Neural Networks

Dr. Arun Chauhan
Machine Learning
IIIT Dharwad,CSE

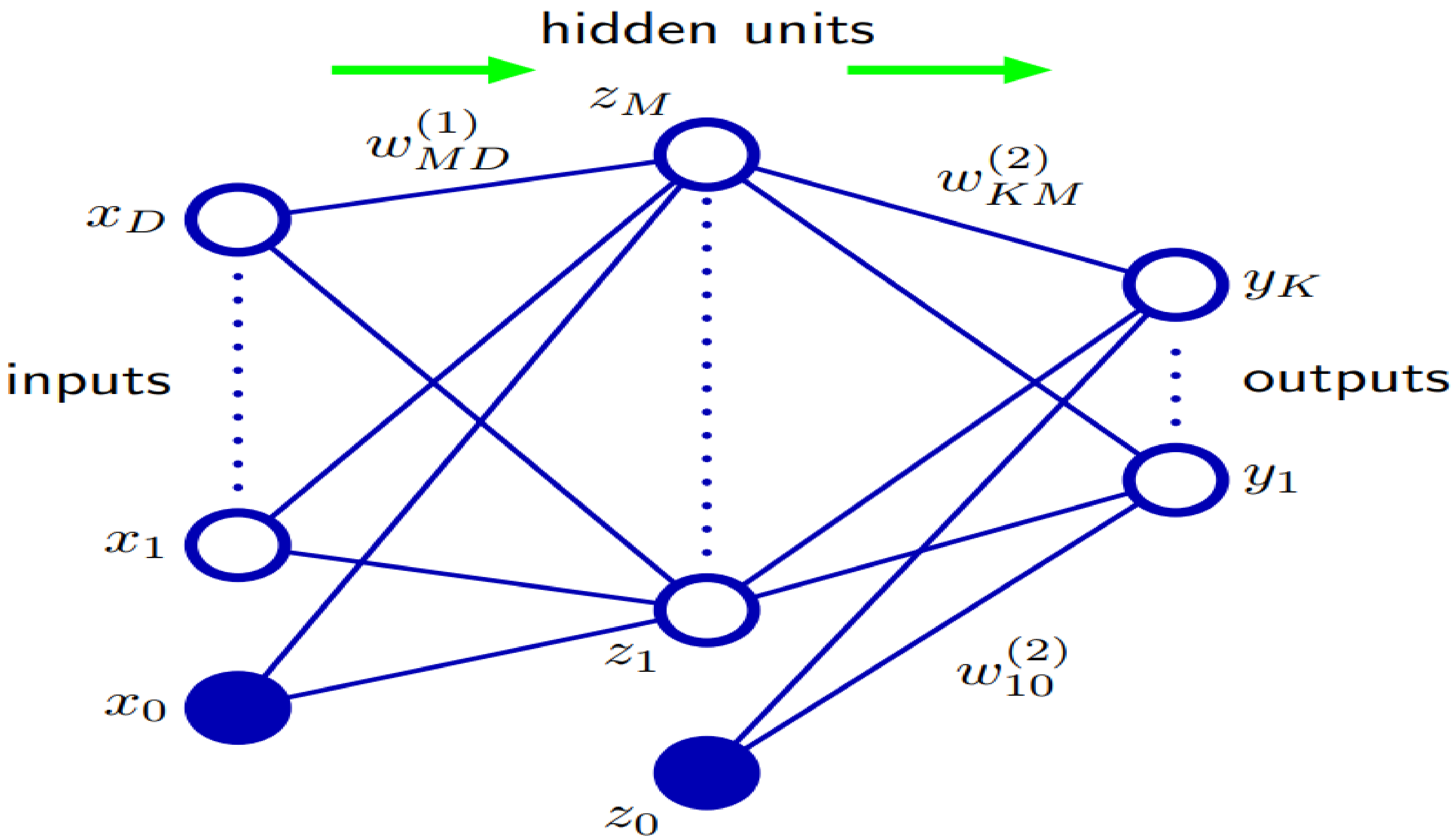
Feed-forward Network Functions

- Functional form for regression and classification so far :

$$y(\mathbf{x}, \mathbf{w}) = f\left(\sum_{j=1}^{M} w_j \phi_j(\mathbf{x})\right)$$

- Now basis function is also parameterized.

Feed-forward Network Functions



Feed-forward Network Functions

- Functional transformation

$$a_j = \sum_{i=1}^{D} w_{ji}^{(1)} x_i + w_{j0}^{(1)} \qquad \text{where } j = 1, \ldots, M$$

a_j are known as *activations*

- Each of them is then transformed using a differentiable, nonlinear *activation function* $h(\cdot)$

$$z_j = h(a_j)$$

Feed-forward Network Functions

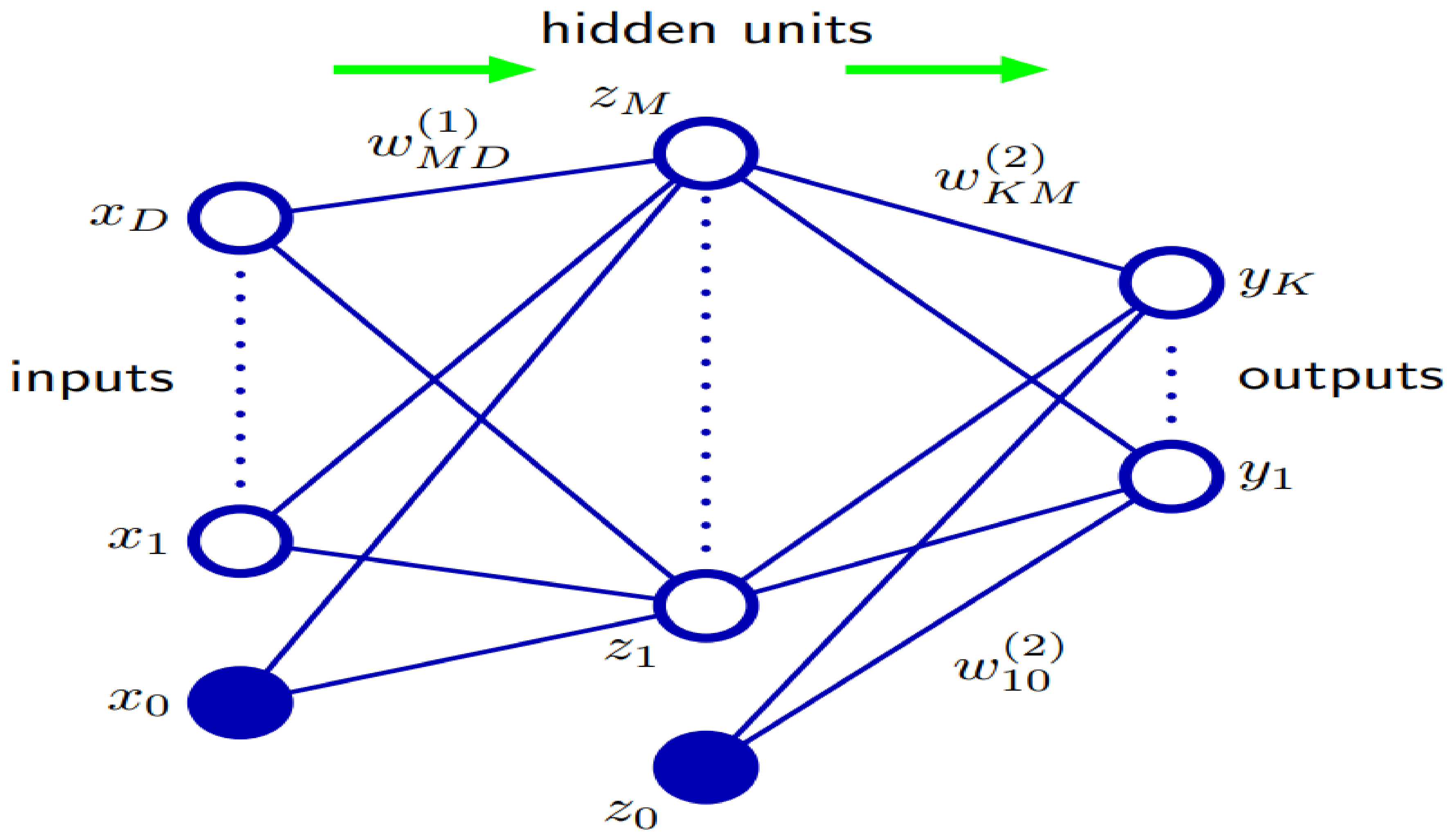
- These values, z_j are again linearly combined to give *output unit activations*

