# 深度网络基本功

## 第二讲

## Neural Network and CNN 计算原理

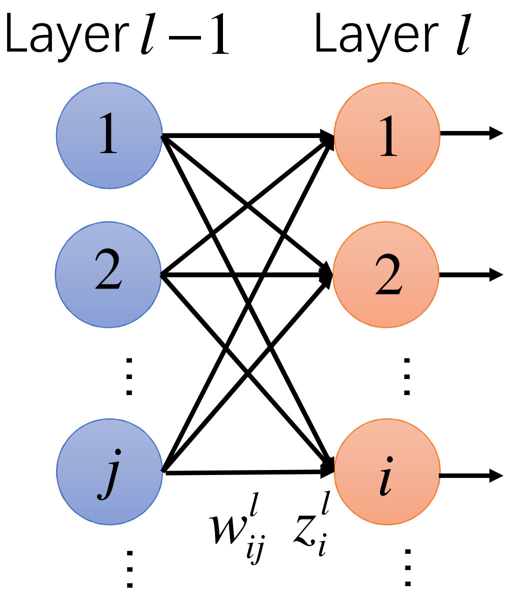
武德安

电子科技大学

# Background

- Cost Function $C(\theta)$
  - Given training examples: $\{(x^1, \hat{y}^1), \cdots, (x^r, \hat{y}^r), \cdots, (x^R, \hat{y}^R)\}$
  - Find a set of parameters θ* minimizing C(θ)
    - $C(\theta) = \frac{1}{R}\sum_r C^r(\theta)$, $C^r(\theta) = \|f(x^r;\theta) - \hat{y}^r\|$
- Gradient Descent
  - $\nabla C(\theta) = \frac{1}{R}\sum_r \nabla C^r(\theta)$
  - Given $w_{ij}^l$ and $b_i^l$, we have to compute $\partial C^r / \partial w_{ij}^l$ and $\partial C^r / \partial b_i^l$
- There is an efficient way to compute the gradients of the network parameters – ***backpropagation***.

# Backpropagation




$$\frac{\partial C}{\partial w_{ij}^{l}} = \boxed{\frac{\partial z_i^l}{\partial w_{ij}^l}} \boxed{\frac{\partial C}{\partial z_i^l}}$$

$$\begin{cases} a_j^{l-1} & l > 1 \\ x_j & l = 1 \end{cases}$$

Error signal

$$\delta_i^l$$

***Forward Pass***

$$z^1 = W^1 x + b^1$$

$$a^1 = \sigma\left(z^1\right)$$

$$\cdots\cdots$$

$$z^{l-1} = W^{l-1} a^{l-2} + b^{l-1}$$

$$a^{l-1} = \sigma\left(z^{l-1}\right)$$

***Backward Pass***

$$\delta^{L} = \sigma'\left(z^L\right) \bullet \nabla_y C$$

$$\delta^{L-1} = \sigma'\left(z^{L-1}\right) \bullet \left(W^L\right)^T \delta^L$$

$$\cdots\cdots$$

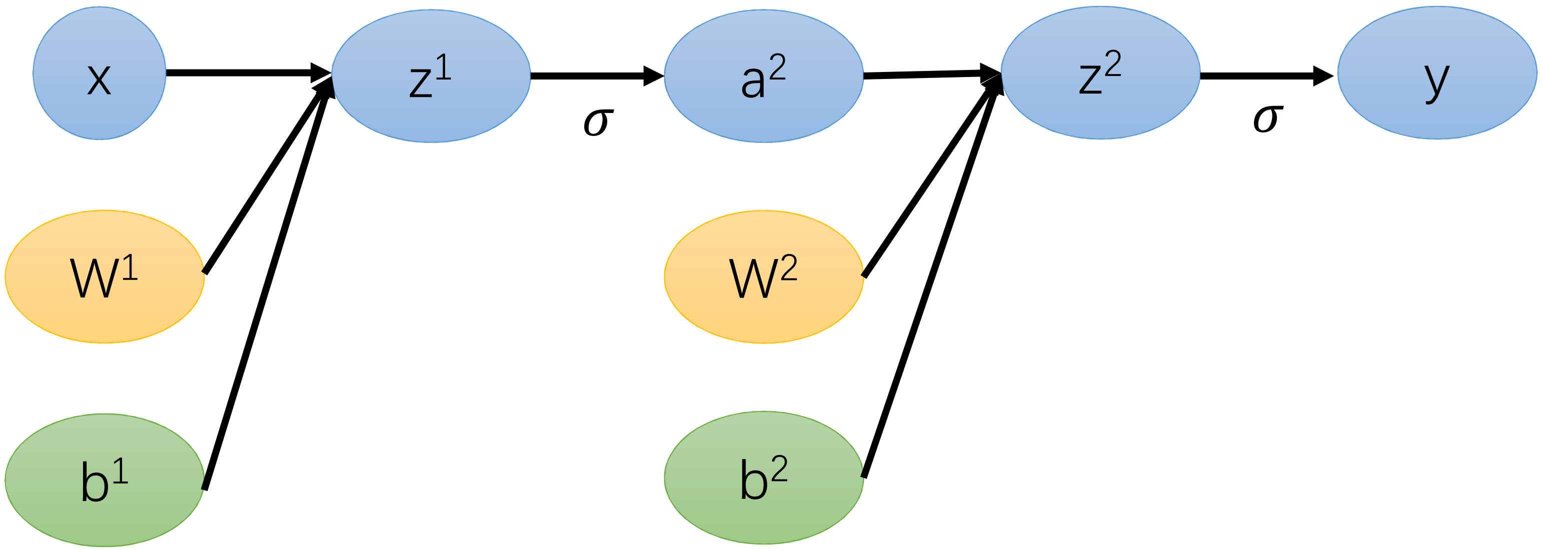
$$\delta^l = \sigma'\left(z^l\right) \bullet \left(W^{l+1}\right)^T \delta^{l+1}$$

$$\cdots\cdots$$

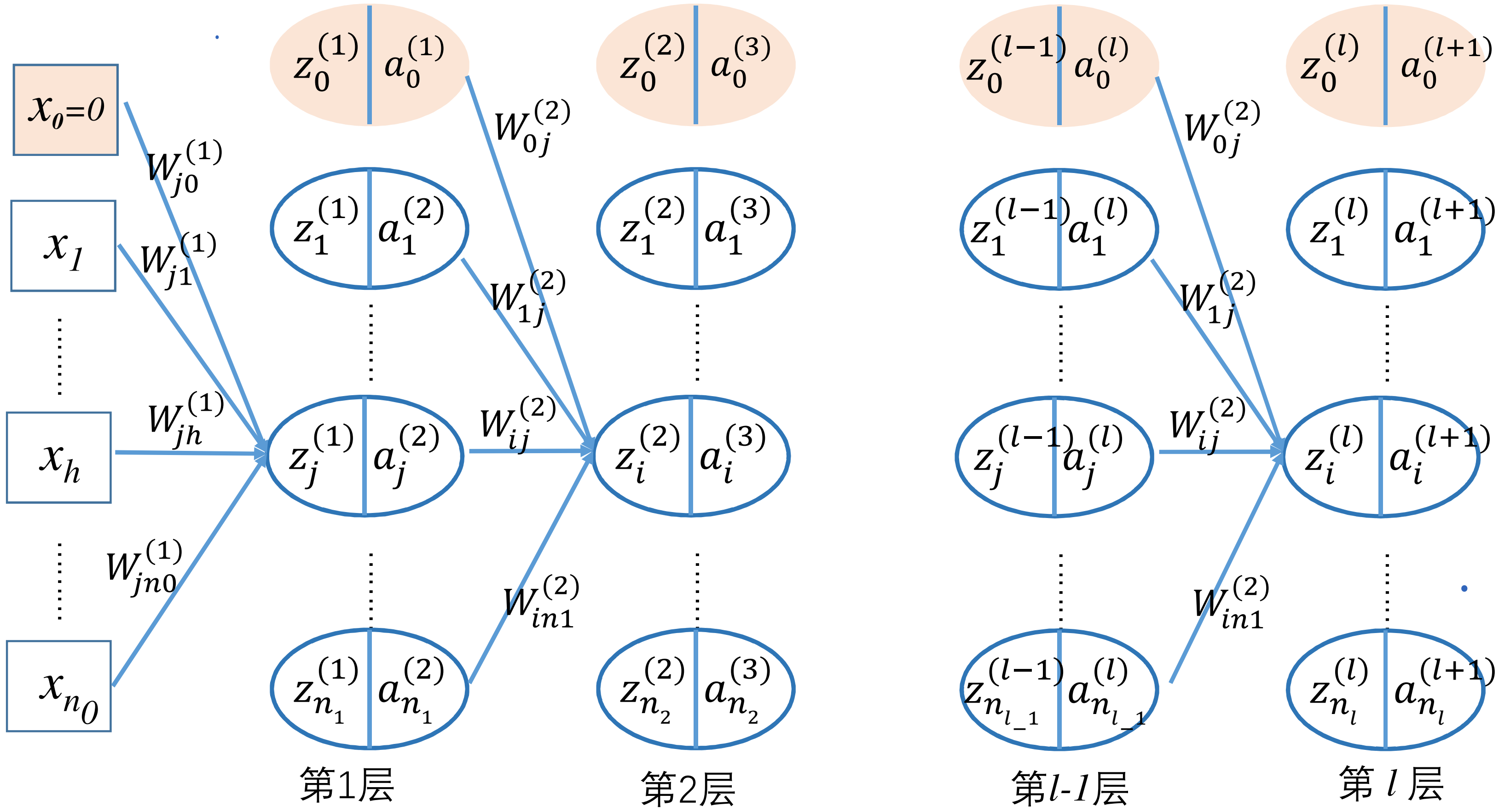
# Feedforward Network

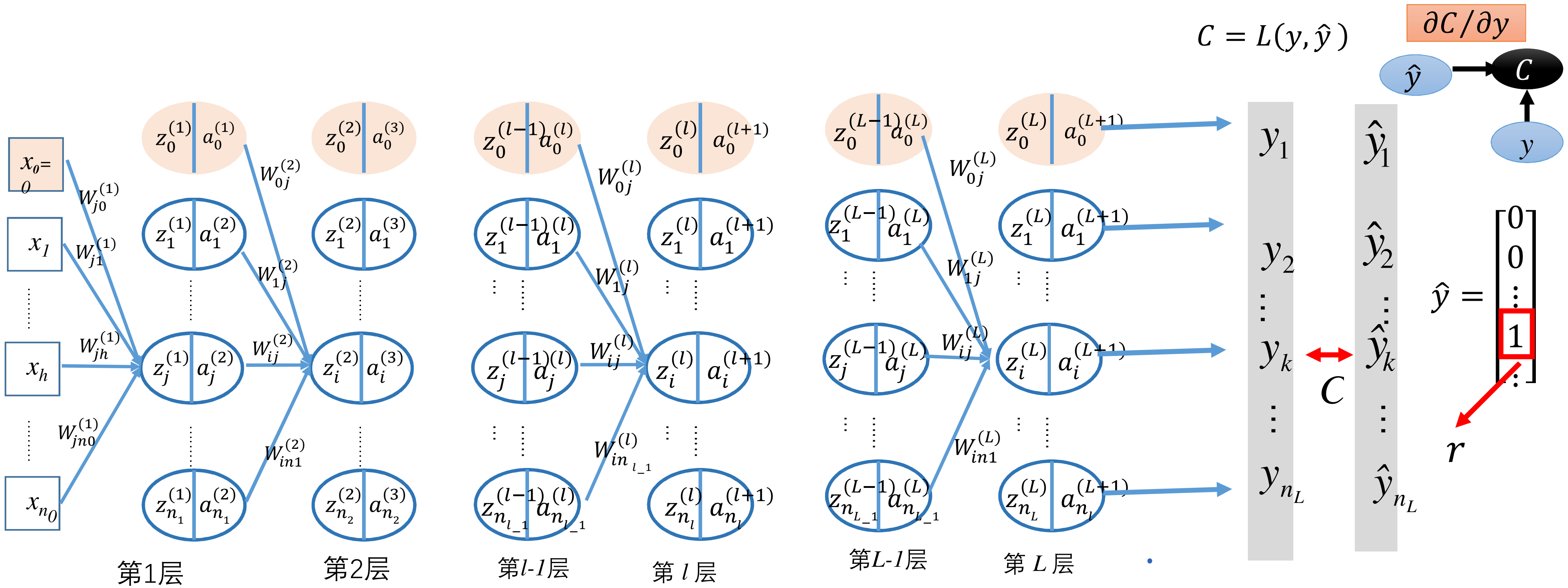
$$y = \sigma(W^L \cdots \sigma(W^2 \, \sigma(W^1 x + b^1) + b^2) \cdots + b^L)$$

$$z^1 = W^1 x + b^1 \qquad z^2 = W^2 a^2 + b^2$$



# NN与CNN BP算法总结






$$C = -log y_r$$

Cross Entropy: $\dfrac{\partial C}{\partial y} = [0 \quad \cdots \quad -1/y_r \quad \cdots]$

$i = r$: $\partial C/\partial y_r = -1/y_r$

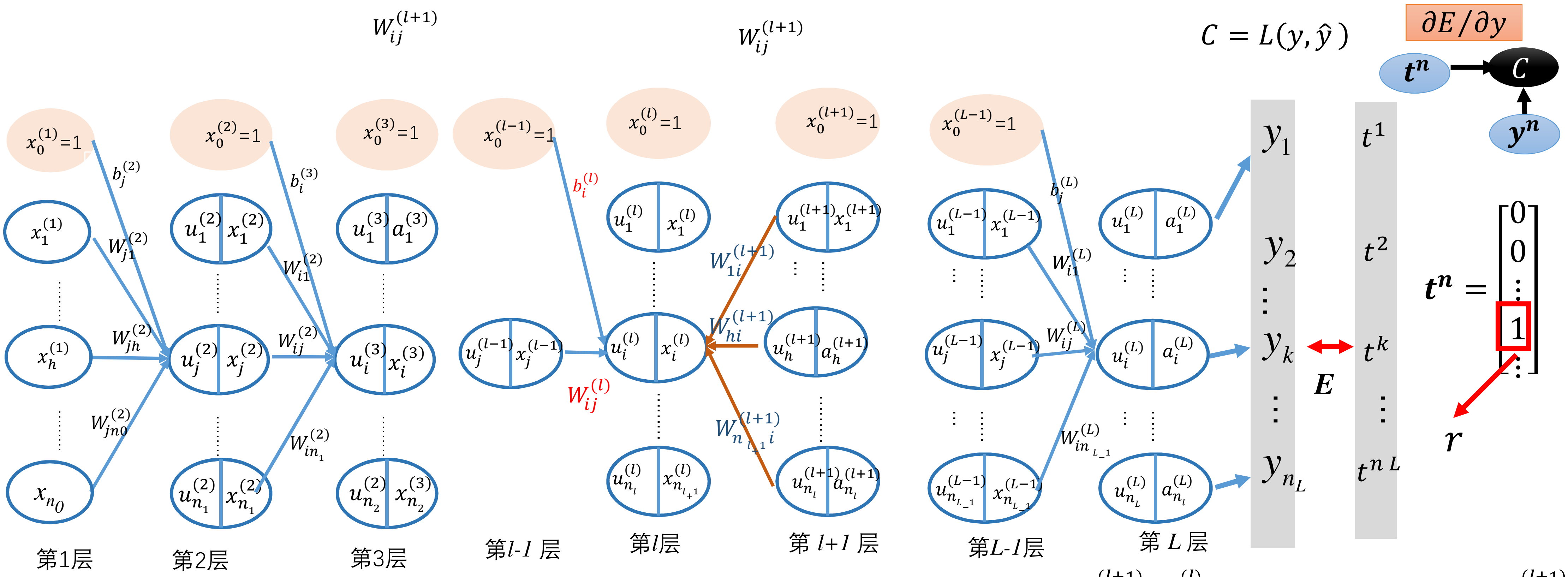
$i \neq r$: $\partial C/\partial y_i = 0$

