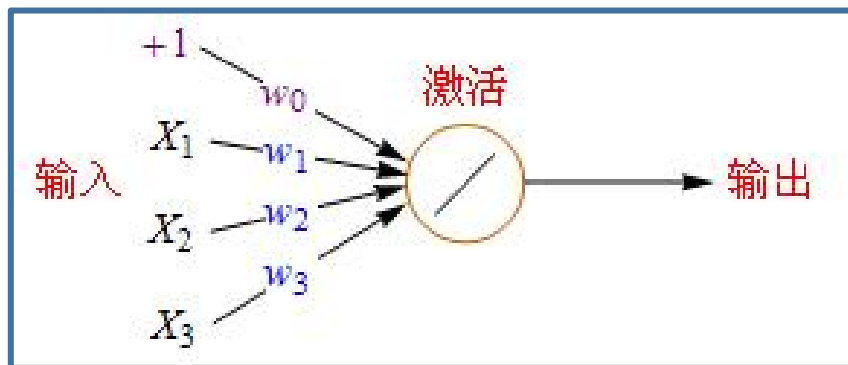

人工神经网络3

□ 神经元构成:



- +1代表偏移值(偏置项, Bias Units);
- X_1, X_2, X_2 代表初始特征;
- w_0, w_1, w_2, w_3 代表权重(Weight), 即参数, 是特征的缩放倍数; 特征经过缩放和偏移后全部累加起来,
- 此后还要经过一次激活运算然后再输出

练习:

- 给定以上神经网络及权重矩阵

$$W_1 = \begin{bmatrix} -5.5921 & 7.5976 \\ -0.5787 & -0.2875 \end{bmatrix}$$

$$W_2 = [-8.4075 \quad 0.4838]$$

$$\theta_1 = [0.5765 \quad -0.2764]$$

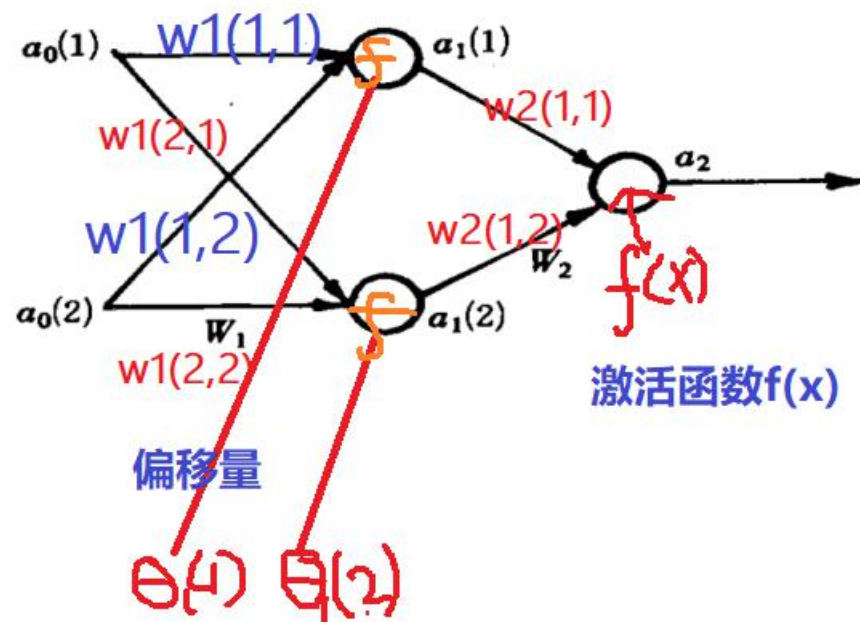
$$\theta_2 = 3.9829$$

$$f(x) = \frac{1}{1 + e^{-x}}$$

- 试计算以下蚊子的输出结果

❖ 1.72 1.24 Af 0.1

❖ 2.00 1.26 Apf 0.9



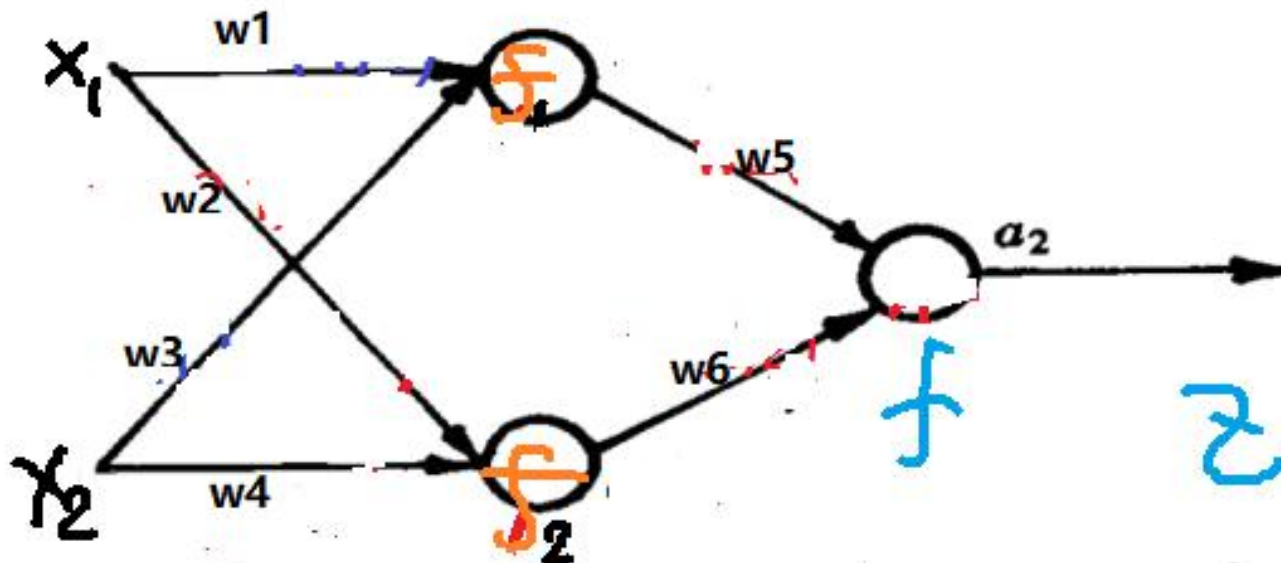
6 误差反向传播算法：让神经网络具备学习功能

- 本质：梯度下降法修改权重信息

$$\Delta \omega_{jk} = -\eta \frac{\partial E}{\partial \omega_{jk}} \quad j = 0, 1, 2, \dots, m; \quad \kappa = 1, 2, \dots, \ell$$

$$\Delta v_{ij} = -\eta \frac{\partial E}{\partial v_{ij}} \quad i = 0, 1, 2, \dots, n; \quad j = 1, 2, \dots, m$$

<https://bfoj>



给定以下神经网络:

输入: x_1, x_2 输出: z 激活函数: $f(x) = \frac{1}{1 + e^{-x}}$

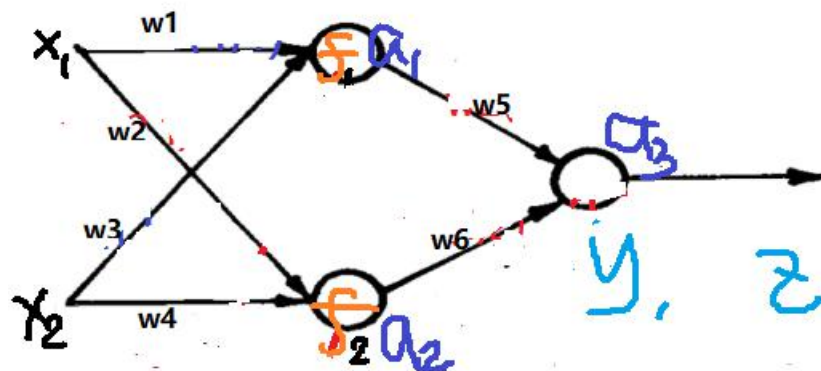
$$f'(x) = \frac{e^{-x}}{(1 + e^{-x})^2} = f(x)(1 - f(x))$$

损失函数定义为:

拟合函数值与给定输出值之间的误差

$$E(w) = (y(w) - z)^2$$

如果 $E < \text{eps}$, 说明拟合权重准确,
如果 E 偏大, 说明权重不准确,
需要反向修改权重, 直到误差满足条件



如何修改权重??

如果E 偏大，说明权重不准确，
需要反向修改权重，直到误差满足条件

$$\min_w E(w) = (y(w) - z)^2$$

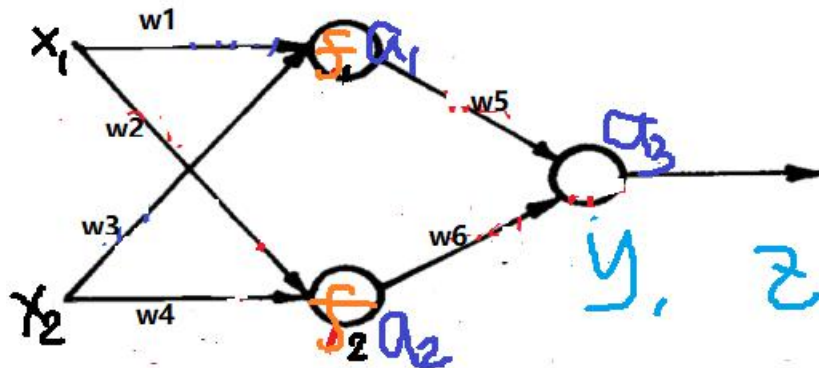
$$y(w) = f(w_5 f_1 + w_6 f_2)$$

$$f_1 = f(w_1 x_1 + w_3 x_2), f_2 = f(w_2 x_1 + w_4 x_2),$$

梯度下降法:

$$\nabla E(w) = 0 \Rightarrow \frac{\partial E}{\partial w_i} = 0$$

$$w_i^{k+1} = w_i^k - \eta \frac{\partial E(w^k)}{\partial w_i}, i = 1, 2, \dots, 6$$

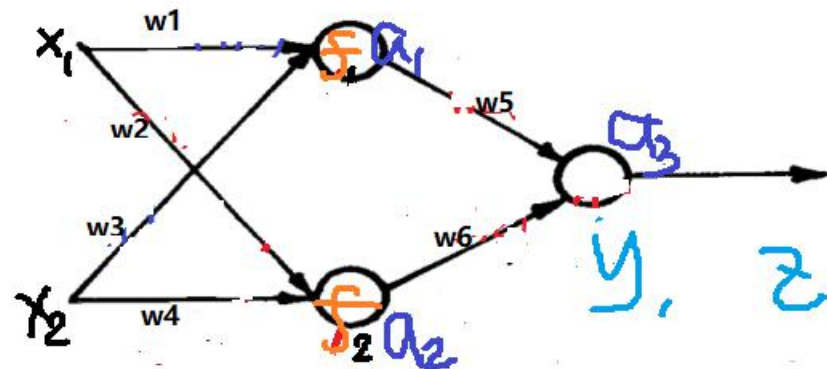


$$\min_{\mathbf{w}} E(\mathbf{w}) = (\mathbf{y}(\mathbf{w}) - \mathbf{z})^2$$

$$\mathbf{y}(\mathbf{w}) = f(\mathbf{w}_5 \mathbf{f}_1 + \mathbf{w}_6 \mathbf{f}_2) = f(\mathbf{a}_3)$$

$$\mathbf{f}_1 = f(\mathbf{w}_1 \mathbf{x}_1 + \mathbf{w}_3 \mathbf{x}_2) = f(\mathbf{a}_1),$$

$$\mathbf{f}_2 = f(\mathbf{w}_2 \mathbf{x}_1 + \mathbf{w}_4 \mathbf{x}_2) = f(\mathbf{a}_2),$$



$$\frac{\partial E(\mathbf{w})}{\partial \mathbf{w}_5} = (\mathbf{y}(\mathbf{w}) - \mathbf{z}) \frac{\partial \mathbf{f}}{\partial \mathbf{a}_3} \frac{\partial \mathbf{a}_3}{\partial \mathbf{w}_5}$$

$$= (\mathbf{y}(\mathbf{w}) - \mathbf{z}) \mathbf{f}(\mathbf{a}_3) (1 - \mathbf{f}(\mathbf{a}_3)) \mathbf{f}_1$$

$$\frac{\partial E(\mathbf{w})}{\partial \mathbf{w}_6} = (\mathbf{y}(\mathbf{w}) - \mathbf{z}) \frac{\partial \mathbf{f}}{\partial \mathbf{a}_3} \frac{\partial \mathbf{a}_3}{\partial \mathbf{w}_6}$$

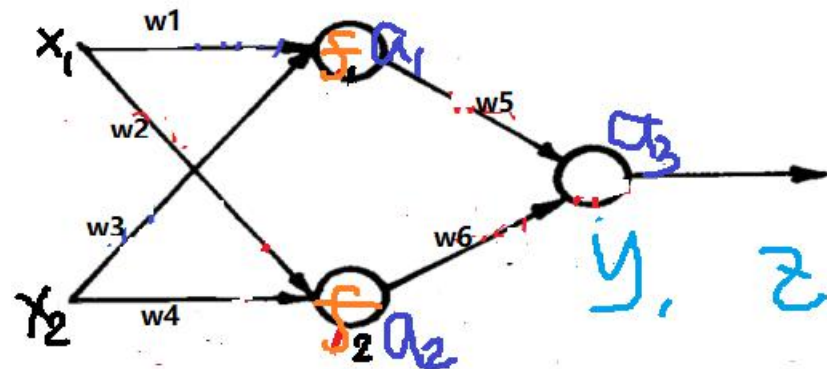
$$= (\mathbf{y}(\mathbf{w}) - \mathbf{z}) \mathbf{f}(\mathbf{a}_3) (1 - \mathbf{f}(\mathbf{a}_3)) \mathbf{f}_2$$

$$\min_{\mathbf{w}} E(\mathbf{w}) = (\mathbf{y}(\mathbf{w}) - \mathbf{z})^2$$

$$\mathbf{y}(\mathbf{w}) = f(\mathbf{w}_5 \mathbf{f}_1 + \mathbf{w}_6 \mathbf{f}_2) = f(\mathbf{a}_3)$$

$$\mathbf{f}_1 = f(\mathbf{w}_1 \mathbf{x}_1 + \mathbf{w}_3 \mathbf{x}_2) = f(\mathbf{a}_1),$$

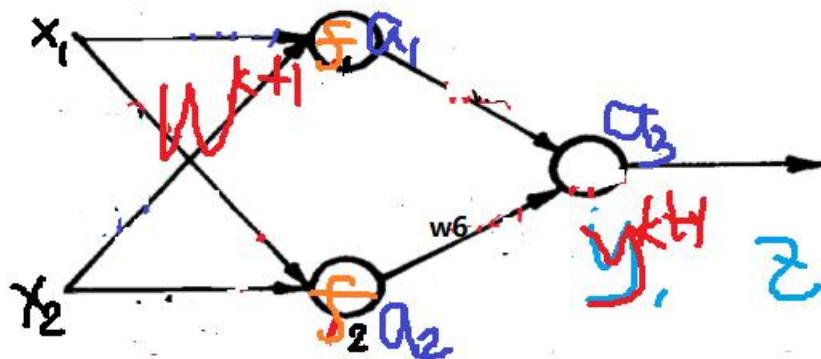
$$\mathbf{f}_2 = f(\mathbf{w}_2 \mathbf{x}_1 + \mathbf{w}_4 \mathbf{x}_2) = f(\mathbf{a}_2),$$



$$\begin{aligned} \frac{\partial E(\mathbf{w})}{\partial \mathbf{w}_1} &= (\mathbf{y}(\mathbf{w}) - \mathbf{z}) \frac{\partial f}{\partial \mathbf{a}_3} \frac{\partial \mathbf{a}_3}{\partial \mathbf{f}_1} \frac{\partial \mathbf{f}_1}{\partial \mathbf{a}_1} \frac{\partial \mathbf{a}_1}{\partial \mathbf{w}_1} \\ &= (\mathbf{y}(\mathbf{w}) - \mathbf{z}) f(\mathbf{a}_3) (1 - f(\mathbf{a}_3)) \mathbf{w}_5 f(\mathbf{a}_1) (1 - f(\mathbf{a}_1)) \mathbf{x}_1 \end{aligned}$$

$$\begin{aligned} \frac{\partial E(\mathbf{w})}{\partial \mathbf{w}_2} &= (\mathbf{y}(\mathbf{w}) - \mathbf{z}) \frac{\partial f}{\partial \mathbf{a}_3} \frac{\partial \mathbf{a}_3}{\partial \mathbf{f}_2} \frac{\partial \mathbf{f}_2}{\partial \mathbf{a}_2} \frac{\partial \mathbf{a}_2}{\partial \mathbf{w}_2} \\ &= (\mathbf{y}(\mathbf{w}) - \mathbf{z}) f(\mathbf{a}_3) (1 - f(\mathbf{a}_3)) \mathbf{w}_6 f(\mathbf{a}_2) (1 - f(\mathbf{a}_2)) \mathbf{x}_1 \end{aligned}$$