$$a_k = \sum_{j=1}^{M} w_{kj}^{(2)} z_j + w_{k0}^{(2)}$$

where $k = 1, \ldots, K$, and K is the total number of outputs

- For regression*:* $y_k = a_k$
- For multiple binary classification problems: $y_k = \sigma(a_k)$
- For multiclass problems, a softmax activation function.

Feed-forward Network Functions



Feed-forward Network Functions

We can combine these various stages to give the overall network function that, for sigmoidal output unit activation functions, takes the form

$$y_k(\mathbf{x}, \mathbf{w}) = \sigma\left(\sum_{j=1}^{M} w_{kj}^{(2)} h\left(\sum_{i=1}^{D} w_{ji}^{(1)} x_i + w_{j0}^{(1)}\right) + w_{k0}^{(2)}\right)$$

Feed-forward Network Functions

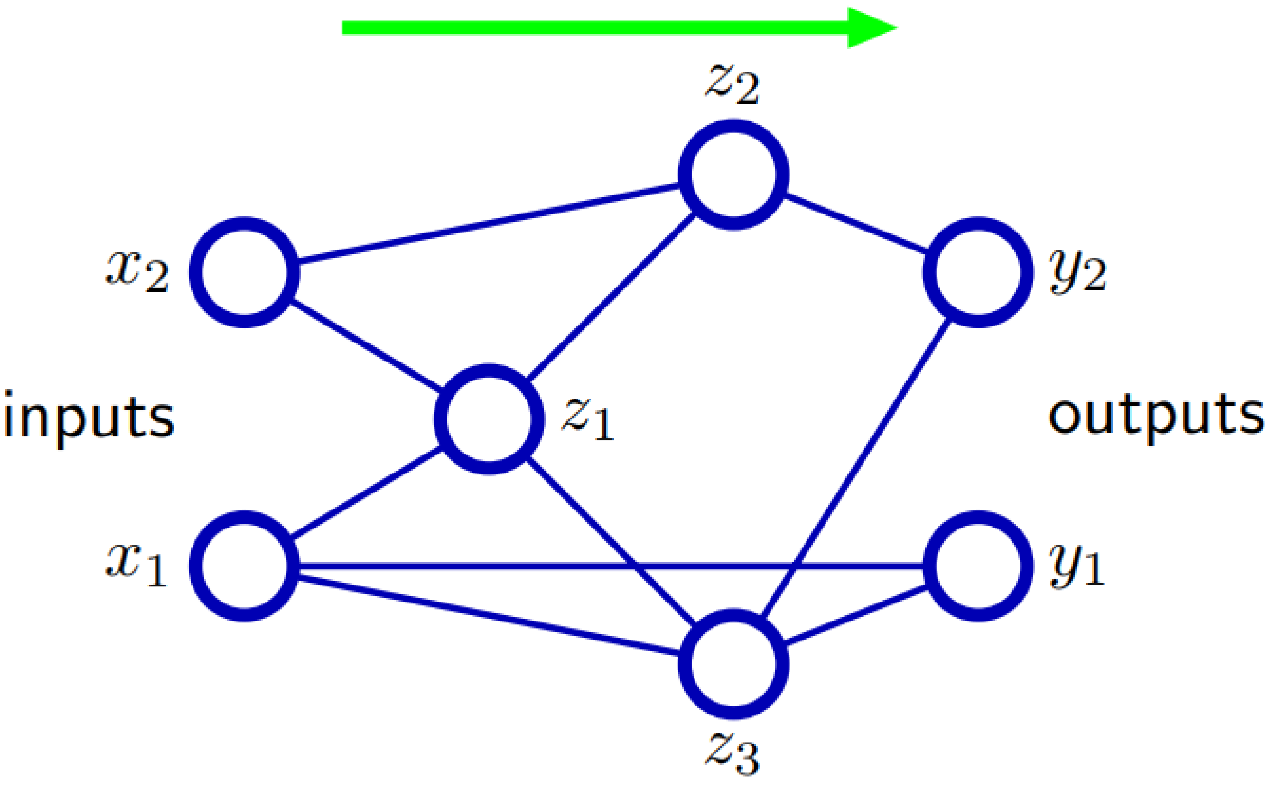
- The bias parameters can be absorbed into the set of weight parameters