Find a set of parameters θ* minimizing C(θ)

$$C(\theta) = \frac{1}{R}\sum_r C^r(\theta),\ C^r(\theta) = \|f(x^r;\theta) - \hat{y}^r\|$$

Gradient Descent

$$\nabla C(\theta) = \frac{1}{R}\sum_r \nabla C^r(\theta)$$




$$\frac{\partial E}{\partial b_i^{(l)}} = \frac{\partial E}{\partial u_i^{(l)}}\frac{\partial u_i^{(l)}}{\partial b_i^{(l)}} = \frac{\partial E}{\partial u_i^{(l)}} \times 1 = \frac{\partial E}{\partial u_i^{(l)}}$$

$$\frac{\partial E}{\partial u_i^{(l)}} = \sum_{k=1}^{n_{l+1}} \frac{\partial E}{\partial u_k^{(l+1)}}\frac{\partial u_k^{(l+1)}}{\partial x_i^{(l)}}\frac{\partial x_i^{(l)}}{\partial u_i^{(l)}} = f'(u_i^{(l)}) \sum_{k=1}^{n_{l+1}} \frac{\partial E}{\partial u_k^{(l+1)}}\frac{\partial u_k^{(l+1)}}{\partial x_i^{(l)}}$$

$$\frac{\partial E}{\partial W_{ij}^{(l)}} = \frac{\partial E}{\partial u_i^{(l)}}\frac{\partial u_i^{(l)}}{\partial W_{ij}^{(l)}} = \frac{\partial E}{\partial u_i^{(l)}} \times x_i^{(l-1)}$$

**Cross Entropy:** $C = -\log y_r$ $\quad \frac{\partial C}{\partial y_i} = ?$

$i = r$: $\partial C/\partial y_r = -1/y_r$

$i \neq r$: $\partial C/\partial y_i = 0$

$$\frac{\partial C}{\partial y} = [0 \quad \cdots \quad -1/y_r \quad \cdots]$$

Find a set of parameters θ* minimizing C(θ)

$$C(\theta) = \frac{1}{R}\sum_r C^r(\theta),\ C^r(\theta) = \|f(x^r;\theta) - \hat{y}^r\|$$

Gradient Descent

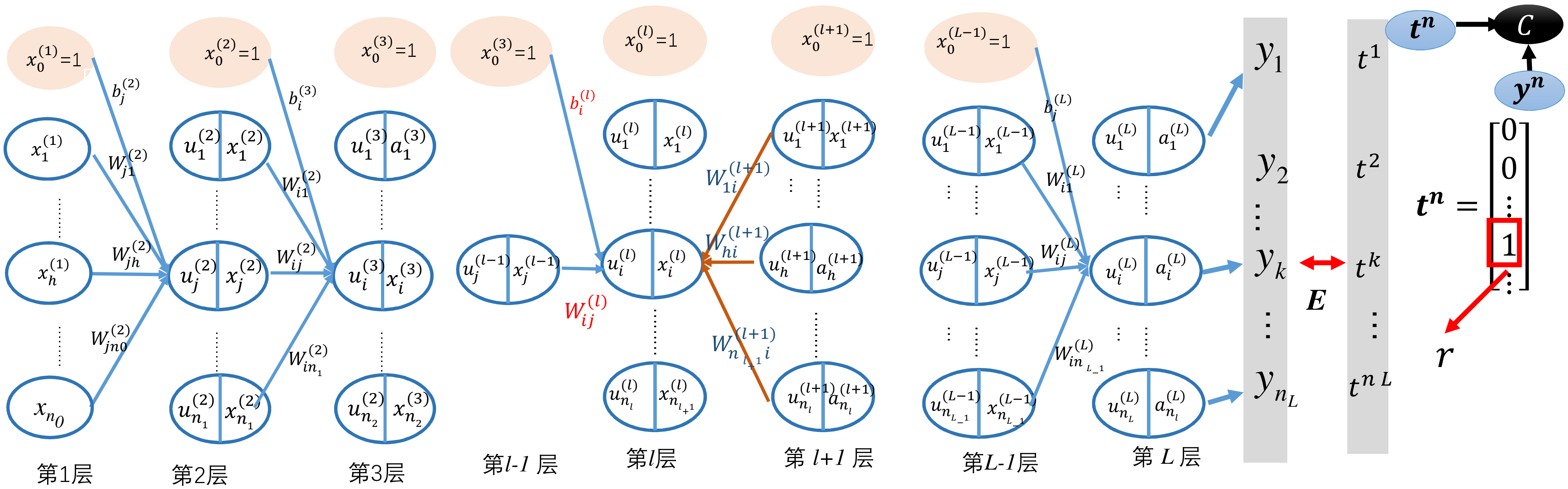
$$\nabla C(\theta) = \frac{1}{R}\sum_r \nabla C^r(\theta)$$

$$\frac{\partial E}{\partial u_i^{(l)}} = \sum_{k=1}^{n_{l+1}} \frac{\partial E}{\partial u_k^{(l+1)}} \frac{\partial u_k^{(l+1)}}{\partial x_i^{(l)}} \frac{\partial x_i^{(l)}}{\partial u_i^{(l)}} = f'(u_i^{(l)}) \sum_{k=1}^{n_{l+1}} \frac{\partial E}{\partial u_k^{(l+1)}} \frac{\partial u_k^{(l+1)}}{\partial x_i^{(l)}}$$

$$\frac{\partial E}{\partial u_i^{(l)}} = \sum_{k=1}^{n_{l+1}} \frac{\partial E}{\partial u_k^{(l+1)}} \frac{\partial u_k^{(l+1)}}{\partial x_i^{(l)}} \frac{\partial x_i^{(l)}}{\partial u_i^{(l)}} = f'(u_i^{(l)}) \sum_{k=1}^{n_{l+1}} \frac{\partial E}{\partial u_k^{(l+1)}} \frac{\partial u_k^{(l+1)}}{\partial x_i^{(l)}}$$

$$\delta_i^{(l)} = \frac{\partial E}{\partial u_i^{(l)}} = \sum_{k=1}^{n_{l+1}} \frac{\partial E}{\partial u_k^{(l+1)}} \frac{\partial u_k^{(l+1)}}{\partial x_i^{(l)}} \frac{\partial x_i^{(l)}}{\partial u_i^{(l)}} = f'\left(u_i^{(l)}\right) \sum_{k=1}^{n_{l+1}} \frac{\partial E}{\partial u_k^{(l+1)}} \frac{\partial u_k^{(l+1)}}{\partial x_i^{(l)}} = f'\left(u_i^{(l)}\right) \sum_{k=1}^{n_{l+1}} \delta_k^{(l+1)} W_{ki}^{(l+1)}$$

$$\frac{\partial E}{\partial u_i^{(l)}} = \sum_{k=1}^{n_{l+1}} \frac{\partial E}{\partial u_k^{(l+1)}} \frac{\partial u_k^{(l+1)}}{\partial x_i^{(l)}} \frac{\partial x_i^{(l)}}{\partial u_i^{(l)}} = f'(u_i^{(l)}) \sum_{k=1}^{n_{l+1}} \frac{\partial E}{\partial u_k^{(l+1)}} \frac{\partial u_k^{(l+1)}}{\partial x_i^{(l)}}$$




$$\delta_i^{(l)} \stackrel{\text{def}}{=} \frac{\partial E}{\partial u_i^{(l)}} \Rightarrow \left\{ \begin{aligned} \frac{\partial E}{\partial b_i^{(l)}} &= \frac{\partial E}{\partial u_i^{(l)}} \frac{\partial u_i^{(l)}}{\partial b_i^{(l)}} = \frac{\partial E}{\partial u_i^{(l)}} \times 1 = \frac{\partial E}{\partial u_i^{(l)}} = \delta_i^{(l)} \\ \frac{\partial E}{\partial W_{ij}^{(l)}} &= \frac{\partial E}{\partial u_i^{(l)}} \frac{\partial u_i^{(l)}}{\partial W_{ij}^{(l)}} = \delta_i^{(l)}\, x_j^{(l-1)} \end{aligned} \right.$$

$$\delta_i^{(L)} = \frac{\partial E}{\partial u_i^{(L)}} = (ti - yi) f'(u_i^{(L)})$$

$$\delta_i^{(l)} = \frac{\partial E}{\partial u_i^{(l)}} = \sum_{k=1}^{n_{l+1}} \frac{\partial E}{\partial u_k^{(l+1)}} \frac{\partial u_k^{(l+1)}}{\partial x_i^{(l)}} \frac{\partial x_i^{(l)}}{\partial u_i^{(l)}} = f'\left(u_i^{(l)}\right) \sum_{k=1}^{n_{l+1}} \frac{\partial E}{\partial u_k^{(l+1)}} \frac{\partial u_k^{(l+1)}}{\partial x_i^{(l)}} = f'\left(u_i^{(l)}\right) \sum_{k=1}^{n_{l+1}} \delta_k^{(l+1)} W_{ki}^{(l+1)}$$

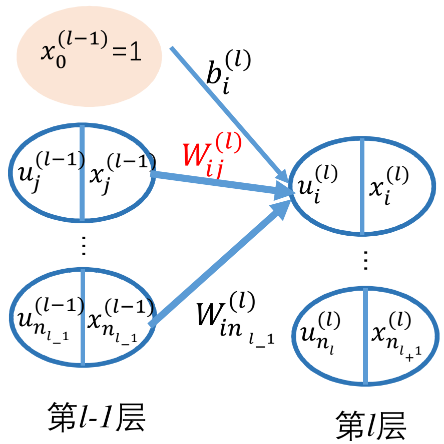
$$\delta_i^{(l)} \overset{\text{def}}{=} \frac{\partial E}{\partial u_i^{(l)}} \Rightarrow \left\{ \begin{array}{l} \dfrac{\partial E}{\partial b_i^{(l)}} = \dfrac{\partial E}{\partial u_i^{(l)}} \dfrac{\partial u_i^{(l)}}{\partial b_i^{(l)}} = \dfrac{\partial E}{\partial u_i^{(l)}} \times 1 = \dfrac{\partial E}{\partial u_i^{(l)}} = \delta_i^{(l)} \\ \dfrac{\partial E}{\partial W_{ij}^{(l)}} = \dfrac{\partial E}{\partial u_i^{(l)}} \dfrac{\partial u_i^{(l)}}{\partial W_{ij}^{(l)}} = \delta_i^{(l)} x_j^{(l-1)} \end{array} \right.$$

$$\delta_i^{(L)} = \frac{\partial E}{\partial u_i^{(L)}} = (ti - yi) f'(u_i^{(L)})$$

$$\boldsymbol{\delta}^{(L)} = \frac{\partial E}{\partial u_i^{(L)}} = (t^i - yi) f'\left(u_i^{(L)}\right) = f'\left(\boldsymbol{u}^{(L)}\right) \odot (\boldsymbol{t} - \boldsymbol{y})$$

$$\delta_i^{(l)} = \frac{\partial E}{\partial u_i^{(l)}} = \sum_{k=1}^{n_{l+1}} \frac{\partial E}{\partial u_k^{(l+1)}} \frac{\partial u_k^{(l+1)}}{\partial x_i^{(l)}} \frac{\partial x_i^{(l)}}{\partial u_i^{(l)}} = f'\left(u_i^{(l)}\right) \sum_{k=1}^{n_{l+1}} \frac{\partial E}{\partial u_k^{(l+1)}} \frac{\partial u_k^{(l+1)}}{\partial x_i^{(l)}} = f'\left(u_i^{(l)}\right) \sum_{k=1}^{n_{l+1}} \delta_k^{(l+1)} W_{ki}^{(l+1)}$$

# 矩阵形式




$$\boldsymbol{u}^{(l)} = \begin{bmatrix} u_1^{(l)} \\ \vdots \\ u_{n_l}^{(l)} \end{bmatrix} \quad \boldsymbol{x}^{(l)} = \begin{bmatrix} x_1^{(l)} \\ \vdots \\ x_{n_l}^{(l)} \end{bmatrix} \quad \boldsymbol{\delta}^{(l)} = \frac{\partial E}{\partial \boldsymbol{u}^{(l)}} \quad \boldsymbol{\delta}^{(l)} = \begin{bmatrix} \delta_1^{(l)} \\ \vdots \\ \delta_{n_l}^{(l)} \end{bmatrix}$$

$$\boldsymbol{u}^{(l)} = \boldsymbol{W}^{(l)} \boldsymbol{x}^{(l-1)} + \boldsymbol{b}^{(l)}$$

$$\boldsymbol{x}^{(l)} = f\left(\boldsymbol{u}^{(l)}\right) = f(\boldsymbol{W}^{(l)} \boldsymbol{x}^{(l-1)} + \boldsymbol{b}^{(l)})$$

$$\frac{\partial E}{\partial \boldsymbol{W}^{(l)}} = \frac{\partial E}{\partial u_i^{(l)}} \frac{\partial u_i^{(l)}}{\partial W_{ij}^{(l)}} = \left(\boldsymbol{\delta}^{(l)}\right)^T \boldsymbol{x}^{(l-1)}$$

$$\boldsymbol{W}^{(l)} \boldsymbol{x}^{(l-1)} + \boldsymbol{b}^{(l)} = \begin{bmatrix} w_{11}^{(l)} & w_{12}^{(l)} \cdots & w_{1n_{l\_1}}^{(l)} \\ w_{21}^{(l)} & w_{22}^{(l)} & w_{2n_{l\_1}}^{(l)} \\ w_{n_l 1}^{(l)} & w_{n_l 2}^{(l)} & w_{n_l n_{l\_1}}^{(l)} \end{bmatrix} \begin{bmatrix} x_1^{(l-1)} \\ x_2^{(l-1)} \\ x_{n_{l\_1}}^{(l-1)} \end{bmatrix} + \begin{bmatrix} b_1^{(l)} \\ b_2^{(l)} \\ b_{n_{l\_1}}^{(l)} \end{bmatrix} = \begin{bmatrix} u_1^{(l)} \\ \vdots \\ u_{n_l}^{(l)} \end{bmatrix}$$

$$\boldsymbol{\delta}^{(l)} = \begin{bmatrix} \delta_1^{(l)} \\ \vdots \\ \delta_{n_l}^{(l)} \end{bmatrix} = \begin{bmatrix} w_{11}^{(l+1)} & w_{21}^{(l+1)} \cdots & w_{n_{l+1}1}^{(l+1)} \\ w_{12}^{(l+1)} & w_{22}^{(l+1)} & w_{n_{l+1}2}^{(l+1)} \\ w_{1n_l}^{(l+1)} & w_{2n_l}^{(l+1)} & w_{n_{l+1}n_l}^{(l+1)} \end{bmatrix} \begin{bmatrix} \delta_1^{(l+1)} \\ \vdots \\ \delta_{n_{l+1}}^{(l+1)} \end{bmatrix} \odot \begin{bmatrix} f'\left(u_1^{(l)}\right) \\ \vdots \\ f'\left(u_{n_l}^{(l)}\right) \end{bmatrix} = f'\left(\boldsymbol{u}^{(l)}\right) \odot \left(\boldsymbol{W}^{(l+1)}\right)^T \boldsymbol{\delta}^{(l+1)}$$

$$\delta_i^{(l)} \stackrel{\text{def}}{=} \frac{\partial E}{\partial u_i^{(l)}} \Rightarrow \begin{cases} \dfrac{\partial E}{\partial b_i^{(l)}} = \dfrac{\partial E}{\partial u_i^{(l)}}\dfrac{\partial u_i^{(l)}}{\partial b_i^{(l)}} = \dfrac{\partial E}{\partial u_i^{(l)}} \times 1 = \dfrac{\partial E}{\partial u_i^{(l)}} = \delta_i^{(l)} \\[2ex] \dfrac{\partial E}{\partial W_{ij}^{(l)}} = \dfrac{\partial E}{\partial u_i^{(l)}}\dfrac{\partial u_i^{(l)}}{\partial W_{ij}^{(l)}} = \delta_i^{(l)}\, x_j^{(l-1)} \end{cases}$$

$$\delta_i^{(L)} = \frac{\partial E}{\partial u_i^{(L)}} = (ti - yi) f'(u_i^{(L)})$$

$$\boldsymbol{\delta}^{(L)} = \frac{\partial E}{\partial u_i^{(L)}} = (t^i - yi) f'\left(u_i^{(L)}\right) = f'\left(\boldsymbol{u}^{(L)}\right) \odot (\boldsymbol{t} - \boldsymbol{y})$$

