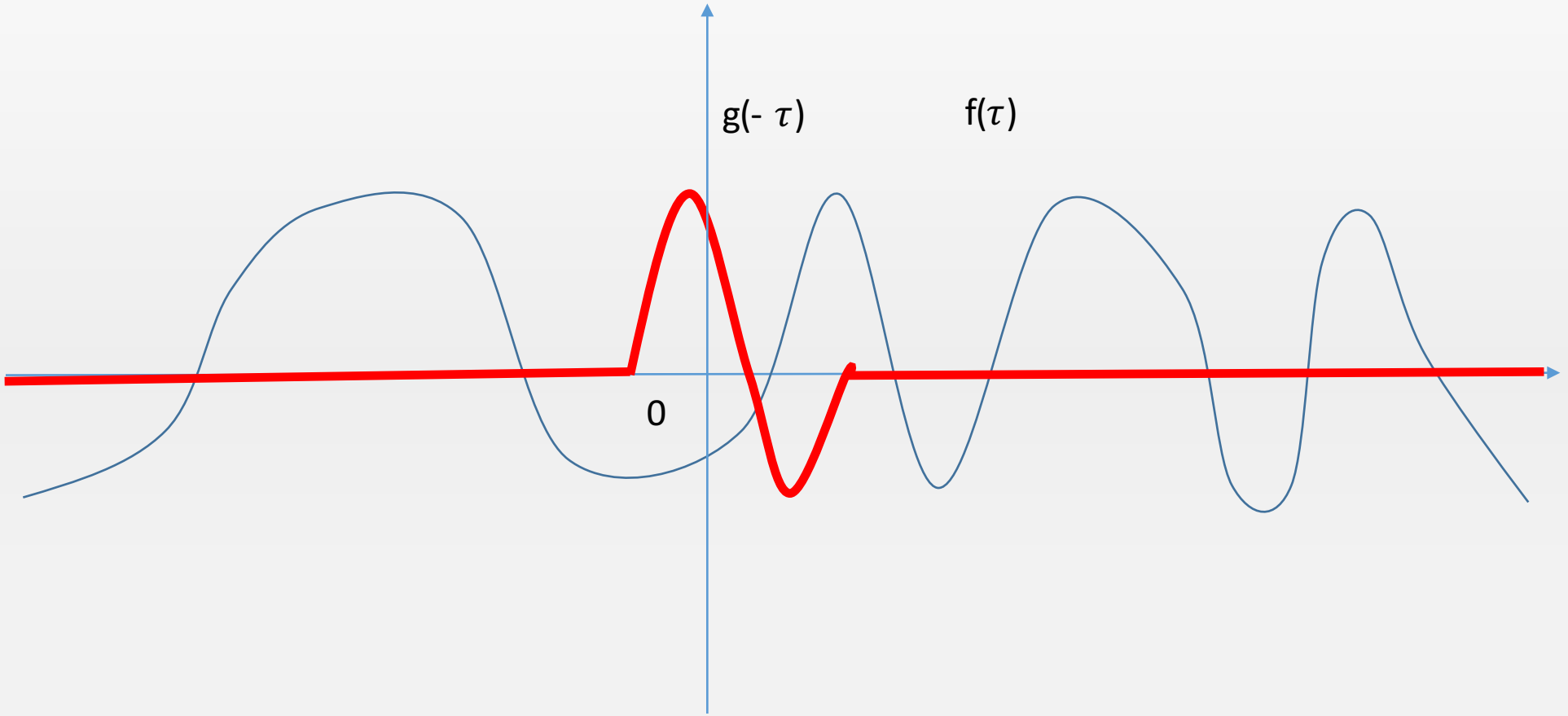
卷积 反传

与交叉熵

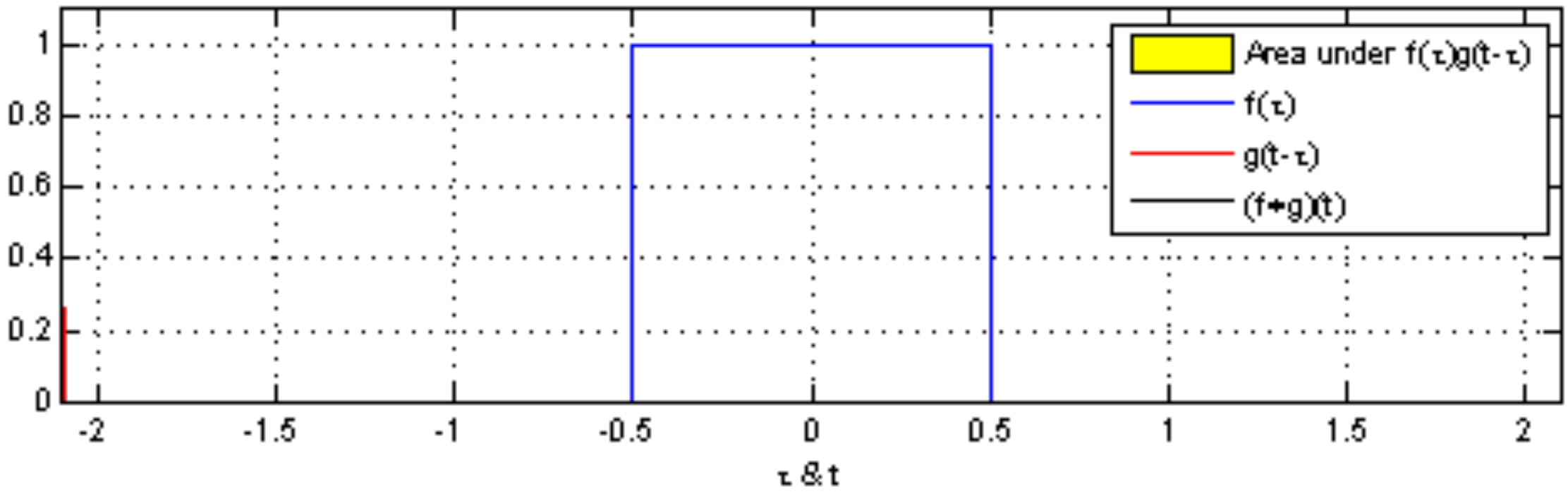
张江

一维的卷积操作



$$(f * g)(t) = \int_{-\infty}^{+\infty} f(\tau)\, g(t-\tau) dt$$

什么是深度学习？



$$(f * g)(t) = \int_{-\infty}^{+\infty} f(\tau)\, g(\tau - t) d\tau$$

连续→离散

$$(f * g)(t) = \int_{-\infty}^{+\infty} f(\tau) g(t-\tau) d\tau$$

$$(f * g)(x) = \sum_{t=-a}^{a} f(t)\, g(x-t) = \sum_{t=-a}^{a} f_t g_t^x = \boldsymbol{f} \cdot \boldsymbol{g^x}$$