$$a_j = \sum_{i=0}^{D} w_{ji}^{(1)} x_i$$

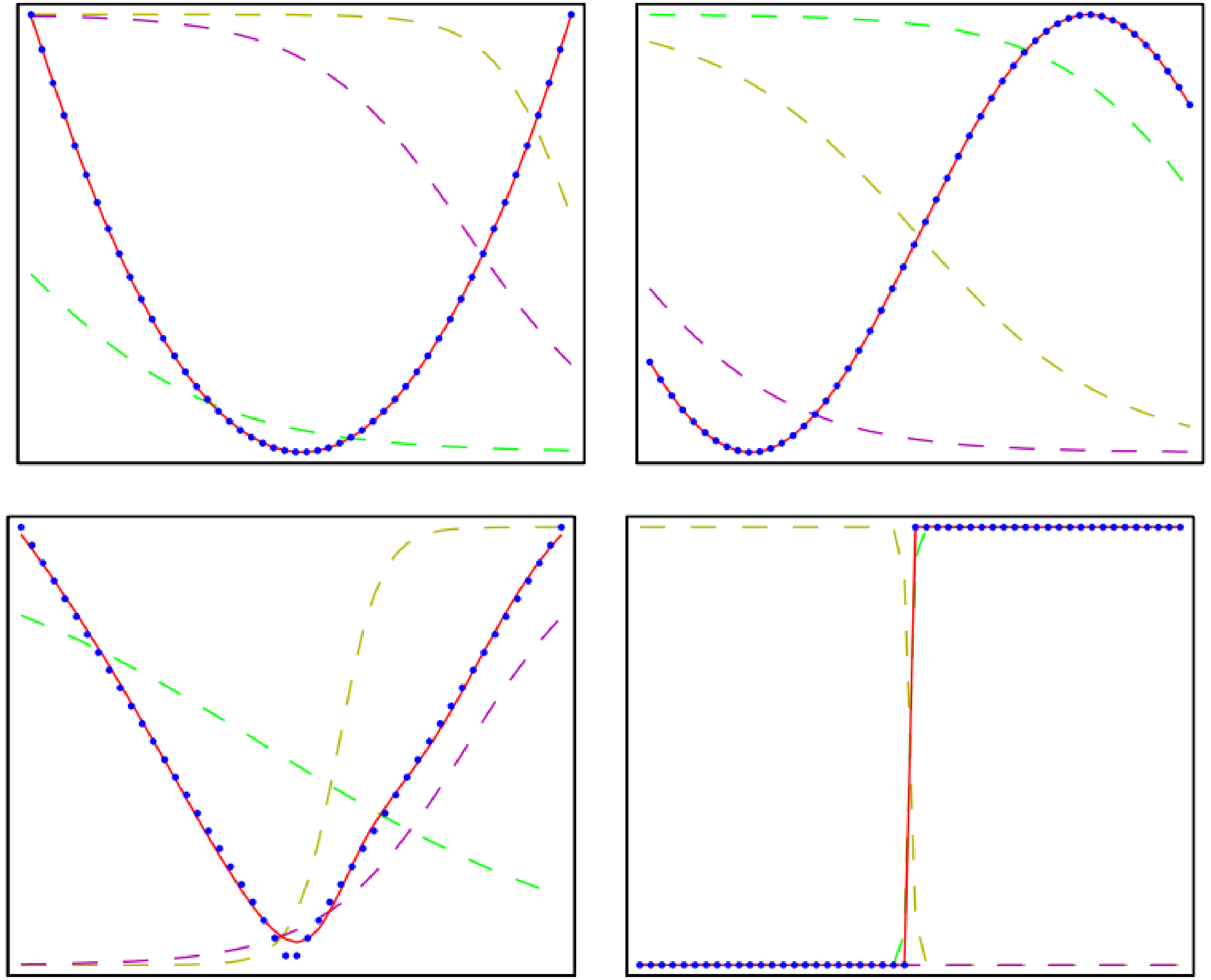
$$y_k(\mathbf{x}, \mathbf{w}) = \sigma\left(\sum_{j=0}^{M} w_{kj}^{(2)} h\left(\sum_{i=0}^{D} w_{ji}^{(1)} x_i\right)\right)$$

Skip-Layer

- Generalization of the network architecture

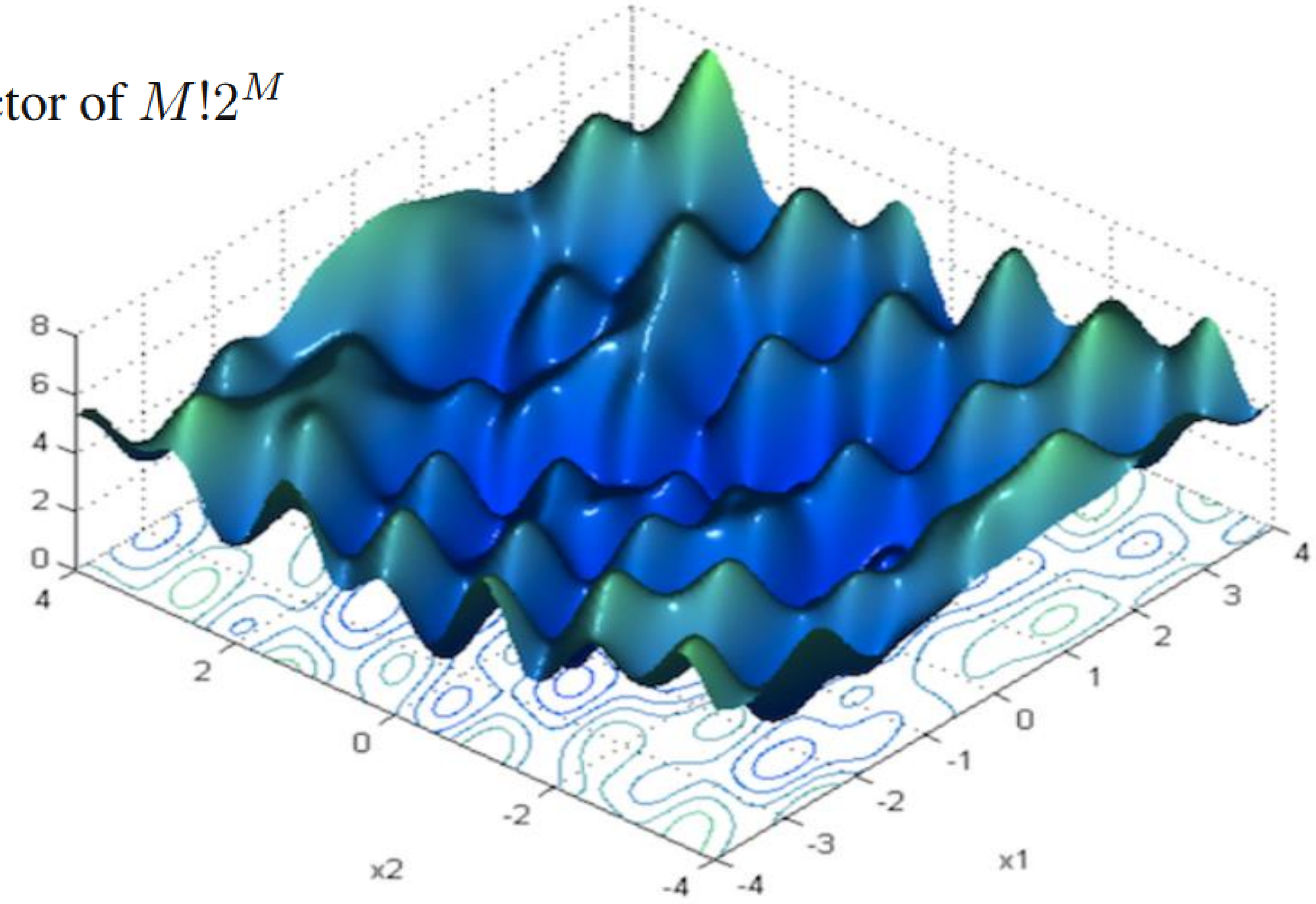


Approximation of function using NN



Error function of Neural Network

an overall weight-space symmetry factor of $M!2^M$
one for each layer of hidden units.



Network Training

- For regression problem with single target variable t

$$p(t|\mathbf{x}, \mathbf{w}) = \mathcal{N}\left(t|y(\mathbf{x}, \mathbf{w}), \beta^{-1}\right)$$

- Take the output unit activation function to be the identity

Likelihood function

- Given a data set of N independent, identically distributed observations

$$\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\},$$

$$\mathbf{t} = \{t_1, \ldots, t_N\}$$

$$p(\mathbf{t}|\mathbf{X}, \mathbf{w}, \beta) = \prod_{n=1}^{N} p(t_n|\mathbf{x}_n, \mathbf{w}, \beta)$$

- Taking the negative logarithm, we obtain the error function

$$\frac{\beta}{2}\sum_{n=1}^{N}\{y(\mathbf{x}_n, \mathbf{w}) - t_n\}^2 - \frac{N}{2}\ln\beta + \frac{N}{2}\ln(2\pi)$$

Maximum likelihood estimation

- Maximizing the likelihood function is equivalent to minimizing the sum-of-squares error function given by

$$E(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N}\{y(\mathbf{x}_n, \mathbf{w}) - t_n\}^2$$

(Local Minima)

