UNIVERSITY OF LEEDS

Class: Machine Learning

Multi-Layer Neural Networks

Instructor: Matteo Leonetti

Learning outcomes

- Define an appropriate error to minimise for Feed-forward neural networks.
- Derive the update rule of the weights of the NN, through backpropagation.
- Apply NNs to real-world data sets

Error definition

$$E(\boldsymbol{X}) = \sum_{\boldsymbol{x}_n \in \boldsymbol{X}} |y_n - t_n|$$

Number of errors on the training set

$$E_p(\boldsymbol{X}) = \sum_{\boldsymbol{x}_n \in \boldsymbol{X}} \boldsymbol{w}^T \boldsymbol{x}_n (y_n - t_n)$$

The Perceptron error

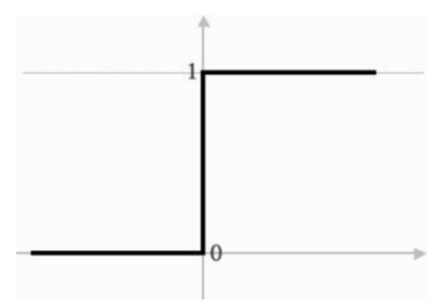
$$E_m(\boldsymbol{X}) = \frac{1}{2} \sum_{\boldsymbol{x}_n \in \boldsymbol{X}} (y_n - t_n)^2$$

Squared error function (differentiable!)
Usually known as the Mean Squared Error (MSE)

Output is differentiable if f is

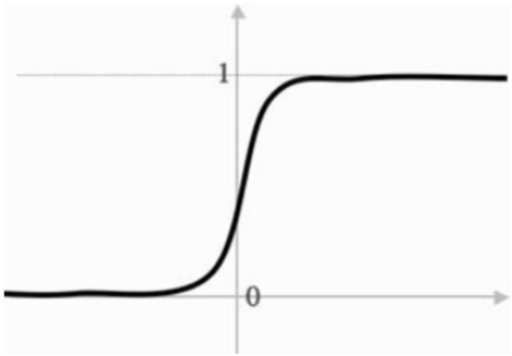
$$y = f\left(\sum_{i=1}^{M} w_i x_i\right)$$

$f =$



Not good

A different activation function



The sigmoid function: $f(x) = \frac{1}{1+e^{-\beta x}} \equiv \sigma_\beta$

$$\sigma_\beta'(x) = ?$$

The derivative of the sigmoid

The sigmoid function: $f(x) = \frac{1}{1+e^{-\beta x}} \equiv \sigma_\beta$

$$\sigma_\beta{}'(x) = ?$$

Two useful properties of derivatives:

$$f(x) = e^x \qquad f'(x) = e^x$$

Example: $\left(e^{x^2}\right)' = e^{x^2} \cdot 2x$

Chain rule: $(f \circ g)'(x) = f'(g(x)) \cdot g'(x)$

Hint: $\frac{1}{1+e^{-\beta x}} = \left(1+e^{-\beta x}\right)^{-1}$

The derivative of the sigmoid

UNIVERSITY OF LEEDS

The sigmoid function: $f(x) = \frac{1}{1+e^{-x}} \equiv \sigma$ where $\beta = 1$ for simplicity

We derive the most external function first

Then this

$$\sigma'(x) = \left(\left(1+e^{-x}\right)^{-1}\right)' = -1\left(1+e^{-x}\right)^{-2} \cdot \left(1+e^{-x}\right)' =$$

$$= -1\left(1+e^{-x}\right)^{-2} \cdot e^{-x} \cdot \left(-x\right)' = -1\left(1+e^{-x}\right)^{-2} \cdot e^{-x} \cdot (-1)$$

and finally this one

$$\sigma'(x) = -1\left(1+e^{-x}\right)^{-2} \cdot e^{-x} \cdot (-1) = \frac{e^{-x}}{\left(1+e^{-x}\right)^2}$$

Let's note that:

$$1-\sigma = 1 - \frac{1}{1+e^{-x}} = \frac{1+e^{-x}-1}{1+e^{-x}} = \frac{e^{-x}}{1+e^{-x}} \qquad \Rightarrow \sigma' = \sigma(1-\sigma)$$

The same thing with β, FYI

UNIVERSITY OF LEEDS

The sigmoid function: $h(x) = \dfrac{1}{1+e^{-\beta x}} \equiv \sigma_\beta$

We derive the most external function first

Then this

$$\sigma_\beta{}'(x) = \left(\left(1+e^{-\beta x}\right)^{-1}\right)' = -1\left(1+e^{-\beta x}\right)^{-2} \cdot \left(1+e^{-\beta x}\right)' =$$

$$= -1\left(1+e^{-\beta x}\right)^{-2} \cdot e^{-\beta x} \cdot \left(-\beta x\right)' = -1\left(1+e^{-\beta x}\right)^{-2} \cdot e^{-\beta x} \cdot \left(-\beta\right)$$

and finally this one

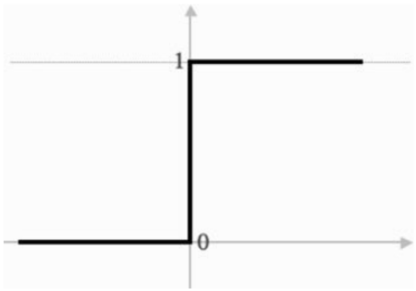
$$\sigma_\beta{}'(x) = -1\left(1+e^{-\beta x}\right)^{-2} \cdot e^{-\beta x} \cdot (-\beta) = \frac{\beta e^{-\beta x}}{\left(1+e^{-\beta x}\right)^2}$$

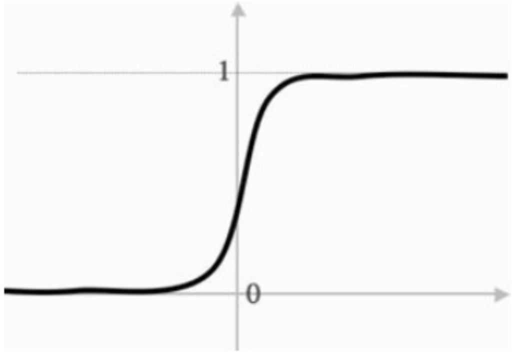
Let’s note that:

$$1-\sigma_\beta = 1 - \frac{1}{1+e^{-\beta x}} = \frac{1+e^{-\beta x}-1}{1+e^{-\beta x}} = \frac{e^{-\beta x}}{1+e^{-\beta x}} \qquad \Rightarrow \sigma_\beta{}' = \beta\,\sigma_\beta\left(1-\sigma_\beta\right)$$

