# Neural Network

Masahiro Ogawa

April 2, 2015

## 1 Notation

- $\boldsymbol{x}$ : vector
- $\boldsymbol{A}$ : matrix
- ${}^t\boldsymbol{A}$ : transpose of $\boldsymbol{A}$
- $f.(\boldsymbol{A}) = \begin{pmatrix} f(a_{1,1}) & f(a_{1,2}) & \cdots & f(a_{1,n}) \\ f(a_{2,1}) & f(a_{2,2}) & \cdots & f(a_{2,n}) \\ \vdots & \vdots & \ddots & \vdots \\ f(a_{m,1}) & f(a_{m,2}) & \cdots & f(a_{m,n}) \end{pmatrix}$
- $(\boldsymbol{A}. * \boldsymbol{B})_{i,j} = a_{i,j} b_{i,j}$
- $\boldsymbol{1}_{m \times n} = \begin{pmatrix} 1 & \cdots & 1 \\ \vdots & \ddots & \vdots \\ 1 & \cdots & 1 \end{pmatrix}$ : $m \times n$ dim
- $\tilde{\boldsymbol{y}} = \begin{pmatrix} 1 \\ \boldsymbol{y} \end{pmatrix}$ : homogeneous vector
- $n$ : data number
- $N$ : total number of data
- $l$ : layer number
- $L$ : output layer number. thus, the total number of layer is $L+1$
- $k$ : learning iteration number
- $K$ : final number of the learning iteration

# 2 Input

parameters

- $L$ : number of layers
- $\{d_l\}_{l=0}^{L}$ : neuron numbers in each layer $l$
- $\{\tilde{\boldsymbol{W}}_l(0)\}_{l=0}^{L} \mid \tilde{\boldsymbol{W}}_l(0) = (\tilde{\boldsymbol{w}}_{l,1}(0), \ldots, \tilde{\boldsymbol{w}}_{l,d_{l+1}}(0))$ : $(d_l + 1) \times d_{l+1}$ dim : initial weights
- $\rho$ : learning rate parameter
- $T_e$ : error threshold
- $T_K$ : maximum iteration threshold number

data

- $\boldsymbol{X}_0 = (\boldsymbol{x}_{0,1}, \ldots, \boldsymbol{x}_{0,N})$ : $d_0 \times N$ dim : input vectors
- $\boldsymbol{B} = (\boldsymbol{b}_1, \ldots, \boldsymbol{b}_N)$ : $d_L \times N$ dim : instruction signals

# 3 Output

- $\{\tilde{\boldsymbol{W}}_l(K)\}_{l=0}^{L} \mid \tilde{\boldsymbol{W}}_l(K) = (\tilde{\boldsymbol{w}}_{l,1}(K), \ldots, \tilde{\boldsymbol{w}}_{l,d_{l+1}}(K))$ : $(d_l + 1) \times d_{l+1}$ dim : final (after $K$th iteration) learned weights
- $\boldsymbol{Y}_L = (\boldsymbol{y}_{L,1}, \ldots, \boldsymbol{y}_{L,N})$ : $dL \times N$ dim : output vectors
- $J = \sum_{n=1}^{N} J_n = \sum_{n=1}^{N} \|\boldsymbol{y}_{L,n} - \boldsymbol{b}_n\|^2$ : final error

# 4 Algorithm(Iterative version)

## 4.1 Iteration

Iterate feed forward and back propagation algorithm for each input datum until the error become smaller than the threshold or the iteration number exceeds the maximum iteration number. That is,

$$J < T_e, \text{ or } k < T_K. \tag{1}$$

## 4.2 Feed Forward

The below are true for all data and all iteration. So we omit the data number $n$ and iteration number $i$ as e.g. $\boldsymbol{x}_{l,n} = \boldsymbol{x}_l, \tilde{\boldsymbol{W}}_l(i) = \tilde{\boldsymbol{W}}_l$.

$$\boldsymbol{x}_l = {}^t\tilde{\boldsymbol{W}}_{l-1}\tilde{\boldsymbol{y}}_{l-1} \tag{2}$$
$$\boldsymbol{y}_l = f.(\boldsymbol{x}_l) \tag{3}$$
$$f(x) = \tanh(x) \tag{4}$$

