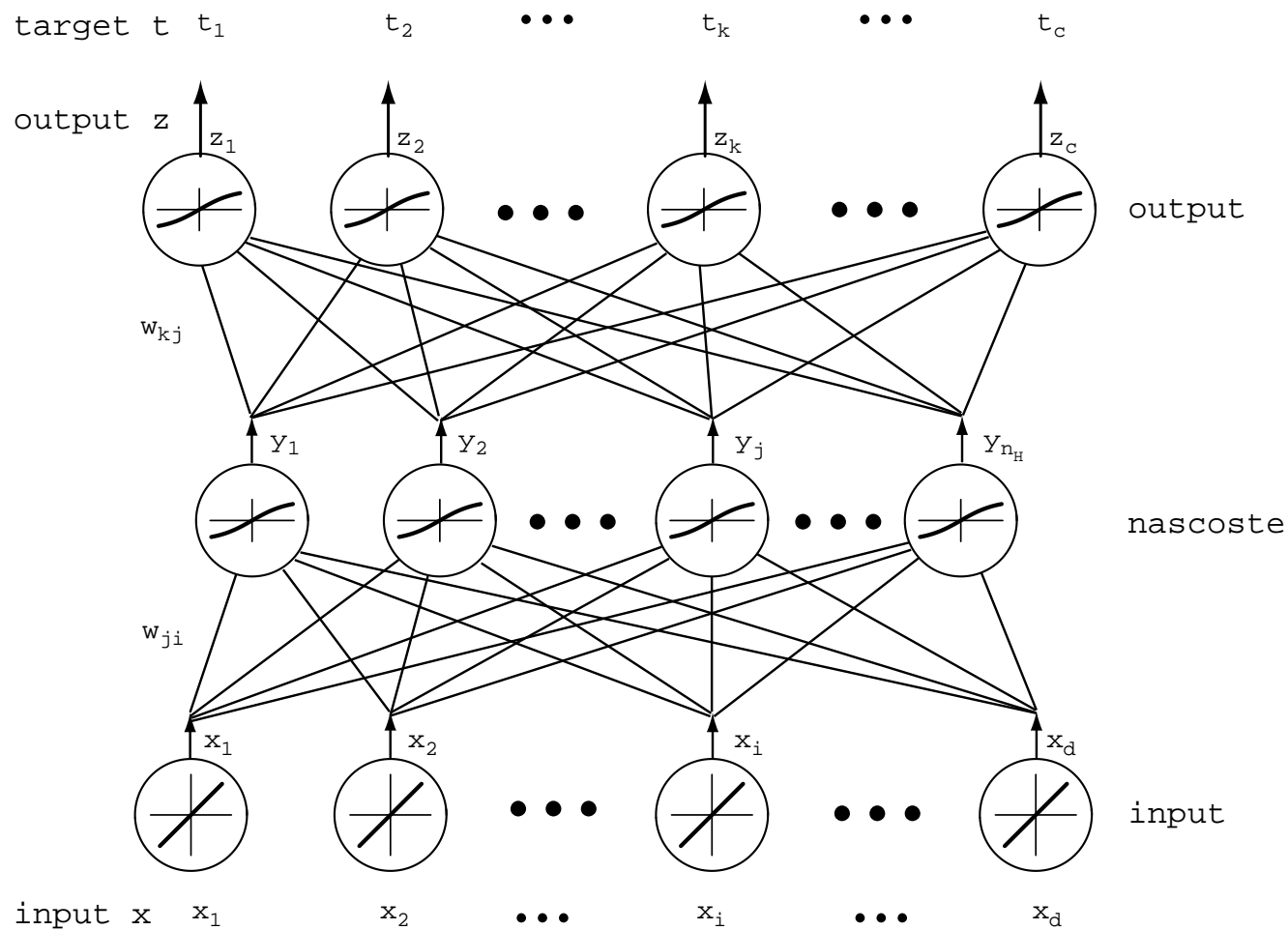


## Reti Neurali Feed-forward: notazione



## Reti Neurali Feed-forward: notazione

- $d$  unità di ingresso, dimensione dei dati in ingresso  $\vec{x} \equiv (x_1, \dots, x_d)$   
( $d + 1$  se si include la soglia nel vettore dei pesi  $\vec{x}' \equiv (x_0, x_1, \dots, x_d)$ )
- $N_H$  unità nascoste (con output  $\vec{y} \equiv (y_1, \dots, y_{N_H})$ )
- $c$  unità di output, dimensione dei dati in output  $\vec{z} \equiv (z_1, \dots, z_c)$
- $c$ , dimensione dei dati desiderati  $\vec{t} \equiv (t_1, \dots, t_c)$
- $w_{ji}$  peso dalla unità di ingresso  $i$  alla unità nascosta  $j$
- $w_{kj}$  peso dalla unità nascosta  $j$  alla unità di output  $k$

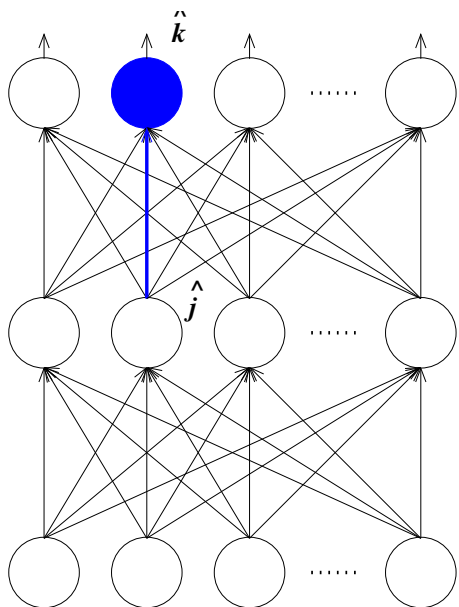
La funzione errore, considerando che si hanno  $c$  unità di output, diventa

$$E[\vec{w}] = \frac{1}{2cN_{Tr}} \sum_{(\vec{x}^{(p)}, \vec{t}^{(p)}) \in Tr} \sum_{k=1}^c \left( t_k^{(p)} - z_k(\vec{x}^{(p)}) \right)^2$$