A different activation function

UNIVERSITY OF LEEDS





before

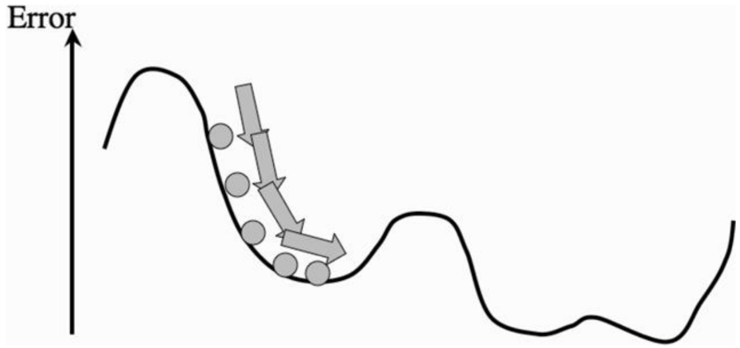
$$y(\boldsymbol{w}^T\boldsymbol{x}) = \begin{cases} 1 & \text{if} \quad \boldsymbol{w}^T\boldsymbol{x} > 0 \\ 0 & \text{if} \quad \boldsymbol{w}^T\boldsymbol{x} \leq 0 \end{cases}$$

after

$$y(\boldsymbol{w}^T\boldsymbol{x}) = \frac{1}{1+e^{-\beta \boldsymbol{w}^T \boldsymbol{x}}}$$

$$\sigma_\beta'(x) = \beta \frac{e^{-\beta x}}{(1+e^{-\beta x})^2} = \beta\, \sigma_\beta(x)(1-\sigma_\beta(x))$$

Gradient descent (again)



$$\boldsymbol{w}_{t+1} = \boldsymbol{w}_t - \eta \nabla E(\boldsymbol{x})$$

Perceptron

$$E_p(\boldsymbol{X}) = \sum_{\boldsymbol{x}_n \in \boldsymbol{X}} \boldsymbol{w}^t \boldsymbol{x}_n (y_n - t_n)$$

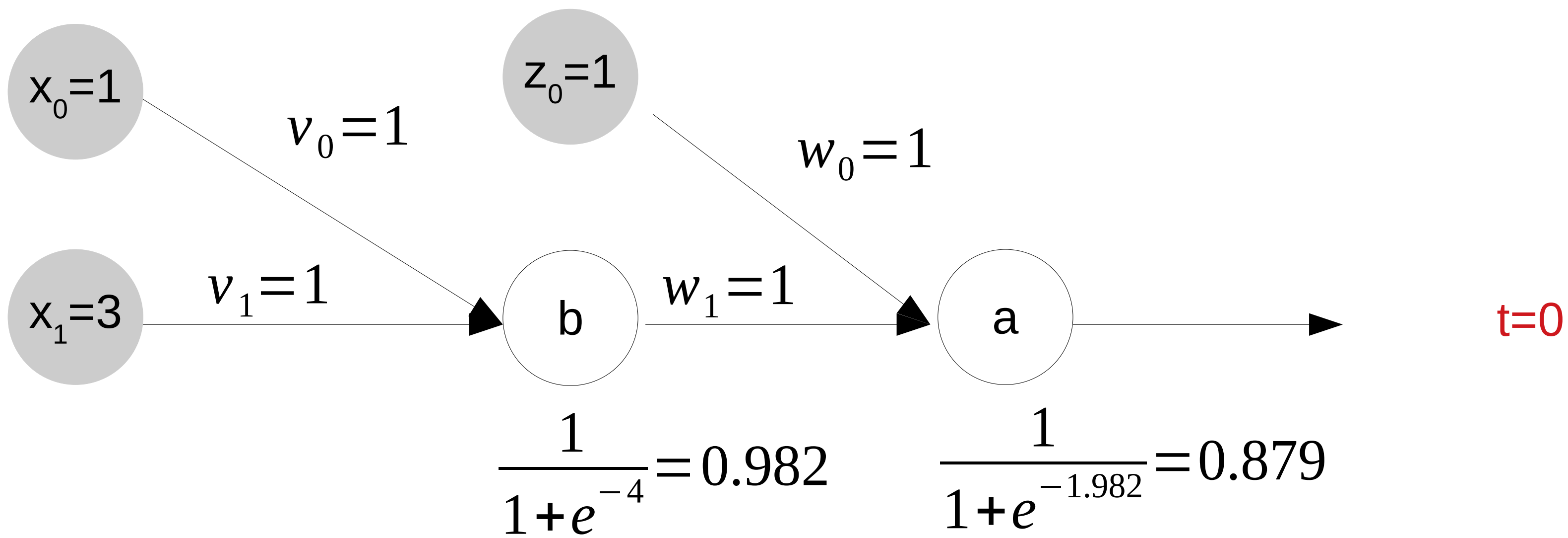
$$\boldsymbol{w}_{t+1} = \boldsymbol{w}_t - \eta \boldsymbol{x}(y - t)$$

Multi-Layer P

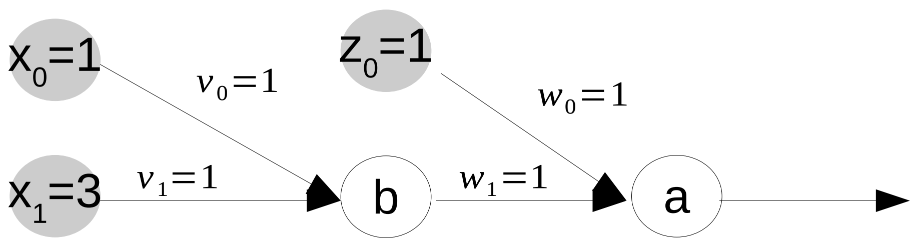
$$E_m(\boldsymbol{X}) = \frac{1}{2} \sum_{\boldsymbol{x}_n \in \boldsymbol{X}} (y_n - t_n)^2$$

?

Example


x_0=1
x_1=3
z_0=1
$v_0=1$
$v_1=1$
$w_0=1$
$w_1=1$
b
a
t=0
$\frac{1}{1+e^{-4}}=0.982$
$\frac{1}{1+e^{-1.982}}=0.879$