- Having found $\mathbf{w}_{\mathrm{ML}}$, the value of β can be found by minimizing the negative log likelihood to give

$$\frac{1}{\beta_{\mathrm{ML}}} = \frac{1}{N}\sum_{n=1}^{N}\{y(\mathbf{x}_n, \mathbf{w}_{\mathrm{ML}}) - t_n\}^2.$$

Maximum likelihood estimation

- For multiple target variables: $p(\mathbf{t}|\mathbf{x}, \mathbf{w}) = \mathcal{N}\left(\mathbf{t}|\mathbf{y}(\mathbf{x}, \mathbf{w}), \beta^{-1}\mathbf{I}\right)$
- Minimize error function: $E(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N}\|\mathbf{y}(\mathbf{x}_n, \mathbf{w}) - \mathbf{t}_n\|^2$
- The noise precision is given by :

$$\frac{1}{\beta_{\mathrm{ML}}} = \frac{1}{NK}\sum_{n=1}^{N}\|\mathbf{y}(\mathbf{x}_n, \mathbf{w}_{\mathrm{ML}}) - \mathbf{t}_n\|^2$$

Regression output activation function

- $y_k = a_k$
- The corresponding sum-of-squares error function has the property

$$\frac{\partial E}{\partial a_k} = y_k - t_k$$

Binary classification using NN

$$y = \sigma(a) \equiv \frac{1}{1 + \exp(-a)}$$

- The conditional distribution of target

$$p(t|\mathbf{x}, \mathbf{w}) = y(\mathbf{x}, \mathbf{w})^t \left\{1 - y(\mathbf{x}, \mathbf{w})\right\}^{1-t}$$

- Error function, which is given by the negative log likelihood

$$E(\mathbf{w}) = -\sum_{n=1}^{N} \{t_n \ln y_n + (1 - t_n) \ln(1 - y_n)\}$$

where y_n denotes $y(\mathbf{x}_n, \mathbf{w})$

Derivative of error function (Binary Classification)

$$E(\mathbf{w}) = -\sum_{n=1}^{N} \{t_n \ln y_n + (1 - t_n) \ln(1 - y_n)\}$$

- Similar to regression function:

$$\frac{\partial E}{\partial a_n} = -t_n \frac{1}{y_n} \frac{\partial y_n}{\partial a_n} + (1 - t_n) \frac{1}{1 - y_n} \frac{\partial y_n}{\partial a_n}$$

$$\frac{\partial y_n}{\partial a_n} = y_n(1 - y_n)$$

$$\begin{aligned} \frac{\partial E}{\partial a_n} &= -t_n \frac{y_n(1 - y_n)}{y_n} + (1 - t_n) \frac{y_n(1 - y_n)}{(1 - y_n)} \\ &= y_n - t_n \end{aligned}$$

K separate binary classifications

$$y_k = \sigma(a) \equiv \frac{1}{1 + \exp(-a)}$$

- The conditional distribution of target

$$p(\mathbf{t}|\mathbf{x}, \mathbf{w}) = \prod_{k=1}^{K} y_k(\mathbf{x}, \mathbf{w})^{t_k} \left[1 - y_k(\mathbf{x}, \mathbf{w})\right]^{1-t_k}$$

- Error function, which is given by the negative log likelihood

$$E(\mathbf{w}) = -\sum_{n=1}^{N} \sum_{k=1}^{K} \{t_{nk} \ln y_{nk} + (1 - t_{nk}) \ln(1 - y_{nk})\}$$

where y_{nk} denotes $y_k(\mathbf{x}_n, \mathbf{w})$.

Multiclass classification problem

1-of-K coding scheme indicating the class

$$E(\mathbf{w}) = -\sum_{n=1}^{N}\sum_{k=1}^{K} t_{kn} \ln y_k(\mathbf{x}_n, \mathbf{w})$$

the output unit activation

$$y_k(\mathbf{x}, \mathbf{w}) = \frac{\exp(a_k(\mathbf{x}, \mathbf{w}))}{\sum_j \exp(a_j(\mathbf{x}, \mathbf{w}))}$$

Parameter optimization

$$\mathbf{w}^{(\tau+1)} = \mathbf{w}^{(\tau)} + \Delta\mathbf{w}^{(\tau)}$$

$$\delta E \simeq \delta\mathbf{w}^{\mathrm{T}} \nabla E(\mathbf{w})$$

Evaluation of minima without gradient info

- If we use gradient information for finding minima in comparison to other techniques such Newton method, then minimum can be found in $O(W^2)$ steps rather then $O(W^3)$.

Gradient descent optimization

$$\mathbf{w}^{(\tau+1)} = \mathbf{w}^{(\tau)} + \Delta\mathbf{w}^{(\tau)}$$

$$\mathbf{w}^{(\tau+1)} = \mathbf{w}^{(\tau)} - \eta\nabla E(\mathbf{w}^{(\tau)})$$

$\eta > 0$ is known as the *learning rate*

- Online(Stochastic) version of gradient decent is more efficient (LeCun 1989)
  - Handle redundancy
  - Escaping from local minima

Error Backpropagation

Two Step.

1. The propagation of errors backwards through the network in order to evaluate derivatives.
2. Weight adjustment using the calculated derivatives.

Evaluation of error-function derivatives

$$\frac{\partial E_n}{\partial w_{ji}} = \frac{\partial E_n}{\partial a_j} \frac{\partial a_j}{\partial w_{ji}}$$

$$\delta_j \equiv \frac{\partial E_n}{\partial a_j}$$

$$\frac{\partial a_j}{\partial w_{ji}} = z_i$$

$$\frac{\partial E_n}{\partial w_{ji}} = \delta_j z_i$$

As we have seen already, for the output units, we have

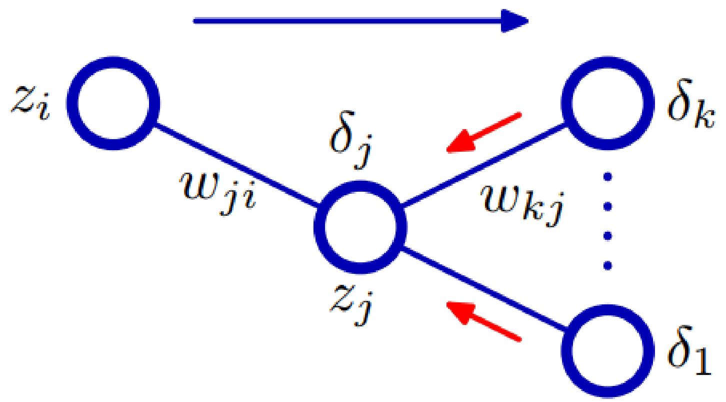
$$\delta_k = y_k - t_k$$

Evaluation of error-function derivatives

evaluate the δ's for hidden units

$$\delta_j \equiv \frac{\partial E_n}{\partial a_j} = \sum_k \frac{\partial E_n}{\partial a_k}\frac{\partial a_k}{\partial a_j}$$