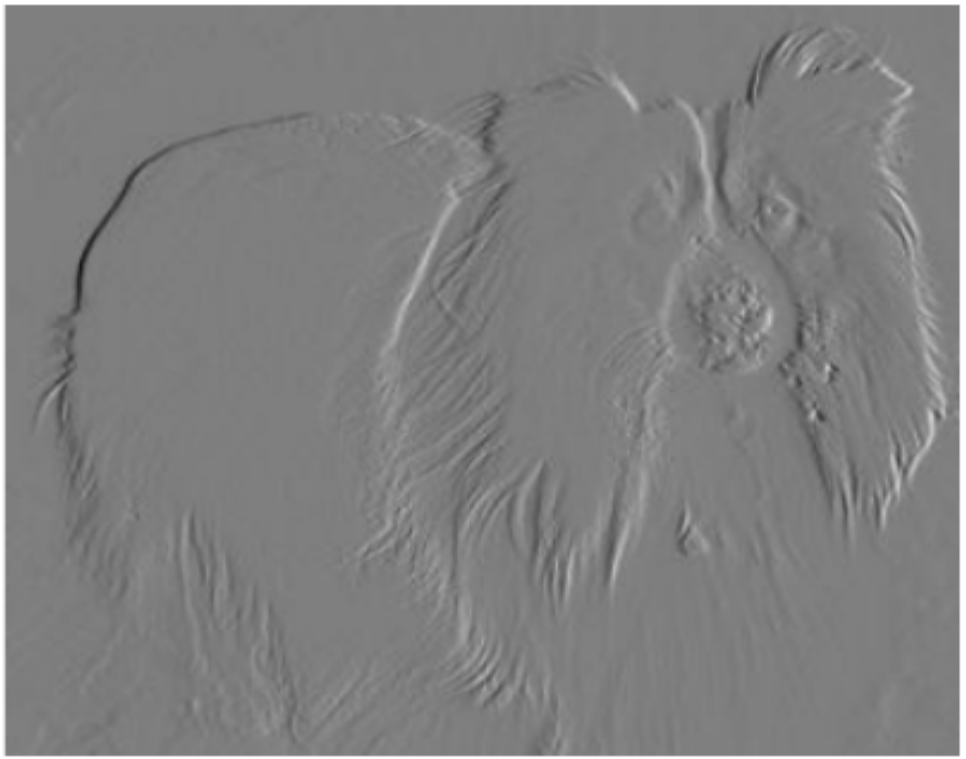
二维情况

$$(f * g)(t) = \sum_{t=-a}^{a} f(\tau) g(t - \tau)$$

$$(f * g)(x, y) = \sum_{u} \sum_{v} f(u, v) g(x - u, y - v)$$

边缘检测

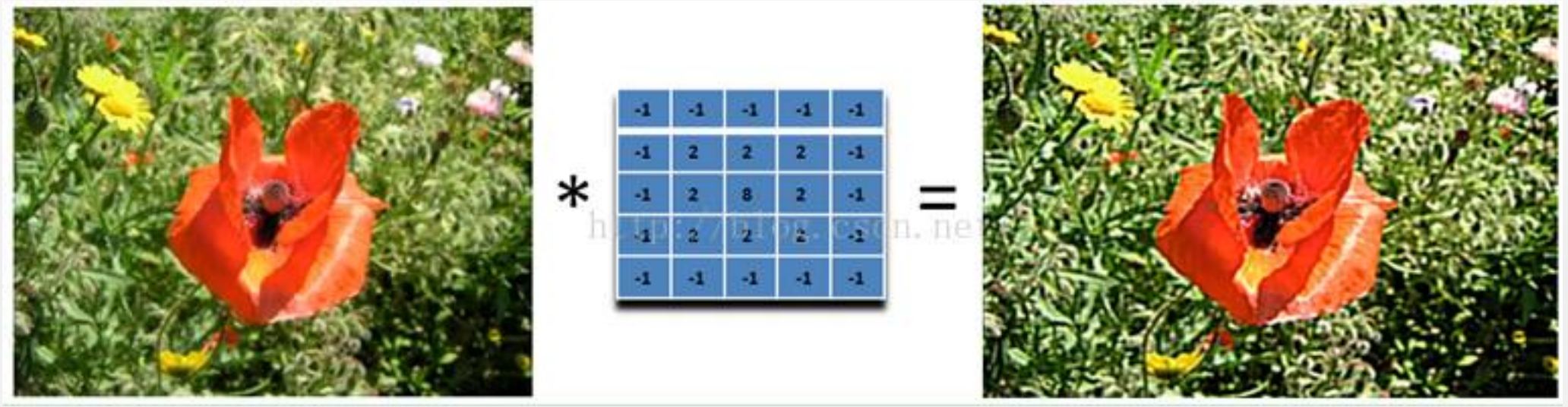