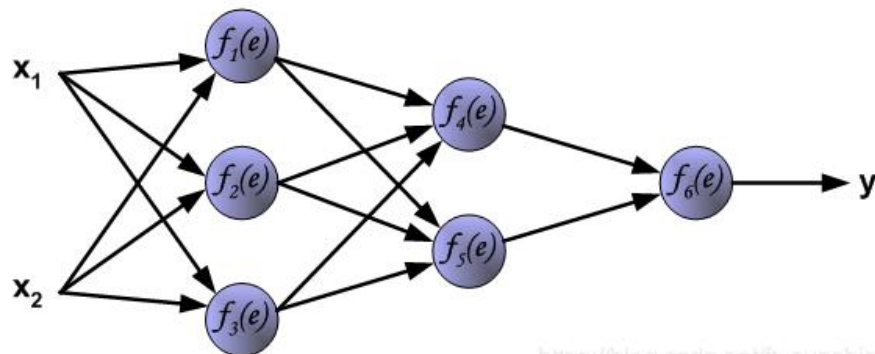
$$\min_w E(w) = (y(w) - z)^2$$



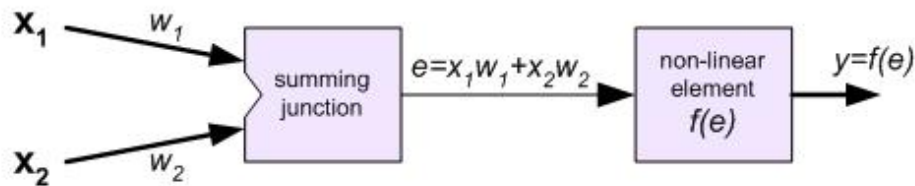
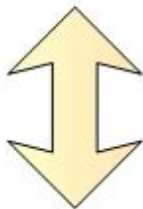
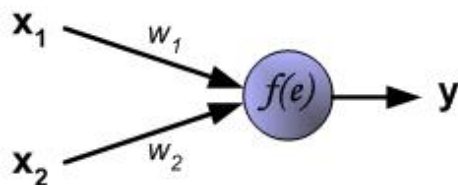
$$w_i^{k+1} = w_i^k - \eta \frac{\partial E(w^k)}{\partial w_i}, i = 1, 2, \dots, 6$$

根据新得到的权重 w^{k+1} , 重 计算输出值 y^{k+1} ,
依次迭代更新, 直到满足损失函数值尽量小

- 针对以下神经网络， $y=f(e)$ ，为输出结果， f 为激活函数

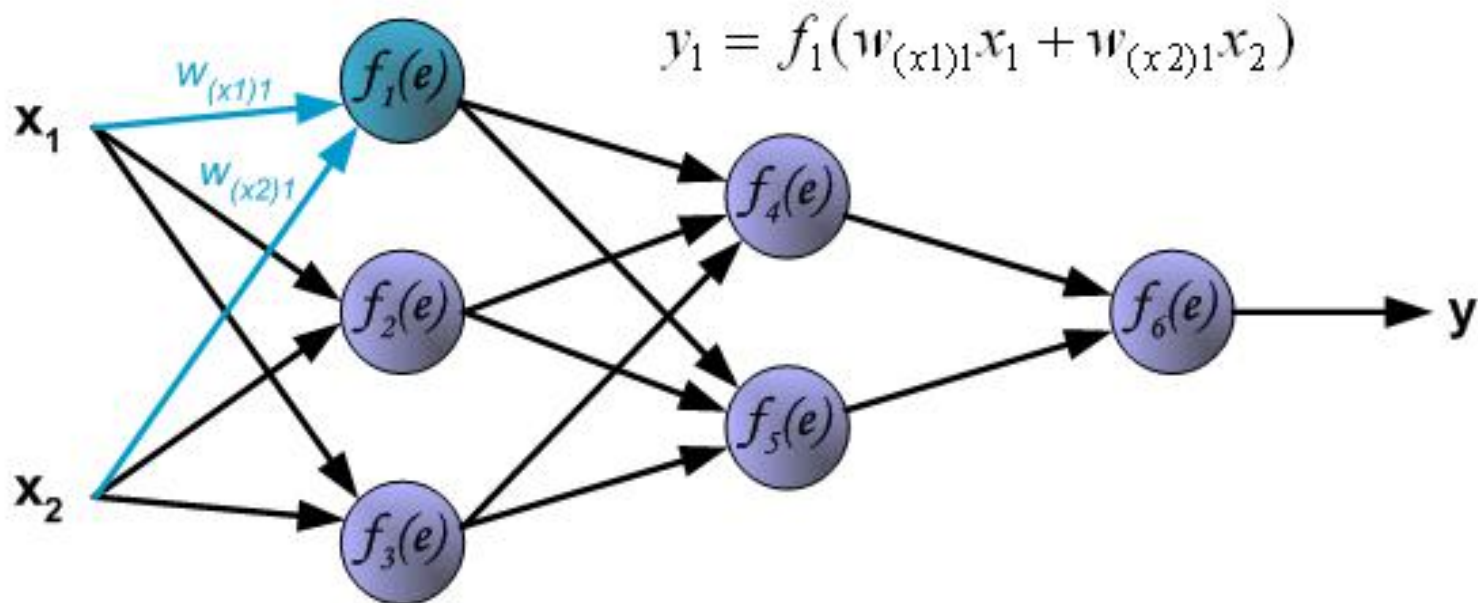


https://blog.csdn.net/it_sunshine

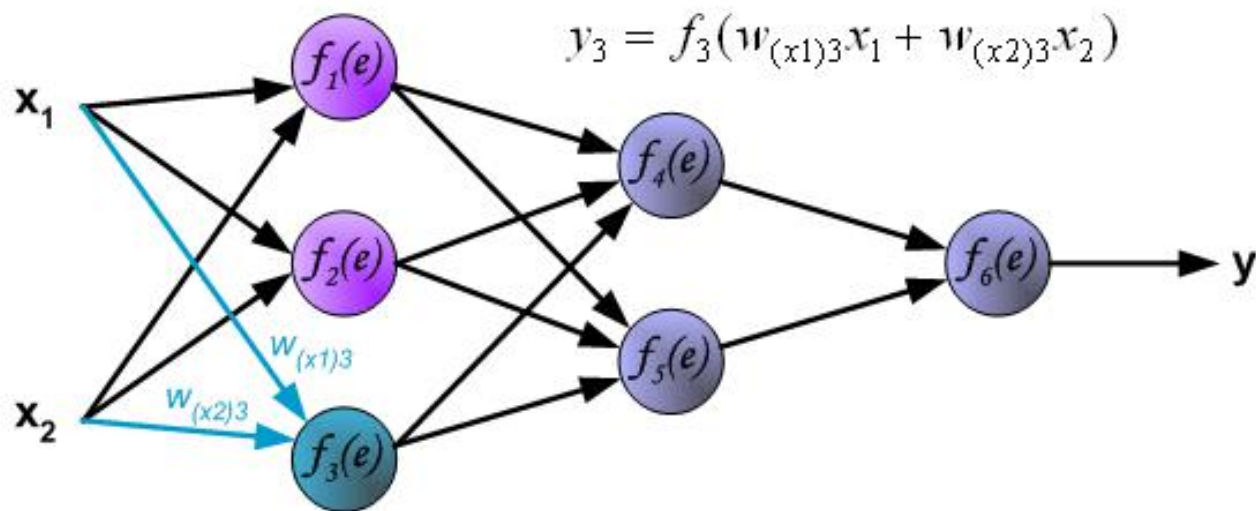
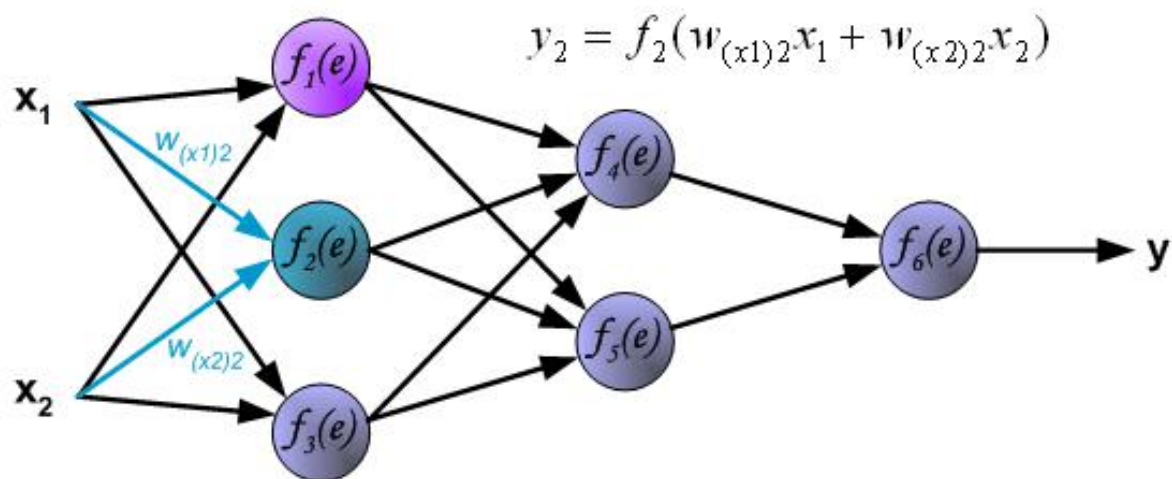


第一步计算前向传播过程

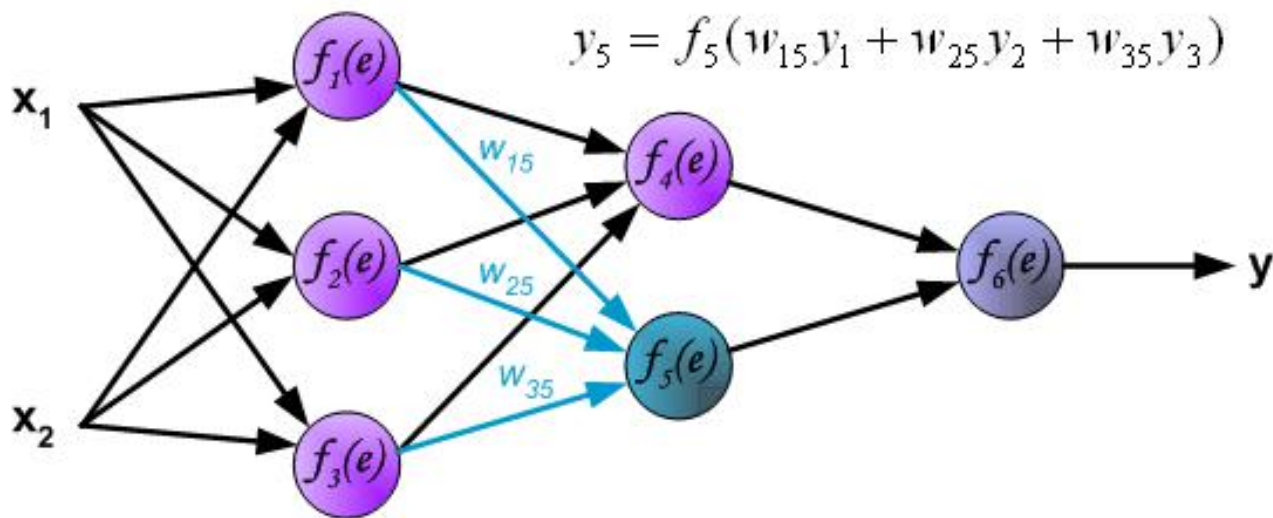
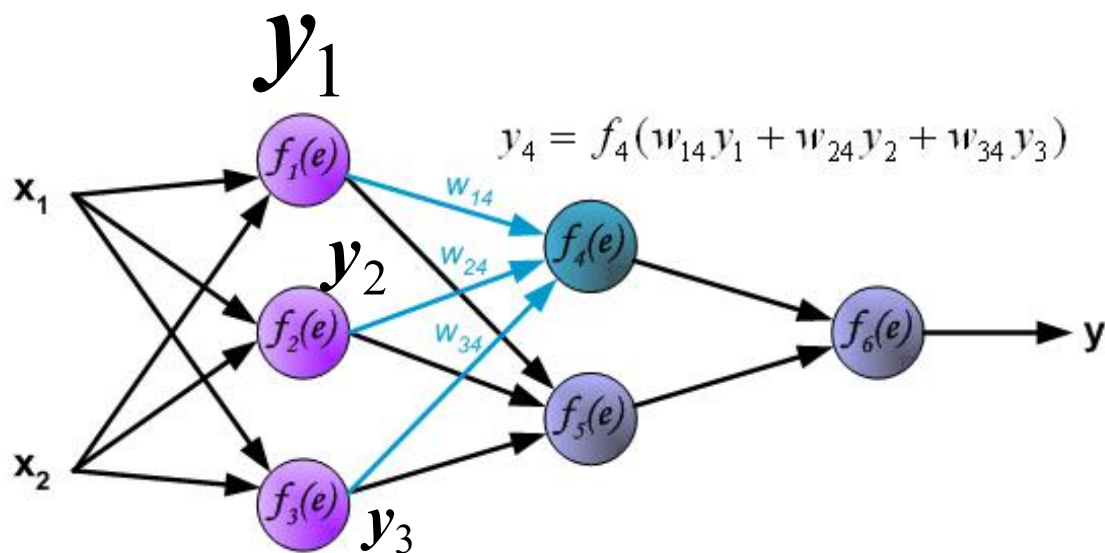
下面是前向传播过程：