## Calcolo del gradiente per i pesi di una unità di output

Fissiamo gli indici  $\hat{k}$  e  $\hat{j}$ :

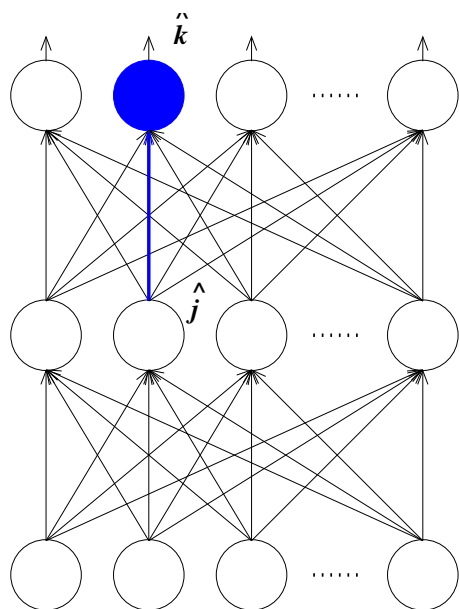
$$\frac{\partial E}{\partial w_{\hat{k}\hat{j}}} = \frac{\partial}{\partial w_{\hat{k}\hat{j}}} \frac{1}{2cN_{Tr}} \sum_{p \in Tr} \sum_{k=1}^c (t_k^{(p)} - z_k^{(p)})^2$$



## Calcolo del gradiente per i pesi di una unità di output

Fissiamo gli indici  $\hat{k}$  e  $\hat{j}$ :

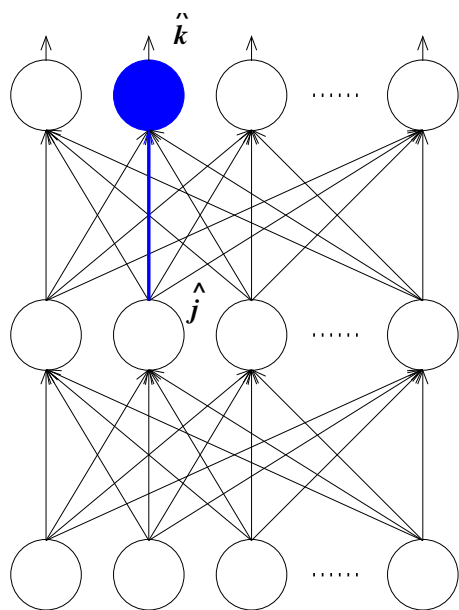
$$\begin{aligned} \frac{\partial E}{\partial w_{\hat{k}\hat{j}}} &= \frac{\partial}{\partial w_{\hat{k}\hat{j}}} \frac{1}{2cN_{Tr}} \sum_{p \in Tr} \sum_{k=1}^c (t_k^{(p)} - z_k^{(p)})^2 \\ &= \frac{1}{2cN_{Tr}} \sum_{p \in Tr} \frac{\partial}{\partial w_{\hat{k}\hat{j}}} \sum_{k=1}^c (t_k^{(p)} - z_k^{(p)})^2 \end{aligned}$$



## Calcolo del gradiente per i pesi di una unità di output

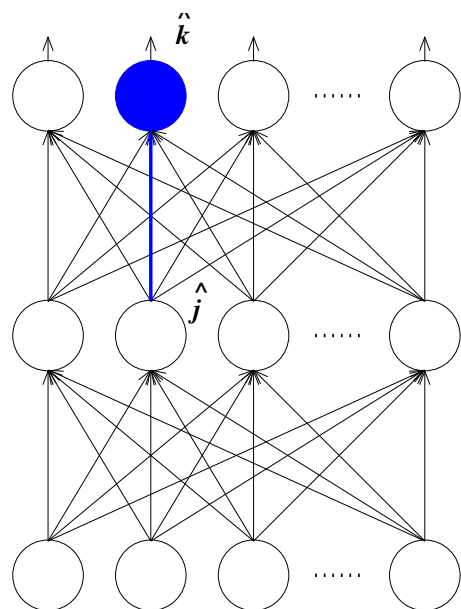
Fissiamo gli indici  $\hat{k}$  e  $\hat{j}$ :

$$\begin{aligned}
 \frac{\partial E}{\partial w_{\hat{k}\hat{j}}} &= \frac{\partial}{\partial w_{\hat{k}\hat{j}}} \frac{1}{2cN_{Tr}} \sum_{p \in Tr} \sum_{k=1}^c (t_k^{(p)} - z_k^{(p)})^2 \\
 &= \frac{1}{2cN_{Tr}} \sum_{p \in Tr} \frac{\partial}{\partial w_{\hat{k}\hat{j}}} \sum_{k=1}^c (t_k^{(p)} - z_k^{(p)})^2 \\
 &= \frac{1}{2cN_{Tr}} \sum_{p \in Tr} 2(t_{\hat{k}}^{(p)} - z_{\hat{k}}^{(p)}) \frac{\partial}{\partial w_{\hat{k}\hat{j}}} (t_{\hat{k}}^{(p)} - z_{\hat{k}}^{(p)})
 \end{aligned}$$