Example



$$b=v_0 x_0+v_1 x \qquad z_1=\sigma(b) \qquad a=w_0 z_0+w_1 z_1 \qquad y=\sigma(a)$$

$$\frac{\partial E}{\partial w_0}=\frac{\partial E}{\partial a}\frac{\partial a}{\partial w_0}$$

chain rule

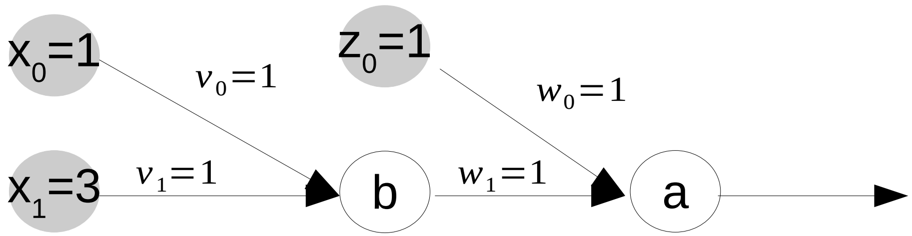
$$\frac{\partial E}{\partial a}=\frac{\partial}{\partial a}\frac{1}{2}(\sigma(a)-t)^2=(\sigma(a)-t)\cdot\ \sigma(a)(1-\sigma(a))$$

$$\frac{\partial a}{\partial w_0}=\frac{\partial}{\partial w_0}w_0 z_0+w_1 z= z_0$$

$$\boxed{\frac{\partial E}{\partial w_0}=(y-t)y(1-y)z_0}$$

$$\boxed{\frac{\partial E}{\partial w_1}=(y-t)y(1-y)z_1}$$

Example



$$b=v_0 x_0+v_1 x \qquad z_1=\sigma(b) \qquad a=w_0 z_0+w_1 z_1 \qquad y=\sigma(a)$$

$$a=w_0 z_0+w_1 \sigma(b)$$

$$\frac{\partial E}{\partial v_0}=\frac{\partial E}{\partial a}\frac{\partial a}{\partial b}\frac{\partial b}{\partial v_0}$$

$$\frac{\partial E}{\partial a}=(y-t)\,y(1-y)$$ from before

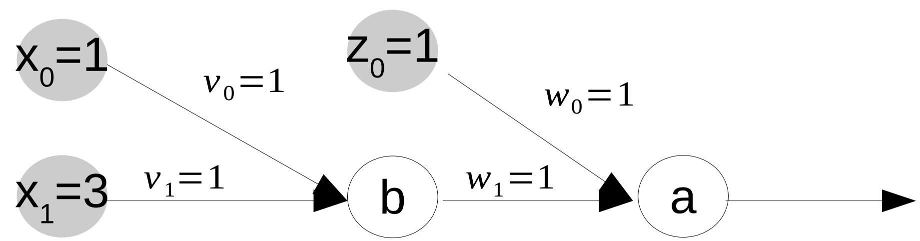
$$\frac{\partial a}{\partial b}=\frac{\partial}{\partial b}w_0 z_0+w_1\sigma(b)=w_1\sigma(b)(1-\sigma(b))=w_1 z_1(1-z_1)$$

$$\frac{\partial}{\partial v_0}v_0 x_0+v_1 x_1=x_0$$

$$\boxed{\frac{\partial E}{\partial v_0}=(y-t)\,y(1-y)\,w_1 z_1(1-z_1)\,x_0}$$

$$\boxed{\frac{\partial E}{\partial v_1}=(y-t)\,y(1-y)\,w_1 z_1(1-z_1)\,x_1}$$

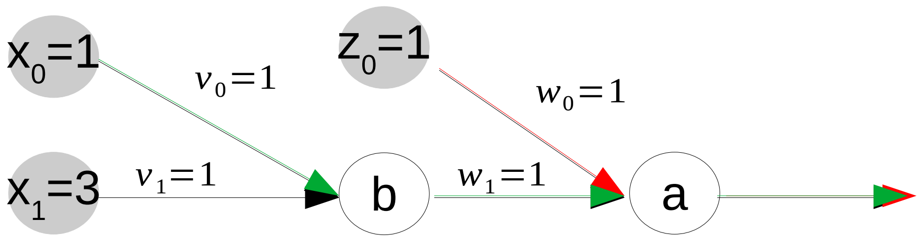
Example



$$b = v_0 x_0 + v_1 x \qquad z_1 = \sigma(b) \qquad a = w_0 z_0 + w_1 z_1 \qquad y = \sigma(a)$$

$$\nabla E = \begin{bmatrix} \dfrac{\partial E}{\partial w_0} \\ \dfrac{\partial E}{\partial w_1} \\ \dfrac{\partial E}{\partial v_0} \\ \dfrac{\partial E}{\partial v_1} \end{bmatrix} = \begin{bmatrix} (y-t)\,y(1-y)\,z_0 \\ (y-t)\,y(1-y)\,z_1 \\ (y-t)\,y(1-y)\,w_1 z_1 (1-z_1)\,x_0 \\ (y-t)\,y(1-y)\,w_1 z_1 (1-z_1)\,x_1 \end{bmatrix} \nabla E(\boldsymbol{w}) = \begin{bmatrix} 0.09 \\ 0.09 \\ 0.002 \\ 0.002 \end{bmatrix}$$

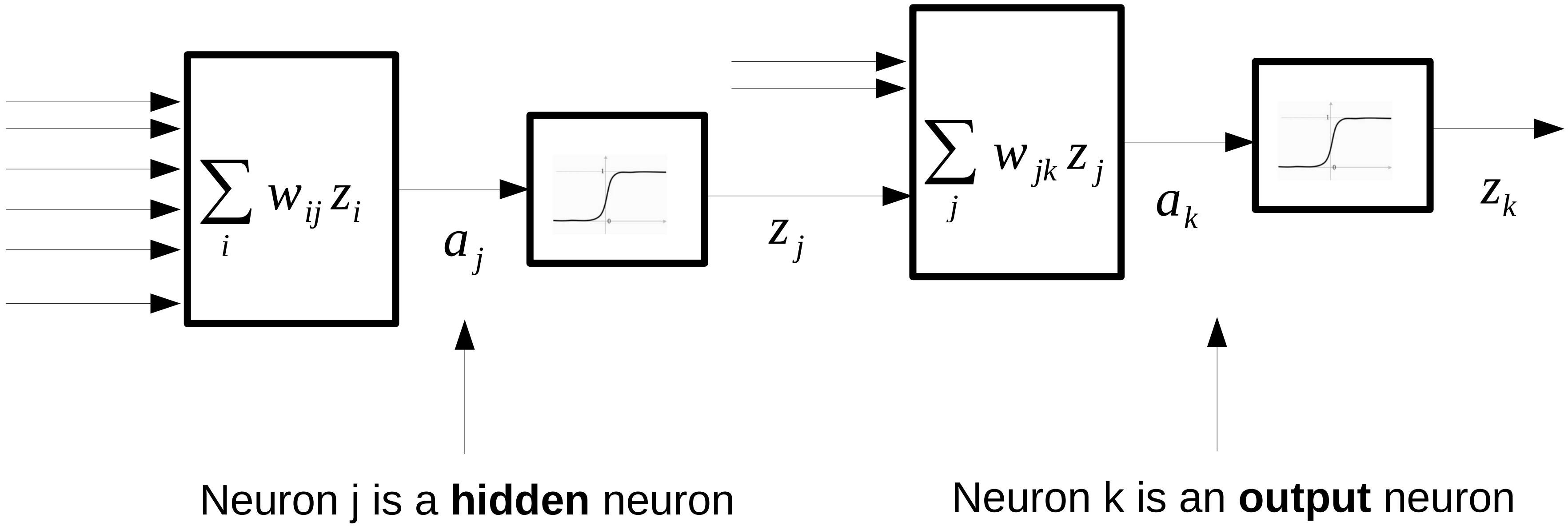
Summary



$$\frac{\partial E}{\partial w_0}=\frac{\partial E}{\partial a}\frac{\partial a}{\partial w_0}$$

$$\frac{\partial E}{\partial v_0}=\frac{\partial E}{\partial a}\frac{\partial a}{\partial b}\frac{\partial b}{\partial v_0}$$