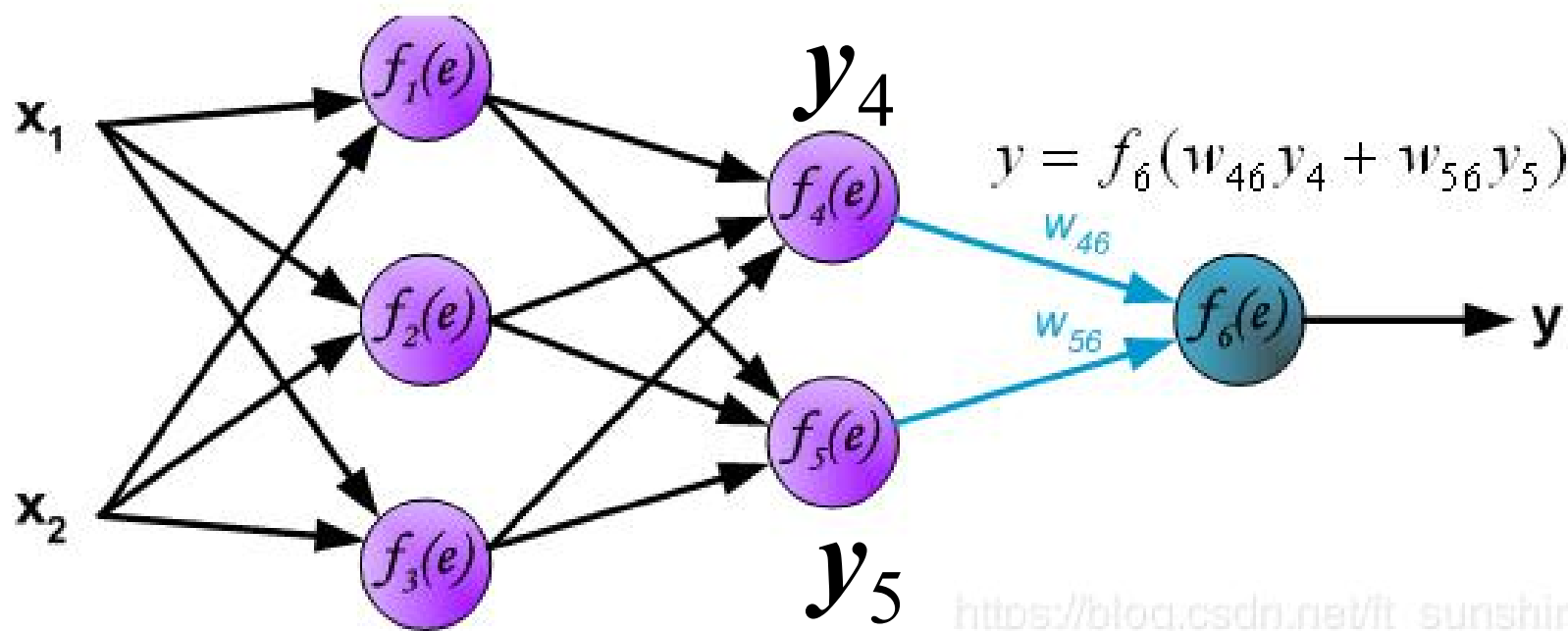
计算第一层



计算第二层，第一层结果为输入



- 第二层的结果为输入，输出正向结果



https://blog.csdn.net/ft_sunshine

- 计算损失函数：正向结果 y 与目标结果 z 差的平方

$$E = (y - z)^2 = (f_6(w_{46}y_4 + w_{56}y_5) - z)^2$$

$$y_1 = f_1(w_{(x1)1}x_1 + w_{(x2)1}x_2)$$

$$y_4 = f_4(w_{14}y_1 + w_{24}y_2 + w_{34}y_3)$$

$$y_2 = f_2(w_{(x1)2}x_1 + w_{(x2)2}x_2)$$

$$y_5 = f_5(w_{15}y_1 + w_{25}y_2 + w_{35}y_3)$$

$$y_3 = f_3(w_{(x1)3}x_1 + w_{(x2)3}x_2)$$

$$f_i(x) = \frac{1}{1 + e^{-x}}, i = 1, 2, 3, 4, 5, 6$$

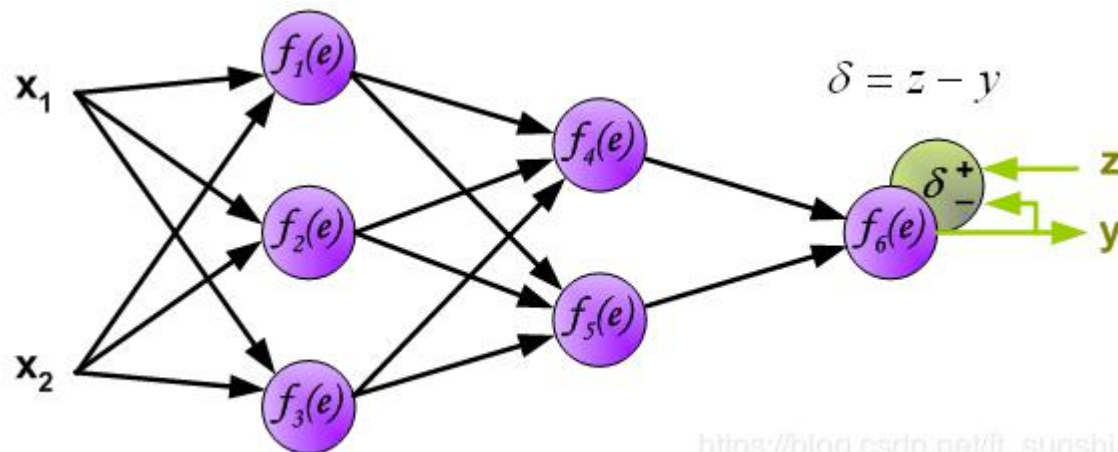
- 因此损失函数 E 为关于权重的复合函数

$$E(w, f_1, \dots, f_6)$$

- 当 E 等于0时，说明权重合适，不需要更新
- 当 E 不等于0时，结果与实际目标不符，需要调整权重，重新计算输出结果
- 更新权重方法，梯度下降法：

$$w_{ij}^{k+1} = w_{ij}^k - \eta \frac{\partial E}{\partial w_{ij}}$$

计算反向误差



记 $e = w_{46}y_4 + w_{56}y_5$

$$E = (z - y)^2 = (z - f_6(w_{46}y_4 + w_{56}y_5))^2 = (z - f_6(e))^2$$

$$\frac{\partial E}{\partial w_{46}} = -(z - y) \frac{\partial f_6(e)}{\partial e} y_4$$

$$\frac{\partial E}{\partial w_{56}} = -(z - y) \frac{\partial f_6(e)}{\partial e} y_5$$

□ 第二层误差

$$E = (y - z)^2 = (f_6(w_{46}y_4 + w_{56}y_5) - z)^2$$

$$y_4 = f_4(w_{14}y_1 + w_{24}y_2 + w_{34}y_3)$$

$$y_5 = f_5(w_{15}y_1 + w_{25}y_2 + w_{35}y_3)$$

$$f_i(x) = \frac{1}{1 + e^{-x}}, i = 1, 2, 3, 4, 5, 6$$

$$\frac{\partial E}{\partial w_{14}} = -(z - y) \frac{\partial f_6(e)}{\partial e} w_{46} \frac{\partial f_4(e)}{\partial e} y_1$$

$$\frac{\partial E}{\partial w_{24}} = -(z - y) \frac{\partial f_6(e)}{\partial e} w_{46} \frac{\partial f_4(e)}{\partial e} y_2$$

$$\frac{\partial E}{\partial w_{34}} = -(z - y) \frac{\partial f_6(e)}{\partial e} w_{46} \frac{\partial f_4(e)}{\partial e} y_3$$

$$\frac{\partial E}{\partial w_{15}} = -(z - y) \frac{\partial f_6(e)}{\partial e} w_{56} \frac{\partial f_5(e)}{\partial e} y_1$$

$$\frac{\partial E}{\partial w_{25}} = -(z - y) \frac{\partial f_6(e)}{\partial e} w_{56} \frac{\partial f_5(e)}{\partial e} y_2$$

$$\frac{\partial E}{\partial w_{35}} = -(z - y) \frac{\partial f_6(e)}{\partial e} w_{56} \frac{\partial f_5(e)}{\partial e} y_3$$

□ 第一层误差

$$E = (y - z)^2 = (f_6(w_{46}y_4 + w_{56}y_5) - z)^2$$

$$y_1 = f_1(w_{(x1)1}x_1 + w_{(x2)1}x_2)$$

$$y_4 = f_4(w_{14}y_1 + w_{24}y_2 + w_{34}y_3)$$

$$y_2 = f_2(w_{(x1)2}x_1 + w_{(x2)2}x_2)$$

$$y_5 = f_5(w_{15}y_1 + w_{25}y_2 + w_{35}y_3)$$

$$y_3 = f_3(w_{(x1)3}x_1 + w_{(x2)3}x_2)$$

$$\frac{\partial E}{\partial w_{(x1)1}} = -(z - y) \frac{\partial f_6(e)}{\partial e} (w_{46} \frac{\partial f_4(e)}{\partial e} w_{14} \frac{\partial f_1(e)}{\partial e} x_1 + w_{56} \frac{\partial f_5(e)}{\partial e} w_{15} \frac{\partial f_1(e)}{\partial e} x_1)$$

$$\frac{\partial E}{\partial w_{(x2)1}} = -(z - y) \frac{\partial f_6(e)}{\partial e} (w_{46} \frac{\partial f_4(e)}{\partial e} w_{14} \frac{\partial f_1(e)}{\partial e} x_2 + w_{56} \frac{\partial f_5(e)}{\partial e} w_{15} \frac{\partial f_1(e)}{\partial e} x_2)$$

$$\frac{\partial E}{\partial w_{(x1)2}} = -(z - y) \frac{\partial f_6(e)}{\partial e} (w_{46} \frac{\partial f_4(e)}{\partial e} w_{24} \frac{\partial f_2(e)}{\partial e} x_1 + w_{56} \frac{\partial f_5(e)}{\partial e} w_{25} \frac{\partial f_2(e)}{\partial e} x_1)$$

$$\frac{\partial E}{\partial w_{(x2)2}} = -(z - y) \frac{\partial f_6(e)}{\partial e} (w_{46} \frac{\partial f_4(e)}{\partial e} w_{24} \frac{\partial f_2(e)}{\partial e} x_2 + w_{56} \frac{\partial f_5(e)}{\partial e} w_{25} \frac{\partial f_2(e)}{\partial e} x_2)$$

$$\frac{\partial E}{\partial w_{(x1)3}} = -(z - y) \frac{\partial f_6(e)}{\partial e} (w_{46} \frac{\partial f_4(e)}{\partial e} w_{34} \frac{\partial f_3(e)}{\partial e} x_1 + w_{56} \frac{\partial f_5(e)}{\partial e} w_{35} \frac{\partial f_3(e)}{\partial e} x_1)$$

$$\frac{\partial E}{\partial w_{(x2)3}} = -(z - y) \frac{\partial f_6(e)}{\partial e} (w_{46} \frac{\partial f_4(e)}{\partial e} w_{34} \frac{\partial f_3(e)}{\partial e} x_2 + w_{56} \frac{\partial f_5(e)}{\partial e} w_{35} \frac{\partial f_3(e)}{\partial e} x_2)$$

常见的几种神经网络

□ 卷积

- 提取图片的特征

□ 卷积核

- 通常为较小尺寸的矩阵, $3 * 3$, $5 * 5$

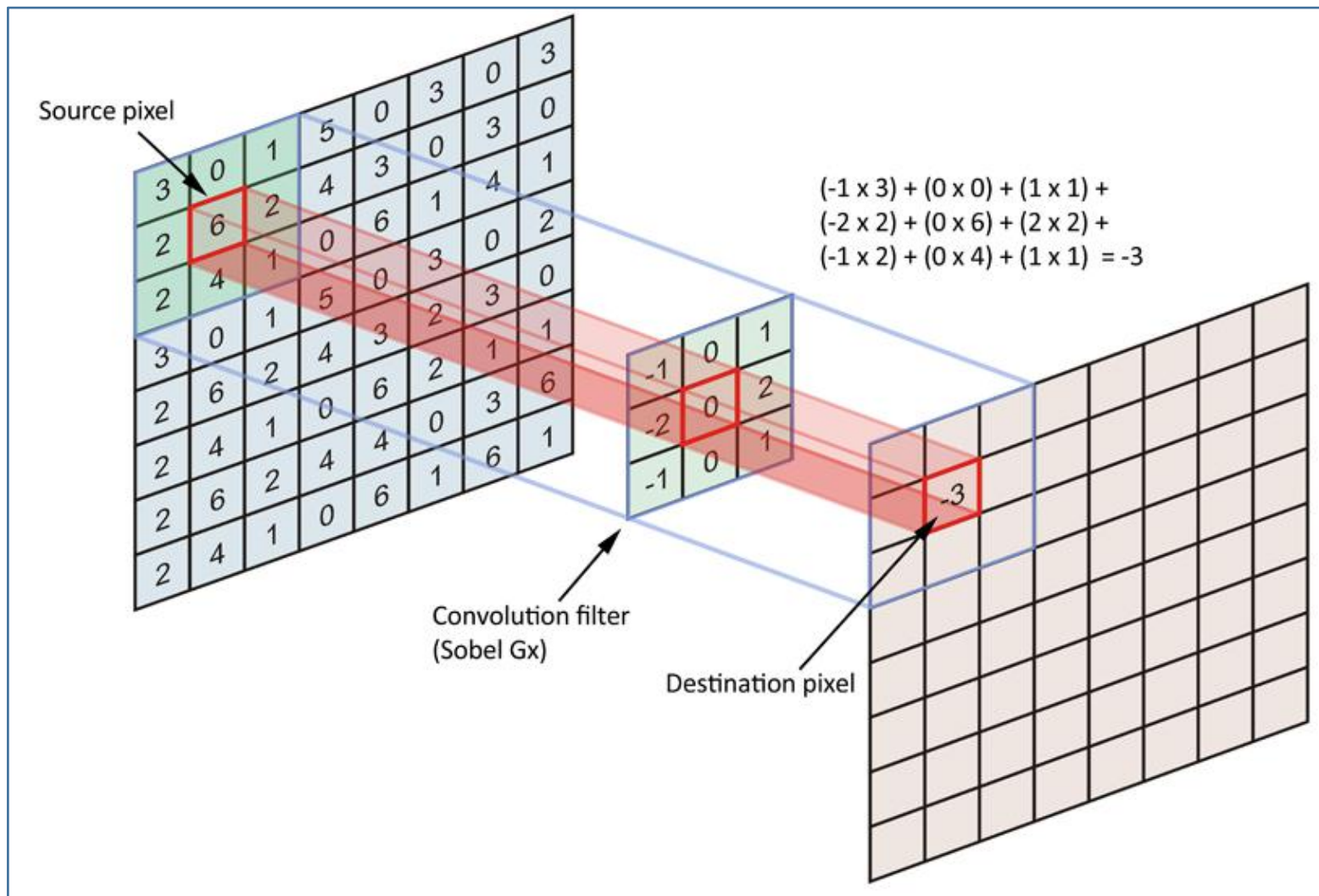
□ 卷积运算

- 使用卷积核自上而下、自左向右在图像上滑动, 将卷积核矩阵的各个元素与它在图像上覆盖的对应位置元素相乘, 求和

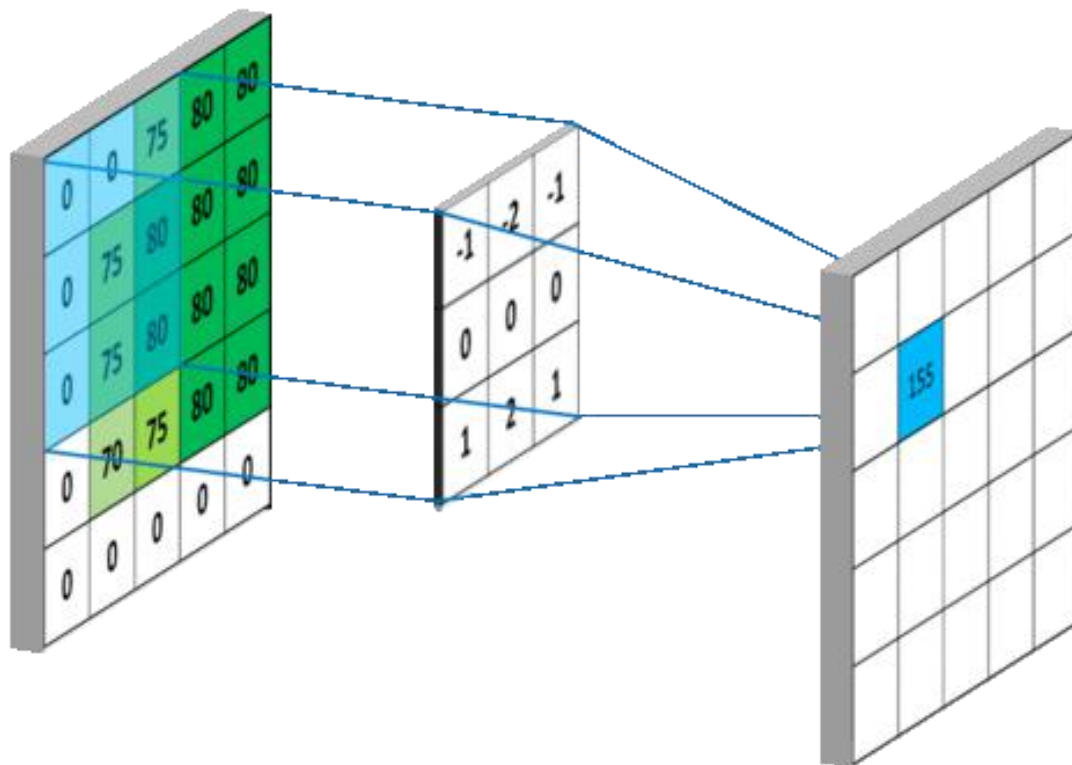
$$\begin{bmatrix} \mathbf{I}_{x-1,y-1} & \mathbf{I}_{x-1,y} & \mathbf{I}_{x-1,y+1} \\ \mathbf{I}_{x,y-1} & \mathbf{I}_{x,y} & \mathbf{I}_{x,y+1} \\ \mathbf{I}_{x+1,y-1} & \mathbf{I}_{x+1,y} & \mathbf{I}_{x+1,y+1} \end{bmatrix}, \text{卷积核} = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