## Calcolo del gradiente per i pesi di una unità di output

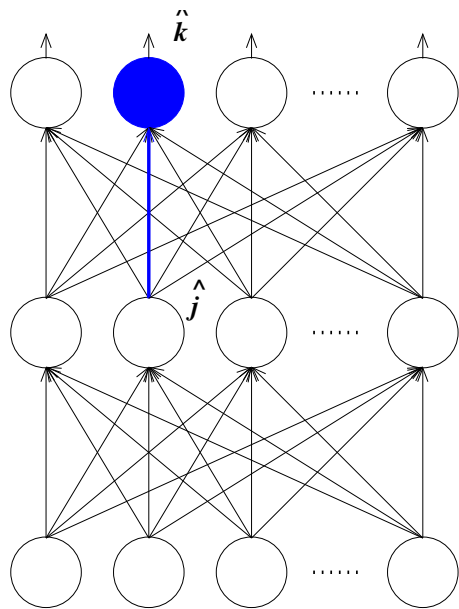
Fissiamo gli indici  $\hat{k}$  e  $\hat{j}$ :



$$\begin{aligned}
 \frac{\partial E}{\partial w_{\hat{k}\hat{j}}} &= \frac{\partial}{\partial w_{\hat{k}\hat{j}}} \frac{1}{2cN_{Tr}} \sum_{p \in Tr} \sum_{k=1}^c (t_k^{(p)} - z_k^{(p)})^2 \\
 &= \frac{1}{2cN_{Tr}} \sum_{p \in Tr} \frac{\partial}{\partial w_{\hat{k}\hat{j}}} \sum_{k=1}^c (t_k^{(p)} - z_k^{(p)})^2 \\
 &= \frac{1}{2cN_{Tr}} \sum_{p \in Tr} 2(t_{\hat{k}}^{(p)} - z_{\hat{k}}^{(p)}) \frac{\partial}{\partial w_{\hat{k}\hat{j}}} (t_{\hat{k}}^{(p)} - z_{\hat{k}}^{(p)}) \\
 &= \frac{1}{cN_{Tr}} \sum_{p \in Tr} (t_{\hat{k}}^{(p)} - z_{\hat{k}}^{(p)}) \frac{\partial}{\partial w_{\hat{k}\hat{j}}} (t_{\hat{k}}^{(p)} - \sigma(\vec{w}_{\hat{k}} \cdot \vec{y}^{(p)}))
 \end{aligned}$$

## Calcolo del gradiente per i pesi di una unità di output

Fissiamo gli indici  $\hat{k}$  e  $\hat{j}$ :

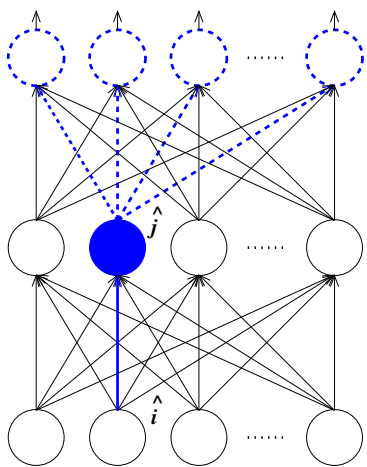


$$\begin{aligned}
 \frac{\partial E}{\partial w_{\hat{k}\hat{j}}} &= \frac{\partial}{\partial w_{\hat{k}\hat{j}}} \frac{1}{2cN_{Tr}} \sum_{p \in Tr} \sum_{k=1}^c (t_k^{(p)} - z_k^{(p)})^2 \\
 &= \frac{1}{2cN_{Tr}} \sum_{p \in Tr} \frac{\partial}{\partial w_{\hat{k}\hat{j}}} \sum_{k=1}^c (t_k^{(p)} - z_k^{(p)})^2 \\
 &= \frac{1}{2cN_{Tr}} \sum_{p \in Tr} 2(t_{\hat{k}}^{(p)} - z_{\hat{k}}^{(p)}) \frac{\partial}{\partial w_{\hat{k}\hat{j}}} (t_{\hat{k}}^{(p)} - z_{\hat{k}}^{(p)}) \\
 &= \frac{1}{cN_{Tr}} \sum_{p \in Tr} (t_{\hat{k}}^{(p)} - z_{\hat{k}}^{(p)}) \frac{\partial}{\partial w_{\hat{k}\hat{j}}} (t_{\hat{k}}^{(p)} - \sigma(\vec{w}_{\hat{k}} \cdot \vec{y}^{(p)})) \\
 &= -\frac{1}{cN_{Tr}} \sum_{p \in Tr} (t_{\hat{k}}^{(p)} - z_{\hat{k}}^{(p)}) \sigma'(\vec{w}_{\hat{k}} \cdot \vec{y}^{(p)}) y_{\hat{j}}^{(p)}
 \end{aligned}$$

## Calcolo del gradiente per i pesi di una unità nascosta

Fissiamo gli indici  $\hat{j}$  e  $\hat{i}$ :

$$\frac{\partial E}{\partial w_{\hat{j}\hat{i}}} = \frac{\partial}{\partial w_{\hat{j}\hat{i}}} \frac{1}{2cN_{Tr}} \sum_{p \in Tr} \sum_{k=1}^c (t_k^{(p)} - z_k^{(p)})^2$$

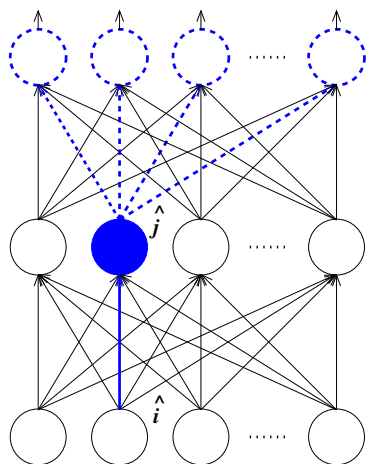




## Calcolo del gradiente per i pesi di una unità nascosta

Fissiamo gli indici  $\hat{j}$  e  $\hat{i}$ :

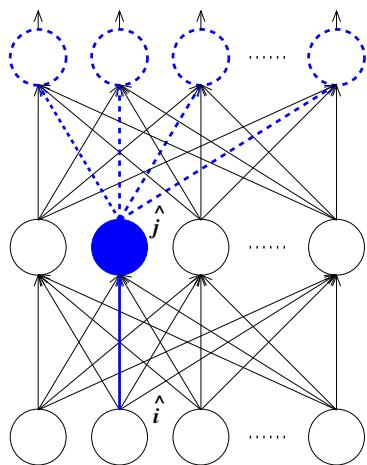
$$\begin{aligned} \frac{\partial E}{\partial w_{\hat{j}\hat{i}}} &= \frac{\partial}{\partial w_{\hat{j}\hat{i}}} \frac{1}{2cN_{Tr}} \sum_{p \in Tr} \sum_{k=1}^c (t_k^{(p)} - z_k^{(p)})^2 \\ &= \frac{1}{2cN_{Tr}} \sum_{p \in Tr} \sum_{k=1}^c \frac{\partial}{\partial w_{\hat{j}\hat{i}}} (t_k^{(p)} - z_k^{(p)})^2 \end{aligned}$$