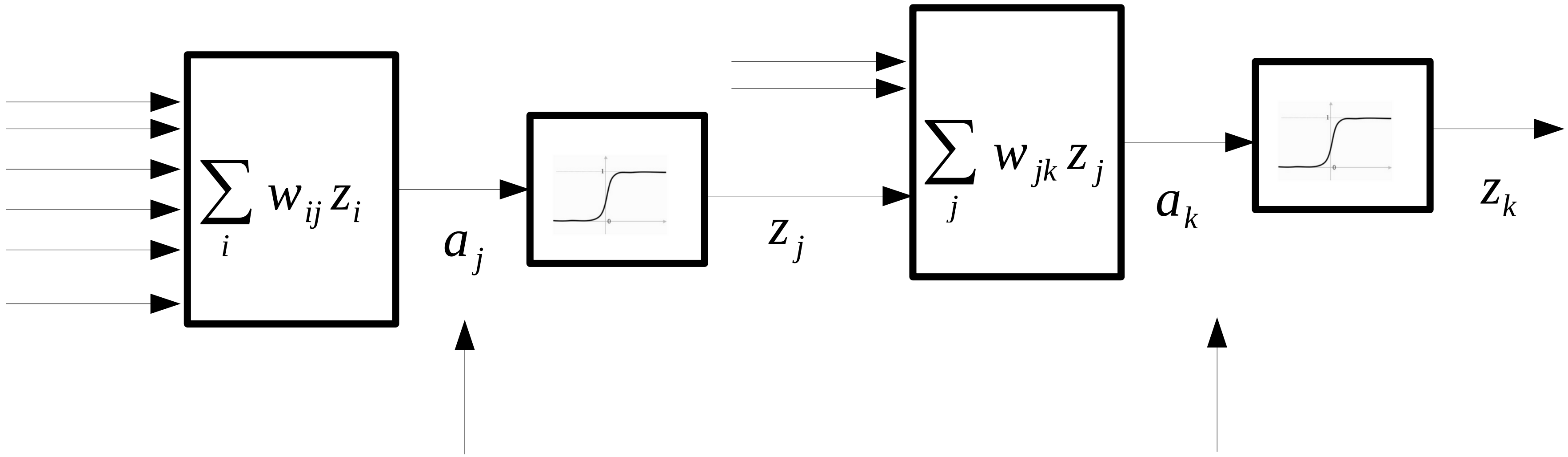
Backpropagation of errors, notation



... $a_j = \sum_{i=1}^{N} w_{ij} z_i$ $z_j = f(a_j)$ $a_k = \sum_{j=1}^{M} w_{jk} z_j$ $z_k = f(a_k)$

Forward pass

UNIVERSITY OF LEEDS



Neuron j is a **hidden** neuron

Neuron k is an **output** neuron

... $a_j = \sum_{i=1}^{N} w_{ij} z_i$ $\quad z_j = f(a_j)$ $\quad a_k = \sum_{j=1}^{M} w_{jk} z_j$ $\quad z_k = f(a_k)$

Forward pass: compute all the z

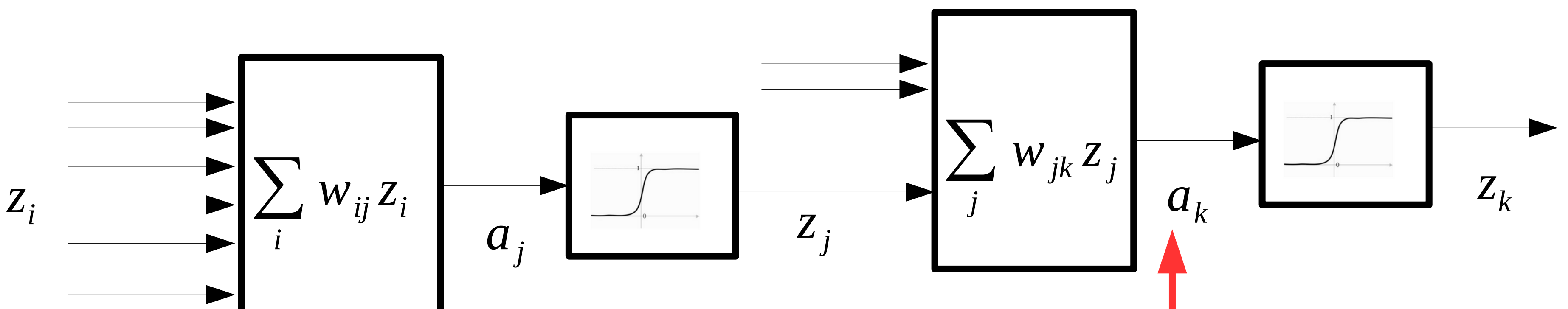
Backward pass, output neuron

How does a_k affect the error?

$$E(\mathbf{x}) = \frac{1}{2}(y - t)^2 = \frac{1}{2}(z_k - t)^2$$

$$\frac{\partial E}{\partial a_k} = \frac{\partial}{\partial a_k}\frac{1}{2}(z_k - t_k)^2 = \frac{\partial}{\partial a_k}\frac{1}{2}(\sigma(a_k) - t_k)^2 = (\sigma(a_k) - t_k)\,\sigma(a_k)(1 - \sigma(a_k))$$