$$\delta_j = h'(a_j)\sum_k w_{kj}\delta_k$$



The simple example

$$E_n = \frac{1}{2}\sum_{k=1}^{K}(y_k - t_k)^2$$

$$y_k = a_k$$

$$h(a) \equiv \tanh(a)$$

$$\tanh(a) = \frac{e^a - e^{-a}}{e^a + e^{-a}}$$

$$h'(a) = 1 - h(a)^2$$

$$a_j = \sum_{i=0}^{D} w_{ji}^{(1)} x_i$$

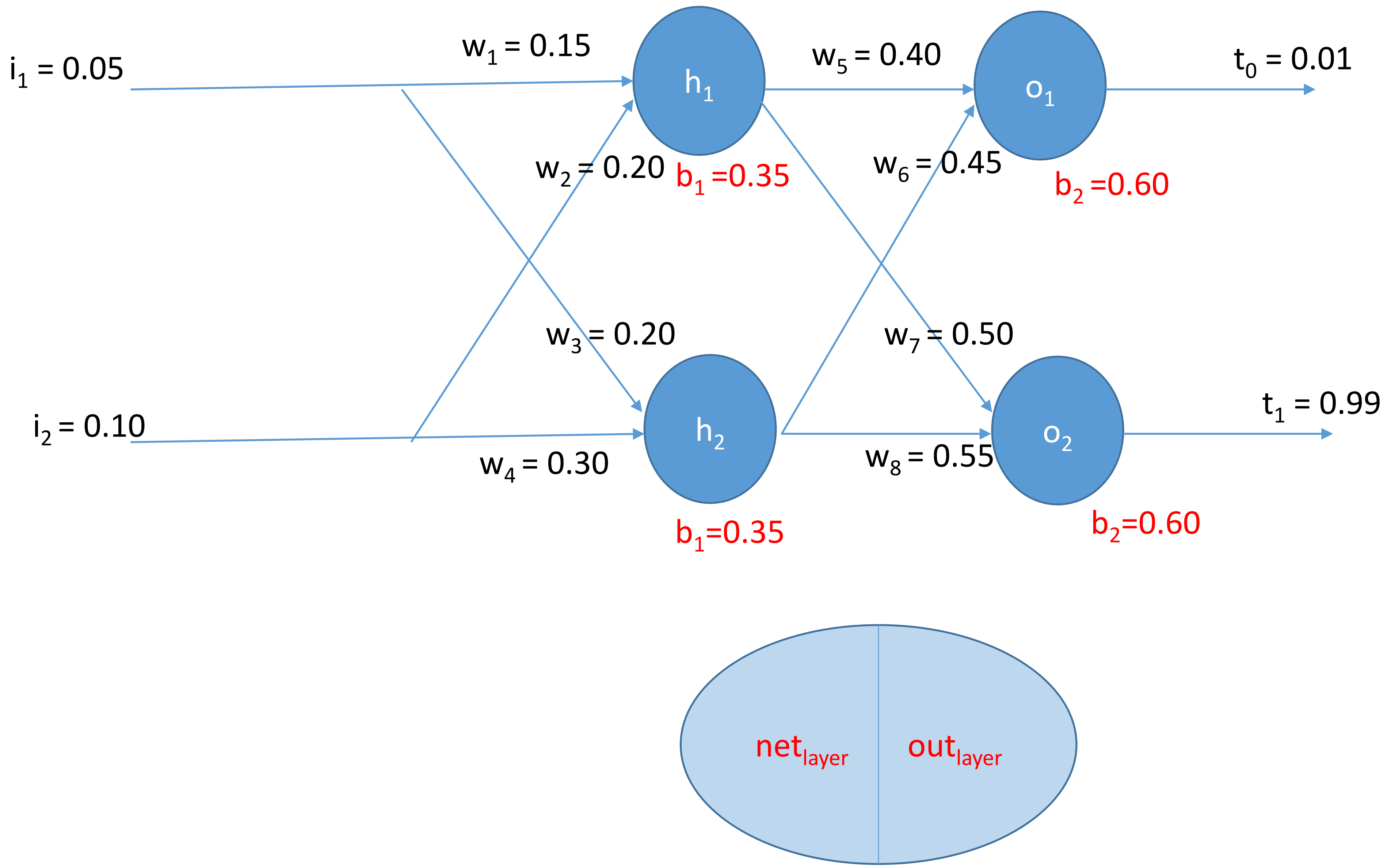
$$z_j = \tanh(a_j)$$

$$y_k = \sum_{j=0}^{M} w_{kj}^{(2)} z_j$$

$$\delta_k = y_k - t_k$$

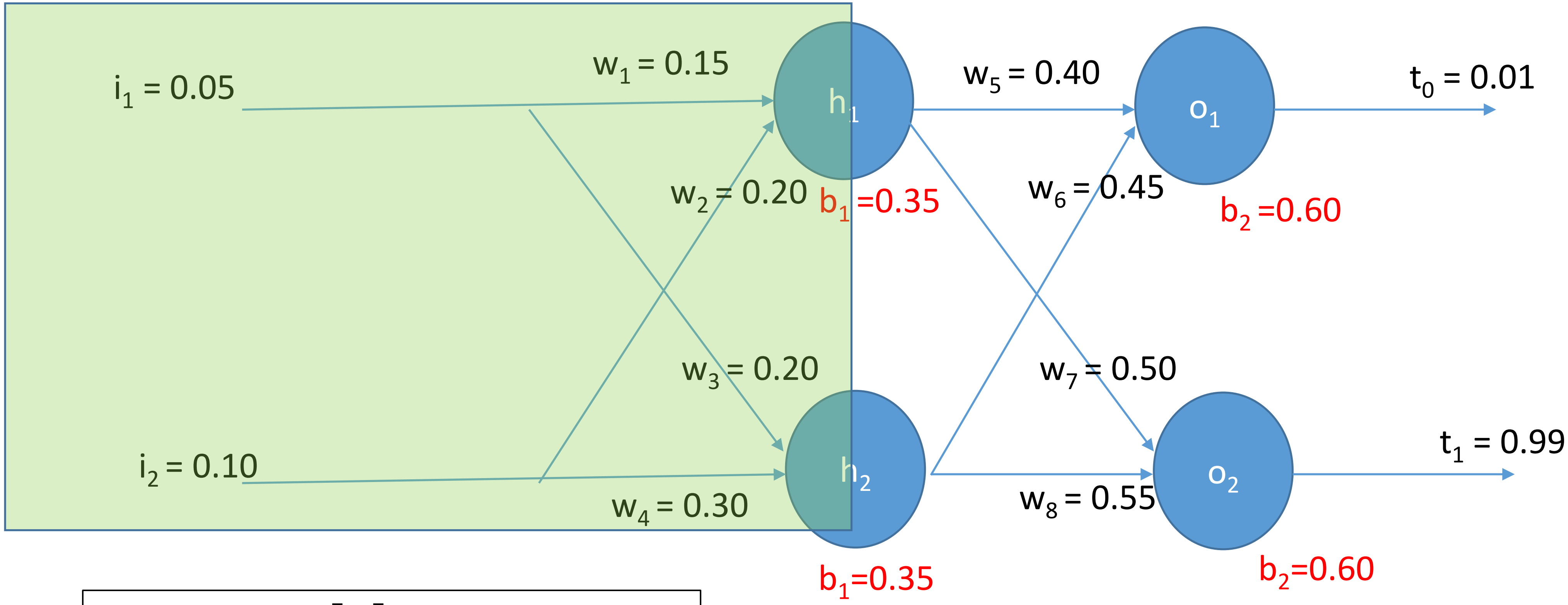
$$\delta_j = (1 - z_j^2)\sum_{k=1}^{K} w_{kj}\delta_k$$

$$\frac{\partial E_n}{\partial w_{ji}^{(1)}} = \delta_j x_i, \qquad \frac{\partial E_n}{\partial w_{kj}^{(2)}} = \delta_k z_j$$