(x, y) 处的卷积结果为:

$$-\mathbf{I}_{x-1,y-1} - 2\mathbf{I}_{x-1,y} - \mathbf{I}_{x-1,y+1} + \mathbf{I}_{x+1,y-1} + 2\mathbf{I}_{x+1,y} + \mathbf{I}_{x+1,y+1}$$

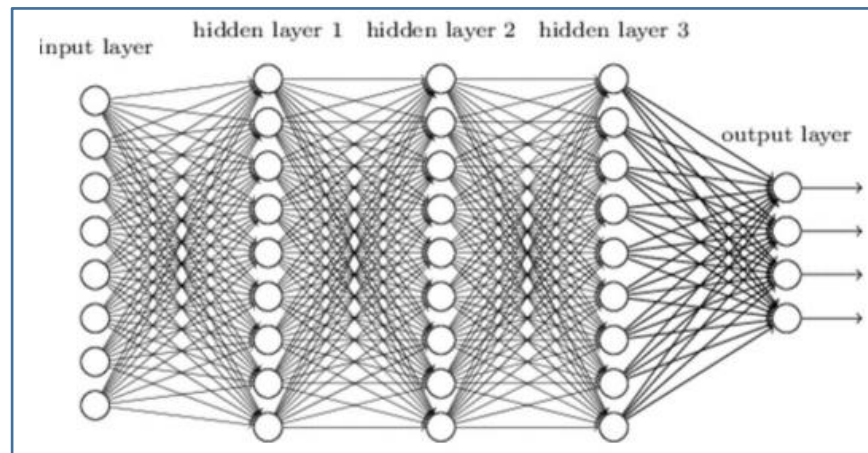
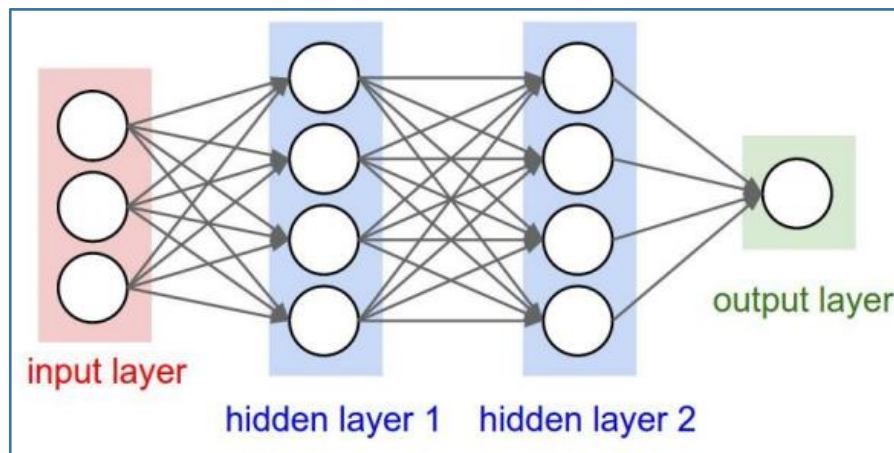


不同的卷积核



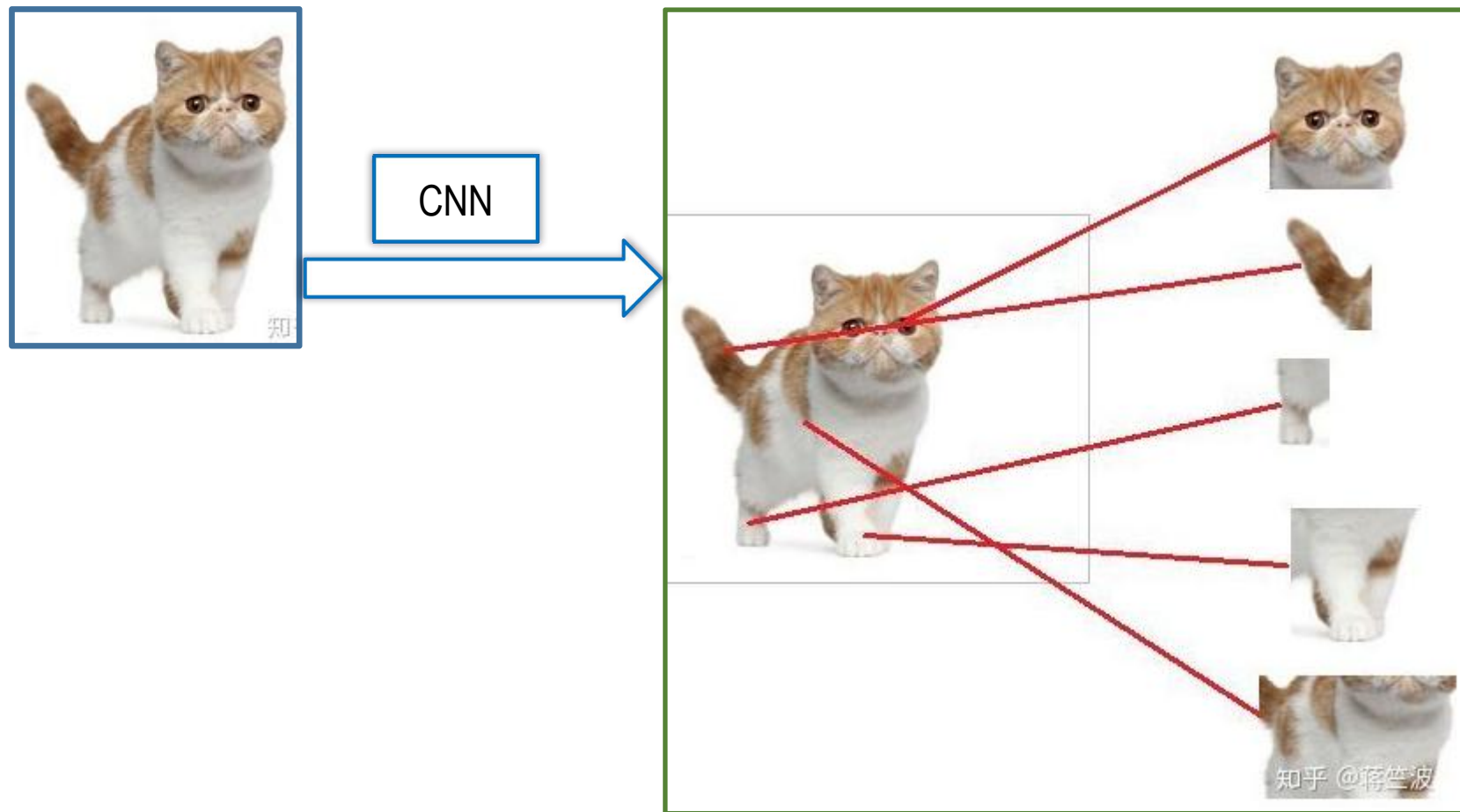
神经网络与深度神经网络

- 神经网络：模拟特征与目标之间的真实关系函数的方法
- 多层神经网络：更多的参数意味着其模拟的函数可以更加的复杂
表示能力大幅度增强



- ▣ 卷积滤波器和神经网络两个思想结合起来
- ▣ 若干底层特征组成上一层特征，最终通过多个层级的组合做出分类
- ▣ 是一种多层神经网络，擅长处理图像相关的机器学习问题

- ▣ 卷积神经网络结构：
 - ①数据输入层/ Input layer: 数据预算理，去均值，归一化，PCA降维
 - ②卷积计算层/ CONV layer: 最重要的一个层次
 - ③ReLU激励层 / ReLU layer: 把卷积层输出结果做非线性映射
 - ④池化层 / Pooling layer: 用于压缩数据和参数的量，减小过拟合
 - ⑤全连接层 / FC layer: 两层之间所有神经元都有权重连接



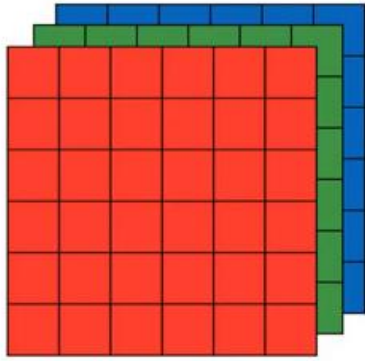
□ 卷积

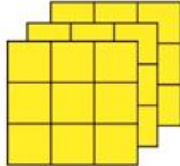
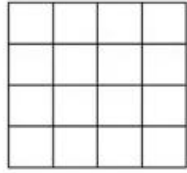
- 设计特定的卷积核，与图像做卷积，提取图片的特征

□ 卷积核

- 通常为较小尺寸的矩阵， $3 * 3$ ， $5 * 5$

卷积核的颜色通道与输入图像颜色通道一致



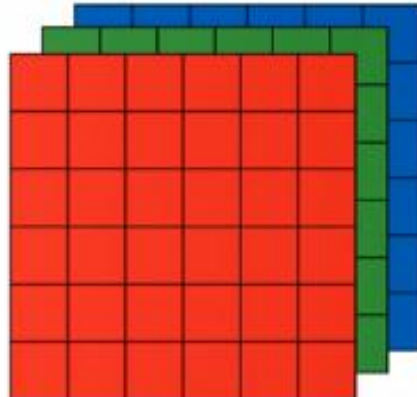
$*$

 $=$


4 x 4

对于一张具有3通道的RGB颜色的图像其大小为6*6*3

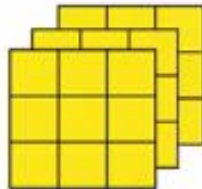
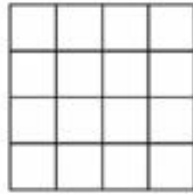
卷积核大小也为3*3*3即具有3个颜色通道的卷积核

生成一个4*4大小的特征图

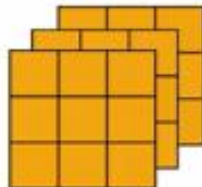
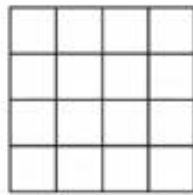


$*$

valid edge


 $=$


3 x 3 x 3 4 x 4

$*$

 $=$


3 x 3 x 3 4 x 4

<http://blog.csdn.net/4x4sec>

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



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

=

0.77	0.11	0.11	0.33	0.55	0.11	0.11
0.11	1.00	0.11	0.33	0.11	0.11	0.11
0.11	0.11	1.00	0.33	0.11	0.11	0.55
0.33	0.33	0.33	0.55	0.33	0.33	0.33
0.55	0.11	0.11	0.33	1.00	0.11	0.11
0.11	0.11	0.11	0.33	0.11	1.00	0.11
0.33	0.11	0.55	0.33	0.11	0.11	0.77

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



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

=

0.33	0.55	0.11	0.11	0.11	0.55	0.33
0.55	0.55	0.55	0.33	0.55	0.55	0.55
0.33	0.55	0.55	0.77	0.55	0.55	0.11
0.11	0.33	0.77	1.00	0.77	0.33	0.11
0.11	0.55	0.55	0.77	0.55	0.55	0.11
0.55	0.55	0.55	0.33	0.55	0.55	0.55
0.33	0.55	0.11	0.11	0.11	0.55	0.33

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



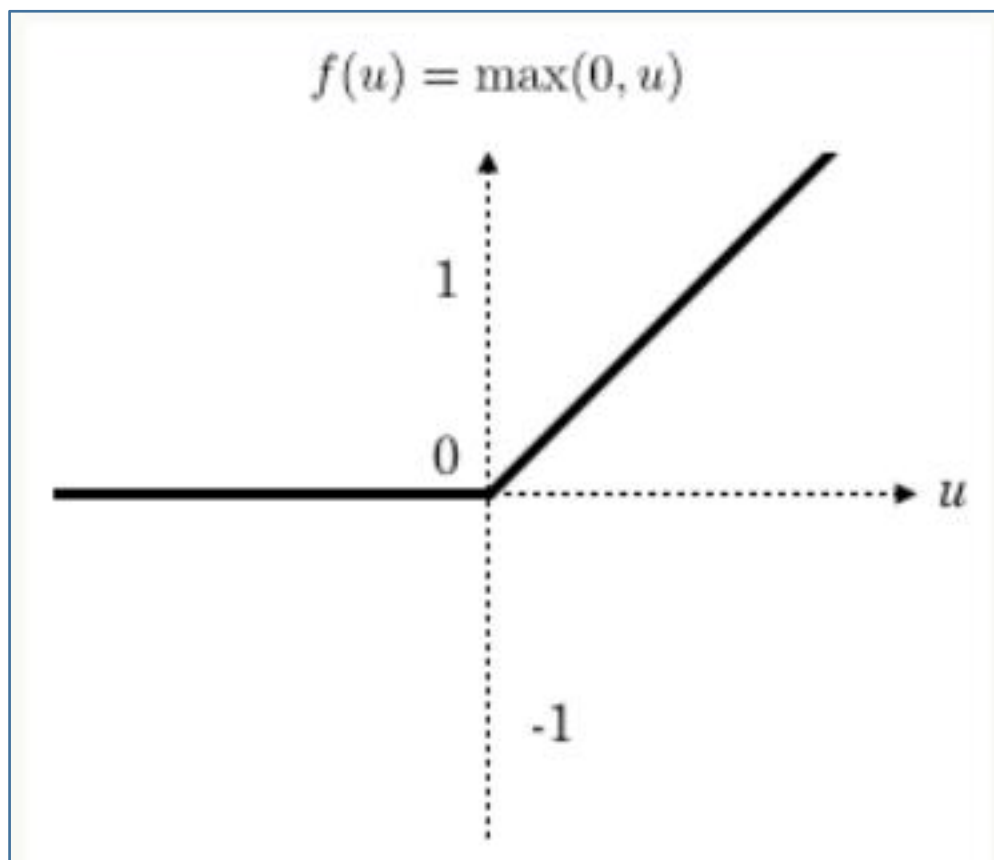
-1	-1	1
-1	1	-1
1	-1	-1

=

0.33	0.11	0.55	0.33	0.11	0.11	0.77
0.11	0.11	0.11	0.33	0.11	1.00	0.11
0.55	0.11	0.11	0.33	1.00	0.11	0.11
0.33	0.33	0.33	0.55	0.33	0.33	0.33
0.11	0.11	1.00	0.33	0.11	0.11	0.55
0.11	1.00	0.11	0.33	0.11	0.11	0.11
0.77	0.11	0.11	0.33	0.55	0.11	0.33