Now this useful, because it cancels out the exponent in the derivation



We call this derivative δ: $\delta_k = (z_k - t_k)\, z_k (1 - z_k)$

Backward pass

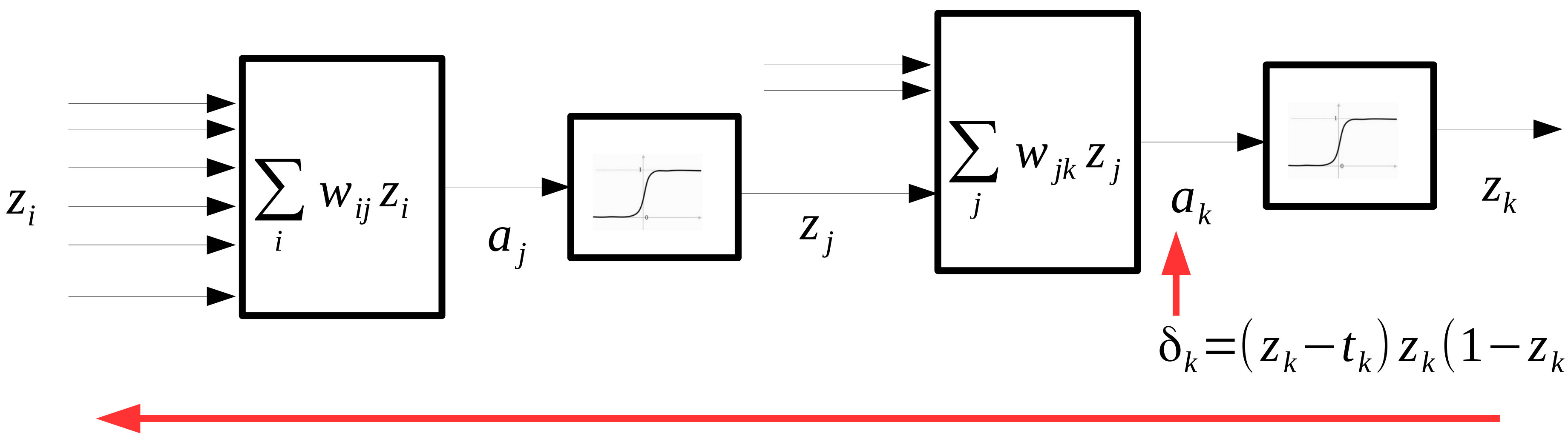
Backward pass, output neuron

One step backward, inside the box: how does w_{jk} affect the error?

$$E(\mathbf{x}) = \frac{1}{2}(z_k - t)^2 = \frac{1}{2}(\sigma(a_k) - t)^2 \qquad a_k = \sum_j w_{jk} z_j$$

We apply the chain rule again: $$\frac{\partial E}{\partial w_{jk}} = \frac{\partial E}{\partial a_k} \frac{\partial a_k}{\partial w_{jk}} = \delta_k \cdot ?$$

$$\frac{\partial a_k}{\partial w_{jk}} = \frac{\partial}{\partial w_{jk}} w_{0k} z_0 + w_{1k} z_1 + w_{2k} z_2 + \cdots + w_{jk} z_j = ?$$



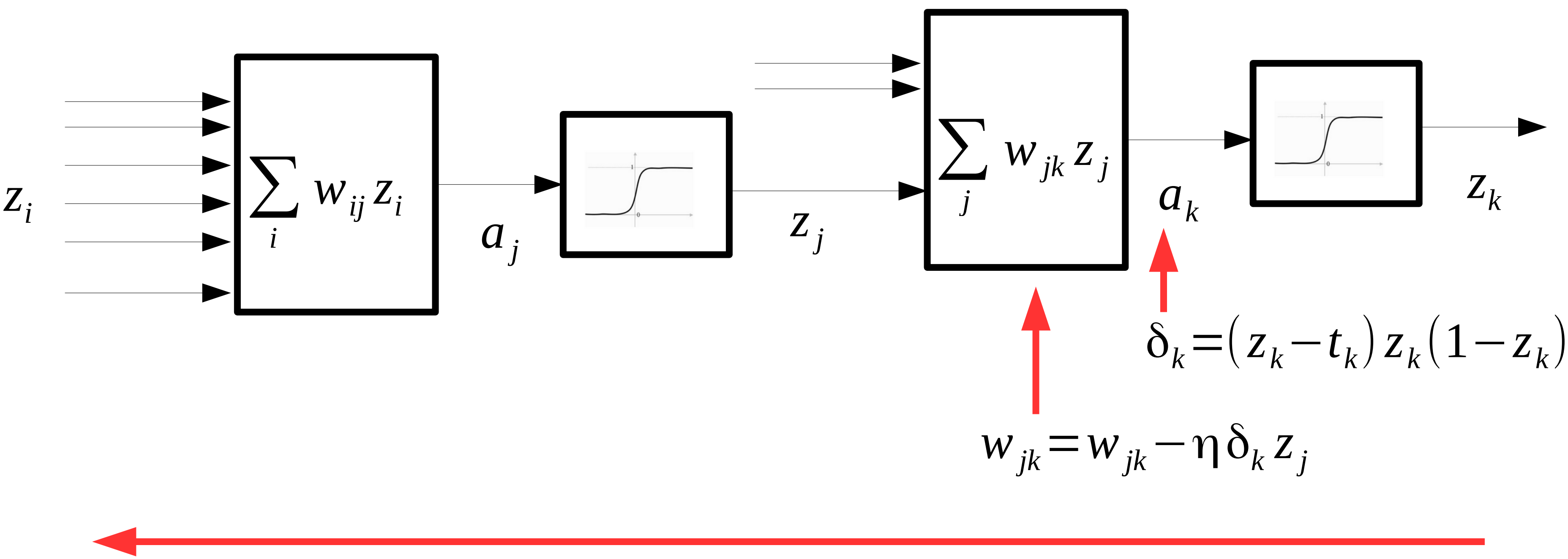
$$\delta_k = (z_k - t_k) z_k (1 - z_k)$$

Backward pass, output neuron

One step backward, inside the box: how does w_{jk} affect the error?

We apply the chain rule again: $$\frac{\partial E}{\partial w_{jk}} = \frac{\partial E}{\partial a_k} \frac{\partial a_k}{\partial w_{jk}} = \delta_k z_j$$

$$\frac{\partial a_k}{\partial w_{jk}} = \frac{\partial}{\partial w_{jk}} w_{0k} z_0 + w_{1k} z_1 + w_{2k} z_2 + \cdots + w_{jk} z_j = z_j$$



