The Derivatives in the Fully Connected Artificial Neural Network

Yaoyu Hu, Ph.D.
Shanghai Jiao Tong University, Shanghai, China.
huyaoyu@sjtu.edu.cn
Jun. 23th, 2017

Two layers of neural networks are illustrated in Fig. 1. b_n^j is not listed in Fig. 1.

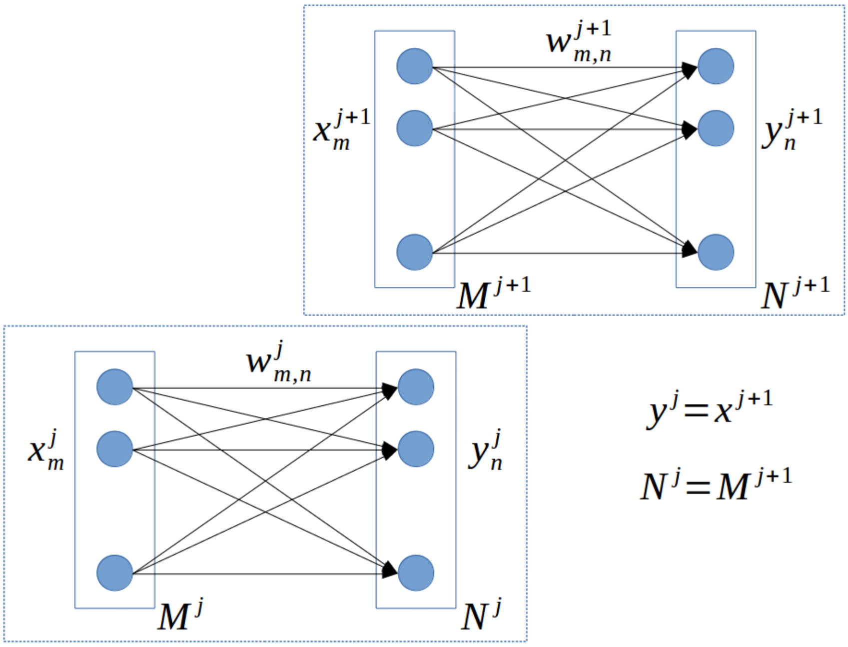

Fig. 1: Two layers of neural network.

The following relations are defined.

$$y_n^j = f\left(\sum_m^{M^j} w_{m,n}^j x_m^j + b_n^j\right) \qquad (1)$$

where *f*() is the activation function.

$$f(x) = \begin{cases} x & x \geqslant 0 \\ 0 & x < 0 \end{cases} \qquad (2)$$

If we define

$$g_n^j = \sum_m^{M^j} w_{m,n}^j x_m^j + b_n^j \qquad (3)$$

Then we have

$$y_n^j = f(g_n^j) \tag{4}$$

Let L be the loss function, J be the last index of neural network layer and $\mathbf{Y}$ be the training value of the neural network. Taking the sum-of-squares loss function as an example.

$$L = \frac{1}{2}\sum_n^{N^J}(y_n^J - Y_n)^2 \tag{5}$$

With $j = J$

$$\frac{\partial L}{\partial y_n^J} = (y_n^J - Y_n) \tag{6}$$

$$\frac{\partial L}{\partial x_m^J} = \sum_n^{N^J}\frac{\partial L}{\partial y_n^J}\frac{\partial y_n^J}{\partial g_n^J}\frac{\partial g_n^J}{\partial x_m^J} = \sum_n^{N^J}\frac{\partial L}{\partial y_n^J}\frac{\partial y_n^J}{\partial g_n^J}w_{m,n}^J \tag{7}$$

$$\frac{\partial L}{\partial w_{m,n}^J} = \frac{\partial L}{\partial y_n^J}\frac{\partial y_n^J}{\partial g_n^J}\frac{\partial g_n^J}{\partial w_{m,n}^J} = \frac{\partial L}{\partial y_n^J}\frac{\partial y_n^J}{\partial g_n^J}x_{m,n}^J \tag{8}$$

$$\frac{\partial L}{\partial b_n^J} = \frac{\partial L}{\partial y_n^J}\frac{\partial y_n^J}{\partial g_n^J}\frac{\partial g_n^J}{\partial b_n^J} = \frac{\partial L}{\partial y_n^J}\frac{\partial y_n^J}{\partial g_n^J} \tag{9}$$