ReLU函数 (Rectified Linear Units)

$$f(x) = \max\{0, x\}$$



卷积后产生的特征图中的值，越靠近1表示与该特征越关联，越靠近-1表示越不关联，进行特征提取时，为了使得数据更少，操作更方便，就直接舍弃掉那些不相关联的数据。

0.77	-0.11	0.11	0.33	0.55	-0.11	0.33
-0.11	1.00	-0.11	0.33	-0.11	0.11	-0.11
0.11	-0.11	1.00	-0.33	0.11	-0.11	0.55
0.33	0.33	-0.33	0.55	-0.33	0.33	0.33
0.55	-0.11	0.11	-0.33	1.00	-0.11	0.11
-0.11	0.11	-0.11	0.33	-0.11	1.00	-0.11
0.33	-0.11	0.55	0.33	0.11	-0.11	0.77

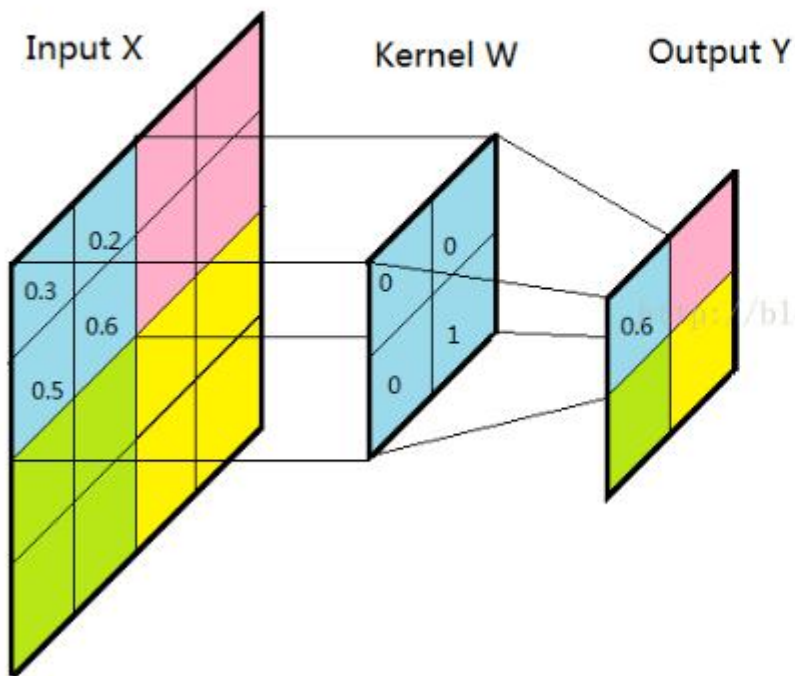


0.77	0	0.11	0.33	0.55	0	0.33
0	1.00	0	0.33	0	0.11	0
0.11	0	1.00	0	0.11	0	0.55
0.33	0.33	0	0.55	0	0.33	0.33
0.55	0	0.11	0	1.00	0	0.11
0	0.11	0	0.33	0	1.00	0
0.33	0	0.55	0.33	0.11	0	0.77

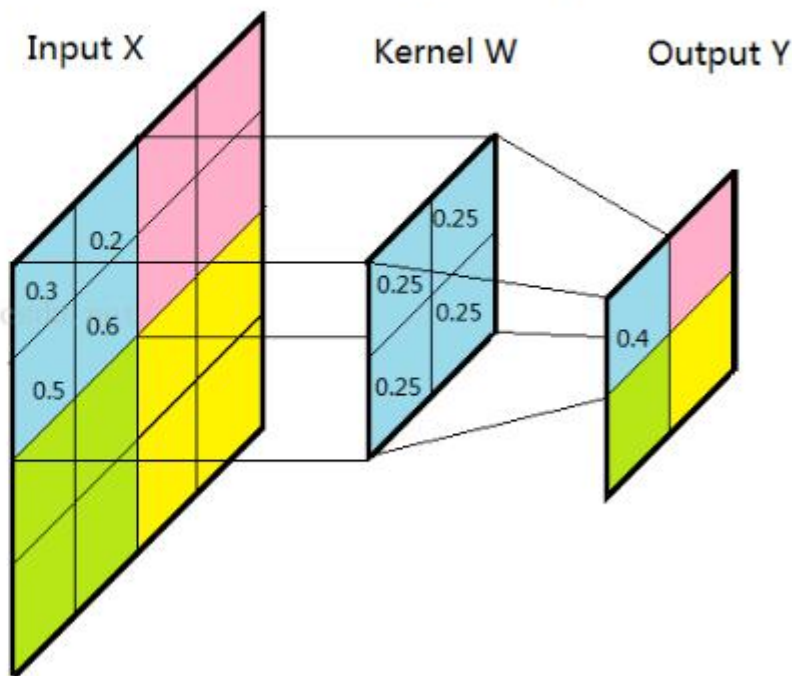
池化层：下采样降维

- 卷积操作后，得到了有着不同值的feature map，尽管数据量比原图少了很多，但还是过于庞大，因此使用接下来的池化操作减少数据量。
- 最大池化：某一个区域用最大值代替
- 平均池化：某一个区域用平均值代替

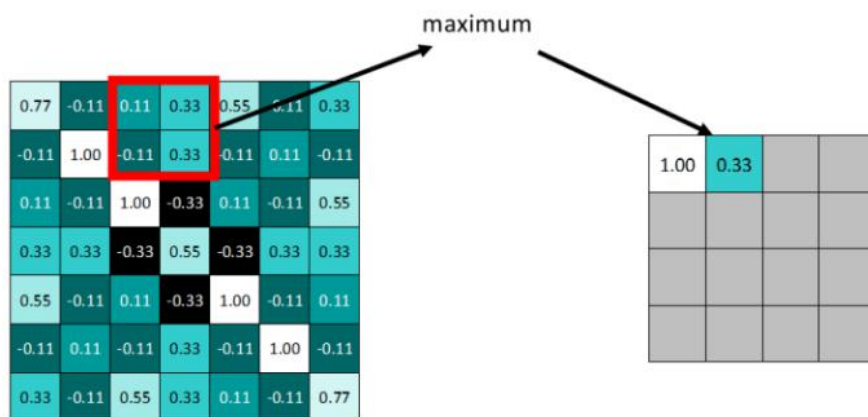
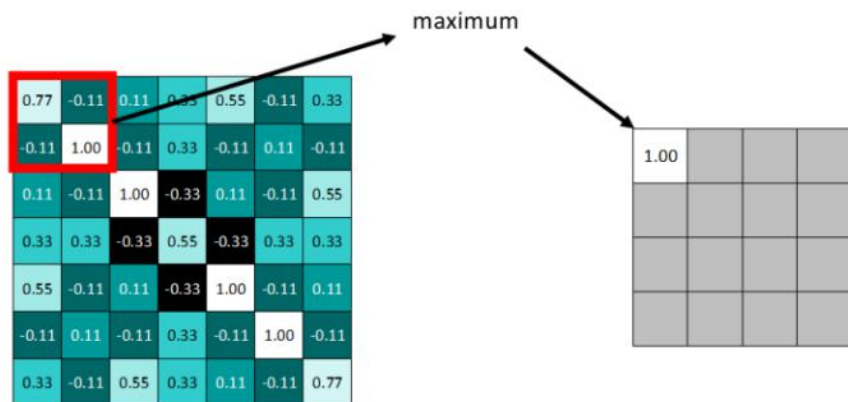
max-pooling



mean-polling



Navigation icons: back, forward, search, etc.



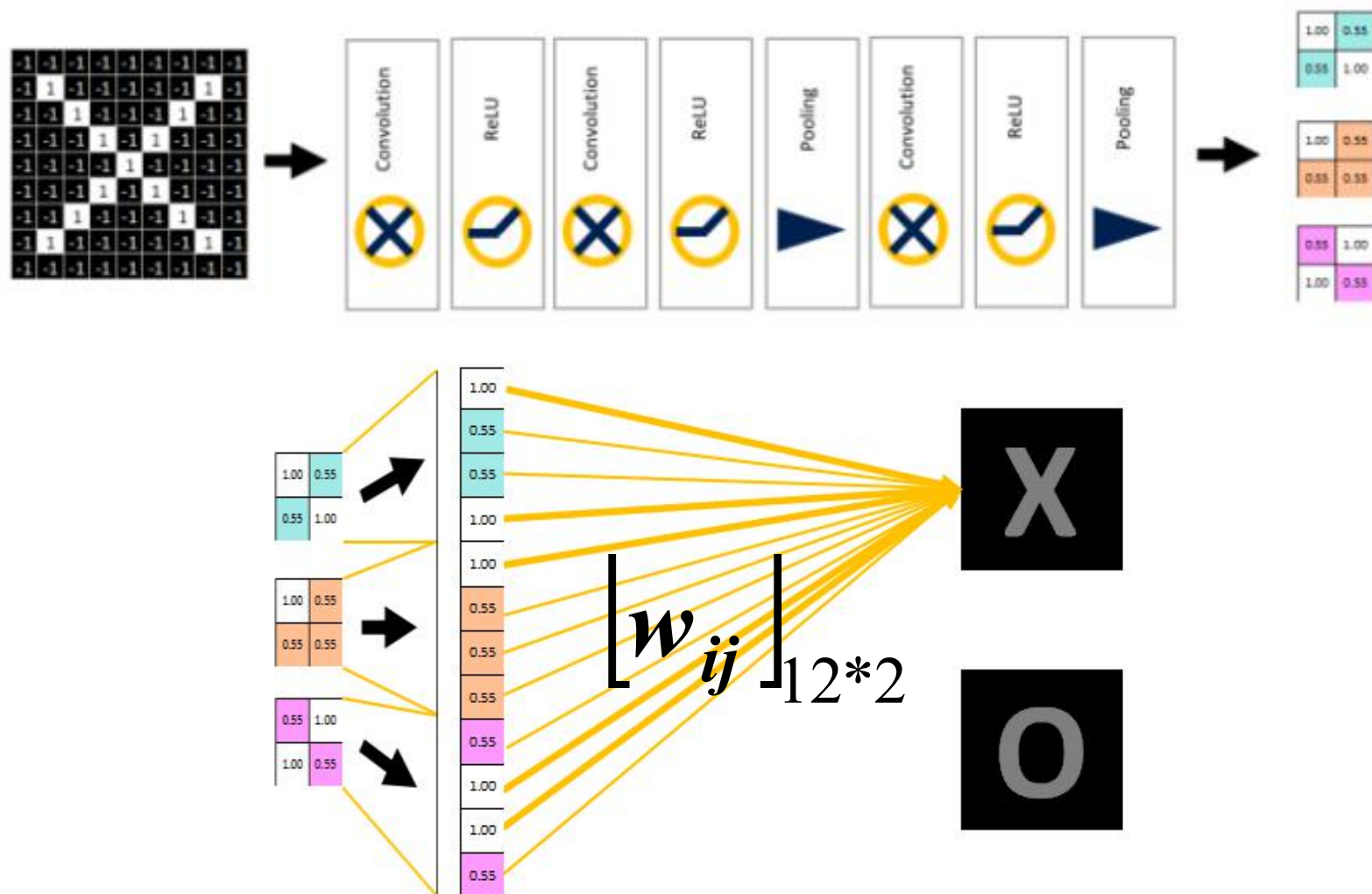
0.77	-0.11	0.11	0.33	0.55	-0.11	0.33
-0.11	1.00	-0.11	0.33	-0.11	0.11	-0.11
0.11	-0.11	1.00	-0.33	0.11	-0.11	0.55
0.33	0.33	-0.33	0.55	-0.33	0.33	0.33
0.55	-0.11	0.11	-0.33	1.00	-0.11	0.11
-0.11	0.11	-0.11	0.33	-0.11	1.00	-0.11
0.33	-0.11	0.55	0.33	0.11	-0.11	0.77

max pooling

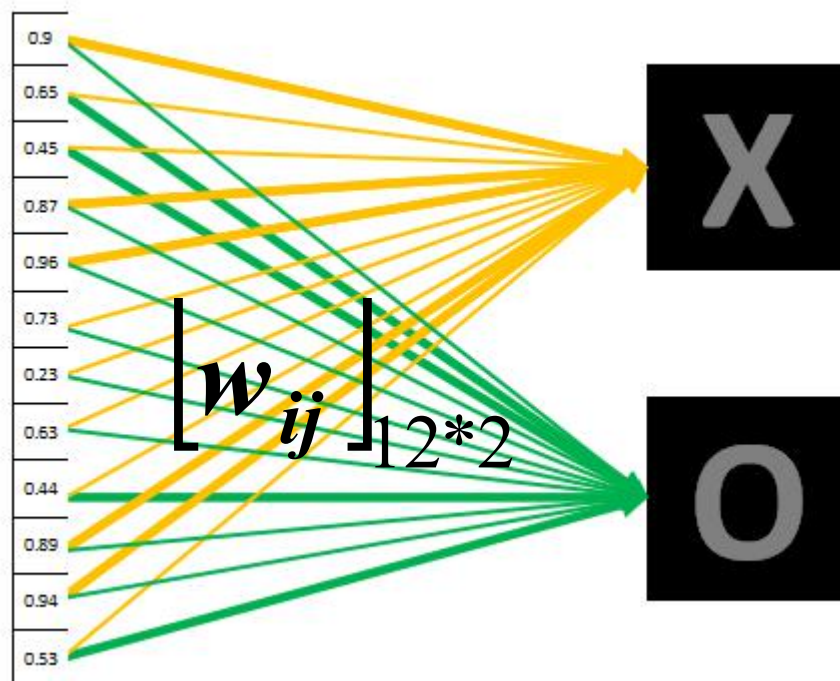
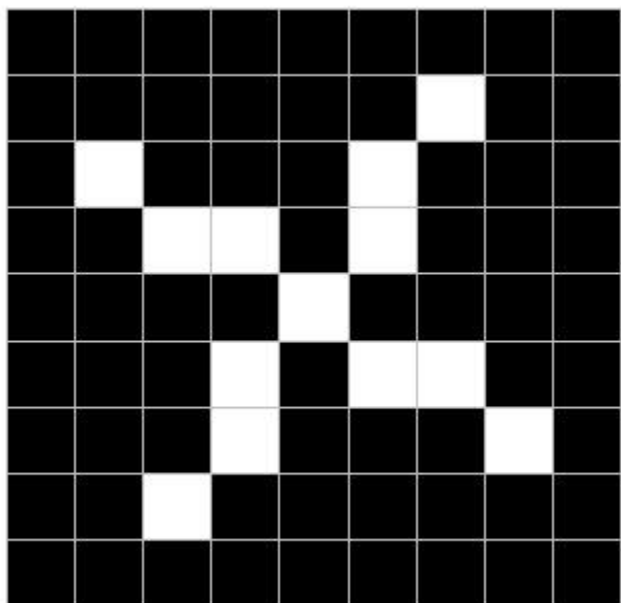
1.00	0.33	0.55	0.33
0.33	1.00	0.33	0.55
0.55	0.33	1.00	0.11
0.33	0.55	0.11	0.77