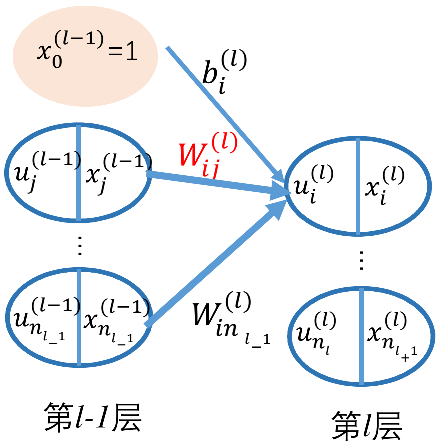
$$\delta_i^{(l)} = \frac{\partial E}{\partial u_i^{(l)}} = \sum_{k=1}^{n_{l+1}} \frac{\partial E}{\partial u_k^{(l+1)}} \frac{\partial u_k^{(l+1)}}{\partial x_i^{(l)}} \frac{\partial x_i^{(l)}}{\partial u_i^{(l)}} = f'\left(u_i^{(l)}\right) \sum_{k=1}^{n_{l+1}} \frac{\partial E}{\partial u_k^{(l+1)}} \frac{\partial u_k^{(l+1)}}{\partial x_i^{(l)}} = f'\left(u_i^{(l)}\right) \sum_{k=1}^{n_{l+1}} \delta_k^{(l+1)} W_{ki}^{(l+1)}$$

# 矩阵形式

$$\boldsymbol{u}^{(l)} = \begin{bmatrix} u_1^{(l)} \\ \vdots \\ u_{n_l}^{(l)} \end{bmatrix} \quad \boldsymbol{x}^{(l)} = \begin{bmatrix} x_1^{(l)} \\ \vdots \\ x_{n_l}^{(l)} \end{bmatrix} \quad \boldsymbol{\delta}^{(l)} = \frac{\partial E}{\partial \boldsymbol{u}^{(l)}} \quad \boldsymbol{\delta}^{(l)} = \begin{bmatrix} \delta_1^{(l)} \\ \vdots \\ \delta_{n_l}^{(l)} \end{bmatrix}$$

$$\boldsymbol{u}^{(l)} = \boldsymbol{W}^{(l)}\, \boldsymbol{x}^{(l-1)} + \boldsymbol{b}^{(l)}$$

$$\boldsymbol{x}^{(l)} = f\left(\boldsymbol{u}^{(l)}\right) = f(\boldsymbol{W}^{(l)}\, \boldsymbol{x}^{(l-1)} + \boldsymbol{b}^{(l)})$$

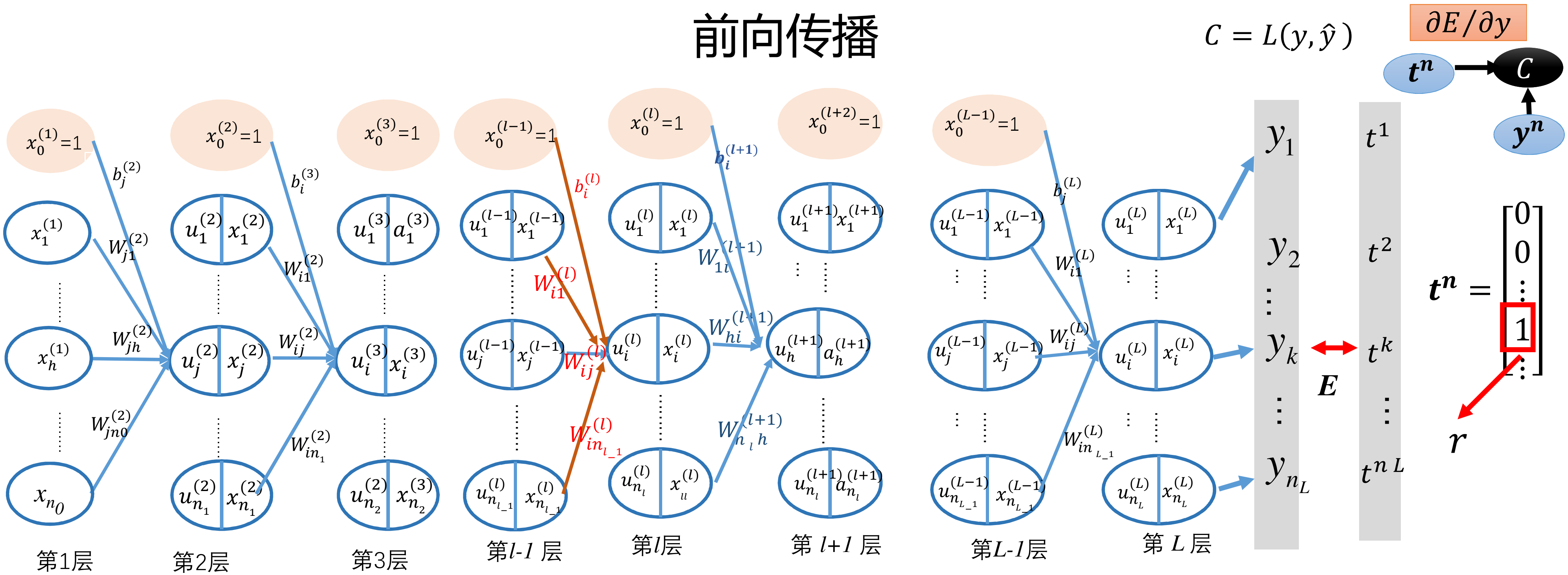



$$\frac{\partial E}{\partial \boldsymbol{W}^{(l)}} = \frac{\partial E}{\partial u_i^{(l)}} \frac{\partial u_i^{(l)}}{\partial W_{ij}^{(l)}} = \left(\boldsymbol{\delta}^{(l)}\right)^T \boldsymbol{x}^{(l-1)}$$

$$\boldsymbol{W}^{(l)}\boldsymbol{x}^{(l-1)} + \boldsymbol{b}^{(l)} = \begin{bmatrix} w_{11}^{(l)} & w_{12}^{(l)} \cdots & w_{1n_{l-1}}^{(l)} \\ w_{21}^{(l)} & w_{22}^{(l)} & w_{2n_{l-1}}^{(l)} \\ w_{n_l1}^{(l)} & w_{n_l2}^{(l)} & w_{n_ln_{l-1}}^{(l)} \end{bmatrix} \begin{bmatrix} x_1^{(l-1)} \\ x_2^{(l-1)} \\ x_{n_{l-1}}^{(l-1)} \end{bmatrix} + \begin{bmatrix} b_1^{(l)} \\ b_2^{(l)} \\ b_{n_{l-1}}^{(l)} \end{bmatrix} = \begin{bmatrix} u_1^{(l)} \\ \vdots \\ u_{n_l}^{(l)} \end{bmatrix}$$

$$\boldsymbol{\delta}^{(l)} = \begin{bmatrix} \delta_1^{(l)} \\ \vdots \\ \delta_{n_l}^{(l)} \end{bmatrix} = \begin{bmatrix} w_{11}^{(l+1)} & w_{21}^{(l+1)} \cdots & w_{n_{l+1}1}^{(l+1)} \\ w_{12}^{(l+1)} & w_{22}^{(l+1)} & w_{n_{l+1}2}^{(l+1)} \\ w_{1n_l}^{(l+1)} & w_{2n_l}^{(l+1)} & w_{n_{l+1}n_l}^{(l+1)} \end{bmatrix} \begin{bmatrix} \delta_1^{(l+1)} \\ \vdots \\ \delta_{n_{l+1}}^{(l+1)} \end{bmatrix} \odot \begin{bmatrix} f'\left(u_1^{(l)}\right) \\ \vdots \\ f'\left(u_{n_l}^{(l)}\right) \end{bmatrix} = f'\left(\boldsymbol{u}^{(l)}\right) \odot \left(\boldsymbol{W}^{(l+1)}\right)^T \boldsymbol{\delta}^{(l+1)}$$

# 前向传播




$$\boldsymbol{W}^{(l)}\boldsymbol{x}^{(l-1)} + \boldsymbol{b}^{(l)} = \begin{bmatrix} w_{11}^{(l)} & w_{12}^{(l)} \cdots & w_{1n_{l\_1}}^{(l)} \\ w_{21}^{(l)} & w_{22}^{(l)} & w_{2n_{l\_1}}^{(l)} \\ w_{n_l 1}^{(l)} & w_{n_l 2}^{(l)} & w_{n_l n_{l\_1}}^{(l)} \end{bmatrix} \begin{bmatrix} x_1^{(l-1)} \\ x_2^{(l-1)} \\ x_{n_{l\_1}}^{(l-1)} \end{bmatrix} + \begin{bmatrix} b_1^{(l)} \\ b_2^{(l)} \\ b_{n_{l\_1}}^{(l)} \end{bmatrix} = \begin{bmatrix} u_1^{(l)} \\ \vdots \\ u_{n_l}^{(l)} \end{bmatrix}$$

$$\boldsymbol{u}^{(l)} = \boldsymbol{W}^{(l)}\ \boldsymbol{x}^{(l-1)} + \boldsymbol{b}^{(l)}$$

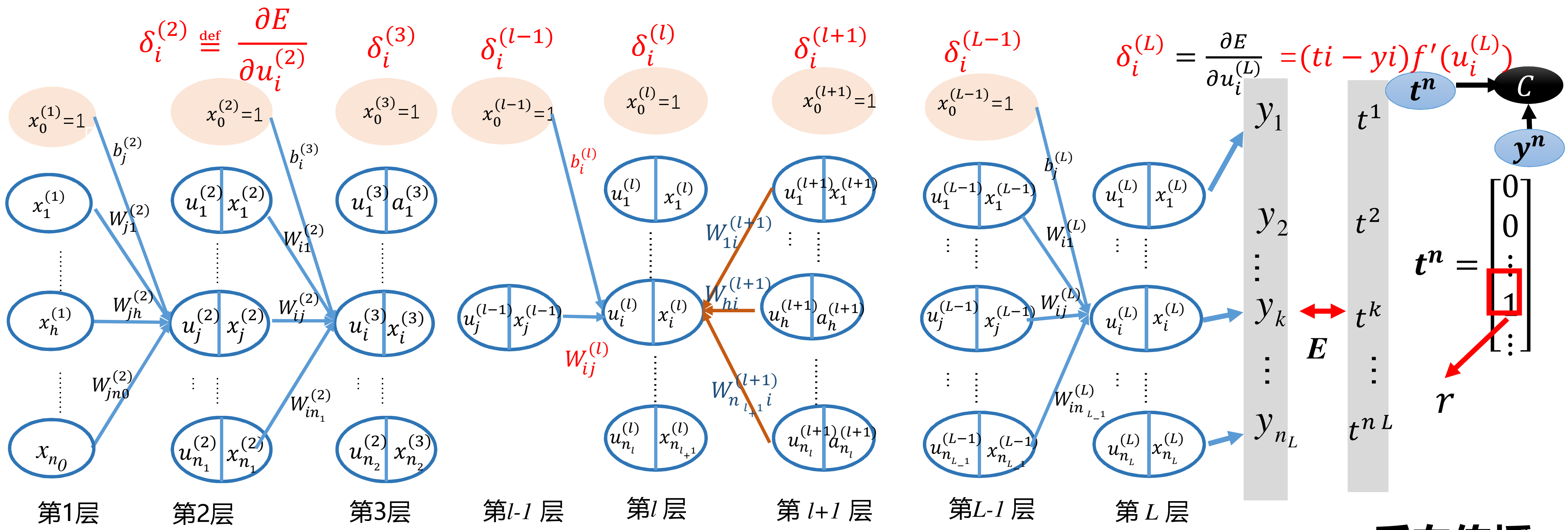
$$\boldsymbol{x}^{(l)} = f\left(\boldsymbol{u}^{(l)}\right) = f(\boldsymbol{W}^{(l)}\ \boldsymbol{x}^{(l-1)} + \boldsymbol{b}^{(l)})$$

$$\boldsymbol{u}^{(2)} = \boldsymbol{W}^{(2)}\ \boldsymbol{x}^{(1)} + \boldsymbol{b}^{(1)}$$

$$\boldsymbol{x}^{(2)} = f\left(\boldsymbol{u}^{(2)}\right) = f(\boldsymbol{W}^{(2)}\ \boldsymbol{x}^{(1)} + \boldsymbol{b}^{(2)})$$

$$\boldsymbol{u}^{(L)} = \boldsymbol{W}^{(L)}\ \boldsymbol{x}^{(L-1)} + \boldsymbol{b}^{(L)}$$

$$\boldsymbol{y} = f\left(\boldsymbol{u}^{(L-1)}\right) = f(\boldsymbol{W}^{(L)}\ \boldsymbol{x}^{(L-1)} + \boldsymbol{b}^{(L)})$$