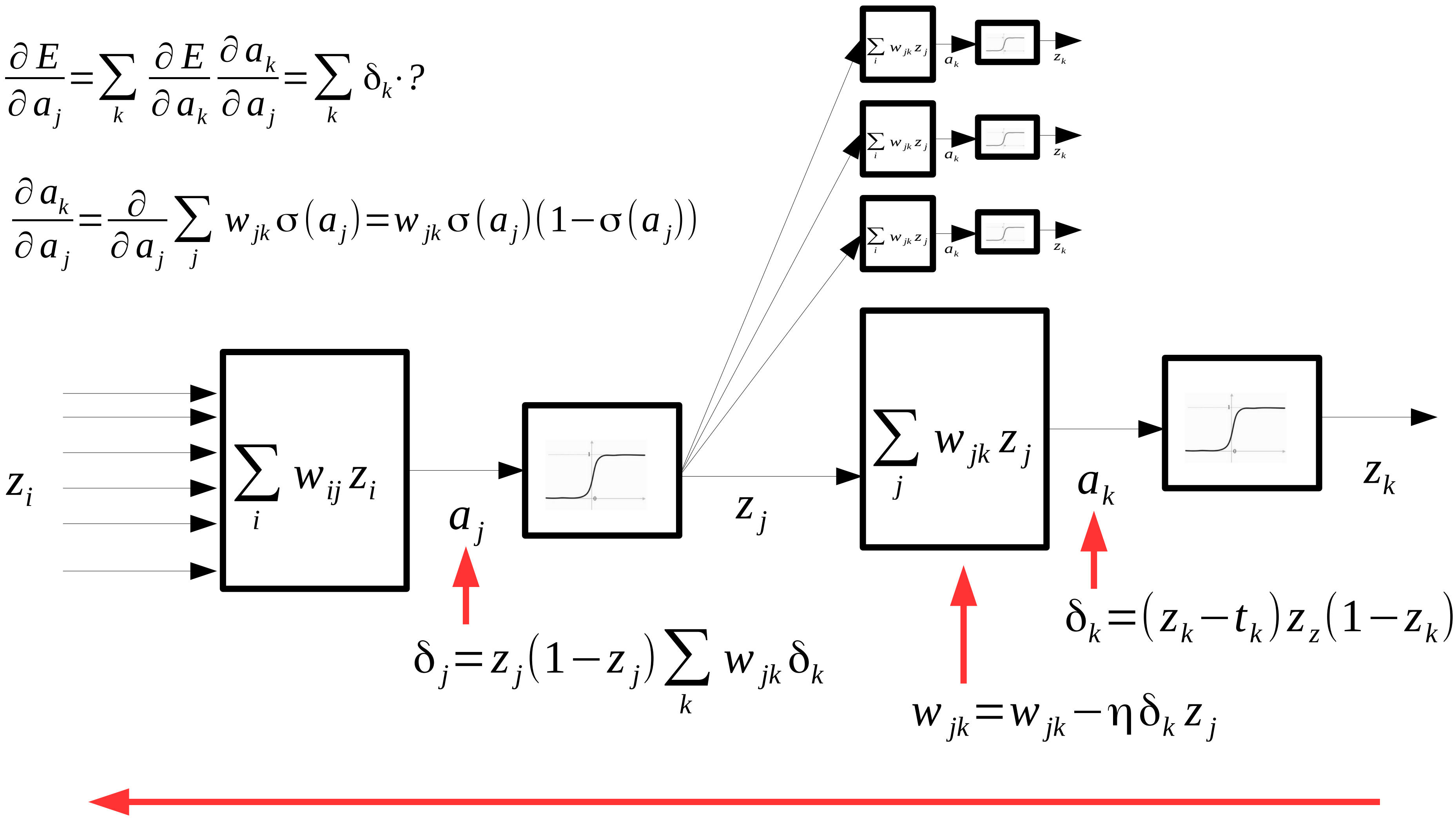
Backward pass, hidden neuron

UNIVERSITY OF LEEDS

One step backward: how does a_j affect the error?

$$\frac{\partial E}{\partial a_j} = \sum_k \frac{\partial E}{\partial a_k} \frac{\partial a_k}{\partial a_j} = \sum_k \delta_k \cdot ?$$

$$\frac{\partial a_k}{\partial a_j} = \frac{\partial}{\partial a_j} \sum_j w_{jk} \sigma(a_j) = w_{jk} \sigma(a_j)(1 - \sigma(a_j))$$



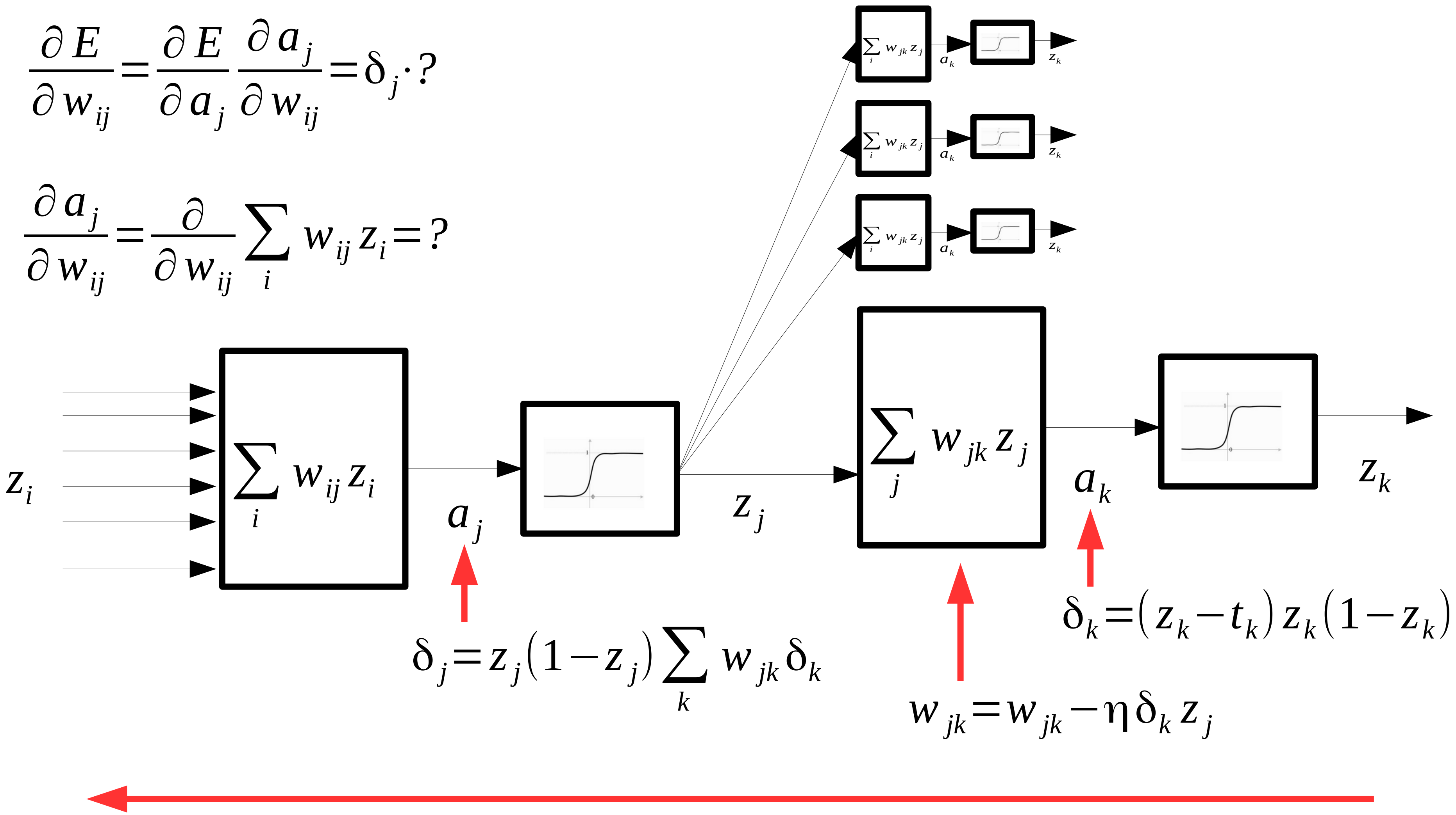
Computing delta

UNIVERSITY OF LEEDS

One step backward, inside the box: how does w_{ij} affect the error?