$$(f * g)(x,y) = \sum_u \sum_v f(u,v)g(x-u, y-v)$$



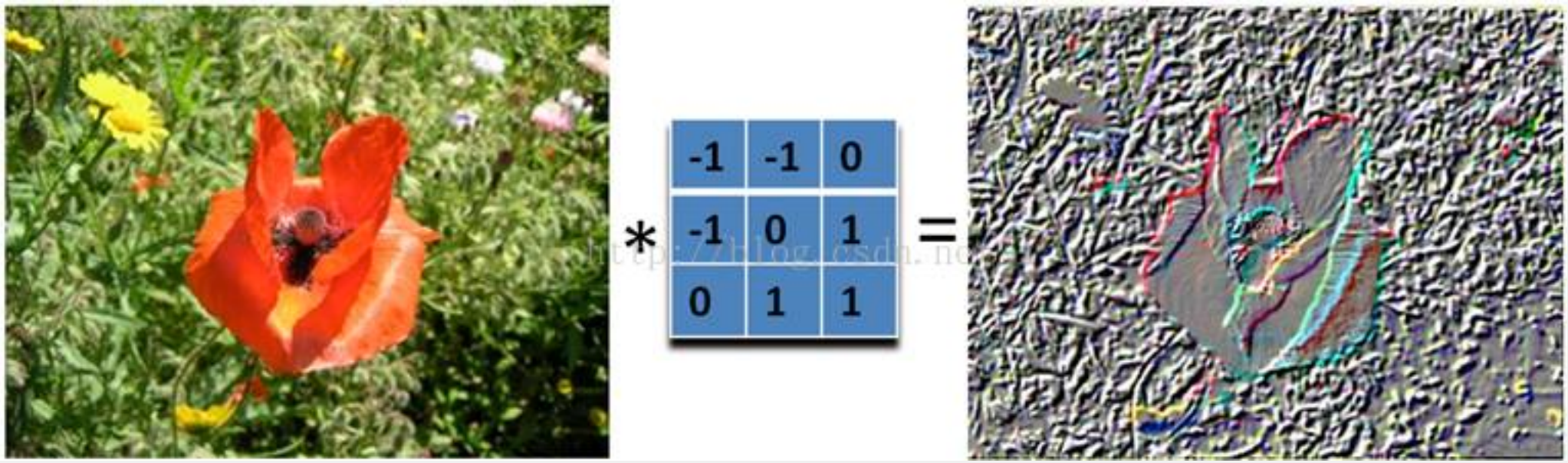
$$g(x,y)\begin{cases} 1, & x=0, y=0 \\ -1, & x=1, y=0 \\ 0, & otherwise \end{cases}$$

f(x,y)-f(x-1,y)