$i_1 = 0.05$
$i_2 = 0.10$
$w_1 = 0.15$
$w_2 = 0.20$
$w_3 = 0.20$
$w_4 = 0.30$
h_1
h_2
$b_1 = 0.35$
$b_1 = 0.35$
$w_5 = 0.40$
$w_6 = 0.45$
$w_7 = 0.50$
$w_8 = 0.55$
o_1
o_2
$b_2 = 0.60$
$b_2 = 0.60$
$t_0 = 0.01$
$t_1 = 0.99$
net_{layer}
out_{layer}

Forward Pass



$$\begin{bmatrix} b_1 & w_1 & w_2 \\ b_1 & w_3 & w_4 \end{bmatrix} \begin{bmatrix} i_0 \\ i_1 \\ i_2 \end{bmatrix} = \begin{bmatrix} net_{h_1} \\ net_{h_2} \end{bmatrix} \quad ;\ i_0 = 1$$

$$b_1 + w_1 i_1 + w_2 i_2 = net_{h_1}$$

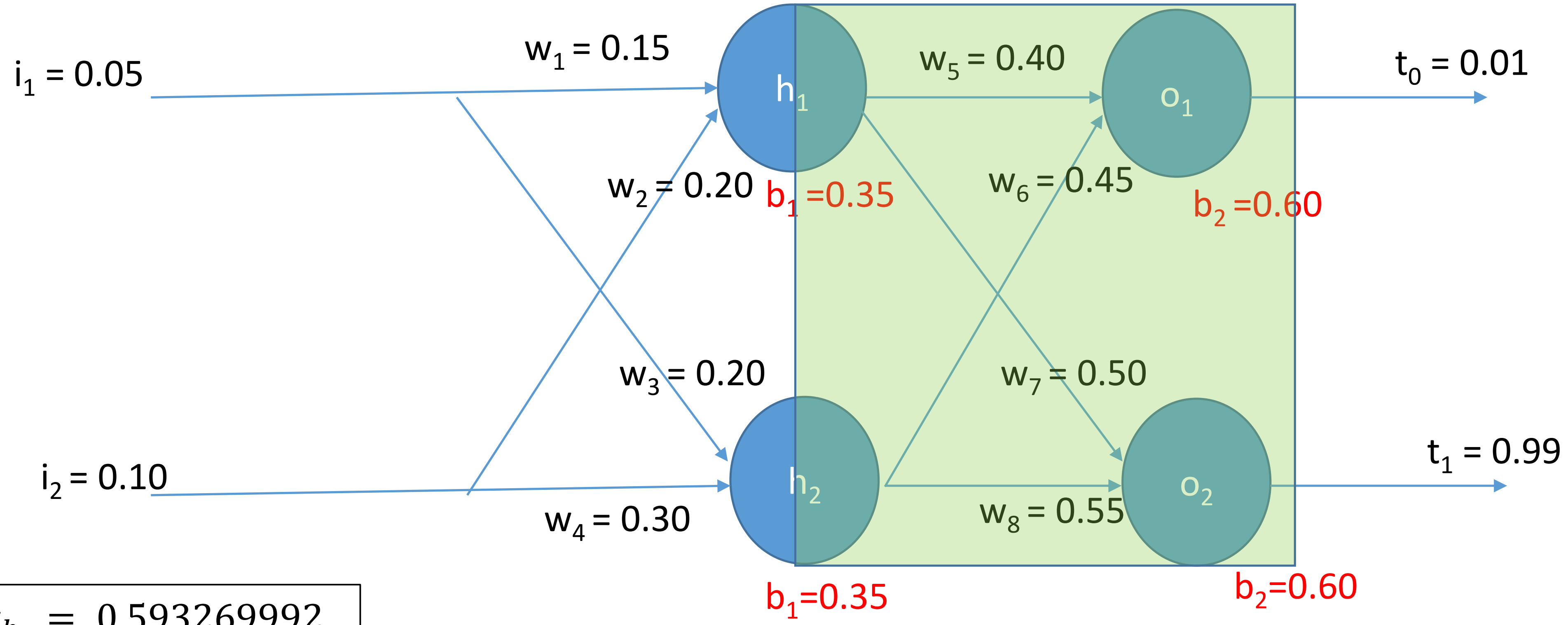
$$b_1 + w_3 i_1 + w_4 i_2 = net_{h_2}$$

$$out_{h_1} = \frac{1}{1 + \exp(-0.3775)} = 0.593269992$$

$$out_{h_2} = \frac{1}{1 + \exp(-0.3925)} = 0.596884378$$

$$net_{h_1} = b_1 i_o + w_1 i_1 + w_2 i_2$$
$$= 0.35 * 1 + 0.15 * 0.05 + 0.20 * 0.10 = 0.3775$$

$$net_{h_2} = b_1 i_o + w_3 i_1 + w_4 i_2$$
$$= 0.35 * 1 + 0.25 * 0.05 + 0.30 * 0.10 = 0.3925$$



$$out_{h_1} = 0.593269992$$
$$out_{h_2} = 0.596884378$$

$$net_{o_1} = 0.60 * 1 + 0.40 * 0.593269992 + 0.45 * 0.596884378 = 1.105905967$$
$$net_{o_2} = 0.60 * 1 + 0.50 * 0.593269992 + 0.55 * 0.596884378 = 1.224921403$$

$$\begin{bmatrix} b_2 & w_5 & w_6 \\ b_2 & w_7 & w_8 \end{bmatrix} \begin{bmatrix} out_{h_0} \\ out_{h_1} \\ out_{h_2} \end{bmatrix} = \begin{bmatrix} net_{o_1} \\ net_{o_2} \end{bmatrix} \; ; out_{h_0} = 1$$

$$b_2 out_{h_0} + w_5 out_{h_1} + w_6 out_{h_2} = net_{o_1}$$
$$b_2 out_{h_0} + w_7 out_{h_1} + w_8 out_{h_2} = net_{o_2}$$

$$out_{o_1} = \frac{1}{1 + \exp(-1.105905967)} = 0.75136507$$
$$out_{o_2} = \frac{1}{1 + \exp(-1.224921403)} = 0.772928465$$

