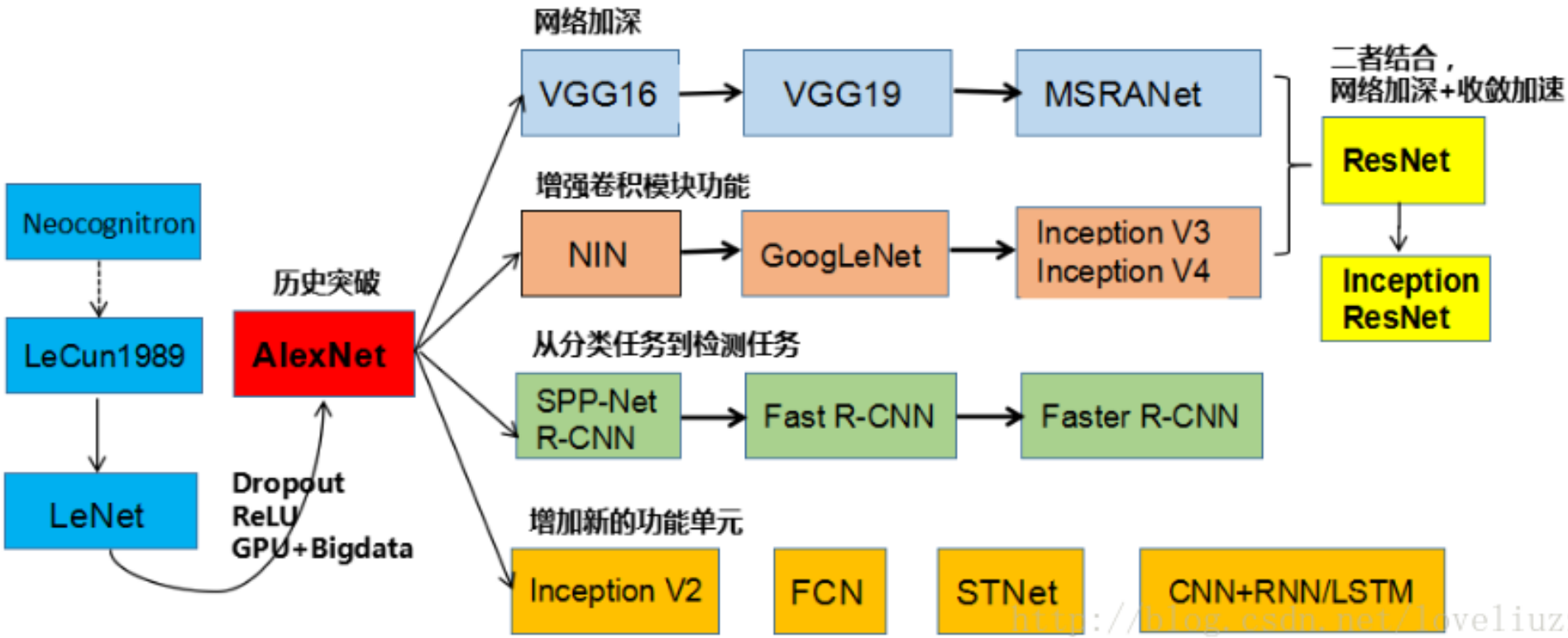
参数和连接数：84x10=840

CNN Application

CNN App and Developments

- LeNet：最早用于数字识别的CNN
- AlexNet：2012年ILSVRC比赛冠军，远超第二名的CNN，比LeNet更深，用多层小卷积叠加来替换单个的大卷积
- ZF Net：2013ILSVRC冠军
- GoogleNet：2014ILSVRC冠军
- VGGNet：2014ILSVRC比赛中算法模型，效果率低于GoogleNet
- ResNet：2015ILSVRC冠军，结构修正以适应更深层次的CNN训练

CNN App and Developments



CNN App and Developments

CNN 应用

A 图像分类

B 物体检测

C 物体追踪

D 姿态预估（Pose Estimatation）

E 文本检测识别

F 视觉 saliency 检测

G 行动识别

H 场景标记

经典的卷积神经网络：

1. LeNet
2. AlexNet
3. ZF
4. VGG
5. GoogLeNet
6. ResNet
7. DenseNet

Thanks!