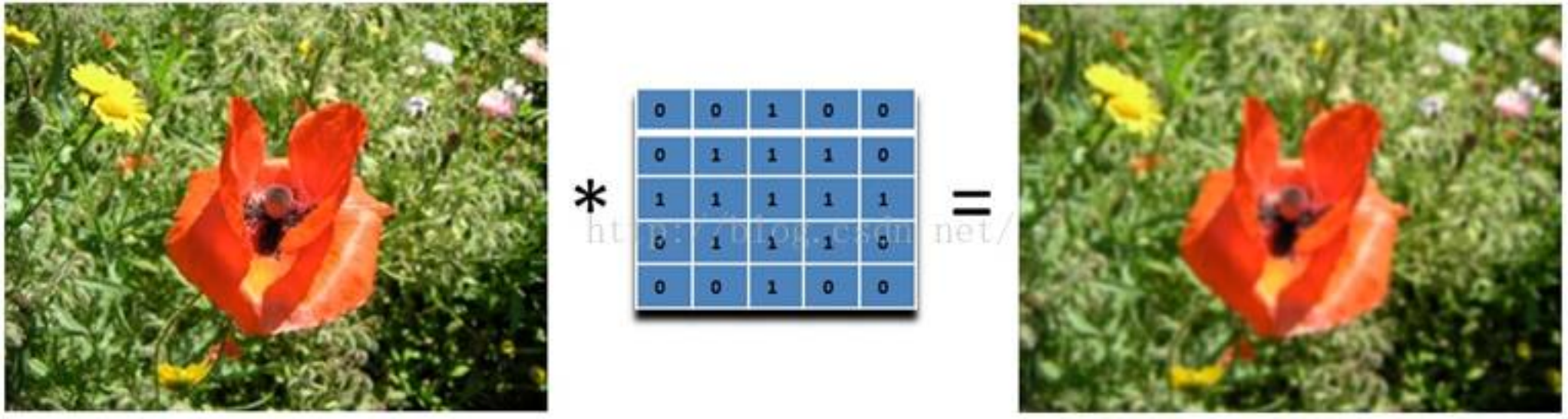
锐化



浮雕



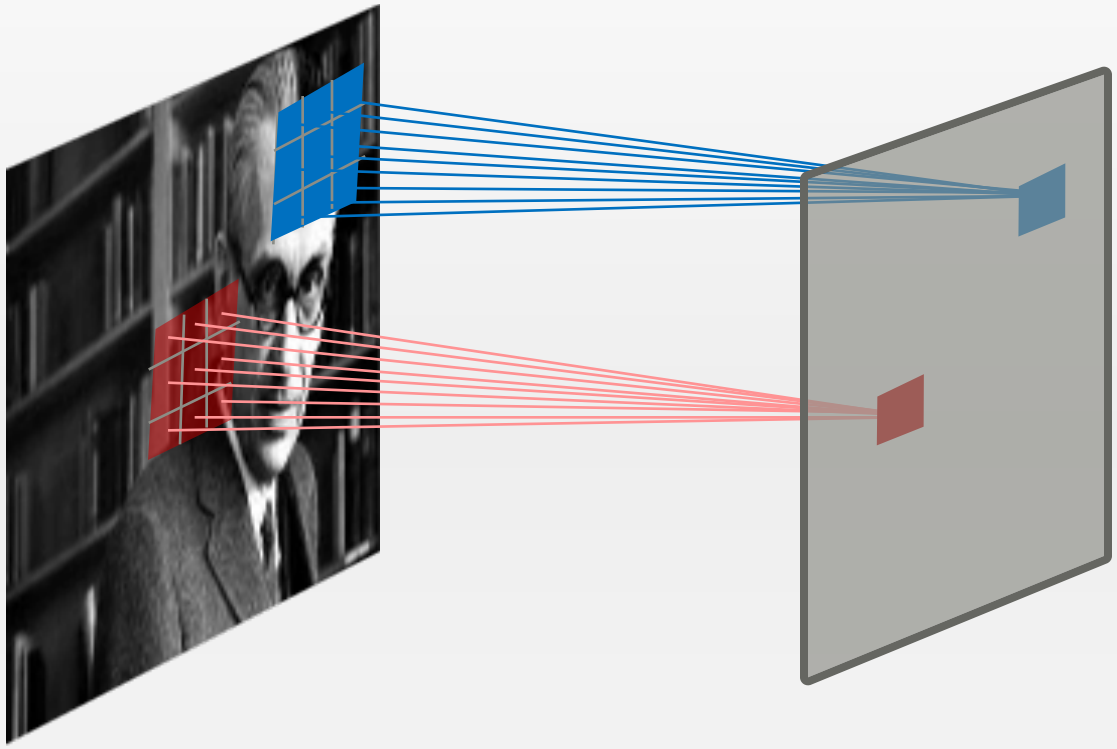
模糊



运动模糊

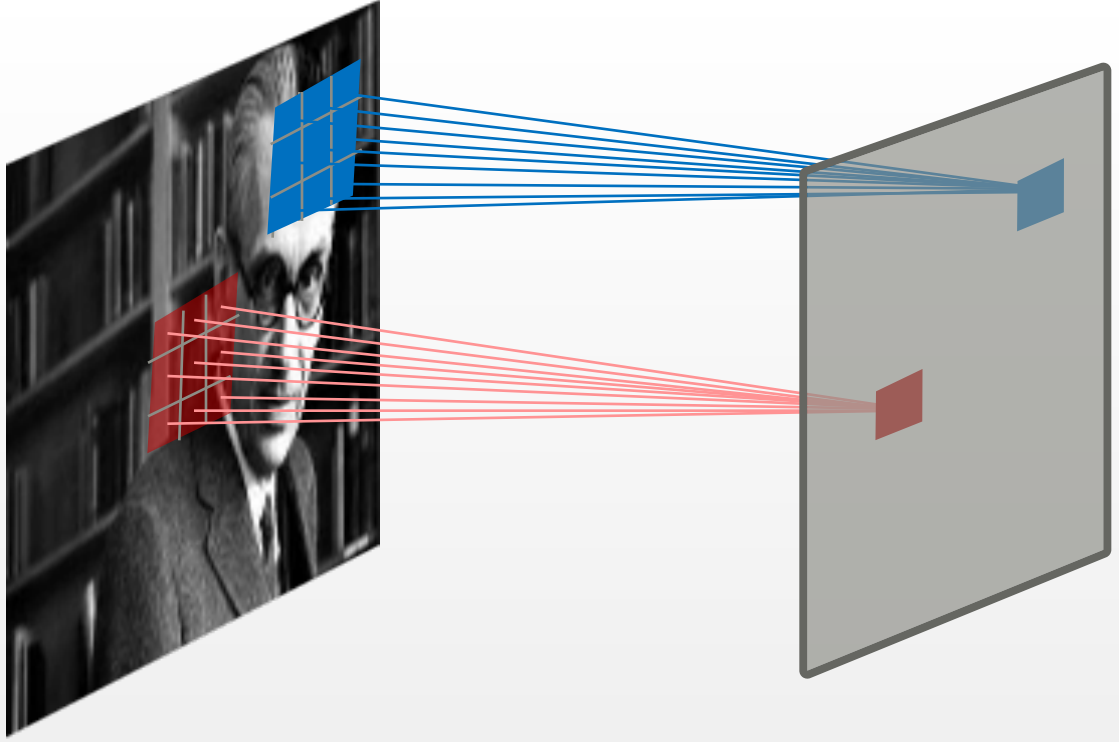
```
1, 0, 0, 0, 0, 0, 0, 0, 0
0, 1, 0, 0, 0, 0, 0, 0, 0
0, 0, 1, 0, 0, 0, 0, 0, 0
0, 0, 0, 1, 0, 0, 0, 0, 0
0, 0, 0, 0, 1, 0, 0, 0, 0
0, 0, 0, 0, 0, 1, 0, 0, 0
0, 0, 0, 0, 0, 0, 1, 0, 0
0, 0, 0, 0, 0, 0, 0, 1, 0
0, 0, 0, 0, 0, 0, 0, 0, 1
```

权值共享



蓝色和红色连边对应位置的权重值相等

手动实现卷积操作



```
input = torch.randn([64, 1, 28, 28])
conv1 = nn.Conv2d(1, 1, 5, padding = 2)
output = conv1(Variable(input))
```

```
# 定义权重张量
weights = torch.zeros(32, 32, 28, 28)# 根据卷积核以及权重共享准则，给手动的权重张量赋值
for i in range(28):
    for j in range(28):
        weights[i:i+5,j:j+5,i,j] = conv1.weight[0,0,...].data      # 将原始的64张28*28的图片扩充为32*32
image = torch.zeros(64, 1, 32, 32)
image[:, :, 2:30, 2:30] = input # 将乘积的维度设为1维
image_in = image.view(image.size()[0], -1)
weights_p = weights.view(32 * 32, 28 * 28)# 计算卷积，此时等价为矩阵乘积
output2 = image_in.mm(weights_p) + conv1.bias.data.expand(image.size()[0], 28 * 28)# 将结果展现为64张图片
output2 = output2.view(output2.size()[0], 1, 28, 28)
```