$$E_{total} = \sum \frac{1}{2}(target - output)^2$$

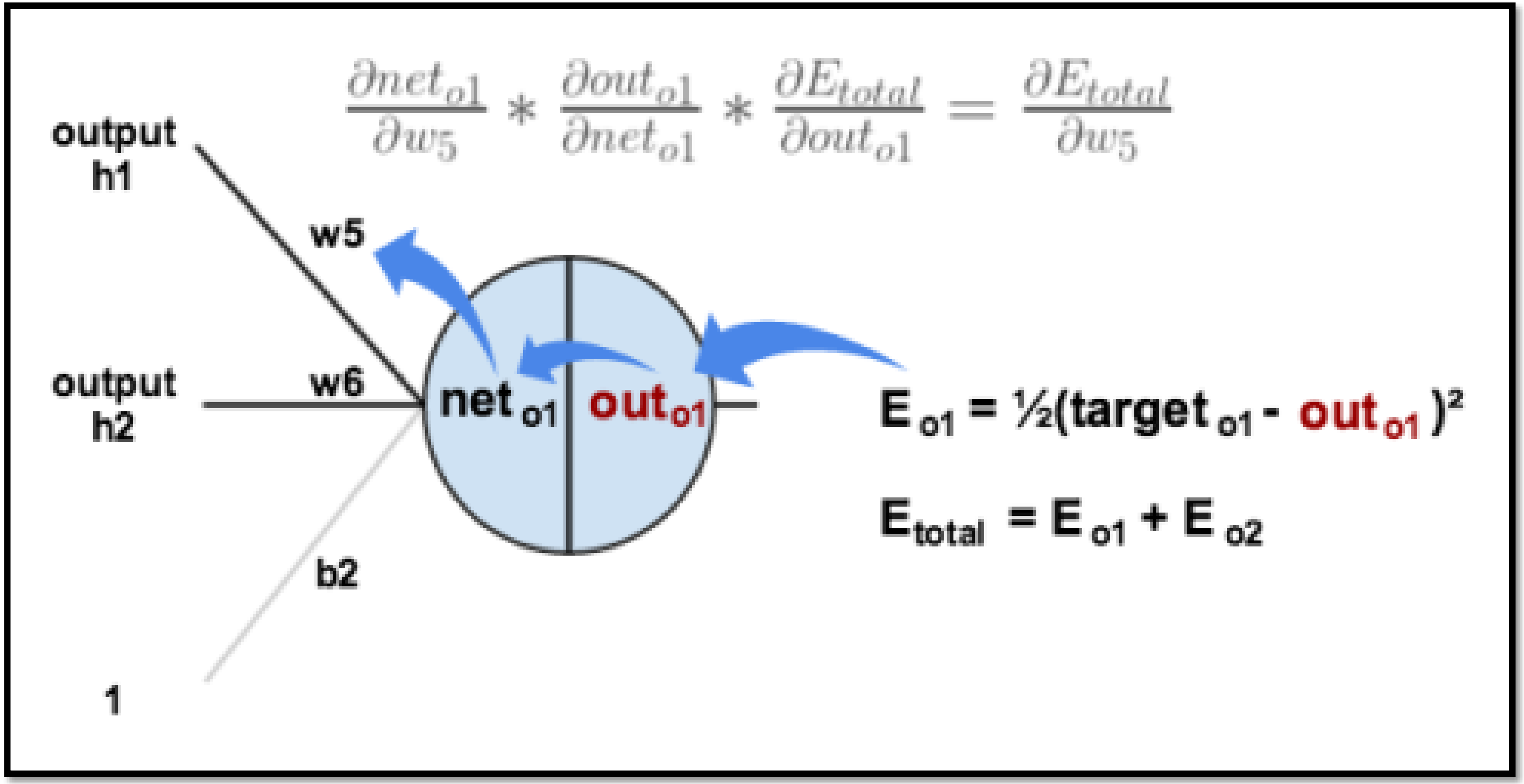
$$E_{total} = E_{o_1} + E_{o_2}$$

$$E_{o_1} = \frac{1}{2}(target_{o_1} - out_{o_1})^2 = \frac{1}{2}(0.01 - 0.75136507)^2 = 0.274811083$$

$$E_{o_2} = \frac{1}{2}(target_{o_2} - out_{o_2})^2 = \frac{1}{2}(0.99 - 0.772928465)^2 = 0.023560026$$

$$E_{total} = E_{o_1} + E_{o_2} = 0.274811083 + 0.023560026 = 0.298371109$$

Backward Pass



$\frac{\partial net_{o1}}{\partial w_5} * \frac{\partial out_{o1}}{\partial net_{o1}} * \frac{\partial E_{total}}{\partial out_{o1}} = \frac{\partial E_{total}}{\partial w_5}$
output h1
w5
output h2
w6
net_{o1}
out_{o1}
b2
1
$E_{o1} = \frac{1}{2}(target_{o1} - out_{o1})^2$
$E_{total} = E_{o1} + E_{o2}$

Change in error due to $w_{5:}$

$$\frac{d\,E_{total}}{d\,w_5} = \frac{d\,E_{total}}{d\,out_{o_1}} * \frac{d\,out_{o_1}}{d\,net_{o_1}} * \frac{d\,net_{o_1}}{d\,w_5}$$

$$\frac{d\,E_{total}}{d\,out_{o_1}} = \frac{d\,(\frac{1}{2}\,(target_{o_1} - out_{o_1})^2 + \frac{1}{2}\,(target_{o_2} - out_{o_2})^2\,)}{d\,out_{o_1}} = -(target_{o_1} - out_{o_1})$$

$$= -(0.01 - 0.75136507\,) = 0.74136507$$

$$\frac{d\,out_{o_1}}{d\,net_{o_1}} = \frac{d\,(\frac{1}{1+\exp(-net_{o_1})})}{d\,net_{o_1}} = out_{o_1}\left(1 - out_{o_1}\right) = 0.75136507(1 - 0.75136507) = 0.186815601$$

$$\frac{d\,net_{o_1}}{d\,w_5} = \frac{d(\mathrm{b}_2 out_{h_0} + \mathrm{w}_5 out_{h_1} + \mathrm{w}_6 out_{h_2})}{d\,w_5} = out_{h_1} = 0.593269992$$

$$\frac{d\,E_{total}}{d\,w_5} = 0.74136507 * 0.186815601 * 0.593269992 = 0.082167041$$

$$\boxed{\frac{d\,E_{total}}{d\,net_{o_1}} = \frac{d\,E_{total}}{d\,out_{o_1}} * \frac{d\,out_{o_1}}{d\,net_{o_1}} = \delta_{o_1}}$$

$$w_5^+ = w_5 - \eta\frac{d\,E_{total}}{d\,w_5} = 0.4 - 0.5 * 0.082167041 = 0.35891648$$