## Calcolo del gradiente per i pesi di una unità nascosta

Fissiamo gli indici  $\hat{j}$  e  $\hat{i}$ :

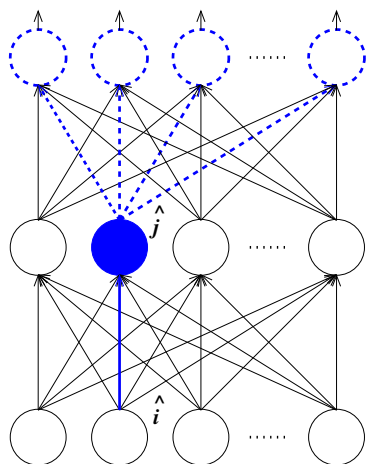
$$\begin{aligned}
 \frac{\partial E}{\partial w_{\hat{j}\hat{i}}} &= \frac{\partial}{\partial w_{\hat{j}\hat{i}}} \frac{1}{2cN_{Tr}} \sum_{p \in Tr} \sum_{k=1}^c (t_k^{(p)} - z_k^{(p)})^2 \\
 &= \frac{1}{2cN_{Tr}} \sum_{p \in Tr} \sum_{k=1}^c \frac{\partial}{\partial w_{\hat{j}\hat{i}}} (t_k^{(p)} - z_k^{(p)})^2 \\
 &= \frac{1}{2cN_{Tr}} \sum_{p \in Tr} \sum_{k=1}^c 2(t_k^{(p)} - z_k^{(p)}) \frac{\partial}{\partial w_{\hat{j}\hat{i}}} (-z_k^{(p)})
 \end{aligned}$$



## Calcolo del gradiente per i pesi di una unità nascosta

Fissiamo gli indici  $\hat{j}$  e  $\hat{i}$ :

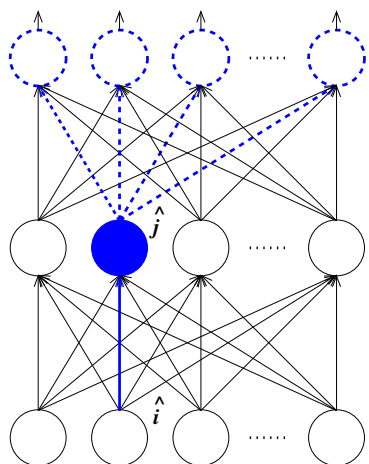
$$\begin{aligned}
 \frac{\partial E}{\partial w_{\hat{j}\hat{i}}} &= \frac{\partial}{\partial w_{\hat{j}\hat{i}}} \frac{1}{2cN_{Tr}} \sum_{p \in Tr} \sum_{k=1}^c (t_k^{(p)} - z_k^{(p)})^2 \\
 &= \frac{1}{2cN_{Tr}} \sum_{p \in Tr} \sum_{k=1}^c \frac{\partial}{\partial w_{\hat{j}\hat{i}}} (t_k^{(p)} - z_k^{(p)})^2 \\
 &= \frac{1}{2cN_{Tr}} \sum_{p \in Tr} \sum_{k=1}^c 2(t_k^{(p)} - z_k^{(p)}) \frac{\partial}{\partial w_{\hat{j}\hat{i}}} (-z_k^{(p)}) \\
 &= -\frac{1}{cN_{Tr}} \sum_{p \in Tr} \sum_{k=1}^c (t_k^{(p)} - z_k^{(p)}) \sigma'(\vec{w}_k \cdot \vec{y}^{(p)}) \frac{\partial}{\partial w_{\hat{j}\hat{i}}} \vec{w}_k \cdot \vec{y}^{(p)}
 \end{aligned}$$



## Calcolo del gradiente per i pesi di una unità nascosta

Fissiamo gli indici  $\hat{j}$  e  $\hat{i}$ :

$$\begin{aligned}
 \frac{\partial E}{\partial w_{\hat{j}\hat{i}}} &= \frac{\partial}{\partial w_{\hat{j}\hat{i}}} \frac{1}{2cN_{Tr}} \sum_{p \in Tr} \sum_{k=1}^c (t_k^{(p)} - z_k^{(p)})^2 \\
 &= \frac{1}{2cN_{Tr}} \sum_{p \in Tr} \sum_{k=1}^c \frac{\partial}{\partial w_{\hat{j}\hat{i}}} (t_k^{(p)} - z_k^{(p)})^2 \\
 &= \frac{1}{2cN_{Tr}} \sum_{p \in Tr} \sum_{k=1}^c 2(t_k^{(p)} - z_k^{(p)}) \frac{\partial}{\partial w_{\hat{j}\hat{i}}} (-z_k^{(p)}) \\
 &= -\frac{1}{cN_{Tr}} \sum_{p \in Tr} \sum_{k=1}^c (t_k^{(p)} - z_k^{(p)}) \sigma'(\vec{w}_k \cdot \vec{y}^{(p)}) \frac{\partial}{\partial w_{\hat{j}\hat{i}}} \sum_{j=1}^{N_H} w_{kj} y_j^{(p)}
 \end{aligned}$$



## Calcolo del gradiente per i pesi di una unità nascosta

Fissiamo gli indici  $\hat{j}$  e  $\hat{i}$ :