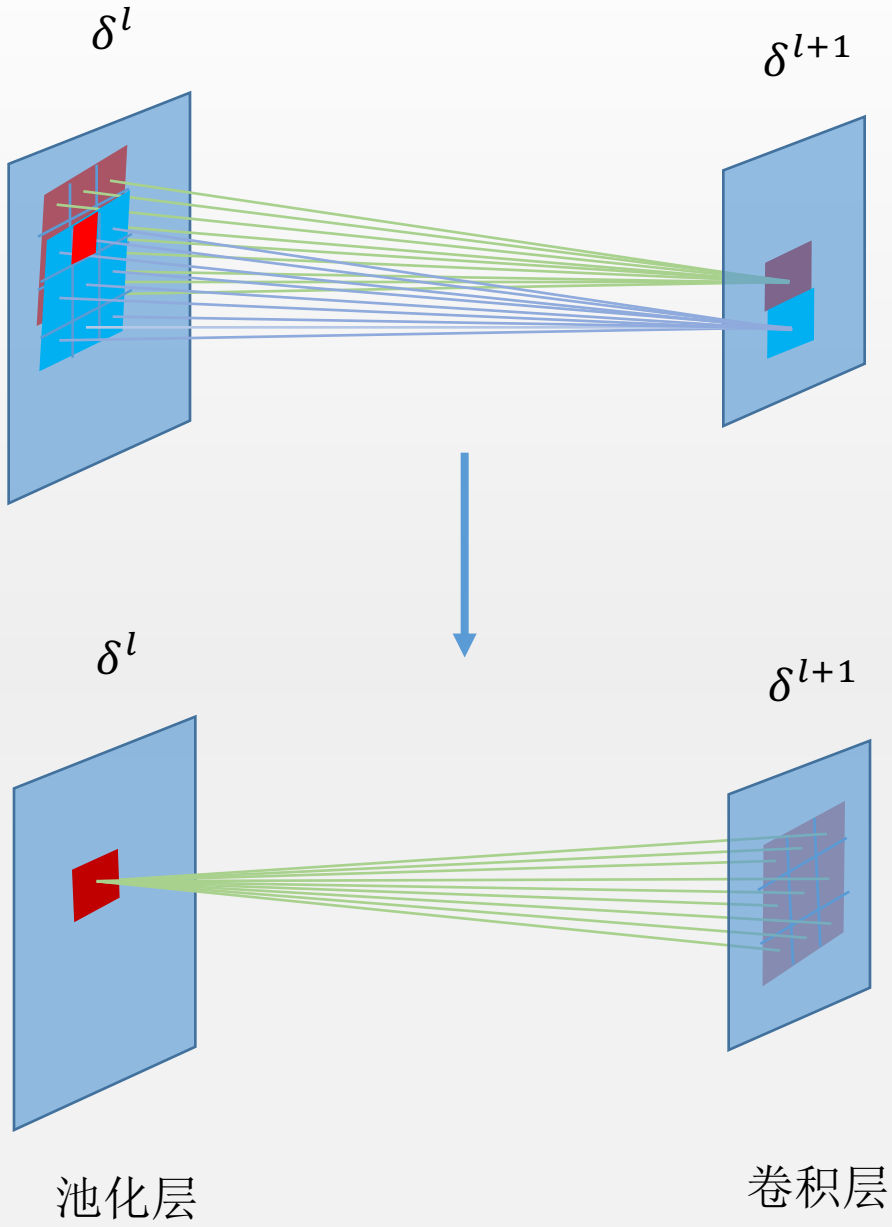
BP算法过程

- 计算每一层，每个神经元的误差
- 更新每一层神经链接的权重

反向传播：池化层

- 如果仅有一个feature map
- 每个池化层像素都与3*3个卷积层相连
- 故而

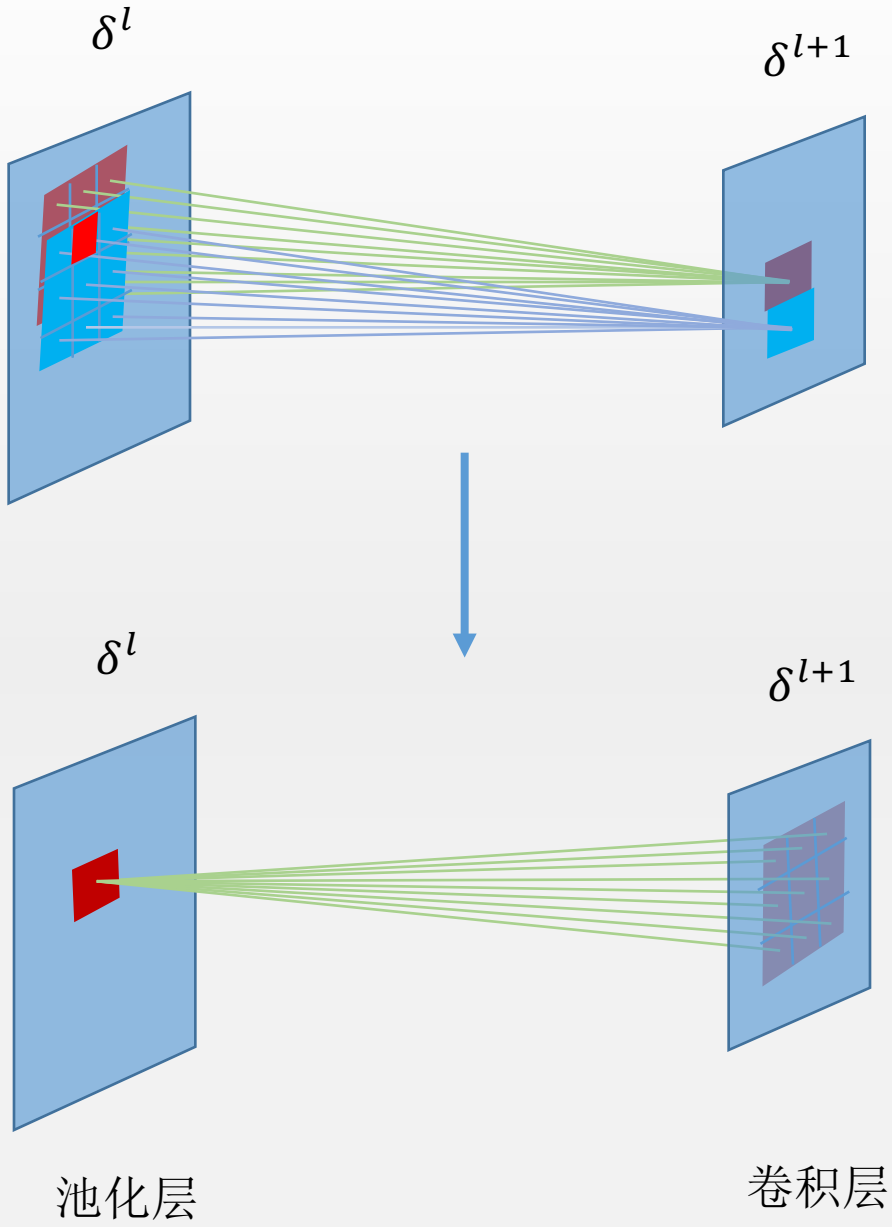
$$\delta^l = rot(K, \pi) * \delta^{l+1}$$