$$w_6^+ = 0.408666186$$

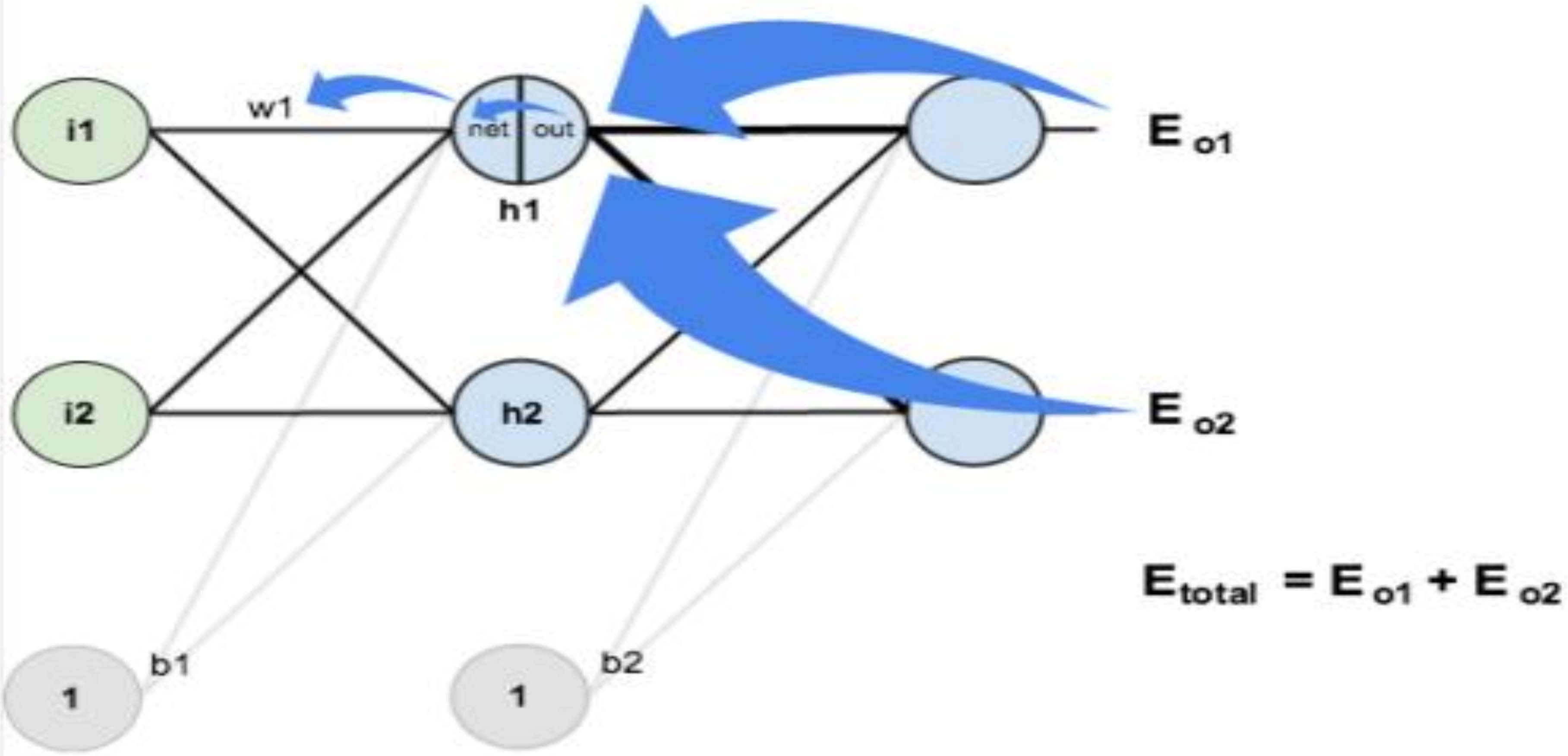
$$w_7^+ = 0.511301270$$

$$w_8^+ = 0.561370121$$

Hidden Layer

$$\frac{\partial E_{total}}{\partial w_1} = \frac{\partial E_{total}}{\partial out_{h1}} * \frac{\partial out_{h1}}{\partial net_{h1}} * \frac{\partial net_{h1}}{\partial w_1}$$

$$\frac{\partial E_{total}}{\partial out_{h1}} = \frac{\partial E_{o1}}{\partial out_{h1}} + \frac{\partial E_{o2}}{\partial out_{h1}}$$

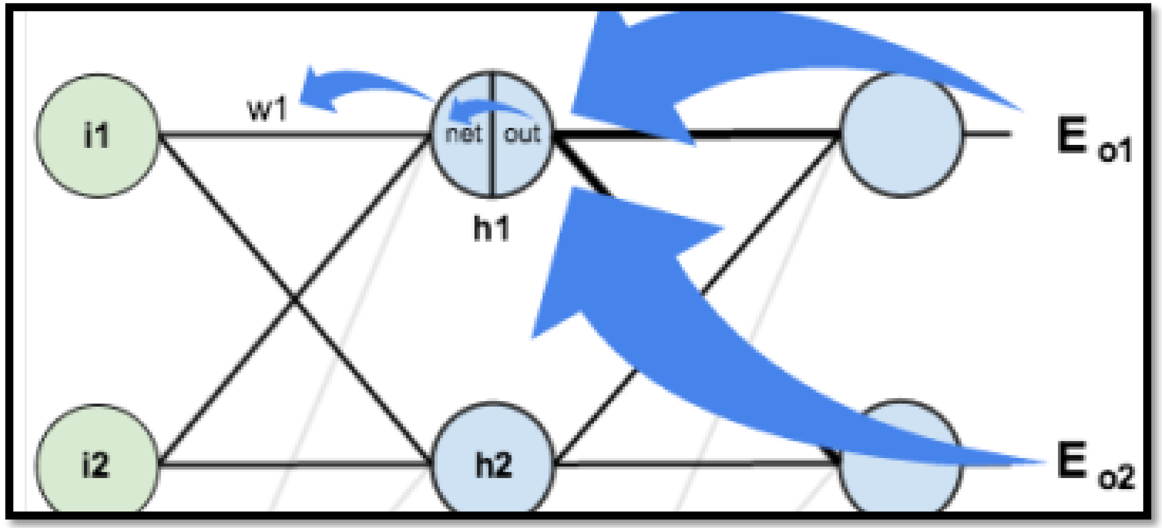


$$\frac{d\,E_{total}}{d\,out_{h_1}} = \frac{d\,E_{o_1}}{d\,out_{h_1}} + \frac{d\,E_{o_2}}{d\,out_{h_1}}$$

$$\frac{d\,E_{o_1}}{d\,out_{h_1}} = \frac{dE_{o_1}}{d\,out_{o_1}} * \frac{d\,out_{o_1}}{d\,net_{o_1}} * \frac{d\,net_{o_1}}{d\,out_{h_1}}$$

$$\frac{d\,net_{o_1}}{d\,out_{h_1}} = \frac{d(\mathrm{b}_2 out_{h_0} + \mathrm{w}_5 out_{h_1} + \mathrm{w}_6 out_{h_2})}{d\,out_{h_1}} = w_5 = 0.4$$

$$\frac{d\,E_{o_1}}{d\,out_{h_1}} = 0.74136507 * 0.186815601 * 0.4 = 0.55399425$$



$$\frac{d\,E_{o_2}}{d\,out_{h_1}} = \frac{dE_{o_2}}{d\,out_{o_2}} * \frac{d\,out_{o_2}}{d\,net_{o_2}} * \frac{d\,net_{o_2}}{d\,out_{h_1}}$$

$$\frac{d\,E_{o_2}}{d\,out_{h_1}} = -0.019049119$$

$$\frac{d\,E_{total}}{d\,out_{h_1}} = \frac{d\,E_{o_1}}{d\,out_{h_1}} + \frac{d\,E_{o_2}}{d\,out_{h_1}}$$

$$\frac{d\,E_{total}}{d\,out_{h_1}} = 0.55399425 - 0.019049119 = 0.036350306$$

$$\frac{d\,E_{total}}{d\,w_1} = \frac{d\,E_{total}}{d\,out_{h_1}} * \frac{d\,out_{h_1}}{d\,net_{h_1}} * \frac{d\,net_{h_1}}{d\,w_1}$$

$$\frac{d\,out_{h_1}}{d\,net_{h_1}} = \frac{d\,(\frac{1}{1+\exp(-net_{h_1})})}{net_{h_1}} = out_{h_1}\left(1 - out_{h_1}\right) = 0.593269992(1 - 0.593269992) = 0.241300709$$

$$\frac{d\,net_{h_1}}{d\,w_1} = \frac{d(\mathrm{b}_1\mathrm{i}_o + \mathrm{w}_1\mathrm{i}_1 + \mathrm{w}_2\mathrm{i}_2)}{d\,w_1} = i_1$$

$$\frac{d\,E_{total}}{d\,w_1} = 0.036350306 * 0.241300709 * 0.05 = 0.000438568$$

$$w_1^+ = w_5 - \eta\frac{d\,E_{total}}{d\,w_1} = 0.15 - 0.5 * 0.000438568 = 0.149780716$$

$$w_2^+ = 0.19956143$$

$$w_3^+ = 0.24975114$$

$$w_8^+ = 0.29950229$$

- At first the total error was 0.298371109
- After updating weights error is 0.291027924

Lecture 5, Patter Recognition using Machine Learning.