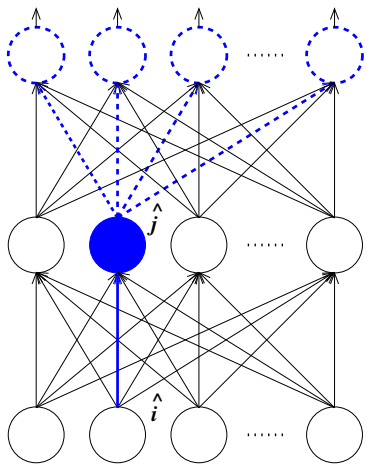
$$\frac{\partial E}{\partial w_{\hat{j}\hat{i}}} = \frac{\partial}{\partial w_{\hat{j}\hat{i}}} \frac{1}{2cN_{Tr}} \sum_{p \in Tr} \sum_{k=1}^c (t_k^{(p)} - z_k^{(p)})^2$$

$$= \frac{1}{2cN_{Tr}} \sum_{p \in Tr} \sum_{k=1}^c \frac{\partial}{\partial w_{\hat{j}\hat{i}}} (t_k^{(p)} - z_k^{(p)})^2$$

$$= \frac{1}{2cN_{Tr}} \sum_{p \in Tr} \sum_{k=1}^c 2(t_k^{(p)} - z_k^{(p)}) \frac{\partial}{\partial w_{\hat{j}\hat{i}}} (-z_k^{(p)})$$

$$= -\frac{1}{cN_{Tr}} \sum_{p \in Tr} \sum_{k=1}^c (t_k^{(p)} - z_k^{(p)}) \sigma'(\vec{w}_k \cdot \vec{y}^{(p)}) \frac{\partial}{\partial w_{\hat{j}\hat{i}}} \sum_{j=1}^{N_H} w_{kj} y_j^{(p)}$$

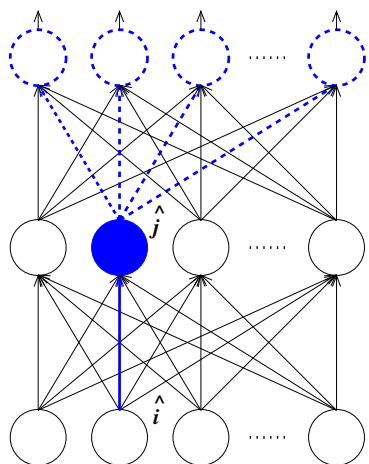
$$= -\frac{1}{cN_{Tr}} \sum_{p \in Tr} \sum_{k=1}^c (t_k^{(p)} - z_k^{(p)}) \sigma'(\vec{w}_k \cdot \vec{y}^{(p)}) w_{k\hat{j}} \frac{\partial}{\partial w_{\hat{j}\hat{i}}} y_{\hat{j}}^{(p)}$$



## Calcolo del gradiente per i pesi di una unità nascosta

Fissiamo gli indici  $\hat{j}$  e  $\hat{i}$ :

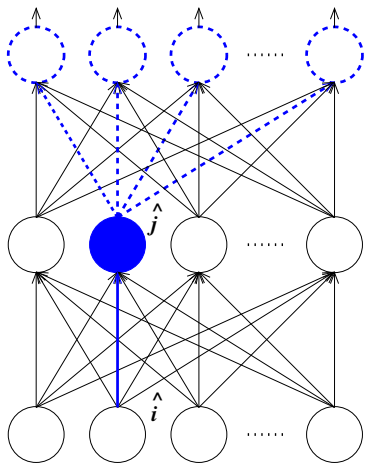
$$\begin{aligned}
 \frac{\partial E}{\partial w_{\hat{j}\hat{i}}} &= \frac{1}{2cN_{Tr}} \sum_{p \in Tr} \sum_{k=1}^c 2(t_k^{(p)} - z_k^{(p)}) \frac{\partial}{\partial w_{\hat{j}\hat{i}}} (-z_k^{(p)}) \\
 &= -\frac{1}{cN_{Tr}} \sum_{p \in Tr} \sum_{k=1}^c (t_k^{(p)} - z_k^{(p)}) \sigma'(\vec{w}_k \cdot \vec{y}^{(p)}) \frac{\partial}{\partial w_{\hat{j}\hat{i}}} \sum_{j=1}^{N_H} w_{kj} y_j^{(p)} \\
 &= -\frac{1}{cN_{Tr}} \sum_{p \in Tr} \sum_{k=1}^c (t_k^{(p)} - z_k^{(p)}) \sigma'(\vec{w}_k \cdot \vec{y}^{(p)}) w_{k\hat{j}} \frac{\partial}{\partial w_{\hat{j}\hat{i}}} y_{\hat{j}}^{(p)} \\
 &= -\frac{1}{cN_{Tr}} \sum_{p \in Tr} \sum_{k=1}^c (t_k^{(p)} - z_k^{(p)}) \sigma'(\vec{w}_k \cdot \vec{y}^{(p)}) w_{k\hat{j}} \sigma'(\vec{w}_{\hat{j}} \cdot \vec{x}^{(p)}) \frac{\partial}{\partial w_{\hat{j}\hat{i}}} (\vec{w}_{\hat{j}} \cdot \vec{x}^{(p)})
 \end{aligned}$$