反向传播：池化层

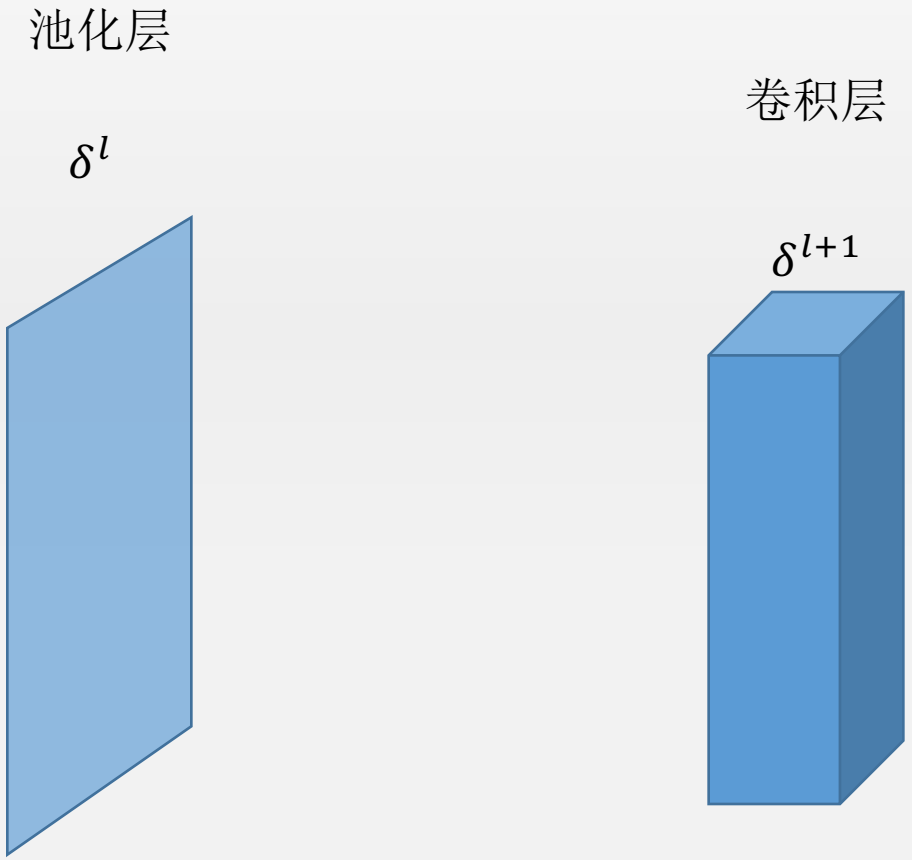
$$x_{i,j}^{l+1} = w_1\ x_{i-1,j-1}^{l} + w_2\ x_{i,j-1}^{l} + w_3\ x_{i+1,j-1}^{l}$$
$$w_4\ x_{i-1,j}^{l} + w_5\ x_{i,j}^{l} + w_6\ x_{i+1,j}^{l}$$
$$w_7\ x_{i-1,j+1}^{l} + w_8\ x_{i,j+1}^{l} + w_9\ x_{i+1,j+1}^{l}$$

$$\delta_{i,j}^{l} = w_9\ \delta_{i-1,j-1}^{l+1} + w_8\ \delta_{i,j-1}^{l+1} + w_7\ \delta_{i+1,j-1}^{l+1}$$
$$w_6\ \delta_{i-1,j}^{l+1} + w_5\ \delta_{i,j}^{l+1} + w_4\ \delta_{i+1,j}^{l+1}$$
$$w_1\ \delta_{i-1}^{l+1} + w_2\ \delta_{i,j+1}^{l+1} + w_1\ \delta_{i+1,j+1}^{l+1}$$



反向传播： 卷积层

- 如果池化层有多个featuremaps
- 则每个featuremap都单独计算一个卷积层的误差δ^l
- 然后将所有的δ^l加起来。



反向传播：卷积层

- 每一个神经元像素都有一个误差值
- 计算卷积层的误差
  - 将池化层任意像素的误差
  - 平均分配给其所对的所有像素

前馈，平均池化：

$$x_{i,j}^{l+1} = \frac{1}{w^2} \sum_{x=-1,0,1} \sum_{y=-1,0,1} x_{i+x,j+y}^{l}$$