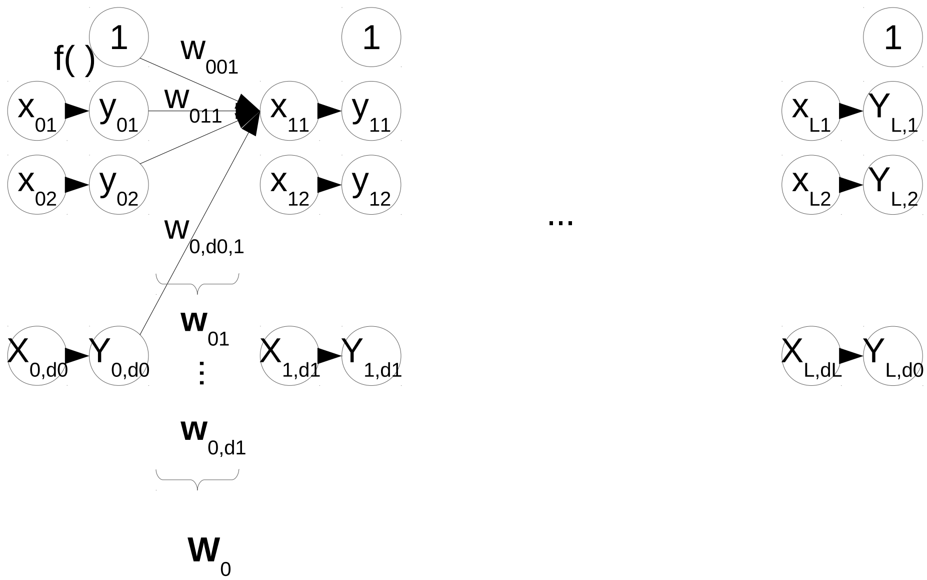


Figure 1: Neural network architecture

### 4.3 Back Propagation

$$\boldsymbol{\epsilon}_l = \begin{cases} (\boldsymbol{y}_L - \boldsymbol{b}).*f'.(\boldsymbol{x}_L) & \text{if } l = L \\ (\boldsymbol{W}_l \boldsymbol{\epsilon}_{l+1}).*f'.(\boldsymbol{x}_l) & 0 \end{cases} \tag{5}$$

$$= \begin{cases} (\boldsymbol{y}_L - \boldsymbol{b}).*(\mathbf{1}_{dL} - \boldsymbol{y}_L.*\boldsymbol{y}_L) \\ (\boldsymbol{W}_l \boldsymbol{\epsilon}_{l+1}).*(\mathbf{1}_{dl} - \boldsymbol{y}_l.*\boldsymbol{y}_l) \end{cases} \tag{6}$$

$$\tilde{\boldsymbol{W}}_{l-1}(i+1) = \tilde{\boldsymbol{W}}_{l-1}(i) - \rho\, \tilde{\boldsymbol{y}}_{l-1}{}^t \boldsymbol{\epsilon}_l \tag{7}$$

## 5 Algorithm(Matrix version)

### 5.1 Iteration

Iterate feed forward algorithm for all input data and back propagation for all data until the error become smaller than the threshold or the iteration number exceeds the maximum iteration number. That is,

$$J < T_e, \text{ or } k < T_K. \tag{8}$$

## 5.2 Feed Forward

The below are true for all iteration. So we omit the iteration number as e.g. $\tilde{\boldsymbol{W}}_l(i) = \tilde{\boldsymbol{W}}_l$.

$$
\begin{aligned}
\boldsymbol{X}_l &= {}^t\tilde{\boldsymbol{W}}_{l-1}\tilde{\mathbf{Y}}_{l-1} && (9)\\
\mathbf{Y}_l &= f.(\boldsymbol{X}_l) && (10)\\
f(x) &= \tanh(x) && (11)
\end{aligned}
$$

## 5.3 Back Propagation

$$
\begin{aligned}
\boldsymbol{E}_l &= \begin{cases} (\mathbf{Y}_L - \boldsymbol{B}).*f'.(\boldsymbol{X}_L) & \text{if } l = L \\ (\boldsymbol{W}_l\boldsymbol{E}_{l+1}).*f'.(\boldsymbol{X}_l) & \text{otherwise} l = L \end{cases} && (12)\\
&= \begin{cases} (\mathbf{Y}_l - \boldsymbol{B}).*(\mathbf{1}_{dL\times N} - \mathbf{Y}_L.*\mathbf{Y}_L) \\ (\boldsymbol{W}_l\boldsymbol{E}_{l+1}).*(\mathbf{1}_{dl\times N} - \mathbf{Y}_l.*\mathbf{Y}_l) \end{cases} && (13)\\
\tilde{\boldsymbol{W}}_{l-1}(i+1) &= \tilde{\boldsymbol{W}}_{l-1}(i) - \rho\,\tilde{\mathbf{Y}}_{l-1}{}^t\boldsymbol{E}_l && (14)
\end{aligned}
$$