The following processing aims at a mathematical and arithmatical reasoning of the backpropagation progress for the network in architecture 1 by Shaun.

gradient of `kernels`

$$\frac{\partial J}{\partial k_{n,i,j}} = \frac{\partial L}{\partial v_k^{(-1)}} \frac{\partial v_k^{(-1)}}{\partial y^{(-2)}} \frac{\partial y^{(-2)}}{\partial y^{(-3)}} \frac{\partial y^{(-3)}}{\partial k_{n,i,j}} \qquad , n \in \{1, \ldots, 20\}, k_{n,i,j} \in [9, 9]$$

The expression extended by **the chain rule** could be calculated in a macro approach, which cnovert the operation upon single value into the operation of the whole kernels and vectors.

$$\left[\frac{\partial J}{\partial k_{n,i,j}} \right]_{9\times 9} = \left[\frac{\partial L}{\partial y^{(-3)}} \right]_{20\times 20} * \left[x \right]_{28\times 28} \qquad , n \in \{1, \ldots, 20\}$$

The matrix $\left[\frac{\partial J}{\partial k_{n,i,j}} \right]_{9\times 9}$ to be convolved is backward propagated (say) from the classifier subnetwork, reshaped from a 2000-dimensional vector to twenty 10 square `backprop feature map' (say) and RE-pooled to another twenty 20 square `backprop feature map', which factually lies the matrix to be convolved.

> Attension: ensure the kernels $\left[k_{n,i,j} \right]_{9\times 9}$ rotted when convolution in the progress of forward propagation.

gradient of `weight1`

$$\begin{aligned} \frac{\partial J}{\partial w_{i,j}^{(-1)}} &= \frac{\partial L}{\partial v_k} \frac{\partial v_k}{\partial y_i^{(-1)}} \frac{\partial y_i^{(-1)}}{\partial v_i^{(-1)}} \frac{\partial v_i^{(-1)}}{\partial w_{i,j}^{(-1)}} \qquad , w_{i,j}^{(-1)} \in [100, 20000] \\ &= (y_i - d_i) \cdot \sum_k w_{k,i} \cdot \varphi'_{(-1)}(v_i^{(-1)}) \cdot x_j^{(-1)} \end{aligned}$$

gradient of `weight`

$$\begin{aligned} \frac{\partial J}{\partial w_{i,j}} &= \frac{\partial L}{\partial y_i} \frac{\partial y_i}{\partial v_i} \frac{\partial v_i}{\partial w_{i,j}} \qquad , w_{i,j} \in [10, 100] \\ &= (y_i - d_i) x_j \qquad , v_i = \sum_j w_{i,j} x_j \end{aligned}$$

processing the gradient of softmax

- functional defination:

$$S(y_k) = \frac{\exp y_k}{\sum\limits_n \exp y_n}$$

- gradient analysis:

$$\frac{\partial S_k}{\partial y_i} = \begin{cases} \dfrac{\partial S_k}{\partial y_k} & , i = k \\ \\ \dfrac{\partial S_k}{\partial y_i} & , i \neq k \end{cases}$$

$$= \begin{cases} \dfrac{\frac{\partial \exp y_k}{\partial y_k} \cdot \sum\limits_n \exp y_n - \exp y_k \cdot \partial\Big(\sum\limits_n \exp y_n\Big)/\partial y_k}{\Big(\sum\limits_n \exp y_n\Big)^2} & , i = k \\ -\dfrac{\exp y_k}{\Big(\sum\limits_n \exp y_n\Big)^2} \cdot \dfrac{\partial \sum\limits_n \exp y_n}{\partial y_i} & , i \neq k \end{cases} \quad , S_k = S(y_k), i, k \in \{1, \cdots, n\}$$

$$= \begin{cases} \dfrac{\exp y_k \cdot \sum\limits_n \exp y_n - \exp^2 y_k}{\Big(\sum\limits_n \exp y_n\Big)^2} & , i = k \\ -\dfrac{\exp y_k}{\Big(\sum\limits_n \exp y_n\Big)^2} \cdot \exp y_i & , i \neq k \end{cases}$$

$$= \begin{cases} \dfrac{\exp y_i}{\sum\limits_n \exp y_n}\left(1 - \dfrac{\exp y_i}{\sum\limits_n \exp y_n}\right) & , i = k \\ -\dfrac{\exp y_k}{\sum\limits_n \exp y_n} \cdot \dfrac{\exp y_i}{\sum\limits_n \exp y_n} & , i \neq k \end{cases}$$

$$= \begin{cases} S_i(1 - S_i) & , i = k \\ \\ -S_k \cdot S_i & , i \neq k \end{cases}$$

> Correcting: the expression y here shall the result of linear transformation from full connnection matrix, which usually remarked as v.

processing the gradient of cross entropy with softmax

- functional defination of cross entropy:

$$L(d, y) = \sum_i -d_i \log y_i$$

- gradient analysis. of cross entropy with softmax:

$$\begin{aligned}
\frac{\partial L}{\partial v_i} &= -d_i \cdot \frac{1}{y_i} \cdot \frac{\partial y_i}{\partial v_i} - \sum_{i \neq k} d_i \cdot \frac{1}{y_k} \cdot \frac{y_k}{v_i} \qquad\qquad , y = S(v) \\
&= -d_i \cdot \frac{S_i(1 - S_i)}{S_i} - \sum_{i \neq k} -d_i \cdot \frac{S_i \cdot S_k}{S_k} \\
&= \sum_i d_i \cdot S_k - d_i \\
&= y_i - d_i
\end{aligned}$$