# 反向传播

$$\frac{\partial E}{\partial b_i^{(l)}} = \frac{\partial E}{\partial u_i^{(l)}}\frac{\partial u_i^{(l)}}{\partial b_i^{(l)}} = \frac{\partial E}{\partial u_i^{(l)}} \times 1 = \frac{\partial E}{\partial u_i^{(l)}} = \delta_i^{(l)}$$

$$\boldsymbol{\delta}^{(l)} = \begin{bmatrix} \delta_1^{(l)} \\ \vdots \\ \delta_{n_l}^{(l)} \end{bmatrix} = \begin{bmatrix} w_{11}^{(l+1)} & w_{21}^{(l+1)} & \cdots & w_{n_{l+1}1}^{(l+1)} \\ w_{12}^{(l+1)} & w_{22}^{(l+1)} & & w_{n_{l+1}2}^{(l+1)} \\ w_{1n_l}^{(l+1)} & w_{2n_l}^{(l+1)} & & w_{n_{l+1}n_l}^{(l+1)} \end{bmatrix} \begin{bmatrix} \delta_1^{(l+1)} \\ \vdots \\ \delta_{n_{l+1}}^{(l+1)} \end{bmatrix} \odot \begin{bmatrix} f'\left(u_1^{(l)}\right) \\ \vdots \\ f'\left(u_{n_l}^{(l)}\right) \end{bmatrix} = f'(\boldsymbol{u}^{(l)}) \odot \left(\boldsymbol{W}^{(l+1)}\right)^T \boldsymbol{\delta}^{(l+1)}$$

$$\frac{\partial E}{\partial W_{ij}^{(l)}} = \frac{\partial E}{\partial u_i^{(l)}}\frac{\partial u_i^{(l)}}{\partial W_{ij}^{(l)}} = \delta_i^{(l)} x_j^{(l-1)}$$

$$\frac{\partial E}{\partial \boldsymbol{W}^{(l)}} = \frac{\partial E}{\partial u_i^{(l)}}\frac{\partial u_i^{(l)}}{\partial W_{ij}^{(l)}} = \left(\boldsymbol{\delta}^{(l)}\right)^T \boldsymbol{x}^{(l-1)} \qquad \frac{\partial E}{\partial \boldsymbol{b}^{(l)}} = \boldsymbol{\delta}^{(l)}$$

$$\delta_i^{(l)} = \frac{\partial E}{\partial u_i^{(l)}} = \sum_{k=1}^{n_{l+1}} \frac{\partial E}{\partial u_k^{(l+1)}} \frac{\partial u_k^{(l+1)}}{\partial x_i^{(l)}} \frac{\partial x_i^{(l)}}{\partial u_i^{(l)}} = f'\left(u_i^{(l)}\right) \sum_{k=1}^{n_{l+1}} \frac{\partial E}{\partial u_k^{(l+1)}} \frac{\partial u_k^{(l+1)}}{\partial x_i^{(l)}} = f'\left(u_i^{(l)}\right) \sum_{k=1}^{n_{l+1}} \delta_k^{(l+1)} W_{ki}^{(l+1)}$$

# 1 Feedforward

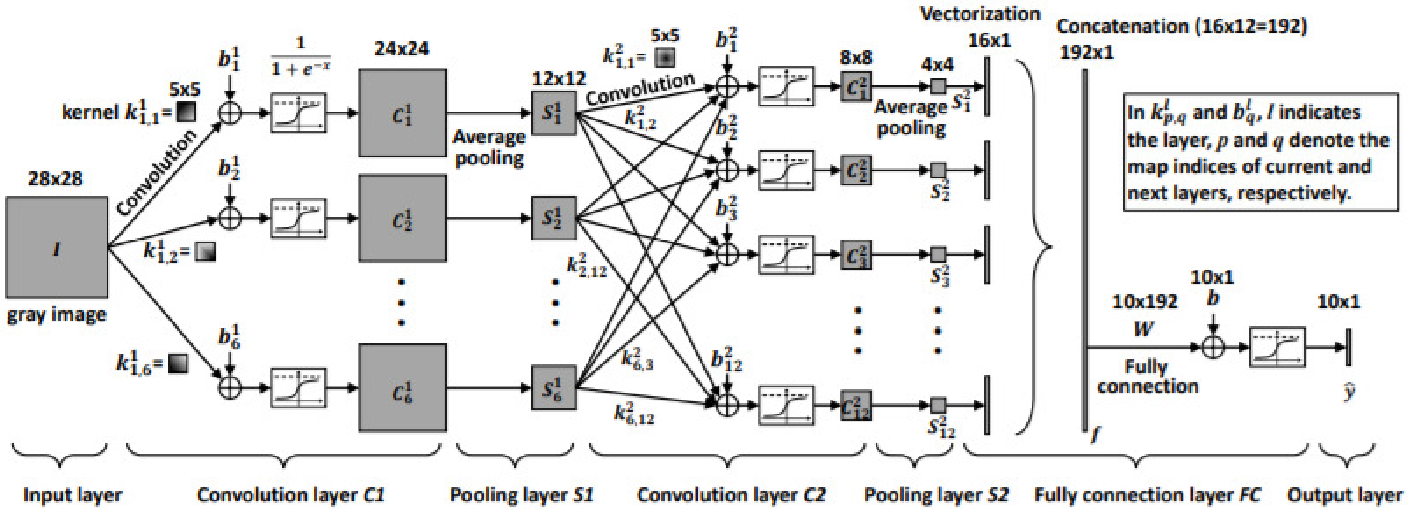



Figure 1: The structure of CNN example that will be discussed in this paper. It is exactly the same to the structure used in the demo of Matlab DeepLearnToolbox [1]. All later derivation will use the same notations in this figure.

- **C1 layer**, $k^1_{1,p}$ (size $5 \times 5$) and $b^1_p$ (size $1 \times 1$), $p = 1, 2, \cdots 6$
- **C2 layer**, $k^2_{p,q}$ (size $5 \times 5$) and $b^2_q$ (size $1 \times 1$), $q = 1, 2, \cdots 12$
- **FC layer**, $W$ (size $10 \times 192$) and $b$ (size $10 \times 1$)

$$k^1_{1,p} \sim U\left(\pm\sqrt{\frac{6}{(1+6) \times 5^2}}\right)$$

$$k^2_{p,q} \sim U\left(\pm\sqrt{\frac{6}{(6+12) \times 5^2}}\right)$$

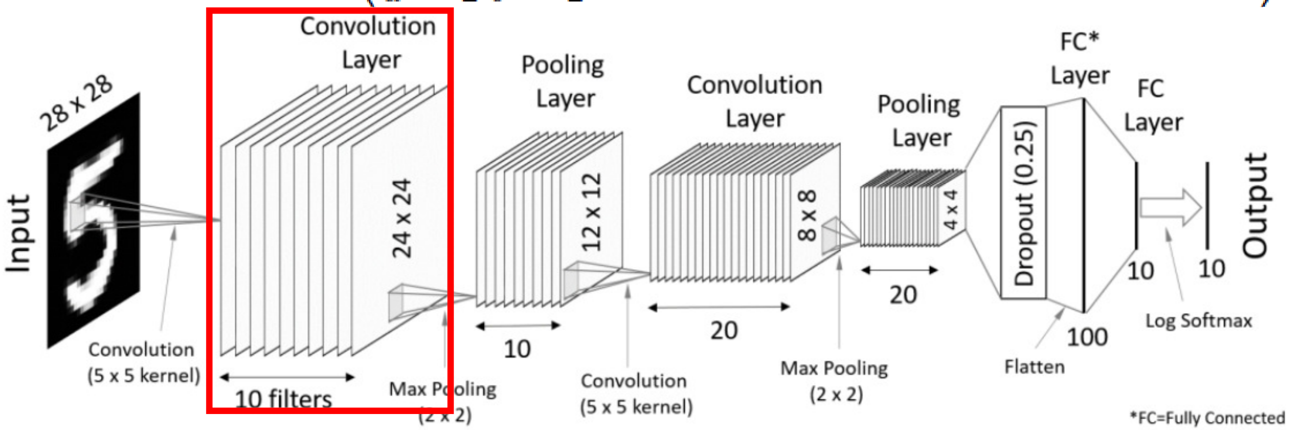
$$W \sim U\left(\pm\sqrt{\frac{6}{192+10}}\right)$$

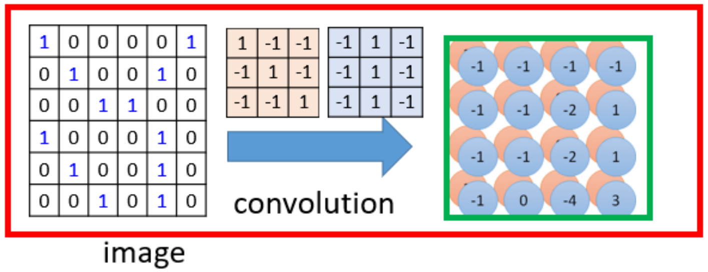
## 1.2 Convolution Layer C1

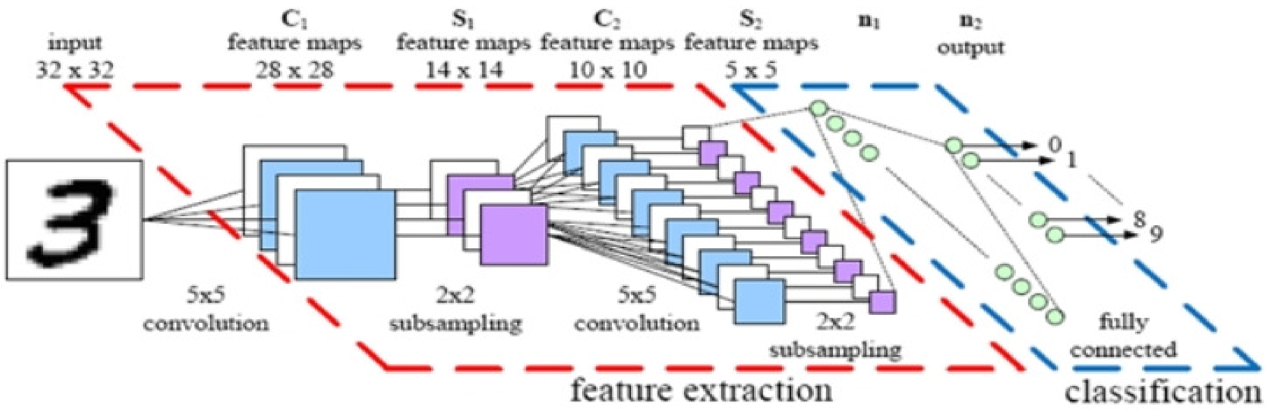
$$C_p^1 = \sigma(I * k_{1,p}^1 + b_p^1), \text{ where } \sigma(x) = \frac{1}{1 + \exp^{-x}}$$

$$C_p^1(i,j) = \sigma\left(\sum_{u=-2}^{2}\sum_{v=-2}^{2} I(i-u, j-v)\cdot k_{1,p}^1(u,v) + b_p^1\right)$$

$p = 1, 2, \cdots, 6$ $\quad$ $C^1{}_p$ is 24×24,



## 1.3 Pooling Layer S1

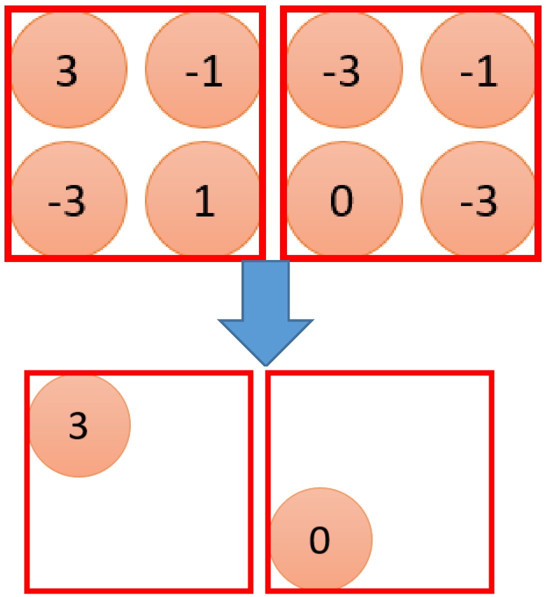
$$p = 1, 2, \cdots, 6$$

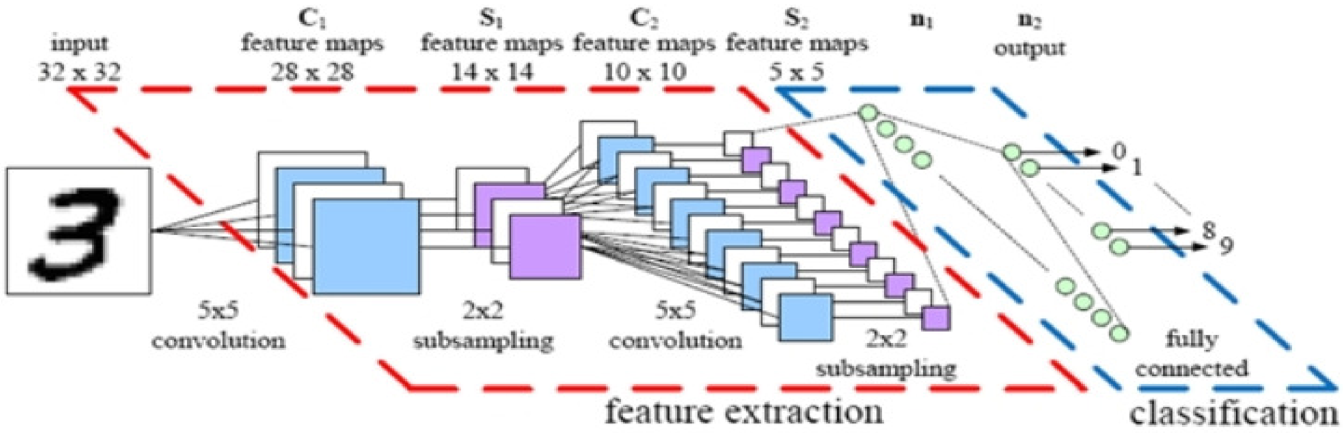
$$S_p^1(i,j) = \frac{1}{4}\sum_{u=0}^{1}\sum_{v=0}^{1} C_p^1(2i-u, 2j-v),\ i,j = 1,2,\cdots,12$$

$$S_1^1(1,1) = \frac{1}{4}\begin{pmatrix} C_1^1(1,1) + C_1^1(1,2) + \\ C_1^1(2,1) + C_1^1(2,2) \end{pmatrix} \qquad S_1^1(1,2) = \frac{1}{4}\begin{pmatrix} C_1^1(1,3) + C_1^1(1,4) + \\ C_1^1(2,3) + C_1^1(2,4) \end{pmatrix}$$

$$S_1^1(11,12) = \frac{1}{4}\begin{pmatrix} C_1^1(21,23) + C_1^1(21,24) + \\ C_1^1(22,23) + C_1^1(22,24) \end{pmatrix} \qquad S_1^1(12,12) = \frac{1}{4}\begin{pmatrix} C_1^1(23,23) + C_1^1(23,24) + \\ C_1^1(24,23) + C_1^1(24,24) \end{pmatrix}$$