$$\frac{\partial E}{\partial w_{ij}} = \frac{\partial E}{\partial a_j} \frac{\partial a_j}{\partial w_{ij}} = \delta_j \cdot ?$$

$$\frac{\partial a_j}{\partial w_{ij}} = \frac{\partial}{\partial w_{ij}} \sum_i w_{ij} z_i = ?$$

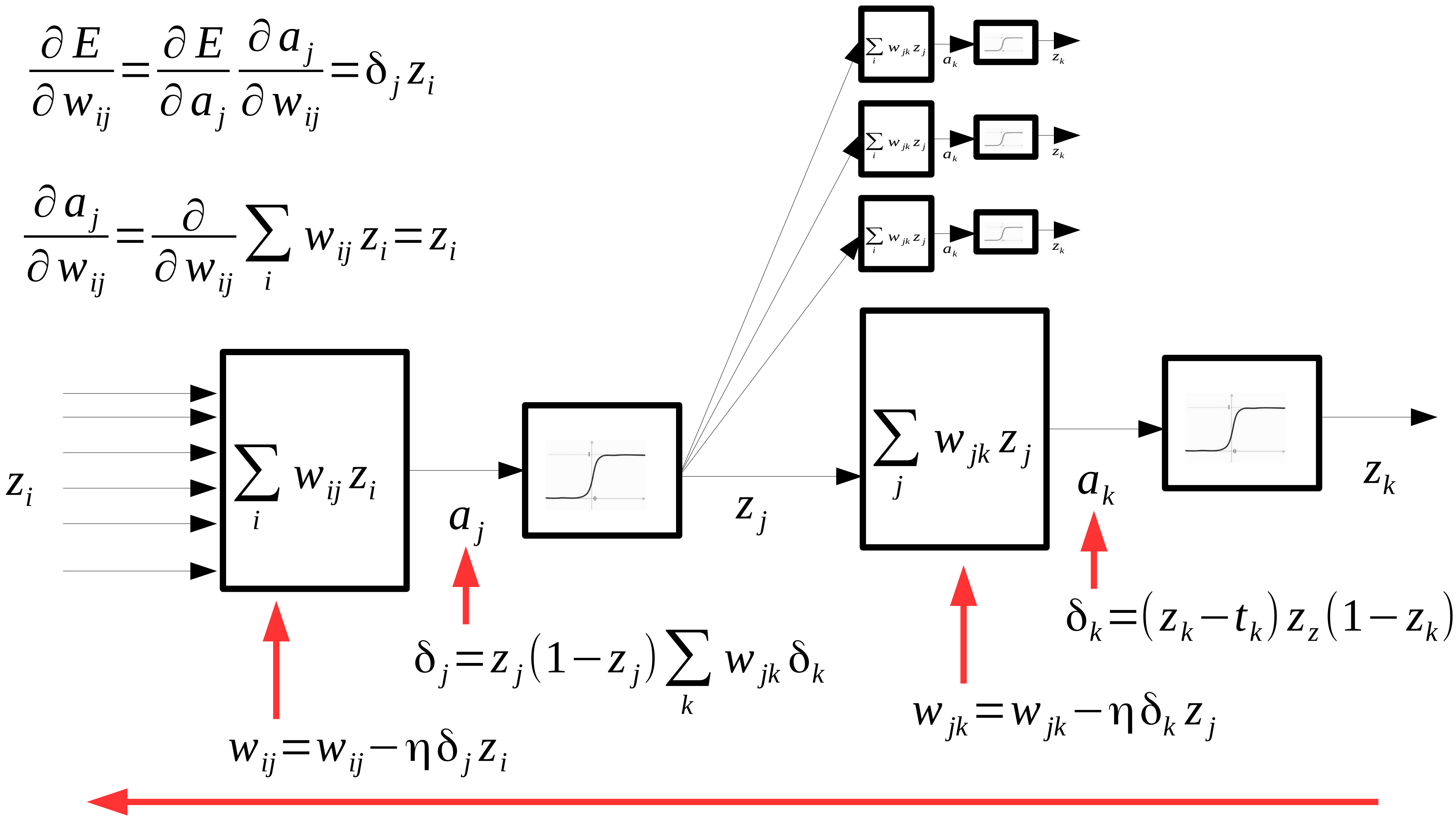


Computing delta

One step backward, inside the box: how does w_{ij} affect the error?

$$\frac{\partial E}{\partial w_{ij}} = \frac{\partial E}{\partial a_j} \frac{\partial a_j}{\partial w_{ij}} = \delta_j z_i$$

$$\frac{\partial a_j}{\partial w_{ij}} = \frac{\partial}{\partial w_{ij}} \sum_i w_{ij} z_i = z_i$$



Gradient descent (again)

UNIVERSITY OF LEEDS

$$\boldsymbol{w}_{t+1} = \boldsymbol{w}_t - \eta \nabla E(\boldsymbol{x})$$

Perceptron

$$E_p(\boldsymbol{X}) = \sum_{\boldsymbol{x}_n \in \boldsymbol{X}} \boldsymbol{w}^t \boldsymbol{x}_n (y_n - t_n)$$

$$\boldsymbol{w}_{t+1} = \boldsymbol{w}_t - \eta (y - t) \boldsymbol{x}$$

Multi-Layer P

$$E_m(\boldsymbol{X}) = \frac{1}{2} \sum_{\boldsymbol{x}_n \in \boldsymbol{X}} (y_n - t_n)^2$$

Output:

$$\delta_{\text{output}} = (y - t) y (1 - y)$$

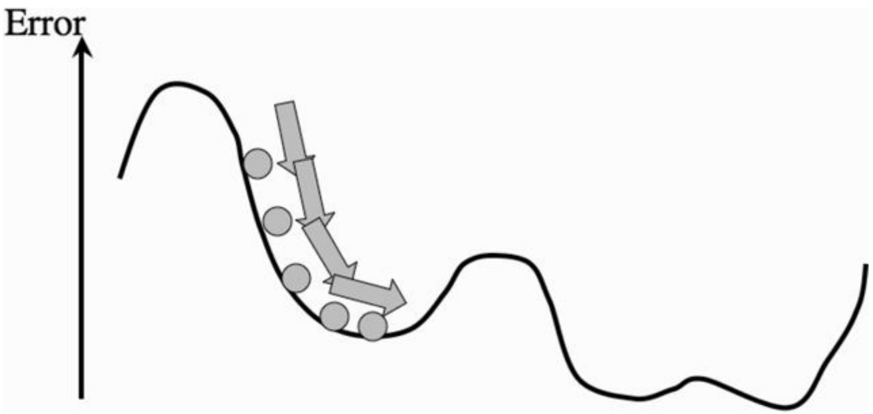
$$\boldsymbol{w}_{t+1} = \boldsymbol{w}_t - \eta \delta_{\text{output}} \boldsymbol{x}$$