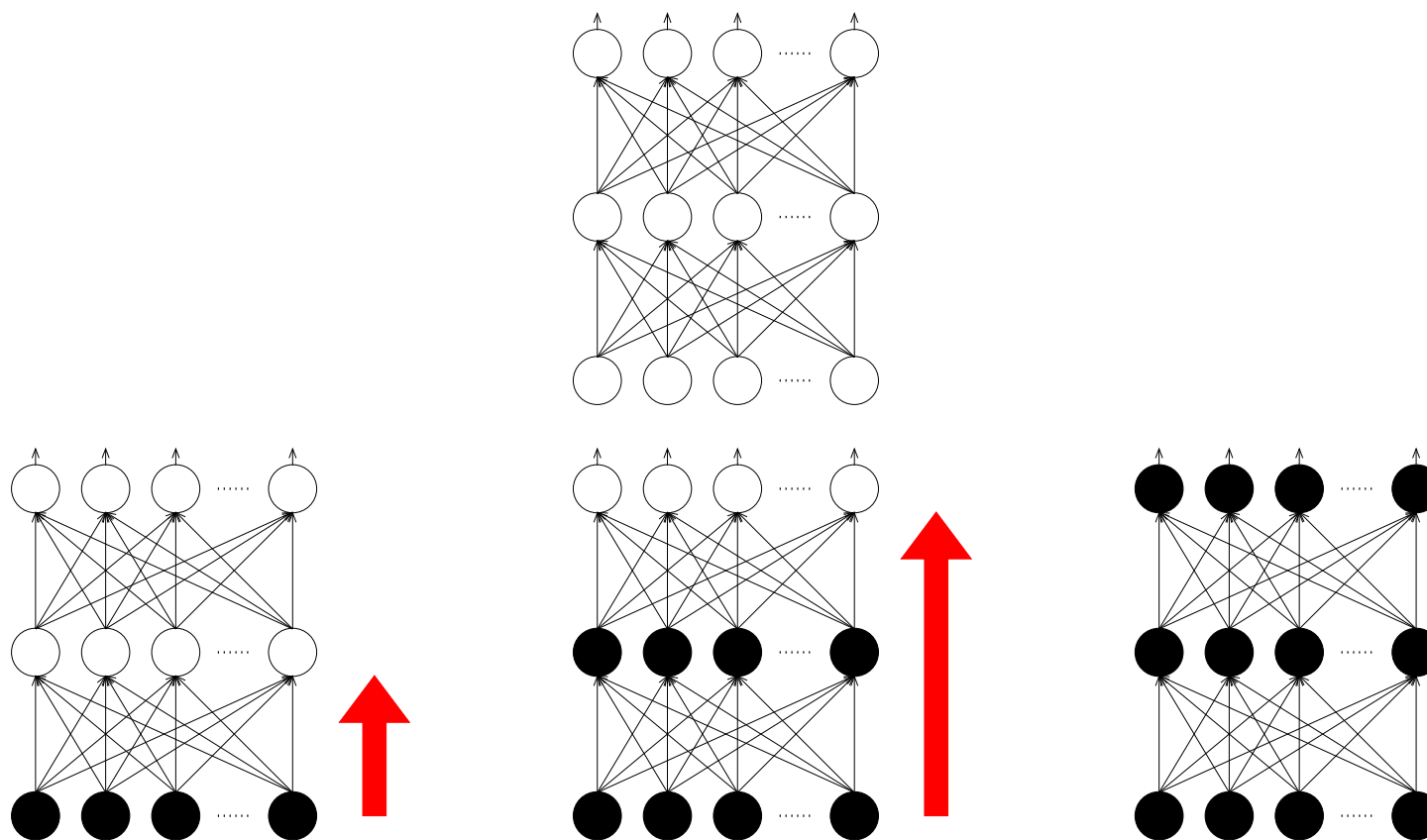
## Calcolo del gradiente per i pesi di una unità nascosta

Fissiamo gli indici  $\hat{j}$  e  $\hat{i}$ :

$$\begin{aligned}
 \frac{\partial E}{\partial w_{\hat{j}\hat{i}}} &= \frac{1}{2cN_{Tr}} \sum_{p \in Tr} \sum_{k=1}^c 2(t_k^{(p)} - z_k^{(p)}) \frac{\partial}{\partial w_{\hat{j}\hat{i}}} (-z_k^{(p)}) \\
 &= -\frac{1}{cN_{Tr}} \sum_{p \in Tr} \sum_{k=1}^c (t_k^{(p)} - z_k^{(p)}) \sigma'(\vec{w}_k \cdot \vec{y}^{(p)}) \frac{\partial}{\partial w_{\hat{j}\hat{i}}} \sum_{j=1}^{N_H} w_{kj} y_j^{(p)} \\
 &= -\frac{1}{cN_{Tr}} \sum_{p \in Tr} \sum_{k=1}^c (t_k^{(p)} - z_k^{(p)}) \sigma'(\vec{w}_k \cdot \vec{y}^{(p)}) w_{k\hat{j}} \frac{\partial}{\partial w_{\hat{j}\hat{i}}} y_{\hat{j}}^{(p)} \\
 &= -\frac{1}{cN_{Tr}} \sum_{p \in Tr} \sum_{k=1}^c (t_k^{(p)} - z_k^{(p)}) \sigma'(\vec{w}_k \cdot \vec{y}^{(p)}) w_{k\hat{j}} \sigma'(\vec{w}_{\hat{j}} \cdot \vec{x}^{(p)}) \frac{\partial}{\partial w_{\hat{j}\hat{i}}} (\vec{w}_{\hat{j}} \cdot \vec{x}^{(p)}) \\
 &= -\frac{1}{cN_{Tr}} \sum_{p \in Tr} \sum_{k=1}^c (t_k^{(p)} - z_k^{(p)}) \sigma'(\vec{w}_k \cdot \vec{y}^{(p)}) w_{k\hat{j}} \sigma'(\vec{w}_{\hat{j}} \cdot \vec{x}^{(p)}) x_{\hat{i}}^{(p)}
 \end{aligned}$$