**CNN – Max Pooling**



$$C_q^2 = \sigma\left(\sum_{p=1}^{6} S_p^1 * k_{p,q}^2 + b_q^2\right)$$

$q = 1, 2, \cdots, 12$ because there are 12 feature maps on C2 layer

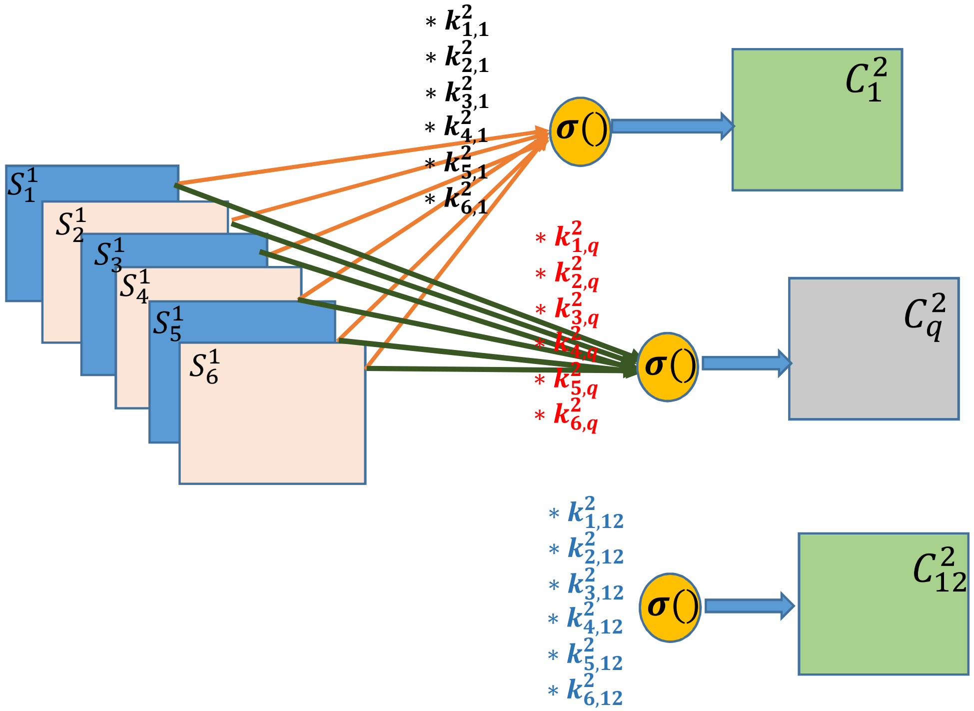
$$C_q^2(i,j) = \sigma\left(\sum_{p=1}^{6}\sum_{u=-2}^{2}\sum_{v=-2}^{2} S_p^1(i-u, j-v)\cdot k_{p,q}^2(u,v) + b_q^2\right)$$

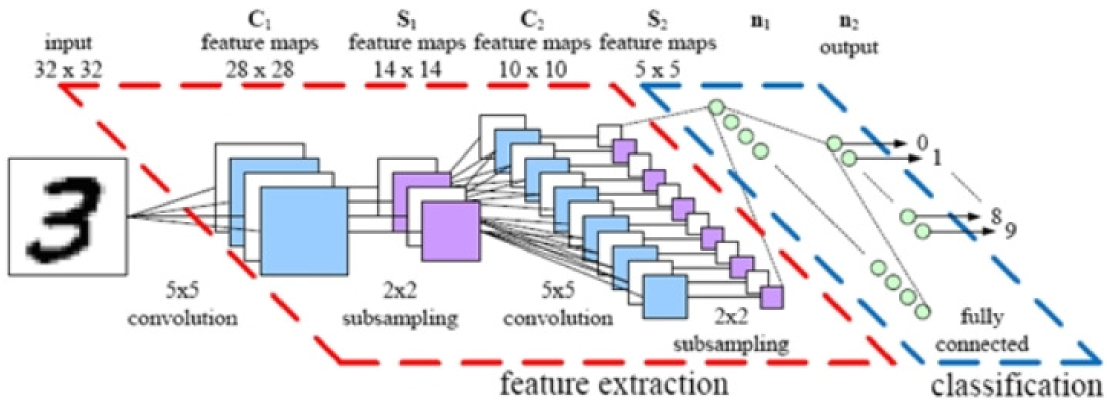
$C^2{}_q$ is $8 \times 8$,

$$C_1^2 = \sigma\begin{pmatrix} S_1^1 * k_{1,1}^2 + S_2^1 * k_{2,1}^2 + S_3^1 * k_{3,1}^2 + \\ S_4^1 * k_{4,1}^2 + S_5^1 * k_{5,1}^2 + S_6^1 * k_{6,1}^2 + b_1^2 \end{pmatrix}$$

$$C_2^2 = \sigma\begin{pmatrix} S_1^1 * k_{1,2}^2 + S_2^1 * k_{2,2}^2 + S_3^1 * k_{3,2}^2 + \\ S_4^1 * k_{4,2}^2 + S_5^1 * k_{5,2}^2 + S_6^1 * k_{6,2}^2 + b_2^2 \end{pmatrix}$$

$$C_{12}^2 = \sigma\begin{pmatrix} S_1^1 * k_{1,12}^2 + S_2^1 * k_{2,12}^2 + S_3^1 * k_{3,12}^2 + \\ S_4^1 * k_{4,12}^2 + S_5^1 * k_{5,12}^2 + S_6^1 * k_{6,12}^2 + b_{12}^2 \end{pmatrix}$$



## 1.5 Pooling Layer S2

$$S_q^2(i,j) = \frac{1}{4}\sum_{u=0}^{1}\sum_{v=0}^{1} C_q^2(2i-u, 2j-v),\ i,j = 1,2,\cdots,4$$

$$S_1^1(1,1) = \frac{1}{4}\begin{pmatrix} C_1^2(1,1) + C_1^2(1,2) + \\ C_1^2(2,1) + C_1^2(2,2) \end{pmatrix} \qquad S_1^1(1,2) = \frac{1}{4}\begin{pmatrix} C_1^2(1,3) + C_1^2(1,4) + \\ C_1^2(2,3) + C_1^2(2,4) \end{pmatrix}$$

$$S_1^1(3,4) = \frac{1}{4}\begin{pmatrix} C_1^2(5,7) + C_1^2(5,8) + \\ C_1^2(6,7) + C_1^2(6,8) \end{pmatrix} \qquad S_1^1(4,4) = \frac{1}{4}\begin{pmatrix} C_1^2(7,7) + C_1^2(7,8) + \\ C_1^2(8,7) + C_1^2(8,8) \end{pmatrix}$$

### 1.6 Vectorization and Concatenation

Each $S_q^2$ is a $4 \times 4$ matrix, and there are 12 such matrices on the S2 layer. First, each $S_q^2$ is vectorized by column scan, then all 12 vectors are concatenated to form a long vector with the length of $4 \times 4 \times 12 = 192$. We denote this process by

$$f = F\left(\{S_q^2\}_{q=1,2,\cdots,12}\right), \tag{10}$$

### 1.7 Fully Connection Layer FC

$$\hat{y} = \sigma(W \times f + b) \tag{12}$$

### 1.8 Loss Function

Assuming the true label is $y$, the loss function is express by

$$L = \frac{1}{2} \sum_{i=1}^{10} \left(\hat{y}(i) - y(i)\right)^2 \tag{13}$$

namely $\boldsymbol{W}$ and $\boldsymbol{b}$, $k^2{}_{p,q}$ and $b^2{}_q$, $k^1{}_{1,p}$ and $b^1{}_p$.

$$\hat{y} = \sigma(W \times f + b) \qquad L = \frac{1}{2}\sum_{i=1}^{10}(\hat{y}(i) - y(i))^2$$

$\Delta W$ **(size $10 \times 192$)**

$$\begin{aligned}
\Delta W(i,j) &= \frac{\partial L}{\partial W(i,j)} \\
&= \frac{\partial L}{\partial \hat{y}(i)} \cdot \frac{\partial \hat{y}(i)}{\partial W(i,j)} \\
&= (\hat{y}(i) - y(i)) \cdot \frac{\partial}{\partial W(i,j)}\sigma\left(\sum_{j=1}^{192} W(i,j) \times f(j) + b(i)\right) \\
&= (\hat{y}(i) - y(i)) \cdot \hat{y}(i)(1 - \hat{y}(i)) \cdot f(j)
\end{aligned}$$

Let $\Delta\hat{y}(i) = (\hat{y}(i) - y(i)) \cdot \hat{y}(i)(1 - \hat{y}(i))$, whose size is $10 \times 1$, then

$$\begin{aligned}
&\Delta W(i,j) = \Delta\hat{y}(i) \cdot f(j) \\
&\Longrightarrow \Delta W = \Delta\hat{y} \times f^T
\end{aligned}$$

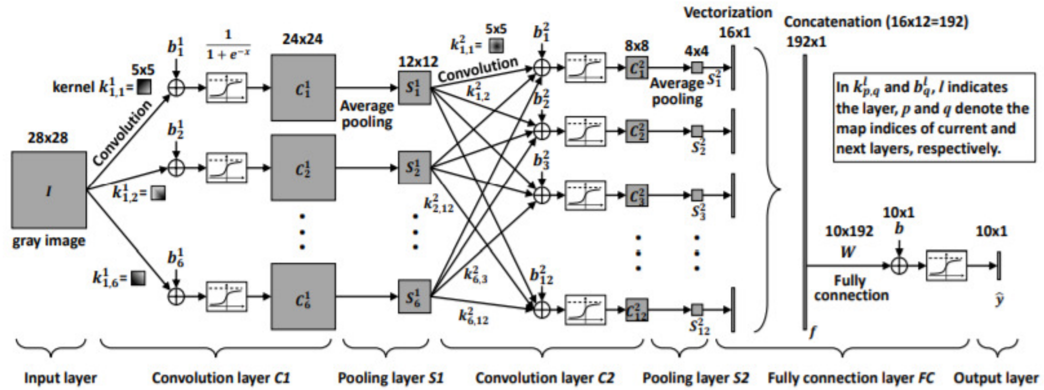
$$\boxed{\Longrightarrow \Delta W = \Delta\hat{y} \times f^T}$$

## $\Delta b$ (size $10 \times 1$)

$$
\begin{aligned}
\Delta b(i) &= \frac{\partial L}{\partial b(i)} \\
&= \frac{\partial L}{\partial \hat{y}(i)} \cdot \frac{\partial \hat{y}(i)}{\partial b(i)} \\
&= (\hat{y}(i) - y(i)) \cdot \frac{\partial}{\partial b(i)} \sigma \left( \sum_{j=1}^{192} W(i,j) \times f(j) + b(i) \right) \\
&= (\hat{y}(i) - y(i)) \cdot \hat{y}(i)(1 - \hat{y}(i))
\end{aligned}
$$

$$\boxed{\Longrightarrow \Delta b = \Delta \hat{y}}$$

1 **Feedforward**

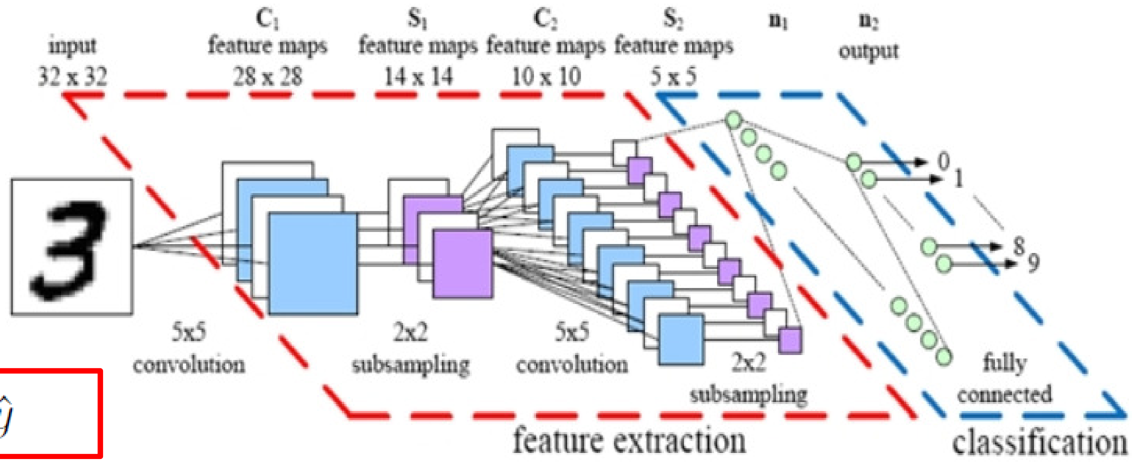



## $\Delta k^2_{p,q}$ (size $5 \times 5$)

Because of concatenation, vectorization, and pooling, we need to compute the backpropagation error $C^2_q$ on C2 layer before calculating $k^2_{p,q}$.

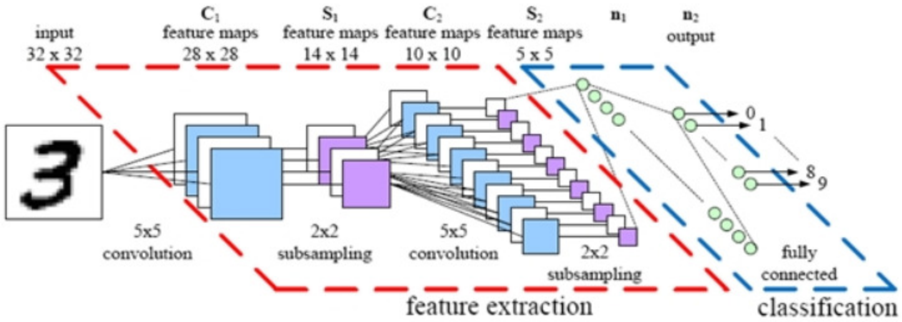
$$
\begin{aligned}
\Delta f(j) &= \frac{\partial L}{\partial f} \\
&= \sum_{i=1}^{10} \frac{\partial L}{\partial \hat{y}(i)} \cdot \frac{\partial \hat{y}(i)}{\partial f(j)} \\
&= (\hat{y}(i) - y(i)) \cdot \frac{\partial}{\partial f(j)} \sigma \left( \sum_{j=1}^{192} W(i,j) \times f(j) + b(i) \right) \\
&= \sum_{i=1}^{10} (\hat{y}(i) - y(i)) \cdot \hat{y}(i)(1 - \hat{y}(i)) \cdot W(i,j) \\
&= \sum_{i=1}^{10} \Delta \hat{y}(i) \cdot W(i,j)
\end{aligned}
$$

$$\boxed{\Longrightarrow \Delta f = W^T \times \Delta \hat{y}}$$



From section 1.6, we reshape the long error vector $\Delta f$ (size $192 \times 1$) by

$$\{\Delta S_q^2\}_{q=1,2,\cdots,12} = F^{-1}(\Delta f),$$




which gets the error on S2 layer (twelve $4 \times 4$ error maps)

upsampling is performed to obtain the error on C2 layer.

$$\Delta C_q^2(i,j) = \frac{1}{4}\Delta S_q^2\left(\lceil i/2 \rceil, \lceil j/2 \rceil\right), \; i,j = 1,2,\cdots,8$$