With arbitrary $j <= J - 1$

$$\frac{\partial L}{\partial y_n^j} = \frac{\partial L}{\partial x_n^{j+1}} \tag{10}$$

$$\frac{\partial L}{\partial x_m^j} = \sum_n^{N^j}\frac{\partial L}{\partial y_n^j}\frac{\partial y_n^j}{\partial g_n^j}\frac{\partial g_n^j}{\partial x_m^j} = \sum_n^{N^j}\underbrace{\frac{\partial L}{\partial y_n^j}\frac{\partial y_n^j}{\partial g_n^j}}_{*}w_{m,n}^j \tag{11}$$

$$\frac{\partial L}{\partial w_{m,n}^j} = \frac{\partial L}{\partial y_n^j}\frac{\partial y_n^j}{\partial g_n^j}\frac{\partial g_n^j}{\partial w_{m,n}^j} = \underbrace{\frac{\partial L}{\partial y_n^j}\frac{\partial y_n^j}{\partial g_n^j}}_{*}x_m^j \tag{12}$$

$$\frac{\partial L}{\partial b_n^j} = \frac{\partial L}{\partial y_n^j}\frac{\partial y_n^j}{\partial g_n^j}\frac{\partial g_n^j}{\partial b_n^j} = \underbrace{\frac{\partial L}{\partial y_n^j}\frac{\partial y_n^j}{\partial g_n^j}}_{*} \tag{13}$$

Let the partial derivative of L respect to y^j be in the matrix form of

$$\left\{\frac{\partial L}{\partial y_n}\right\}^j_{N^j\times 1} \tag{14}$$

where $N^j \times 1$ is the dimension specification. Similarly, we could write a matrix expression related to $\partial y_n / \partial g_n$

$$\left[\frac{\partial y_n}{\partial g_n}\right]^j_{N\times N} = \begin{bmatrix} \frac{\partial y_1}{\partial g_1} & 0 & \cdots & 0 \\ 0 & \frac{\partial y_2}{\partial g_2} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \frac{\partial y_N}{\partial g_N} \end{bmatrix}^j_{N^j\times N^j} \tag{15}$$

We define four parameter matrices

$$\{y_n\}^j_{N^j\times 1} \tag{16}$$

$$\{Y_n\}^J_{N^J\times 1} \tag{17}$$

$$\{x_m\}^j_{M^j\times 1} \tag{18}$$

$$[w_{m,n}]^j_{M^j\times N^j} \tag{19}$$

Then we have

$$\left\{\frac{\partial L}{\partial x_m}\right\}^j_{M^j\times 1} = \left\{\sum_n^{N^j} \frac{\partial L}{\partial y_n^j}\frac{\partial y_n^j}{\partial g_n^j} w_{m,n}^j\right\}^j_{M^j\times 1} = [w_{m,n}]^j_{M^j\times N^j}\left[\frac{\partial y_n}{\partial g_n}\right]^j_{N^j\times N^j}\left\{\frac{\partial L}{\partial y_n}\right\}^j_{N^j\times 1} \tag{20}$$

$$\left[\frac{\partial L}{\partial w_{m,n}}\right]^j_{M^j\times N^j} = \left[\frac{\partial L}{\partial y_n^j}\frac{\partial y_n^j}{\partial g_n^j} x_m^j\right]^j_{M^j\times N^j} = \{x_m\}^j_{M^j\times 1}\left[\left[\frac{\partial y_n}{\partial g_n}\right]^j_{N^j\times N^j}\left\{\frac{\partial L}{\partial y_n}\right\}^j_{N^j\times 1}\right]^{\mathrm{T}} \tag{21}$$

$$\left\{\frac{\partial L}{\partial b_n}\right\}^j_{N^j\times 1} = \left\{\frac{\partial L}{\partial y_n}\frac{\partial y_n}{\partial g_n}\right\}^j_{N\times 1} = \left[\frac{\partial y_n}{\partial g_n}\right]^j_{N^j\times N^j}\left\{\frac{\partial L}{\partial y_n}\right\}^j_{N^j\times 1} \tag{22}$$