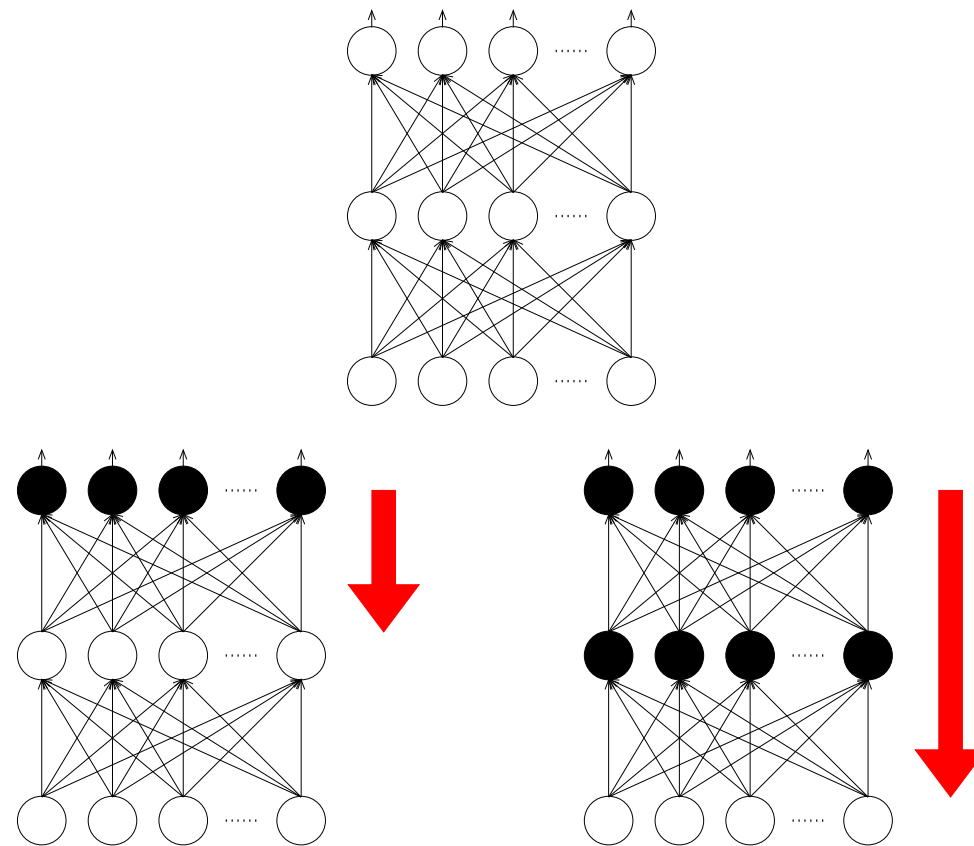


## Reti Neurali Feed-forward: Fase Forward





## Reti Neurali Feed-forward: Fase Backward



## Algoritmo Back-Propagation (uno strato nascosto, stocastico)

**Back-Propagation-1hl-stocastico**( $Tr, \eta, \text{topologia rete}$ )

- Inizializza tutti i pesi a valori random piccoli
- **Finché** la condizione di terminazione non è verificata, **fai**

– **Per ogni**  $(\vec{x}, \vec{t})$  in  $Tr$ , **fai**

1. presenta  $\vec{x}$  alla rete e calcola il corrispondente output
2. **Per ogni** unità di output  $k$

$$\delta_k \leftarrow o_k(1 - o_k)(t_k - o_k)$$

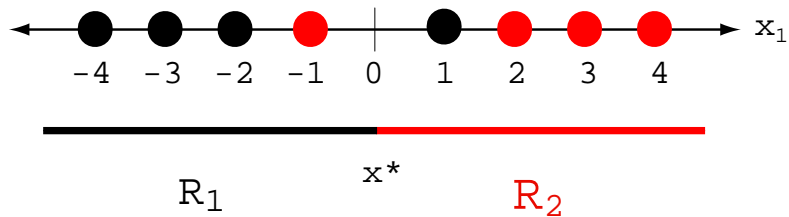
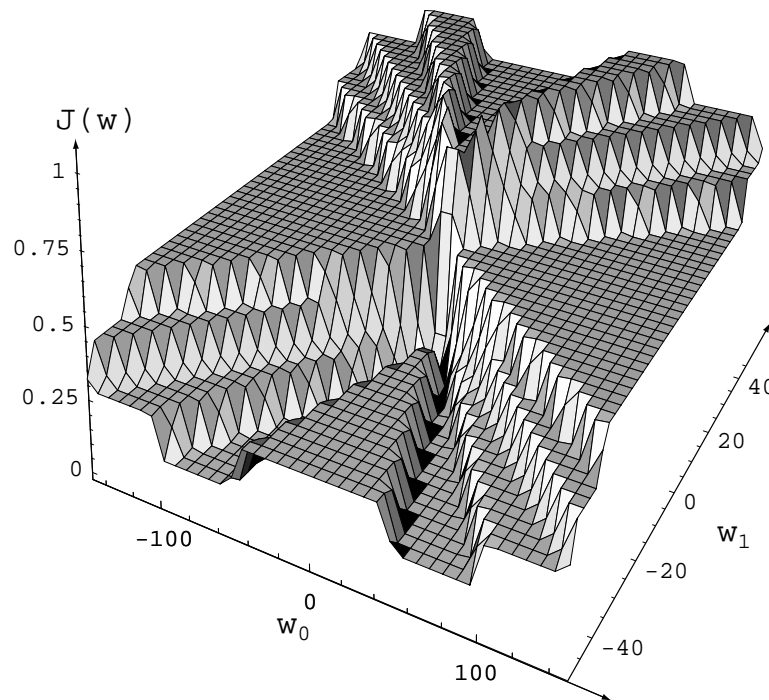
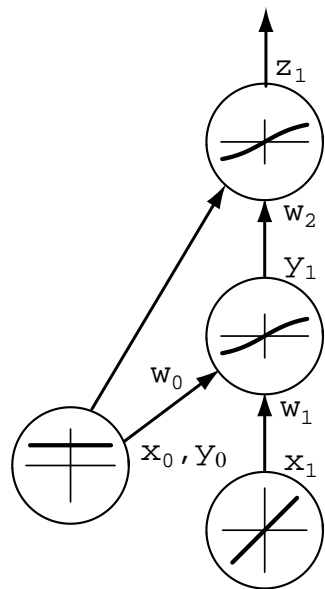
3. **Per ogni** unità nascosta  $j$

$$\delta_j \leftarrow o_j(1 - o_j) \sum_{k \in \text{outputs}} w_{k,j} \delta_k$$

4. aggiorna tutti i pesi  $w_{p,q}$  della rete

$$w_{s,q} \leftarrow w_{s,q} + \eta \Delta w_{s,q} \quad \text{dove} \quad \Delta w_{s,q} = \begin{cases} \delta_s x_q & \text{se } s \in \text{nascoste} \\ \delta_s y_q & \text{se } s \in \text{outputs} \end{cases}$$