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| $\Delta C_q^2(1,1)$; $\Delta C_q^2(1,2)$ $\Delta C_q^2(2,1)$; $\Delta C_q^2(2,2)$ | $= \frac{1}{4}\Delta S_q^2(1,1)$ | $\Delta C_q^2(1,3)$; $\Delta C_q^2(1,4)$ $\Delta C_q^2(2,3)$; $\Delta C_q^2(2,4)$ | $= \frac{1}{4}\Delta S_q^2(1,2)$ | $\Delta C_q^2(1,5)$; $\Delta C_q^2(1,6)$ $\Delta C_q^2(2,5)$; $\Delta C_q^2(2,6)$ | $= \frac{1}{4}\Delta S_q^2(1,3)$ | $\Delta C_q^2(1,7)$; $\Delta C_q^2(1,8)$ $\Delta C_q^2(2,7)$; $\Delta C_q^2(2,8)$ | $= \frac{1}{4}\Delta S_q^2(1,4)$ |
| $\Delta C_q^2(3,1)$; $\Delta C_q^2(3,2)$ $\Delta C_q^2(4,1)$; $\Delta C_q^2(4,2)$ | $= \frac{1}{4}\Delta S_q^2(2,1)$ | $\Delta C_q^2(3,3)$; $\Delta C_q^2(3,4)$ $\Delta C_q^2(4,3)$; $\Delta C_q^2(4,4)$ | $= \frac{1}{4}\Delta S_q^2(2,2)$ | $\Delta C_q^2(3,5)$; $\Delta C_q^2(1,6)$ $\Delta C_q^2(4,5)$; $\Delta C_q^2(2,6)$ | $= \frac{1}{4}\Delta S_q^2(2,3)$ | $\Delta C_q^2(3,7)$; $\Delta C_q^2(3,8)$ $\Delta C_q^2(4,7)$; $\Delta C_q^2(4,8)$ | $= \frac{1}{4}\Delta S_q^2(2,4)$ |
| $\Delta C_q^2(5,1)$; $\Delta C_q^2(5,2)$ $\Delta C_q^2(6,1)$; $\Delta C_q^2(6,2)$ | $= \frac{1}{4}\Delta S_q^2(3,1)$ | $\Delta C_q^2(5,3)$; $\Delta C_q^2(5,4)$ $\Delta C_q^2(6,3)$; $\Delta C_q^2(6,4)$ | $= \frac{1}{4}\Delta S_q^2(3,2)$ | $\Delta C_q^2(5,5)$; $\Delta C_q^2(5,6)$ $\Delta C_q^2(6,5)$; $\Delta C_q^2(6,6)$ | $= \frac{1}{4}\Delta S_q^2(3,3)$ | $\Delta C_q^2(5,7)$; $\Delta C_q^2(5,8)$ $\Delta C_q^2(6,7)$; $\Delta C_q^2(6,8)$ | $= \frac{1}{4}\Delta S_q^2(3,4)$ |
| $\Delta C_q^2(7,1)$; $\Delta C_q^2(7,2)$ $\Delta C_q^2(8,1)$; $\Delta C_q^2(8,2)$ | $= \frac{1}{4}\Delta S_q^2(4,1)$ | $\Delta C_q^2(7,3)$; $\Delta C_q^2(7,4)$ $\Delta C_q^2(8,3)$; $\Delta C_q^2(8,4)$ | $= \frac{1}{4}\Delta S_q^2(4,2)$ | $\Delta C_q^2(7,5)$; $\Delta C_q^2(7,6)$ $\Delta C_q^2(8,5)$; $\Delta C_q^2(8,6)$ | $= \frac{1}{4}\Delta S_q^2(4,3)$ | $\Delta C_q^2(7,7)$; $\Delta C_q^2(7,8)$ $\Delta C_q^2(8,7)$; $\Delta C_q^2(8,8)$ | $= \frac{1}{4}\Delta S_q^2(4,4)$ |

$$\Delta k^2_{p,q}(u,v) = \frac{\partial L}{\partial k^2_{p,q}(u,v)} \tag{33}$$

$$= \sum_{i=1}^{8}\sum_{j=1}^{8} \frac{\partial L}{\partial C^2_q(i,j)} \cdot \frac{\partial C^2_q(i,j)}{\partial k^2_{p,q}(u,v)} \tag{34}$$

$$= \sum_{i=1}^{8}\sum_{j=1}^{8} \Delta C^2_q(i,j) \cdot \frac{\partial}{\partial k^2_{p,q}(u,v)} \sigma\left(\sum_{p=1}^{6}\sum_{u=-2}^{2}\sum_{v=-2}^{2} S^1_p(i-u,j-v)\cdot k^2_{p,q}(u,v) + b^2_q\right) \tag{35}$$

$$= \sum_{i=1}^{8}\sum_{j=1}^{8} \Delta C^2_q(i,j) \cdot C^2_q(i,j)\left(1 - C^2_q(i,j)\right)\cdot S^1_p(i-u,j-v) \tag{36}$$

$$\Delta C^2_{q,\sigma}(i,j) = \Delta C^2_q(i,j)\cdot C^2_q(i,j)\left(1 - C^2_q(i,j)\right) \qquad C^2_{q,\sigma}(i,j) = \sum_{p=1}^{6}\sum_{u=-2}^{2}\sum_{v=-2}^{2} S^1_p(i-u,j-v)\cdot k^2_{p,q}(u,v) + b^2_q$$

Rotating $S^1_p$ 180 degrees, we get $S^1_{p,rot180}$, thus $S^1_{p,rot180}(u-i, v-j) = S^1_p(i-u, j-v)$. $ROT180(w^{l+1}_{x,y}) = w^{l+1}_{-x,-y}$ .

$$\Delta k^2_{p,q}(u,v) = \frac{\partial L}{\partial k^2_{p,q}(u,v)} = \sum_{i=1}^{8}\sum_{j=1}^{8} \Delta C^2_q(i,j) \cdot C^2_q(i,j)\left(1 - C^2_q(i,j)\right)\cdot S^1_p(i-u,j-v)$$

$$= \sum_{i=1}^{8}\sum_{j=1}^{8} \Delta C^2_{q,\sigma}(i,j) \cdot S^1_p(i-u,j-u)$$

$\boldsymbol{W} =$

| | 1 | 2 | 3 | 4 |
|---|---|---|---|---|
| 1 | $W_{11}$ | $W_{12}$ | $W_{13}$ | $W_{14}$ |
| 2 | $W_{21}$ | $W_{22}$ | $W_{23}$ | $W_{24}$ |
| 3 | $W_{31}$ | $W_{32}$ | $W_{33}$ | $W_{34}$ |
| 4 | $W_{41}$ | $W_{42}$ | $W_{43}$ | $W_{44}$ |

$$W_{i,j} = \mathrm{Rot180}(W_{-i,-j})$$

$\mathrm{Rot180}(\boldsymbol{W}) =$

| | -4 | -3 | -2 | -1 |
|---|---|---|---|---|
| -4 | $W_{44}$ | $W_{43}$ | $W_{42}$ | $W_{41}$ |
| -3 | $W_{34}$ | $W_{33}$ | $W_{32}$ | $W_{31}$ |
| -2 | $W_{24}$ | $W_{23}$ | $W_{22}$ | $W_{21}$ |
| -1 | $W_{14}$ | $W_{13}$ | $W_{12}$ | $W_{11}$ |

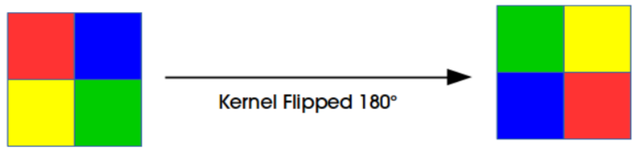
$$\mathrm{Rot180}(w_{-i,-j}) = w_{i,j}$$

Rotating $S^1{}_p$ 180 degrees, we get $S^1{}_{p,rot180}$, thus $S^1{}_{p,rot180}(u-i,\ v-j) = S^1{}_p\,(i-u,\ j-v)$.

$$\Delta k^2_{p,q}(u,v) = \frac{\partial L}{\partial k^2_{p,q}(u,v)} = \sum_{i=1}^{8}\sum_{j=1}^{8} \Delta C^2_q(i,j)\cdot C^2_q(i,j)\left(1 - C^2_q(i,j)\right)\cdot S^1_p(i-u,j-v)$$

$$= \sum_{i=1}^{8}\sum_{j=1}^{8} \Delta C^2_{q,\sigma}(i,j)\cdot S^1_p(i-u,j-v) \qquad \Delta C^2_{q,\sigma}(i,j) = \Delta C^2_q(i,j)\cdot C^2_q(i,j)\left(1 - C^2_q(i,j)\right)$$

$$= \sum_{i=1}^{8}\sum_{j=1}^{8} \Delta C^2_{q,\sigma}(i,j)\cdot S^1_p(-(u-i),-(v-j))$$




$$= \sum_{i=1}^{8}\sum_{j=1}^{8} \Delta C^2_{q,\sigma}(i,j)\cdot \mathrm{Rot180}(S^1_p)\big((u-i),(v-j)\big)$$

$$ROT180(w^{l+1}_{x,y}) = w^{l+1}_{-x,-y}\,.$$

the size of $\Delta S^2_q$ and $\Delta C^2_q$ are $4\times 4$ and $8\times 8$.

$$\Delta k^2_{p,q}(u,v) = \sum_{i=1}^{8}\sum_{j=1}^{8} S^1_{p,rot180}(u-i,v-j)\cdot \Delta C^2_{q,\sigma}(i,j)$$

$$\Longrightarrow \Delta k^2_{p,q} = S^1_{p,rot180} * \Delta C^2_{q,\sigma}$$

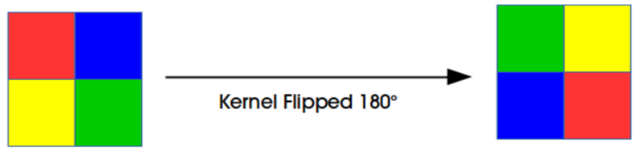
Rotating $S^1_p$ 180 degrees, we get $S^1_{p,rot180}$, thus $S^1_{p,rot180}(u-i,\ v-j) = S^1_p(i-u,\ j-v)$.

$$\Delta k^2_{p,q}(u,v) = \frac{\partial L}{\partial k^2_{p,q}(u,v)} = \sum_{i=1}^{8}\sum_{j=1}^{8} \Delta C^2_{q,\sigma}(i,j)\cdot S^1_p(i-u, j-v) = \sum_{i=1}^{8}\sum_{j=1}^{8} \Delta C^2_{q,\sigma}(i,j)\cdot S^1_p(-(u-i), -(v-j))$$

$$= \sum_{i=1}^{8}\sum_{j=1}^{8} \Delta C^2_{q,\sigma}(i,j)\cdot \mathrm{Rotation}(S^1_p)\big((u-i),(v-j)\big)$$

$$\Delta C^2_{q,\sigma}(i,j) = \Delta C^2_q(i,j)\cdot C^2_q(i,j)\left(1 - C^2_q(i,j)\right)$$

$$= \sum_{i=1}^{8}\sum_{j=1}^{8} \Delta C^2_{q,\sigma}(i,j)\cdot \mathrm{Rotation}(S^1_p)(i-u, j-v)$$




$$\Delta k^2_{p,q}(1,1) = \textstyle\sum_{i=1}^{8}\sum_{j=1}^{8} \Delta C^2_{q,\sigma}(i,j)\cdot S^1_p(i-1, j-1) =$$

$$\Delta C^2_{p,\sigma}(2,2)S^1_p(1,1) + \Delta C^2_{p,\sigma}(2,3)S^1_p(1,2) + \Delta C^2_{p,\sigma}(2,4)S^1_p(1,3) + \Delta C^2_{p,\sigma}(2,5)S^1_p(1,4) + \Delta C^2_{p,\sigma}(2,6)S^1_p(1,5)$$
$$+ \Delta C^2_{p,\sigma}(2,7)S^1_p(1,6) + \Delta C^2_{p,\sigma}(2,8)S^1_p(1,7) + \cdots\cdots +$$
$$\Delta C^2_{p,\sigma}(8,2)S^1_p(7,1) + \Delta C^2_{p,\sigma}(8,3)S^1_p(7,2) + \Delta C^2_{p,\sigma}(8,4)S^1_p(7,3) + \Delta C^2_{p,\sigma}(8,5)S^1_p(7,4) + \Delta C^2_{p,\sigma}(8,6)S^1_p(7,5)$$
$$+ \Delta C^2_{p,\sigma}(8,7)S^1_p(7,6) + \Delta C^2_{p,\sigma}(8,8)S^1_p(7,7)$$

$$\Delta k^2_{p,q}(1,1) = \textstyle\sum_{i=1}^{8}\sum_{j=1}^{8} \Delta C^2_{q,\sigma}(i,j)\cdot \mathrm{R}(S^1_p)(1-i, 1-j) =$$

$$\Delta C^2_{p,\sigma}(2,2)R(S^1_p)(-1,-1) + \Delta C^2_{p,\sigma}(2,3)R(S^1_p)(-1,-2) + \Delta C^2_{p,\sigma}(2,4)R(S^1_p)(-1,-3) + \Delta C^2_{p,\sigma}(2,5)R(S^1_p)(-1,-4)$$
$$+ \Delta C^2_{p,\sigma}(2,6)R(S^1_p)(-1,-5) + \Delta C^2_{p,\sigma}(2,7)R(S^1_p)(-1,-6) + \Delta C^2_{p,\sigma}(2,8)R(S^1_p)(-1,-7) + \cdots\cdots +$$
$$\Delta C^2_{p,\sigma}(8,2)R(S^1_p)(-7,-1) + \Delta C^2_{p,\sigma}(8,3)R(S^1_p)(-7,-2) + \Delta C^2_{p,\sigma}(8,4)R(S^1_p)(-7,-3) + \Delta C^2_{p,\sigma}(8,5)R(S^1_p)(-7,-4)$$
$$+ \Delta C^2_{p,\sigma}(8,6)R(S^1_p)(-7,-5) + \Delta C^2_{p,\sigma}(8,7)R(S^1_p)(-7,-6) + \Delta C^2_{p,\sigma}(8,8)R(S^1_p)(-7,-7)$$

### 2.4 $\Delta b_q^2$ (size $1 \times 1$)

$$
\begin{aligned}
\Delta b_q^2 &= \frac{\partial L}{\partial b_q^2} \\
&= \sum_{i=1}^{8} \sum_{j=1}^{8} \frac{\partial L}{\partial C_q^2(i,j)} \cdot \frac{\partial C_q^2(i,j)}{\partial b_q^2} \\
&= \sum_{i=1}^{8} \sum_{j=1}^{8} \Delta C_q^2(i,j) \cdot \frac{\partial}{\partial b_q^2} \sigma \left( \sum_{p=1}^{6} \sum_{u=-2}^{2} \sum_{v=-2}^{2} S_p^1(i-u, j-v) \cdot k_{p,q}^2(u,v) + b_q^2 \right) \\
&= \sum_{i=1}^{8} \sum_{j=1}^{8} \Delta C_q^2(i,j) \cdot C_q^2(i,j) \left(1 - C_q^2(i,j)\right) \\
&= \sum_{i=1}^{8} \sum_{j=1}^{8} \Delta C_{q,\sigma}^2(i,j)
\end{aligned}
$$

$$
\begin{aligned}
\Delta S_p^1(i,j) &= \frac{\partial L}{\partial S_p^1(i,j)} && (46)\\
&= \sum_{q=1}^{12}\sum_{u=-2}^{2}\sum_{v=-2}^{2} \frac{\partial L}{\partial C_{q,\sigma}^2(i+u,j+v)} \cdot \frac{\partial C_{q,\sigma}^2(i+u,j+v)}{\partial S_p^1(i,j)} && (47)\\
&= \sum_{q=1}^{12}\sum_{u=-2}^{2}\sum_{v=-2}^{2} \Delta C_{q,\sigma}^2(i+u,j+v) \cdot \frac{\partial}{\partial S_p^1(i,j)} \left( \sum_{p=1}^{6}\sum_{u=-2}^{2}\sum_{v=-2}^{2} S_p^1(i,j) \cdot k_{p,q}^2(u,v) + b_q^2 \right) && (48)\\
&= \sum_{q=1}^{12}\sum_{u=-2}^{2}\sum_{v=-2}^{2} \Delta C_{q,\sigma}^2(i+u,j+v) \cdot k_{p,q}^2(u,v) && (49)
\end{aligned}
$$