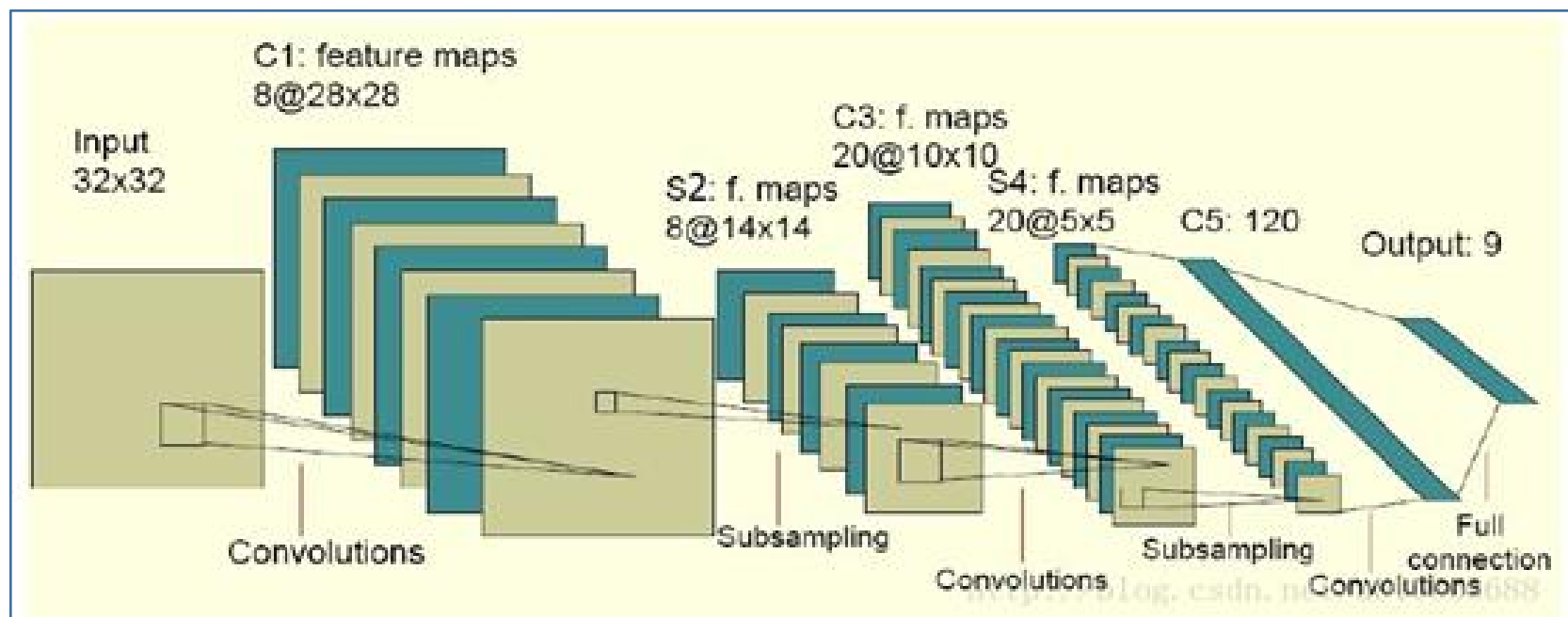
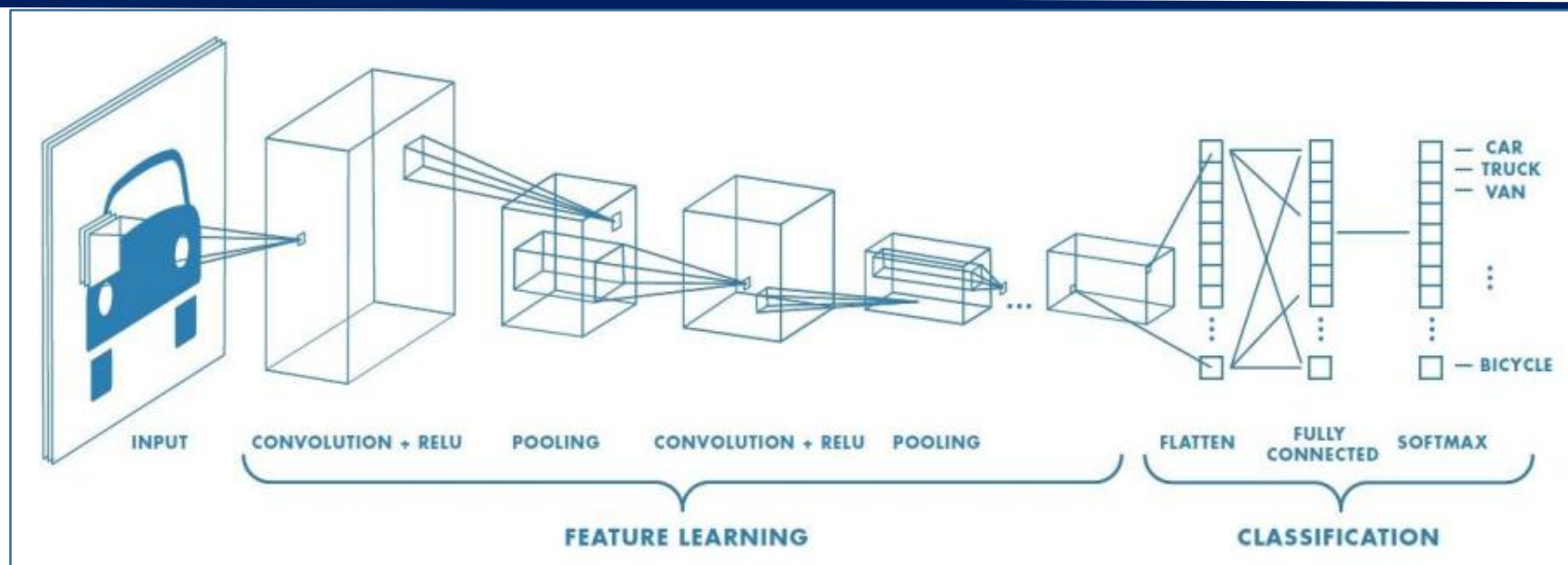
全连接层:

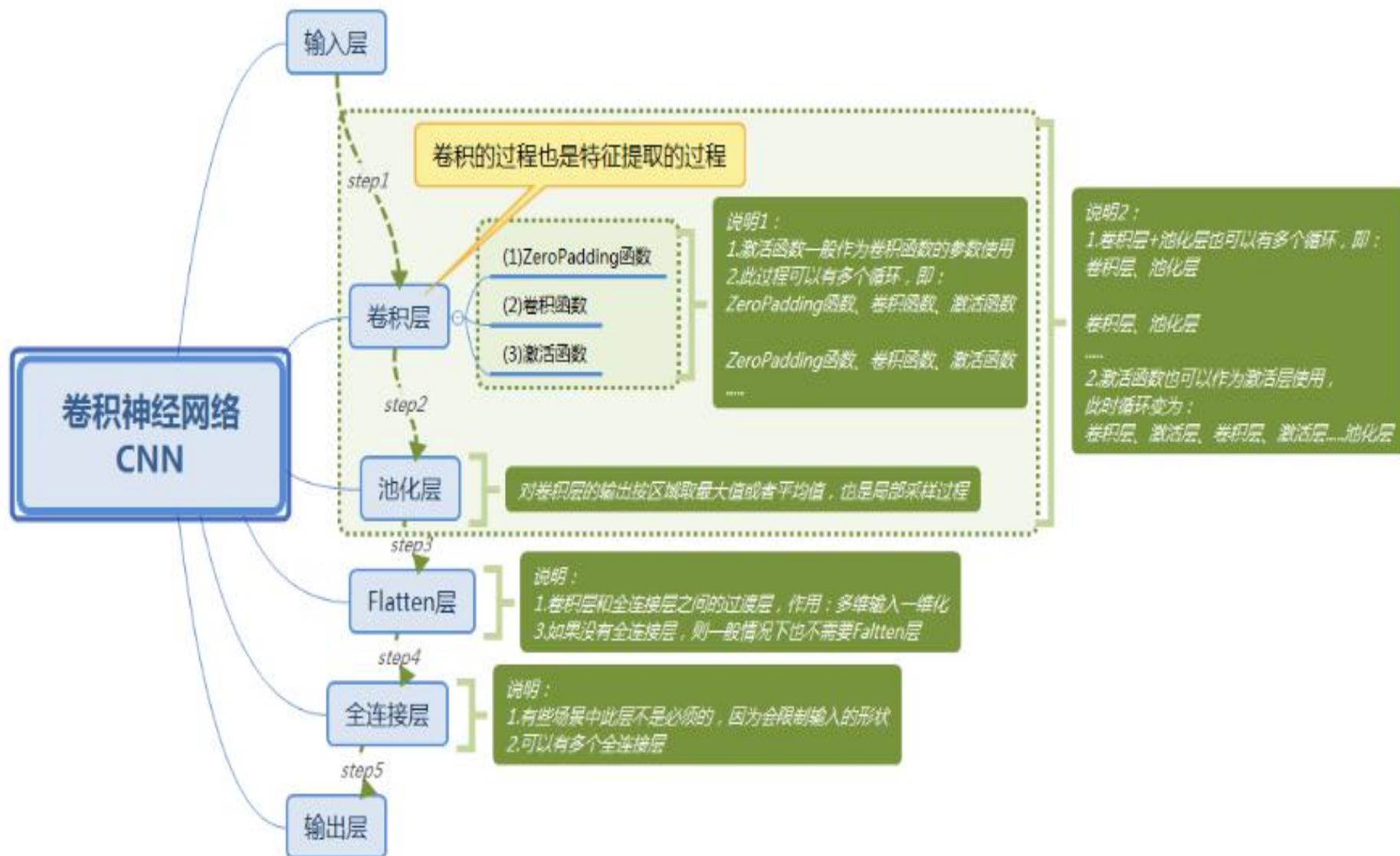
- 以上过程可以重复使用多层，最将终图片压缩为2*2大小
- 全连接：将图片展开对分类目标一一对应，最终得到的数字为概率



对于不太精确的样本进行归类：







卷积神经网络优缺点

□ 优点

- 共享卷积核，对高维数据处理无压力
- 无需手动选取特征，训练好权重，分类效果好

□ 缺点

- 需要调参，需要大样本量，训练最好要GPU
- 物理含义不明确（也就是说，我们并不知道每个卷积层到底提取到的是什么特征，而且神经网络本身就是一种难以解释的“黑箱模型”：无法给出理论上严格的解释）

循环神经网络：RNN

- 经典的神经网络：如 CNN，
所有的输出都是独立的，对于数据具有依赖性的，效果不太理想
解决了计算机的视觉问题，让机器具备视觉上识别的能力
- 缺陷：
单靠机器的视觉能力，并不能实现自主的智能，还有其它能力也很重要
- 例如，人类的分析能力
人类可以根据一个故事的开头猜到一个故事的结尾；
可以根据对方说的话，揣测他背后的目的；
智者往往处理事情有理有据，层次分明，
我们期待计算机也有这样的能力。
所以学者们设计了神奇的**循环神经网络**。

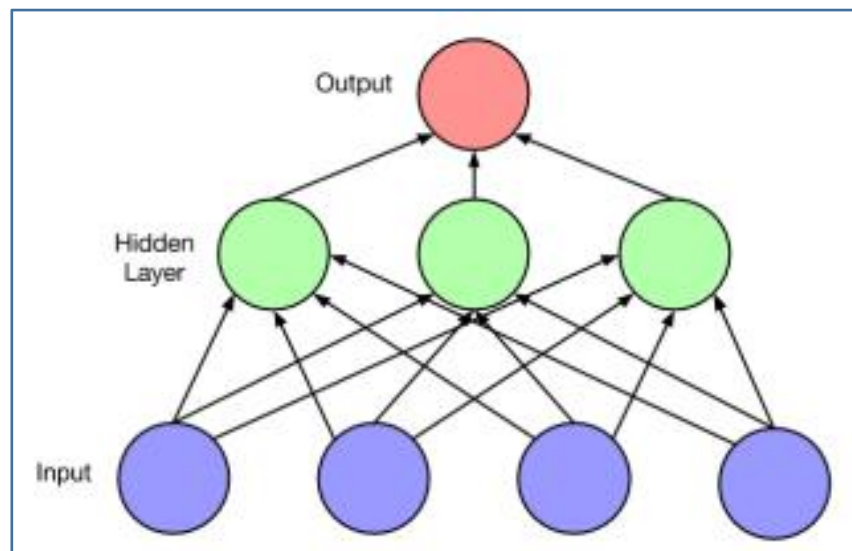
循环神经网络(Recurrent Neural Networks: RNN)

□ 循环神经网络：

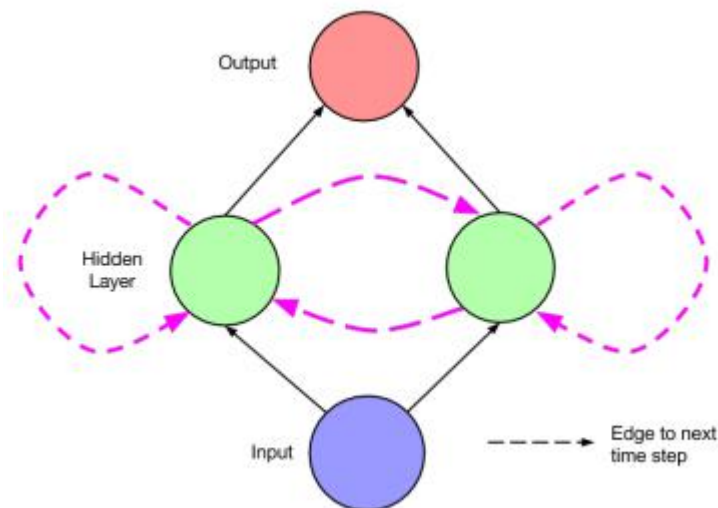
是一类以序列（sequence）数据为输入，在序列的演进方向进行递归（recursion）且所有节点按链式连接的递归神经网络

□ 目的：使机器具备分辨因果的能力，具有记忆功能

□ 应用：机器翻译系统，语音识别



CNN



RNN

简单RNN的结构

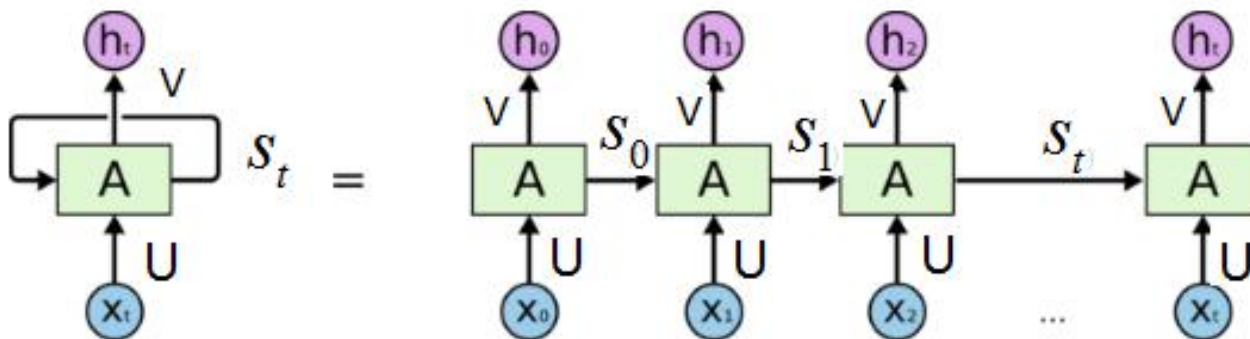
- 简单RNN：输入层，隐藏层、输出层
- 如下面左图所示，右图为对应展开图
- 输出层与隐藏层计算方法