And it is obvious that

$$\left\{\frac{\partial L}{\partial y_n}\right\}^J_{N^J\times 1} = \{y_n - Y_n\}^J_{N^J\times 1} \tag{23}$$

Note that in Eq. (23) j = J. We could conveniently define the following column vector

$$\left\{\frac{\partial L}{\partial g_n}\right\}^j_{N^j\times 1}=\left[\frac{\partial y_n}{\partial g_n}\right]^j_{N^j\times N^j}\left\{\frac{\partial L}{\partial y_n}\right\}^j_{N^j\times 1}=\left\{\frac{\partial L}{\partial y_n}\frac{\partial y_n}{\partial g_n}\right\}^j_{N^j\times 1} \tag{24}$$

Then

$$\left\{\frac{\partial L}{\partial x_m}\right\}^j_{M^j\times 1}=\left[w_{m,n}\right]^j_{M^j\times N^j}\left\{\frac{\partial L}{\partial g_n}\right\}^j_{N^j\times 1} \tag{25}$$

$$\left[\frac{\partial L}{\partial w_{m,n}}\right]^j_{M^j\times N^j}=\left\{x_m\right\}^j_{M^j\times 1}\left[\left\{\frac{\partial L}{\partial g_n}\right\}^j_{N^j\times 1}\right]^{\mathrm{T}} \tag{26}$$

$$\left\{\frac{\partial L}{\partial b_n}\right\}^j_{N^j\times 1}=\left\{\frac{\partial L}{\partial g_n}\right\}^j_{N^j\times 1} \tag{27}$$

It turns out that if we take the terms marked by * in Eq. (12) to (14) as whole column vectors, then the results of Eq. (25) to (26) should fall out naturally.

Pseudo-code of Backscatter Propagation

Table 1.Pseudo-code for calculating the gradients.

```
Calculate Eq. (23) with j = J.
Initialize  ∂L/∂y_n^j
Start from j = J loop until j < 0:
    Evaluate Eq. (24) based on current  ∂L/∂y_n^j .
    Use Eq. (26) and Eq. (27) to obtain the gradients.
    if j != 0:
        Use Eq. (25) to get  ∂L/∂y_n^(j-1)  for the previous pair of neural networks.
    j = j - 1
```

For cross entropy loss function with Sotfmax function, the Sotfmax function is defined as

$$z_n=\frac{e^{y_n^J}}{\sum_n e^{y_n^J}} \tag{28}$$

Then the cross entropy loss function is expressed as

$$L=-\sum_n Y_n^J \ln\left(z_n\right) \tag{29}$$

In order to evaluate $\partial L / \partial y_n^J$ the derivative of the Softmax function should be obtained.

$$\frac{\partial z_n}{\partial y_i^J} = \begin{cases} z_i - z_i^2 & n = i \\ -z_n z_i & n \neq i \end{cases} \tag{30}$$

The partial derivative of *L* respect to y_n^J is

$$\begin{aligned} \frac{\partial L}{\partial y_n^J} &= -Y_n \frac{1}{z_n}(z_n - z_n^2) + \sum_{i \neq n} Y_i z_n \\ &= \sum_i Y_i z_n - Y_n \\ &= z_n - Y_n \end{aligned} \tag{31}$$

with the fact that

$$\sum_i Y_i = 1 \tag{32}$$

Because in the context of classification, the vector **Y** is represent in such a way that only one of its components is 1 and all other components are 0.