Rotating $k_{p,q}^2$ 180 degrees, we get $k_{p,q,rot180}^2(-u,-v) = k_{p,q}^2(u,v)$

$$
\Delta S_p^1(i,j) = \sum_{q=1}^{12}\sum_{u=-2}^{2}\sum_{v=-2}^{2} \Delta C_{q,\sigma}^2(i-(-u), j-(-v)) \cdot k_{p,q,rot180}^2(-u,-v)
$$

$$
\Longrightarrow \Delta S_p^1 = \sum_{q=1}^{12} \Delta C_{q,\sigma}^2 * k_{p,q,rot180}^2 \qquad \Delta C_p^1(i,j) = \frac{1}{4}\Delta S_p^1\left(\lceil i/2\rceil, \lceil j/2\rceil\right),\ i,j = 1,2,\cdots,24
$$

$$
\begin{aligned}
\Delta k^1_{1,p}(u,v) &= \frac{\partial L}{\partial k^1_{1,p}(u,v)} \\
&= \sum_{i=1}^{24}\sum_{j=1}^{24} \frac{\partial L}{\partial C^1_p(i,j)} \cdot \frac{\partial C^1_p(i,j)}{\partial k^1_{1,p}(u,v)} \\
&= \sum_{i=1}^{24}\sum_{j=1}^{24} \Delta C^1_p(i,j) \cdot \frac{\partial}{\partial k^1_{1,p}(u,v)} \sigma\left(\sum_{u=-2}^{2}\sum_{v=-2}^{2} I(i-u, j-v) \cdot k^1_{1,p}(u,v) + b^1_p\right) \\
&= \sum_{i=1}^{24}\sum_{j=1}^{24} \Delta C^1_p(i,j) \cdot C^1_p(i,j)\left(1 - C^1_p(i,j)\right) \cdot I(i-u, j-v)
\end{aligned}
$$

$$
\Delta C^1_{p,\sigma}(i,j) = \Delta C^1_p(i,j) \cdot C^1_p(i,j)\left(1 - C^1_p(i,j)\right)
$$

$$
\begin{aligned}
&\Delta k^1_{1,p}(u,v) = \sum_{i=1}^{24}\sum_{j=1}^{24} I_{rot180}(u-i, v-j) \cdot \Delta C^1_{p,\sigma}(i,j) \\
&\implies \Delta k^1_{1,p} = I_{rot180} * \Delta C^1_{p,\sigma}
\end{aligned}
$$

## 2.6 $\Delta b_p^1$ (size $1 \times 1$)

$$
\begin{aligned}
\Delta b_p^1 &= \frac{\partial L}{\partial b_p^1} \\
&= \sum_{i=1}^{24} \sum_{j=1}^{24} \frac{\partial L}{\partial C_p^1(i,j)} \cdot \frac{\partial C_p^1(i,j)}{\partial b_p^1} \\
&= \sum_{i=1}^{24} \sum_{j=1}^{24} \Delta C_p^1(i,j) \cdot \frac{\partial}{\partial b_p^1} \sigma \left( \sum_{u=-2}^{2} \sum_{v=-2}^{2} I(i-u, j-v) \cdot k_{1,p}^1(u,v) + b_p^1 \right) \\
&= \sum_{i=1}^{24} \sum_{j=1}^{24} \Delta C_p^1(i,j) \cdot C_p^1(i,j) \left( 1 - C_p^1(i,j) \right) \\
&= \sum_{i=1}^{24} \sum_{j=1}^{24} \Delta C_{p,\sigma}^1(i,j)
\end{aligned}
$$

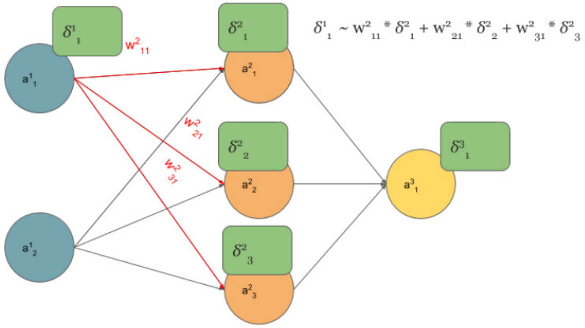
# Question

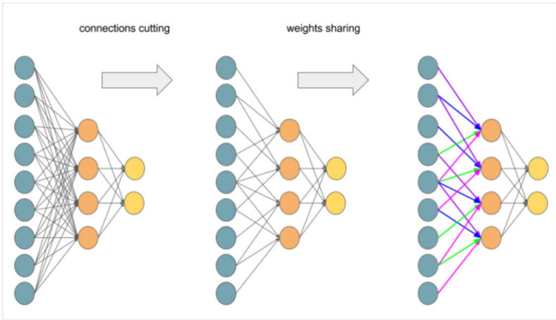
It was a little consoling, when I found out that I am not alone, for example: Hello, when computing the gradients CNN, the weights need to be rotated, Why ?

$$\delta_j^\ell = f'(\mathbf{u}_j^\ell) \circ \mathrm{conv2}(\delta_j^{\ell+1}, \mathrm{rot180}(\mathbf{k}_j^{\ell+1}), \text{'full'}).$$

The answer on above question, that concerns the need of rotation on weights in gradient computing, will be a result of this long post.

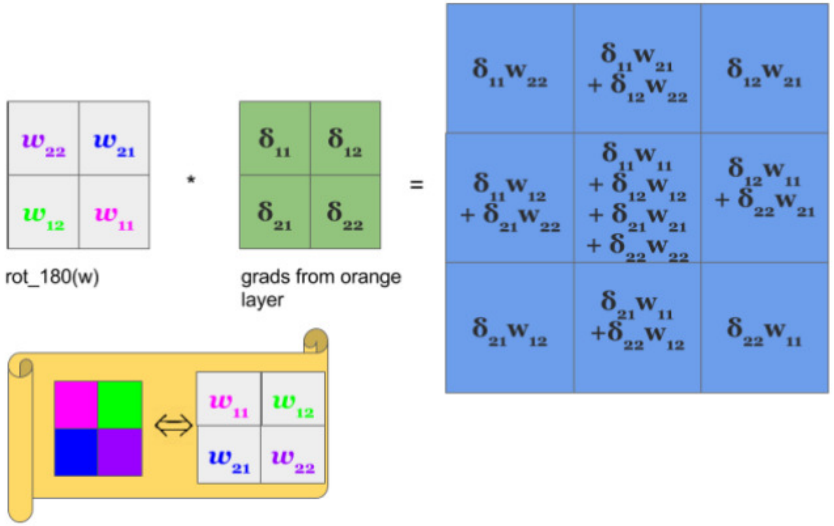
https://grzegorzgwardys.wordpress.com/2016/04/22/8/

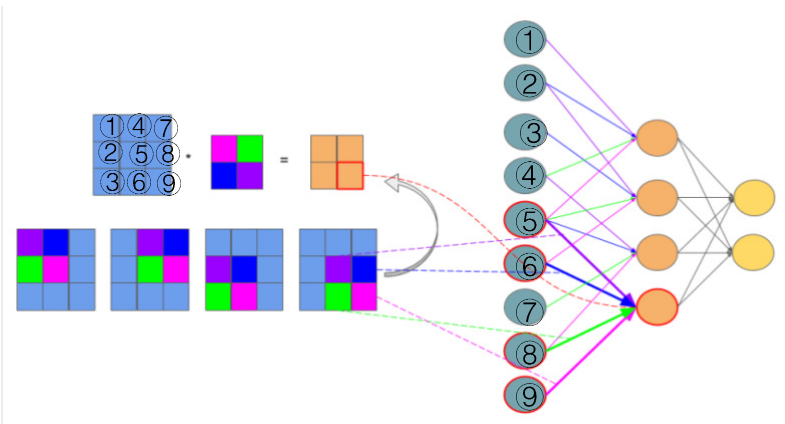


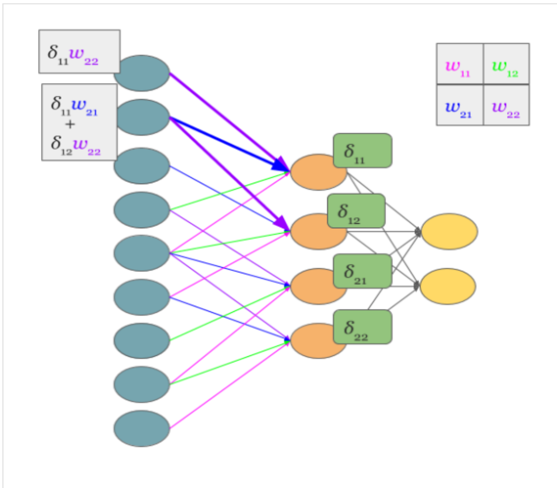



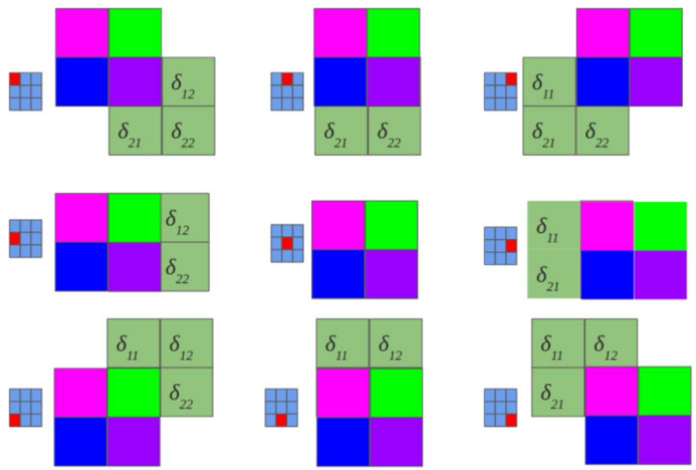
Transforming Multilayer Perceptron to Convolutional Neural Network

If you are not sure that after connections cutting and weights sharing we get





Feedforward in CNN is identical with convolution operation


Backpropagation also results with convolution



$z_j$ $z_{x,y}$

In the standard MLP, we can define an error of neuron j as:

$$\delta_j^l = \frac{\partial C}{\partial z_j^l} \qquad z_j^l = \sum_k w_{jk}^l a_k^{l-1} + b_j^l \qquad a_j^l = \sigma(z_j^l)$$

But here, we do not have MLP but CNN and matrix multiplications are replaced by convolutions as we discussed before. So instead of we do have a

$$z_{x,y}^{l+1} = w^{l+1} * \sigma(z_{x,y}^l) + b_{x,y}^{l+1} = \sum_a \sum_b w_{a,b}^{l+1} \sigma(z_{x-a,y-b}^l) + b_{x,y}^{l+1}$$

$$\delta_{x,y}^l = \frac{\partial C}{\partial z_{x,y}^l} = \sum_{x'} \sum_{y'} \frac{\partial C}{\partial z_{x',y'}^{l+1}} \frac{\partial z_{x',y'}^{l+1}}{\partial z_{x,y}^l}$$

$$\frac{\partial C}{\partial z_{x,y}^l} = \sum_{x'} \sum_{y'} \frac{\partial C}{\partial z_{x',y'}^{l+1}} \frac{\partial z_{x',y'}^{l+1}}{\partial z_{x,y}^l} = \sum_{x'} \sum_{y'} \delta_{x',y'}^{l+1} \frac{\partial (\sum\limits_a \sum\limits_b w_{a,b}^{l+1} \sigma(z_{x'-a,y'-b}^l) + b_{x',y'}^{l+1})}{\partial z_{x,y}^l}$$

$$\sum_{x'} \sum_{y'} \delta_{x',y'}^{l+1} \frac{\partial (\sum\limits_a \sum\limits_b w_{a,b}^{l+1} \sigma(z_{x'-a,y'-b}^l) + b_{x',y'}^{l+1})}{\partial z_{x,y}^l} = \sum_{x'} \sum_{y'} \delta_{x',y'}^{l+1} w_{a,b}^{l+1} \sigma'(z_{x,y}^l)$$

$$\sum_{x'}\sum_{y'}\delta^{l+1}_{x',y'}\frac{\partial(\sum_a\sum_b w^{l+1}_{a,b}\sigma(z^l_{x'-a,y'-b})+b^{l+1}_{x',y'})}{\partial z^l_{x,y}} = \sum_{x'}\sum_{y'}\delta^{l+1}_{x',y'}w^{l+1}_{a,b}\sigma'(z^l_{x,y})$$

NN error 传递

$$\boxed{\delta^l = \sigma'(z^l)\bullet(W^{l+1})^T\delta^{l+1}}$$

$$\sum_{x'}\sum_{y'}\delta^{l+1}_{x',y'}w^{l+1}_{a,b}\sigma'(z^l_{x,y}) = \sum_{x'}\sum_{y'}\delta^{l+1}_{x',y'}w^{l+1}_{x'-x,y'-y}\sigma'(z^l_{x,y})$$

$$\sum_{x'}\sum_{y'}\delta^{l+1}_{x',y'}w^{l+1}_{x'-x,y'-y}\sigma'(z^l_{x,y}) = \boxed{\delta^{l+1} * w^{l+1}_{-x,-y}\sigma'(z^l_{x,y})} = \delta^l_{x,y} = \frac{\partial C}{\partial z^l_{x,y}}$$

**the rotation of the weights just results from derivation of delta error in Convolution Neural Network.**

CNN error 传递

$$\frac{\partial C}{\partial w^l_{a,b}} = \sum_x\sum_y \frac{\partial C}{\partial z^l_{x,y}}\frac{\partial z^l_{x,y}}{\partial w^l_{a,b}} = \sum_x\sum_y \delta^l_{x,y}\frac{\partial(\sum_{a'}\sum_{b'} w^l_{a',b'}\sigma(z^l_{x-a',y-b'})+b^l_{x,y})}{\partial w^l_{a,b}} =$$

$$\sum_x\sum_y \delta^l_{x,y}\sigma(z^{l-1}_{x-a,y-b}) = \delta^l_{a,b}*\sigma(z^{l-1}_{-a,-b}) = \delta^l_{a,b}*\sigma(ROT180(z^{l-1}_{a,b}))$$

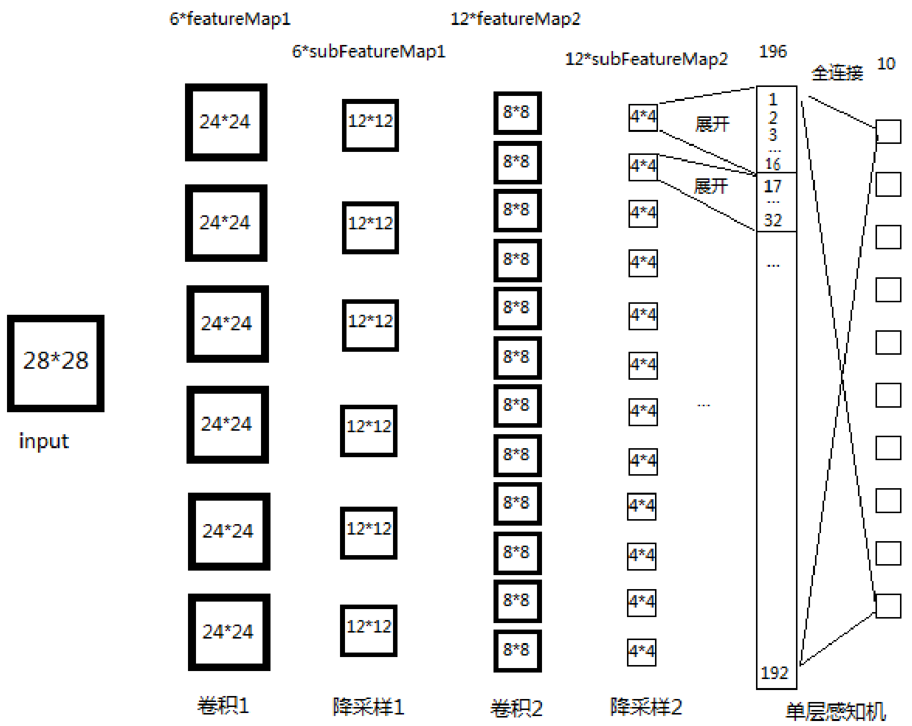
$$ROT180(w^{l+1}_{x,y}) = w^{l+1}_{-x,-y}$$

So **the answer** on question Hello, when computing the gradients CNN, the weights need to be rotated, Why ? is simple: **the rotation of the weights just results from derivation of delta error in Convolution Neural Network**.