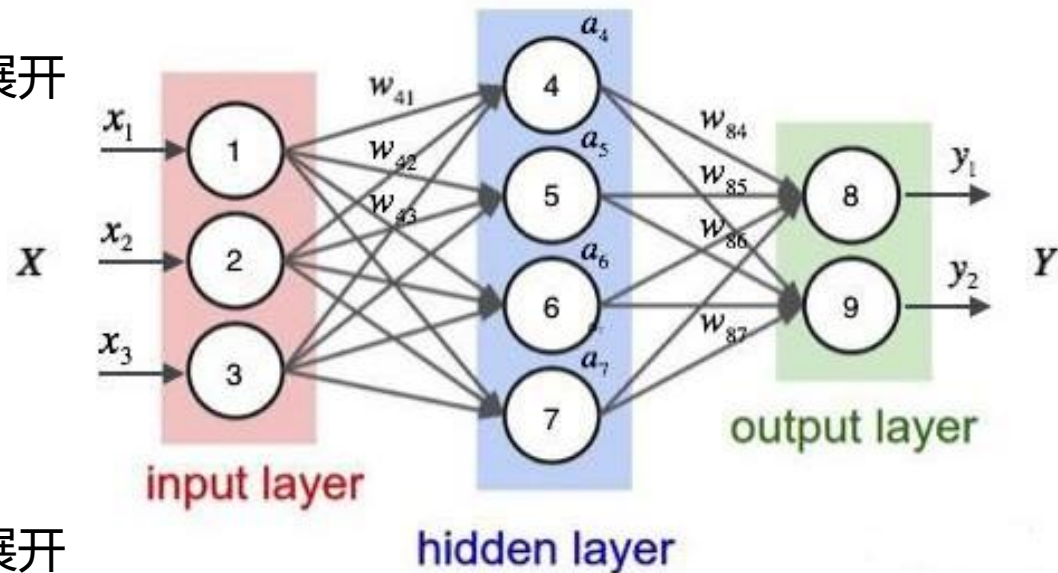
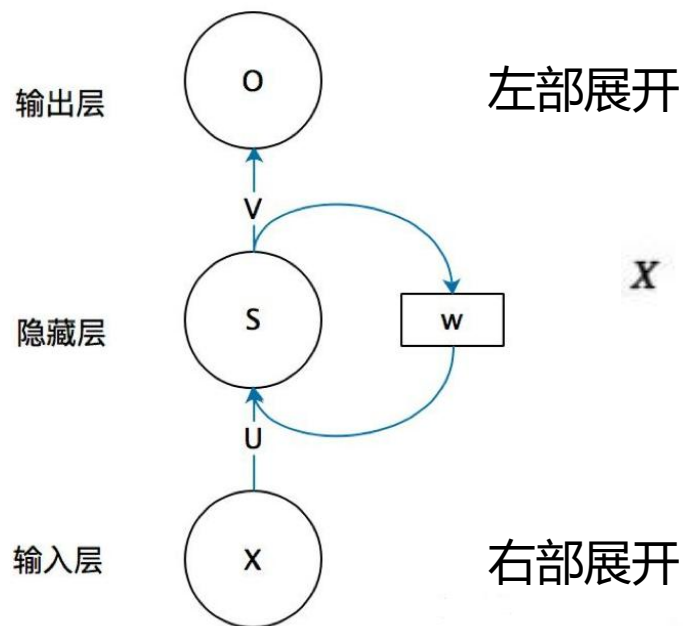
x_t : 输入向量
 h_t : 输出向量
 s_t : 隐藏层向量
 U, V, A : 权重矩阵

循环层： $s_t = f(Ux_t + Ws_{t-1})$
 输出层： $h_t = g(Vs_t)$,
 f, g 为激活函数

输出值 h_t 受前面的输入值影响 x_t

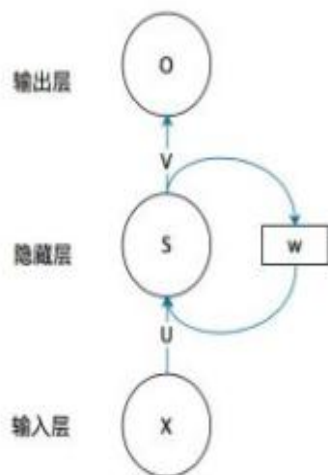
$$\begin{aligned}
 h_t &= g(Vs_t) = g(Vf(Ux_t + Ws_{t-1})) \\
 &= g(Vf(Ux_t + Wf(Ux_{t-1} + Wf(Ux_{t-2} + Wf(Ux_{t-3} + \dots))))
 \end{aligned}$$





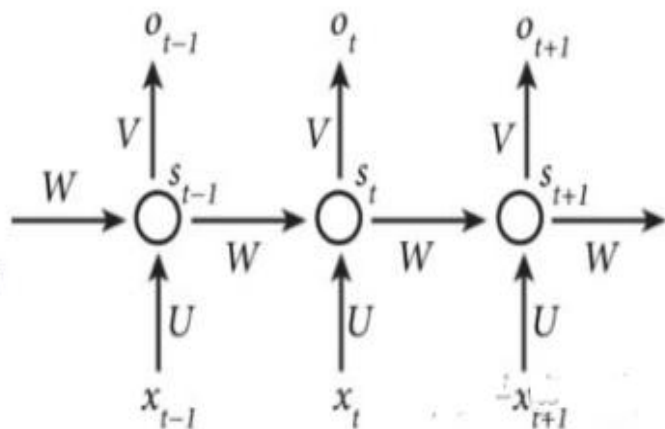
$$O_t = g(V \cdot S_t)$$

$$S_t = f(U \cdot X_t + W \cdot S_{t-1})$$



循环层

按照时间线展开



所有权重的值都为1且没有偏差

当我们输入第一个序列, 【1,1】

$$S_t = f(U \cdot X_t + W \cdot S_{t-1})$$

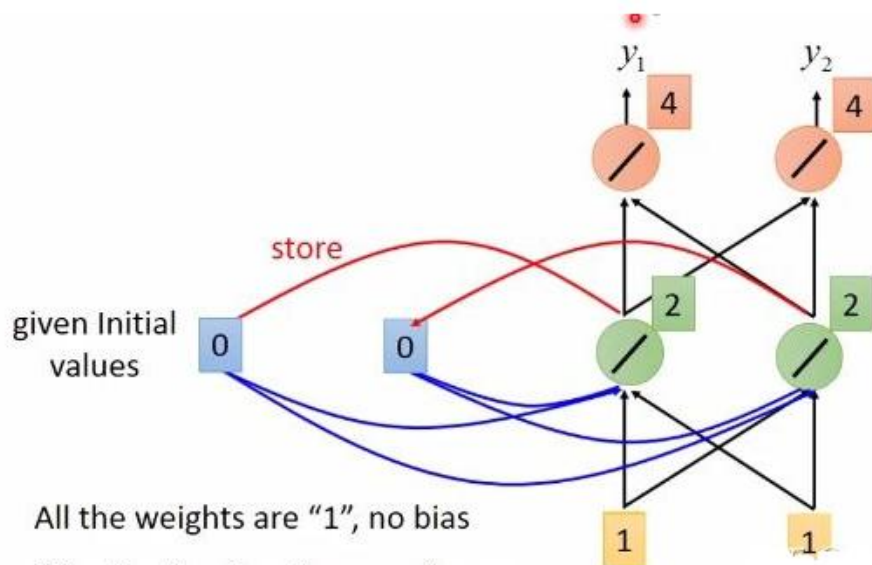
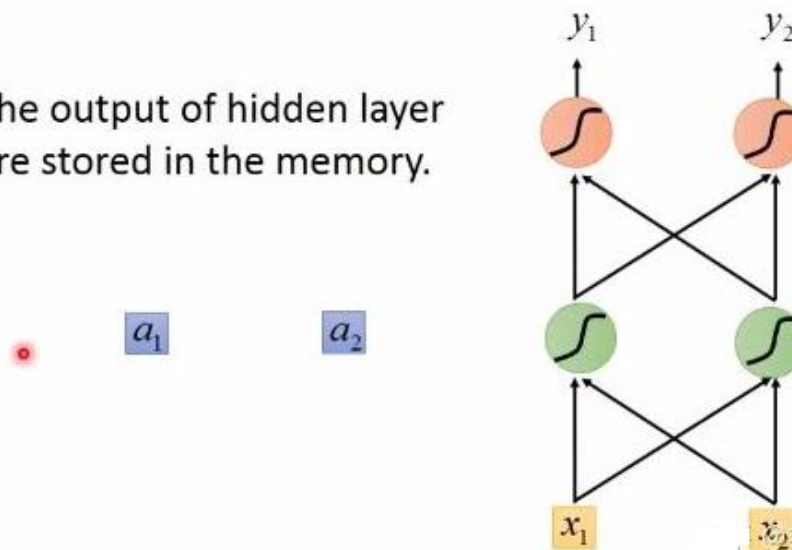
$$1 * 1 + 1 * 1 + 1 * 0 + 1 * 0 = 2$$

$$O_t = g(V \cdot S_t)$$

$$2 * 1 + 2 * 1 = 4$$

得到输出向量 **【4,4】**

The output of hidden layer are stored in the memory.



- All the weights are "1", no bias
- All activation functions are linear

输入下一个向量【1,1】

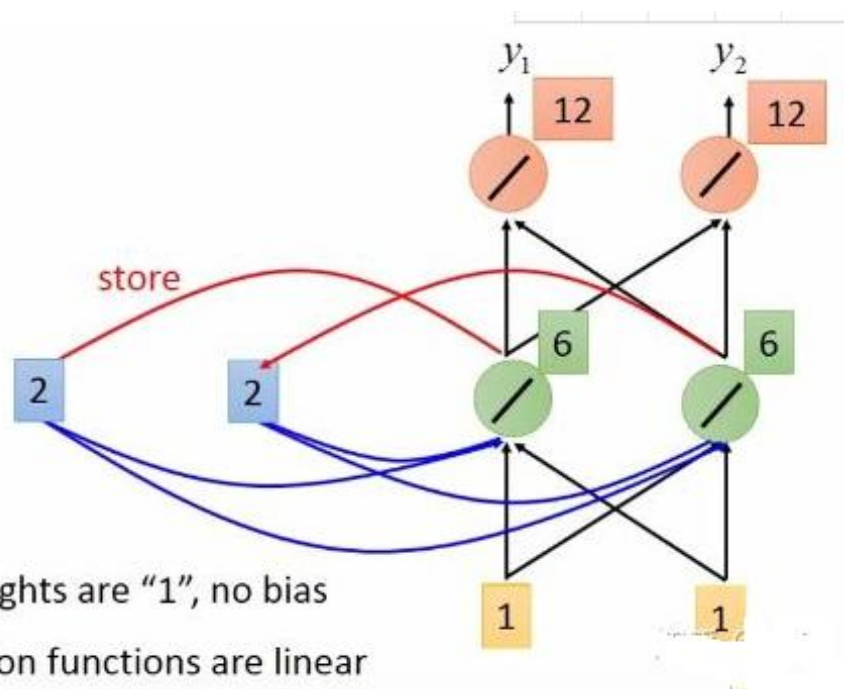
$$S_t = f(U \cdot X_t + W \cdot S_{t-1})$$

$$1 * 1 + 1 * 1 + 1 * 2 + 1 * 2 = 6$$

$$O_t = g(V \cdot S_t)$$

$$6 * 1 + 6 * 1 = 12$$

最终得到输出向量【12,12】

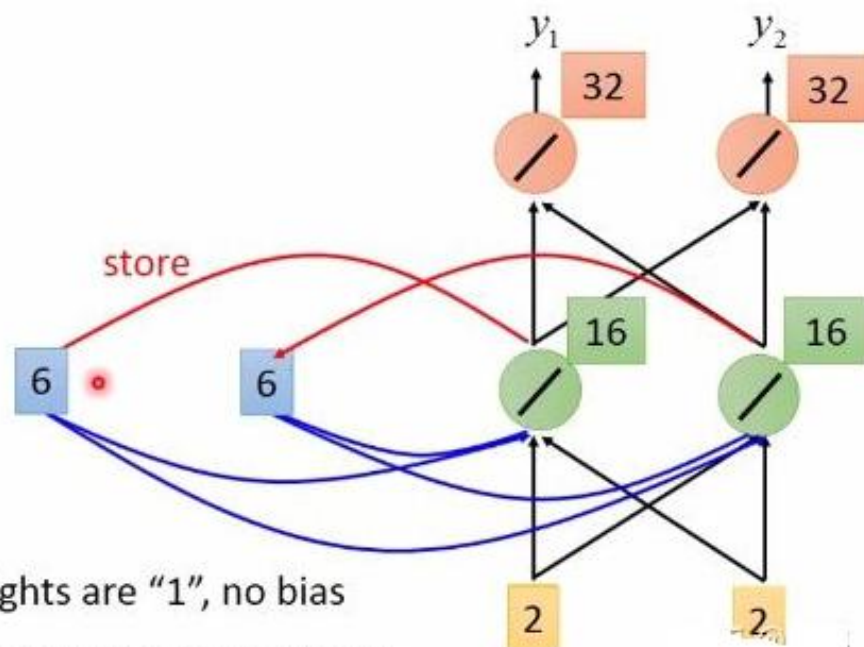


a_1, a_2 的值变成了6,

第三个向量【2,2】

得到输出向量【32,32】

output sequence: $\begin{bmatrix} 4 \\ 4 \end{bmatrix} \begin{bmatrix} 12 \\ 12 \end{bmatrix} \begin{bmatrix} 32 \\ 32 \end{bmatrix}$