Hidden:

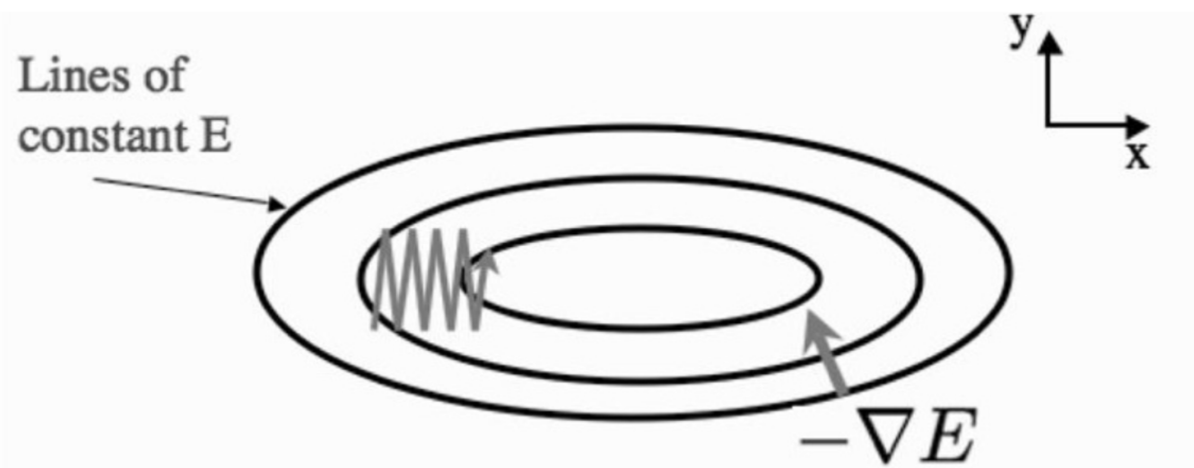
$$\delta_{\text{hidden}} = \sum_k w_k \delta_k$$

$$\boldsymbol{v}_{t+1} = \boldsymbol{v}_t - \eta \delta_{\text{hidden}} \boldsymbol{x}$$

Local Minima



Start with weights close to 0: where the decision is actually made

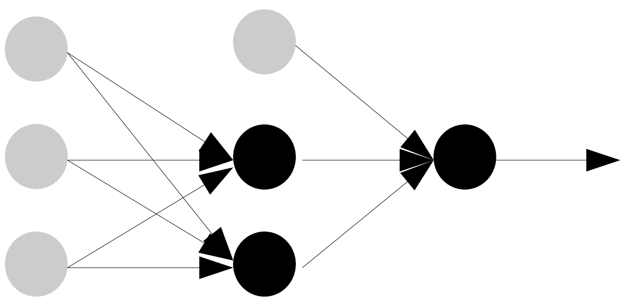


Multiple random restarts

Using MLPs

UNIVERSITY OF LEEDS

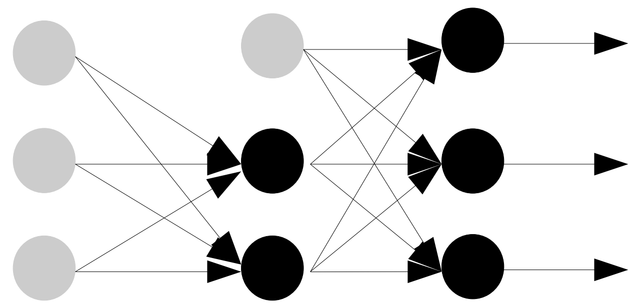
Regression



$$y(a) = a$$

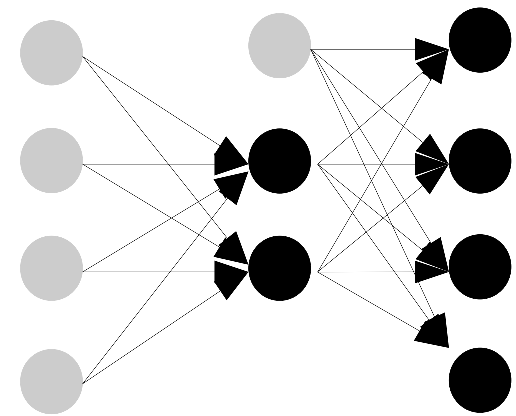
Last neuron
linear

Classification



One output
per class, pick
highest

Compression
(Autoencoder)



Middle
“bottleneck”
layer

Training "recipe"

Choose features

Normalize (rescale) data:

$$x' = \frac{x - \bar{x}}{\sigma}$$

Zero mean, unit variance

or

$$x' = \frac{x - min(x)}{max(x) - min(x)}$$

in [0,1]

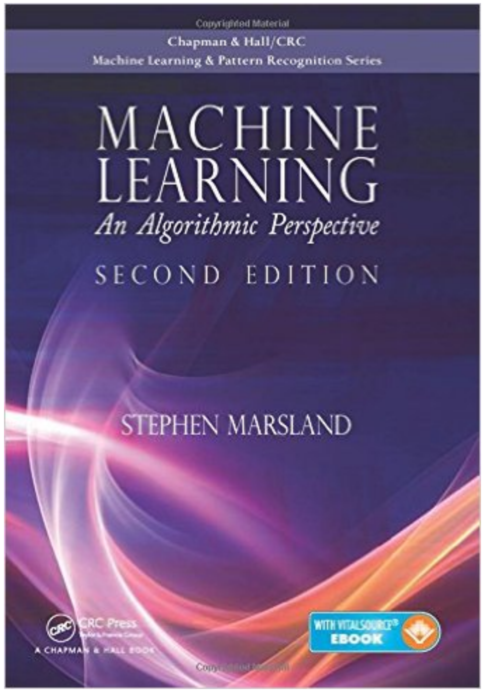
Create training, validation, and test sets

Decide whether you need hidden layers and how big. Try several ones.

Train

Test

Conclusion


Copyrighted Material
Chapman & Hall/CRC
Machine Learning & Pattern Recognition Series
MACHINE LEARNING
An Algorithmic Perspective
SECOND EDITION
STEPHEN MARSLAND
CRC Press
A CHAPMAN & HALL BOOK
WITH VITALSOURCE® EBOOK
Copyrighted Material

Chapter 4