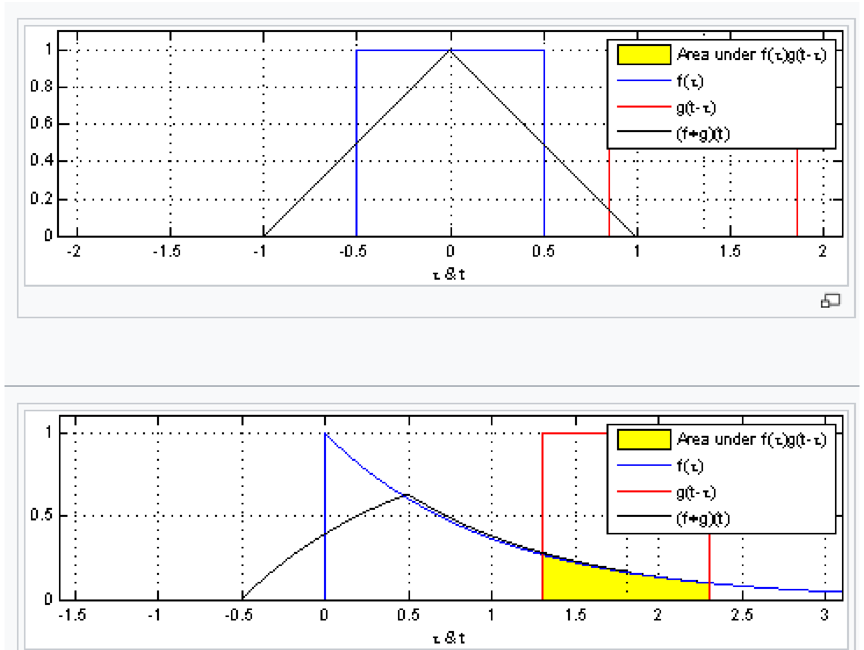
$$\frac{\partial C}{\partial w^l_{a,b}} = \sum_x \sum_y \frac{\partial C}{\partial z^l_{x,y}} \frac{\partial z^l_{x,y}}{\partial w^l_{a,b}} = \sum_x \sum_y \delta^l_{x,y} \frac{\partial(\sum_{a'} \sum_{b'} w^l_{a',b'} \sigma(z^l_{x-a',y-b'}) + b^l_{x,y})}{\partial w^l_{a,b}} =$$
$$\sum_x \sum_y \delta^l_{x,y} \sigma(z^{l-1}_{x-a,y-b}) = \delta^l_{a,b} * \sigma(z^{l-1}_{-a,-b}) = \delta^l_{a,b} * \sigma(ROT180(z^{l-1}_{a,b}))$$

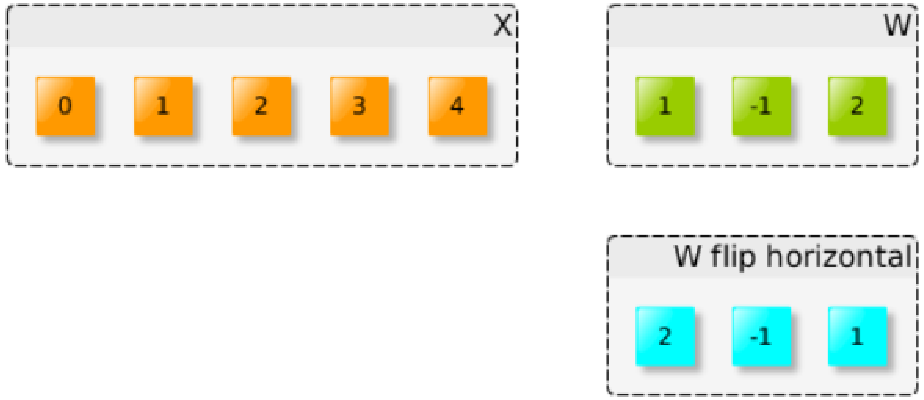
So paraphrasing the backpropagation algorithm for CNN:

1. Input x: set the corresponding activation for the input layer.
2. Feedforward: for each l = 2,3, ...,L compute $z^l_{x,y} = w^l * \sigma(z^{l-1}_{x,y}) + b^l_{x,y}$ and $a^l_{x,y} = \sigma(z^l_{x,y})$
3. Output error : Compute the vector $\delta^L = \nabla_a C \odot \sigma'(z^L)$
4. Backpropagate the error: For each l=L-1,L-2,...,2 compute $\delta^l_{x,y} = \delta^{l+1} * ROT180(w^{l+1}_{x,y}) \sigma'(z^l_{x,y})$
5. Output: The gradient of the cost function is given by $\frac{\partial C}{\partial w^l_{a,b}} = \delta^l_{a,b} * \sigma(ROT180(z^{l-1}_{a,b}))$

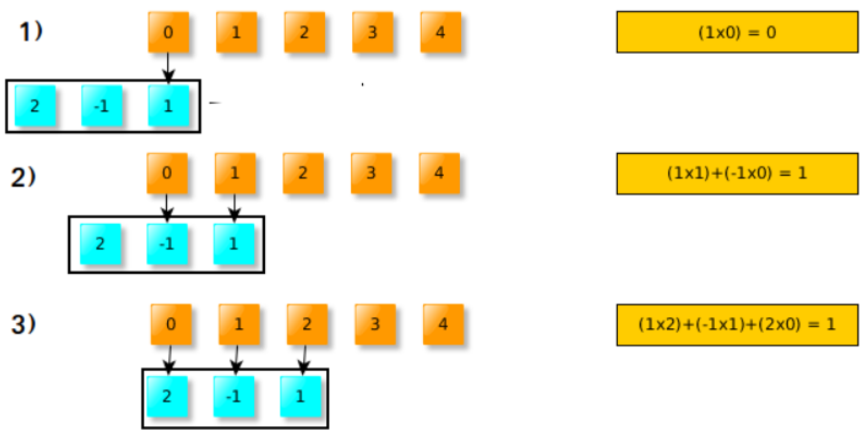


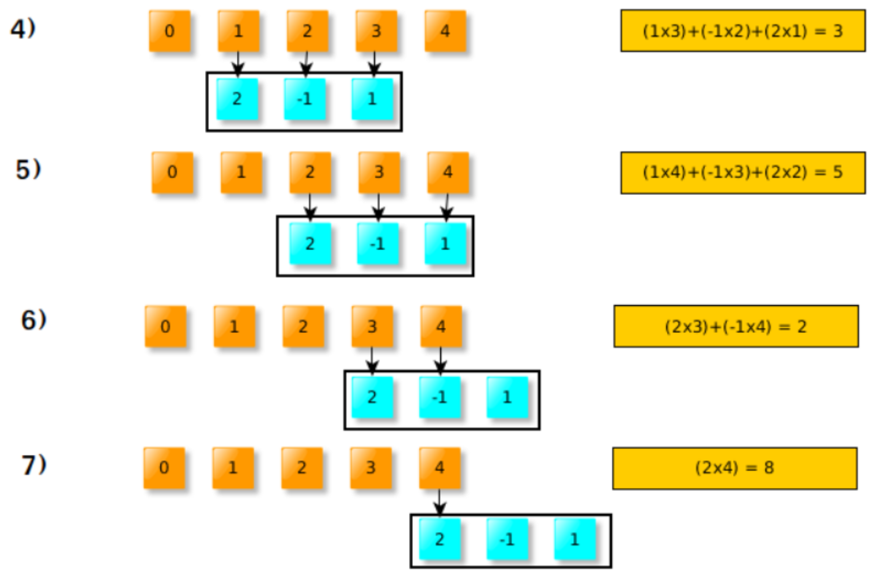

```
cnn.layers = {
    struct('type','i') %input layer
    struct('type','c','outputmaps',6,'kernelsize',5) % convolution layer
    struct('type','s','scale',2) %sub sampling layer
    struct('type','c','outputmaps',12,'kernelsize',5) % convolutional layer
    struct('type','s','scale',2) % sub sampling layer
%% 训练选项，alpha学习效率（不用），batchsiaze批训练总样本的数量，numepoches迭代次数
opts.alpha = 1;
opts.batchsize = 50;
```

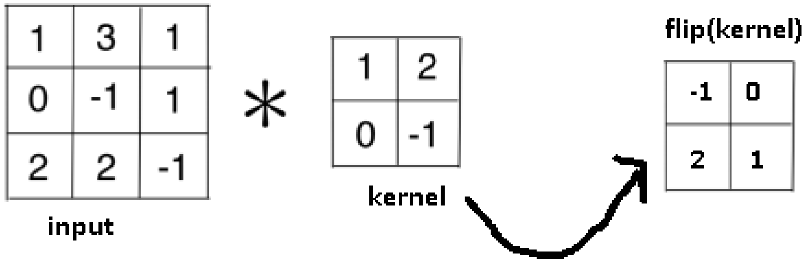


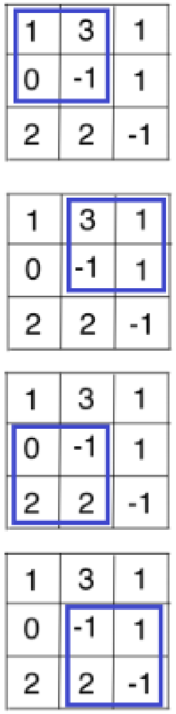

After that we need to slide the flipped W over the input X






Multiply the window element by element with flip(kernel), then sum the results



=> 1(-1)+3(0)+0(2)-1(1) = -2

=> 3(-1)+1(0)-1(2)+1(1) = -4

=> 0(-1)-1(0)+2(2)+2(1) = 6

=> -1(-1)+1(0)+2(2)-1(1) = 4

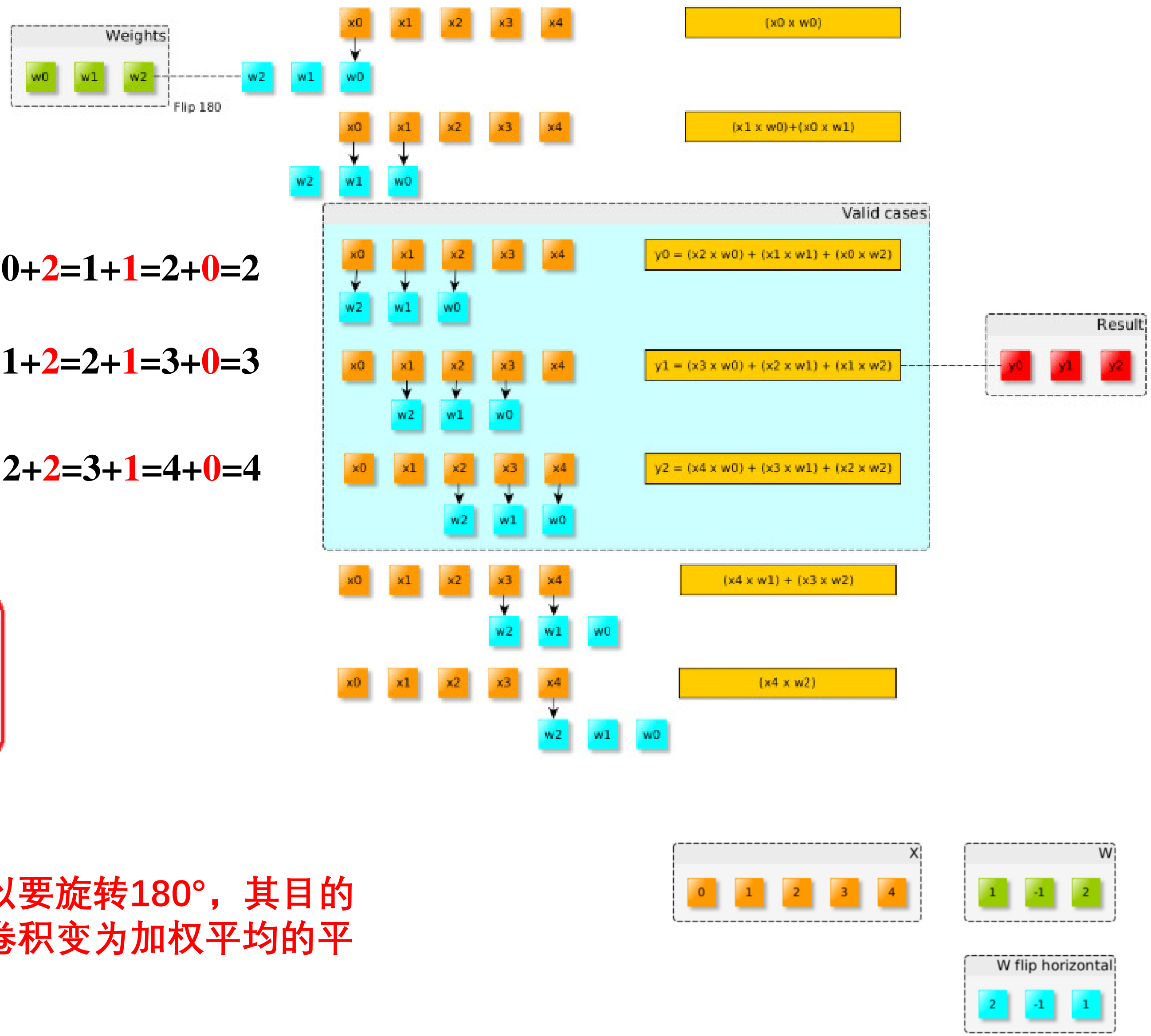
result(valid)

| | |
|---|---|
| -2 | -4 |
| 6 | 4 |

0+2=1+1=2+0=2

1+2=2+1=3+0=3

2+2=3+1=4+0=4




注：之所以要旋转180°，其目的无非是把卷积变为加权平均的平移