All the weights are "1", no bias

All activation functions are linear

双向循环神经网络

- 隐藏层包含：正向计算与反向计算
- 计算方法为：

$$y_2 = g(VA_2 + V'A'_2)$$

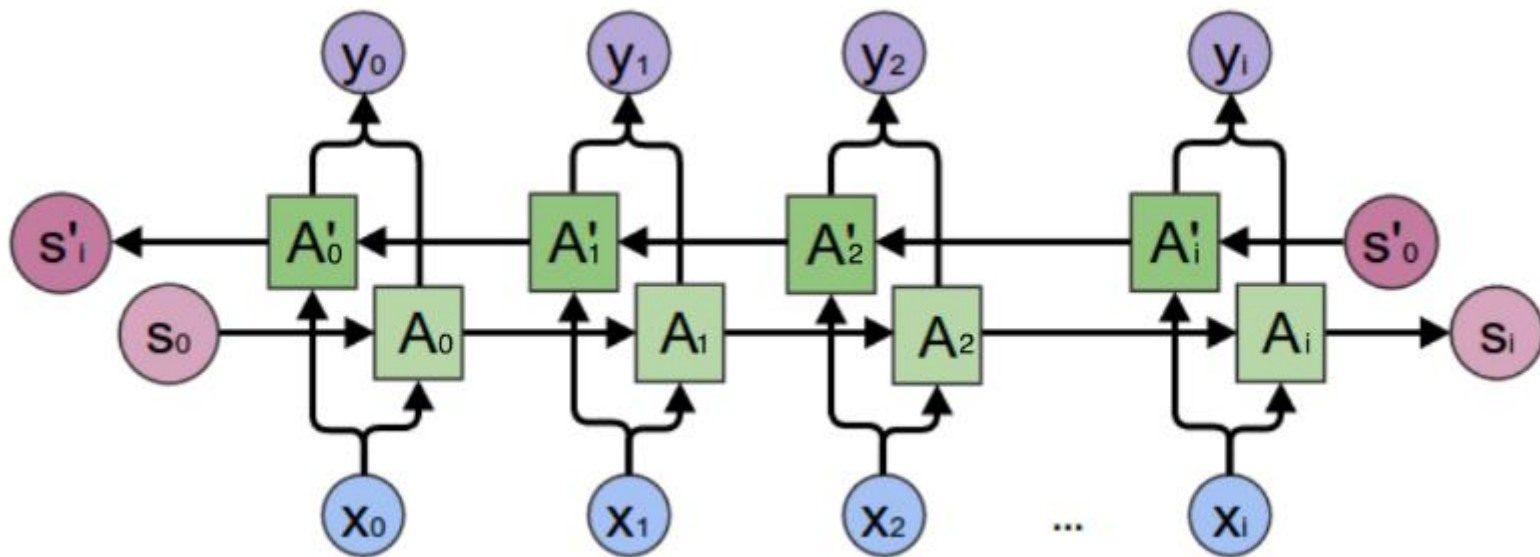
$$A_2 = f(WA_1 + Ux_2)$$

$$A'_2 = f(W'A'_3 + U'x_2)$$

$$y_t = g(Vs_t + V's'_t)$$

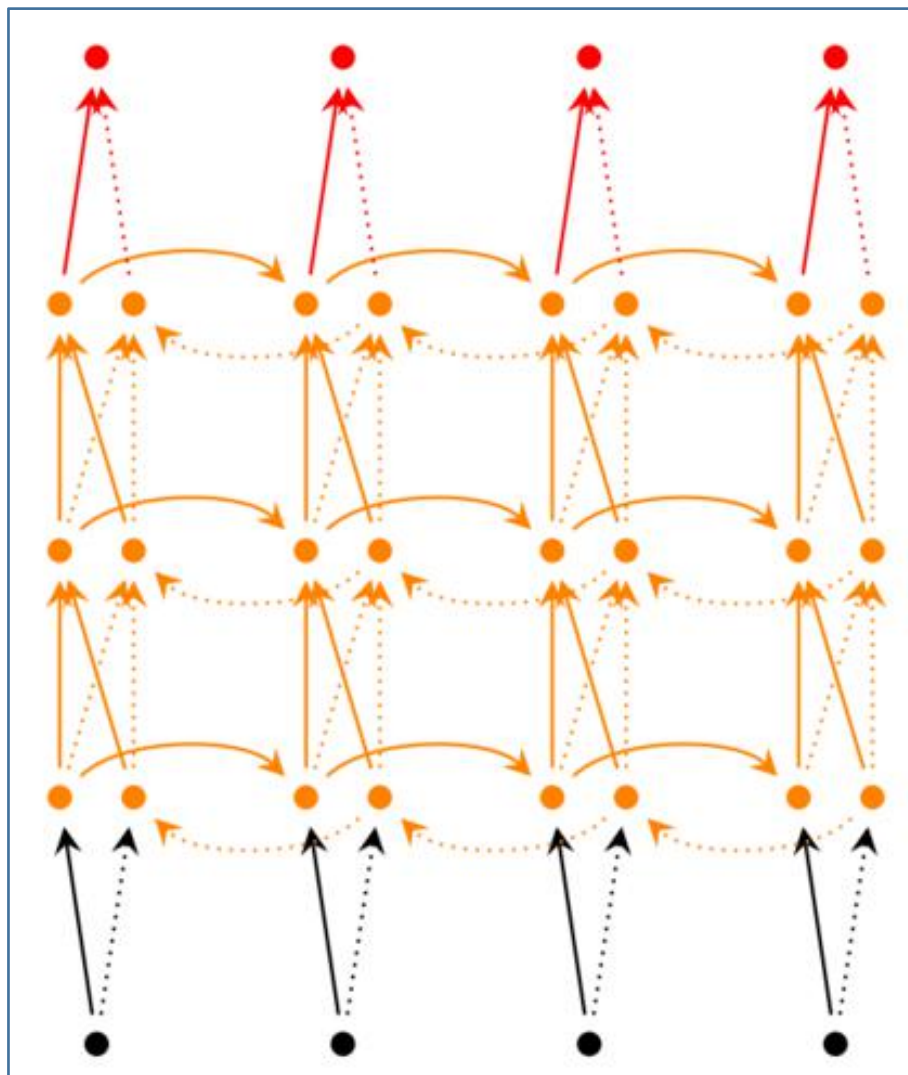
$$s_t = f(Ux_t + Ws_{t-1})$$

$$s'_t = f(U'x_t + W's'_{t+1})$$



深度循环神经网络计算方式为：

$$\begin{aligned}
 y_t &= g(V^{(i)} s_t^{(i)} + V'^{(i)} s_t'^{(i)}) \\
 s_t^{(i)} &= f(U^{(i)} s_t^{(i-1)} + W^{(i)} s_{t-1}^{(i)}) \\
 s_t'^{(i)} &= f(U'^{(i)} s_t'^{(i-1)} + W'^{(i)} s_{t+1}'^{(i)}) \\
 &\dots \\
 s_t^{(1)} &= f(U^{(1)} x_t + W^{(1)} s_{t-1}^{(1)}) \\
 s_t'^{(1)} &= f(U'^{(1)} x_t + W'^{(1)} s_{t+1}'^{(1)})
 \end{aligned}$$



2. 生成对抗网络：GAN，背景分析

- 人类除了识别与分析能力，还具备创造能力；
- 使机器具备创造的能力
 - ①通过学习过去的文章，训练一个可以撰写文章人工智能作者
 - ②可不可以创造一个人工智能画家，通过从画家过去的作品中学习，然后像任何艺术家一样画画？
- 这些任务此前确实难以自动化，GAN已经使部分任务变成可能
- 所以简单来说，GAN，要获得一个强大的英雄（即生成器generator），需要一个更强大的对手（即鉴别器discriminator）。

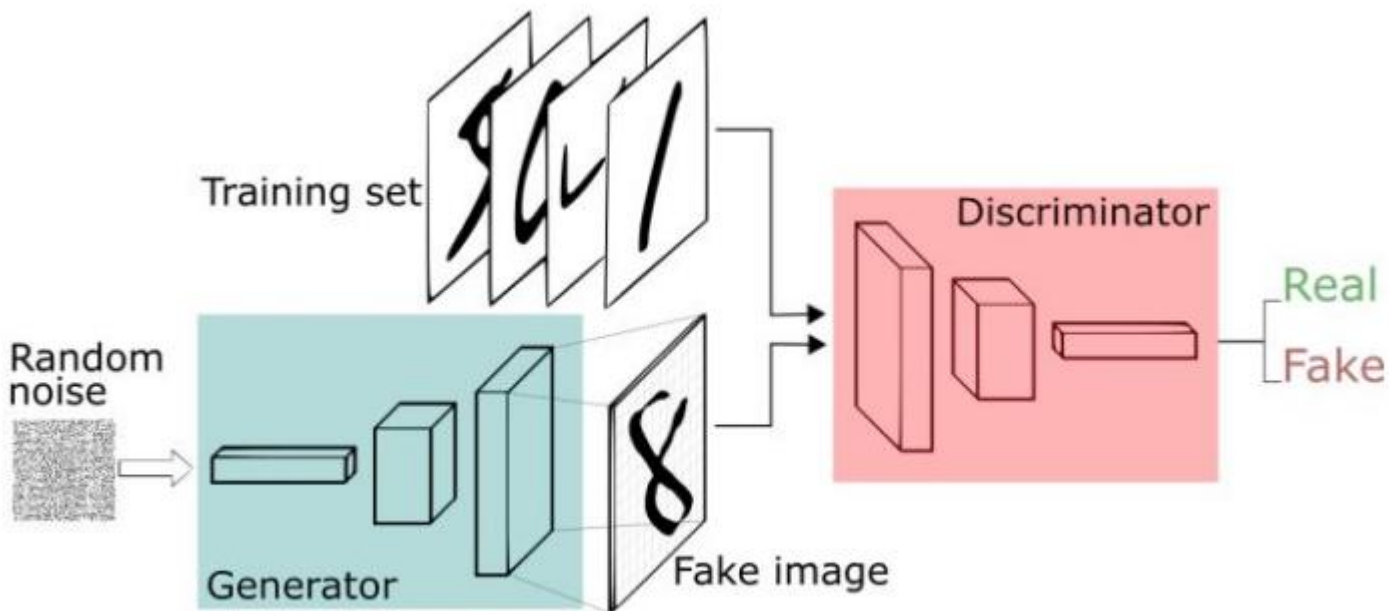
生成对抗网络：GAN

- GAN: Generative Adversarial Networks, 是一种博弈思想;
- 包含两种结构:

生成式模型G (generative model), 判别式模型D (discriminative model)

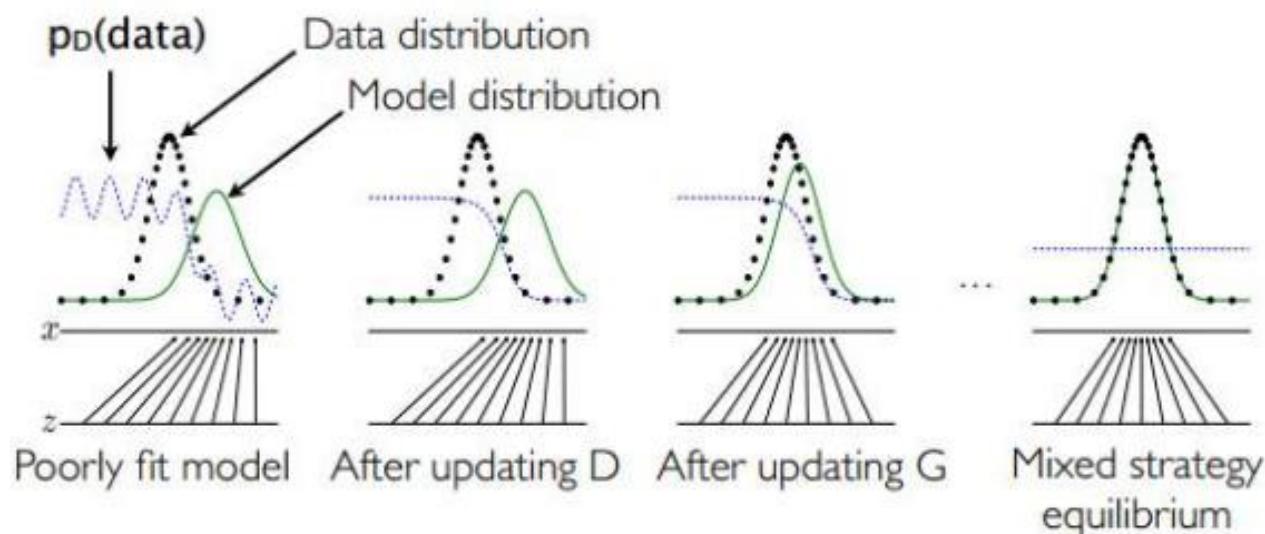
①G是生成图片网络, 接收随机噪声 z , 通过这个噪声生成图片, 记做 $G(z)$ 。

②D是判别网络, 判别一张图片是不是“真实”。



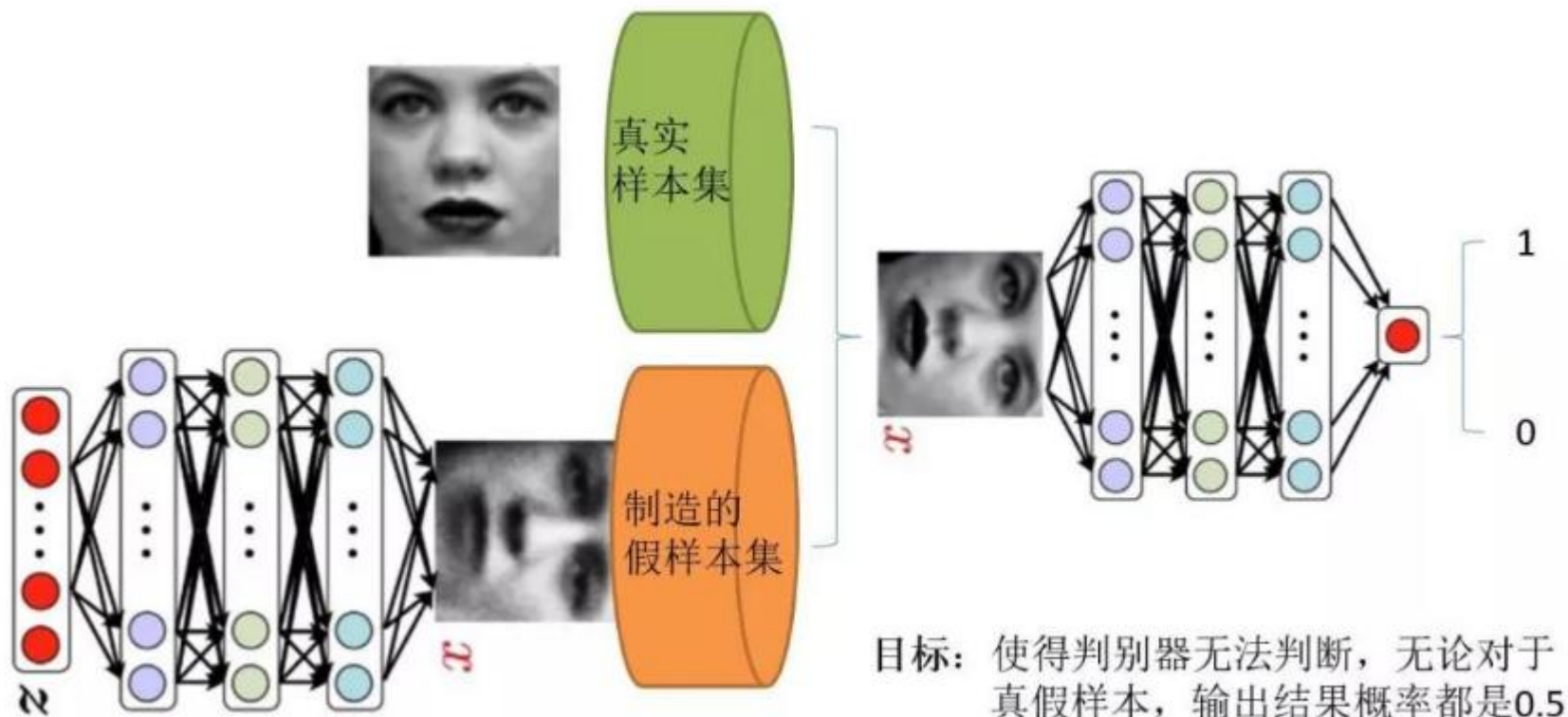
图三 GAN基本结构

- 训练过程中:
G的目标就是尽量生成真实的图片去欺骗判别网络D。
D的目标就是尽量把G生成的图片和真实的图片分别开来,
- G和D构成了一个动态的 **“博弈过程”**。
- 最后博弈的结果是: G可以生成足以“以假乱真”的图片G(z)。



注：图中的**黑色虚线**表示真实的样本的分布情况，**蓝色虚线**表示判别器判别概率的分布情况，**绿色实线**表示生成样本的分布。 z 表示噪声， z 到 x 表示通过生成器之后的分布的映射情

- ▣ 判别网络(下图右半部分)
- ▣ 生成网络(下图左下部分)



目标：使得判别器无法判断，无论对于真假样本，输出结果概率都是0.5

<http://blog.csdn.net/on2way>

- GAN数学优化问题,

$$\min_G \max_D V(D, G) = E_{x \sim p_{data}(x)} [\log(D(x))] + E_{z \sim p_z(z)} [\log(1 - D(G(z)))]$$

- 以上问题交替迭代求解,

优化D:

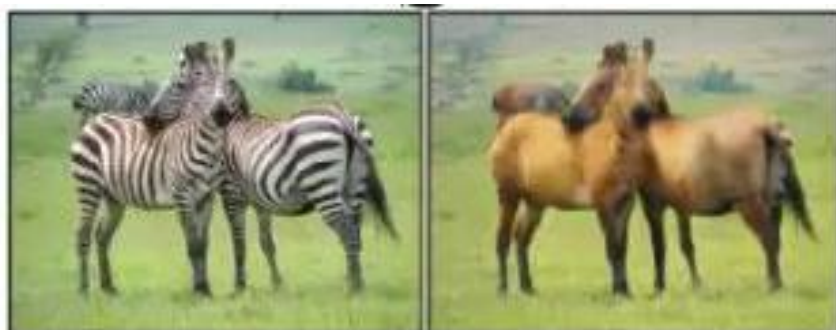
$$\max_D V(D, G) = E_{x \sim p_{data}(x)} [\log(D(x))] + E_{z \sim p_z(z)} [\log(1 - D(G(z)))]$$

优化G:

$$\min_G V(D, G) = E_{z \sim p_z(z)} [\log(1 - D(G(z)))]$$

□ 图像生成:

「输入」满足一个输入分布, 「输出」满足一个预期的期望 分布
学习这两种图像之间的映射



zebra → horse



summer → winter



卷积神经网络: **CNN**

卷积核学习特征，让机器具备视觉上的识别能力

循环神经网络: **RNN**

让机器具备分辨因果的能力和记忆功能，序列问题

生成对抗网络: **GAN**

让机器具备创造能力