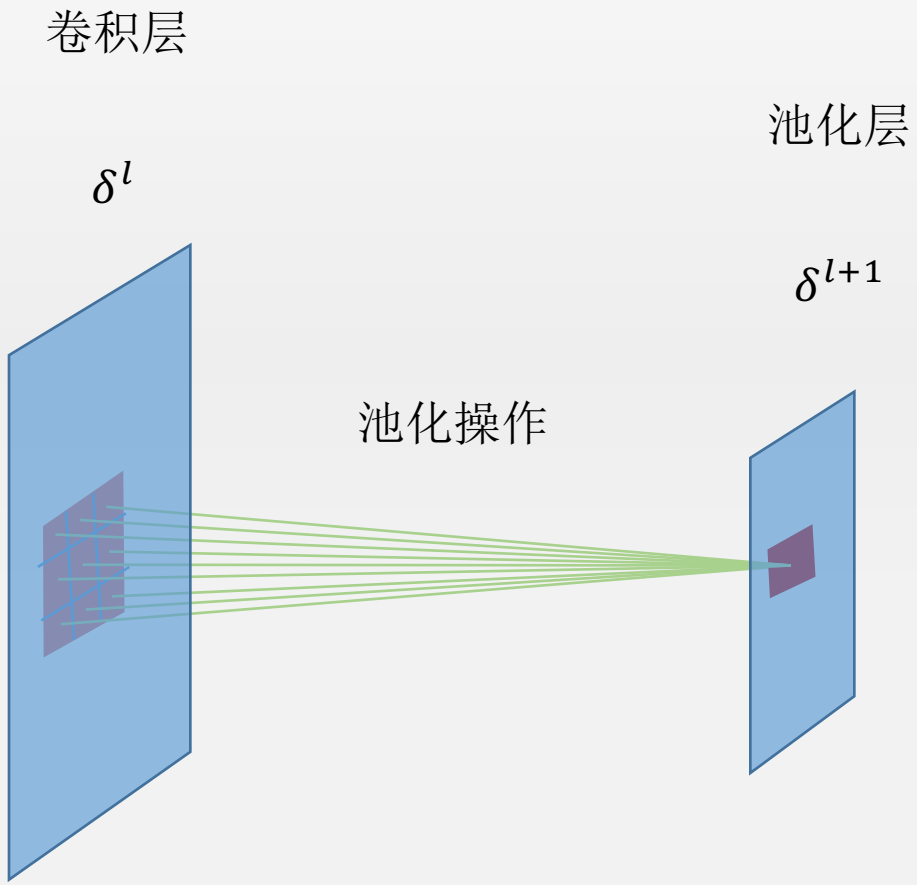
反向，平均池化的误差计算：

$$\delta_{i,j}^{l} = \frac{\delta_{i,j}^{l+1}}{w^2}$$

反向，最大池化的误差反传：

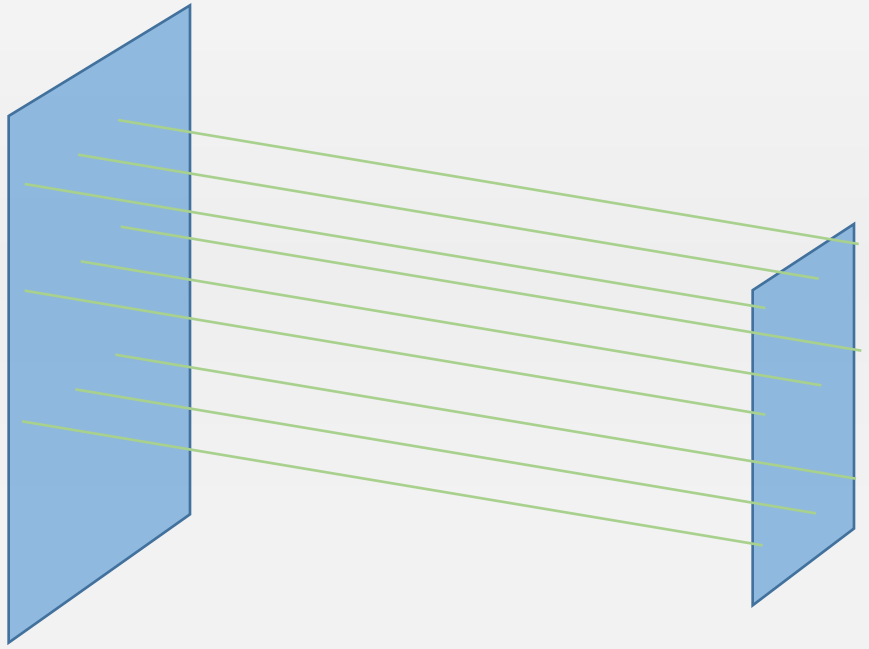
$$\delta_{max}^{l} = \delta_{i,j}^{l+1}$$

正向最大的像素获得误差



权值更新

- 与一般的神经网络不同，卷积神经网络需要考虑权值共享
- 对任意一个权重的更新等于所有共享该权值的像素对的权值更新之和
- 对于任意一个像素对，权值更新与普通神经网络相同



交叉熵

- 原始定义：

$$H(p, q) = -\sum_{x} p(x) \log q(x).$$

$$H(p, q) = \mathrm{E}_p[-\log q] = H(p) + D_{\mathrm{KL}}(p \| q),$$

交叉熵

- 在分类问题中，我们假设根据已有的特征数据F得到数据的分类X={x1,x2,...xn}。于是神经网络在学习一个条件概率Pr{X|F}。
- 我们设真实的条件概率为函数p(X)=Pr{X|F}，模型给出的估计为q(X)=Pr'{X|F}，则H(p,q)度量了两者的相似度。
- 但是由于p(X)无法直接写出来，故而我们需要用频率逼近概率，也就是p(x)~n(x|f)/N，其中n(x|f)为在特征f下类别x出现的次数，N为总样本数。
- 我们知道

$$n(x|f) = \sum_{i=1}^{N} I_i(x|f) \qquad I_i(x|f) = \begin{cases} 1 & \text{当样本i的类别为x的时候} \\ 0 & \text{当样本i的类别不为x的时候} \end{cases}$$

$$H(p,q) = -\sum_{i=1}^{N}\sum_{x} I_i(x|f)\log q(x) = -\sum_{i=1}^{N}\log q(x = l_i)$$

l_i是第i个数据的标签