# Esempio di Funzione Errore



## Discesa di Gradiente Batch e Stocastica

### Batch:

Fai finché condizione di terminazione non soddisfatta

1. calcola  $\nabla E_{Tr}[\vec{w}]$
2.  $\vec{w} \leftarrow \vec{w} - \eta \nabla E_{Tr}[\vec{w}]$

### Stocastica (Incrementale):

Fai finché condizione di terminazione non soddisfatta

- Per ogni esempio di apprendimento  $p$  in  $Tr$ 
  1. calcola  $\nabla E_{p \in Tr}[\vec{w}]$
  2.  $\vec{w} \leftarrow \vec{w} - \eta \nabla E_{p \in Tr}[\vec{w}]$

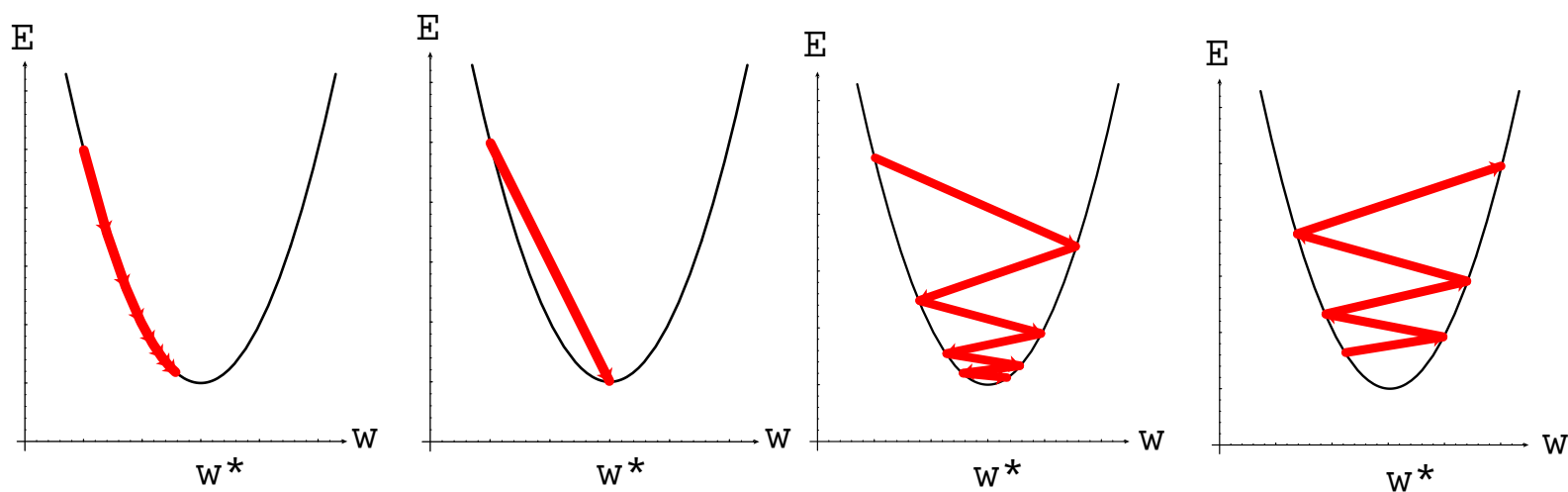
dove

$$E_{Tr}[\vec{w}] \equiv \frac{1}{2cN_{Tr}} \sum_{p \in Tr} \sum_{k=1}^c (t_k^{(p)} - z_k^{(p)})^2 \quad E_{p \in Tr}[\vec{w}] \equiv \frac{1}{2c} \sum_{k=1}^c (t_k^{(p)} - z_k^{(p)})^2$$

La discesa di gradiente *Stocastica* (gradiente istantaneo) può approssimare quella *Batch* (gradiente esatto) con precisione arbitraria se  $\eta$  è sufficientemente piccolo

## Alcuni Problemi ...

- Scelta della topologia della rete  $\rightarrow$  determina lo Spazio delle Ipotesi;
- Scelta del passo di discesa (valore di  $\eta$ ):

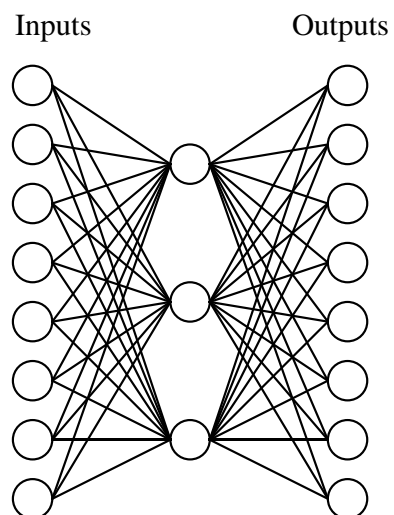


- apprendimento lento..., ma calcolo di output veloce
- **MINIMI LOCALI !!**

**Bias Induttivo: sia nella rappresentazione che nella ricerca**

## Esempio di Apprendimento per Rete Feed-forward

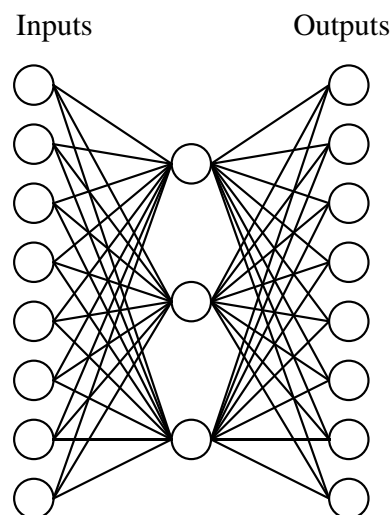
### Compressione di Dati



Input	Output
00000001	00000001
00000010	00000010
00000100	00000100
00001000	00001000
00010000	00010000
00100000	00100000
01000000	01000000
10000000	10000000

## Esempio di Apprendimento per Rete Feed-forward

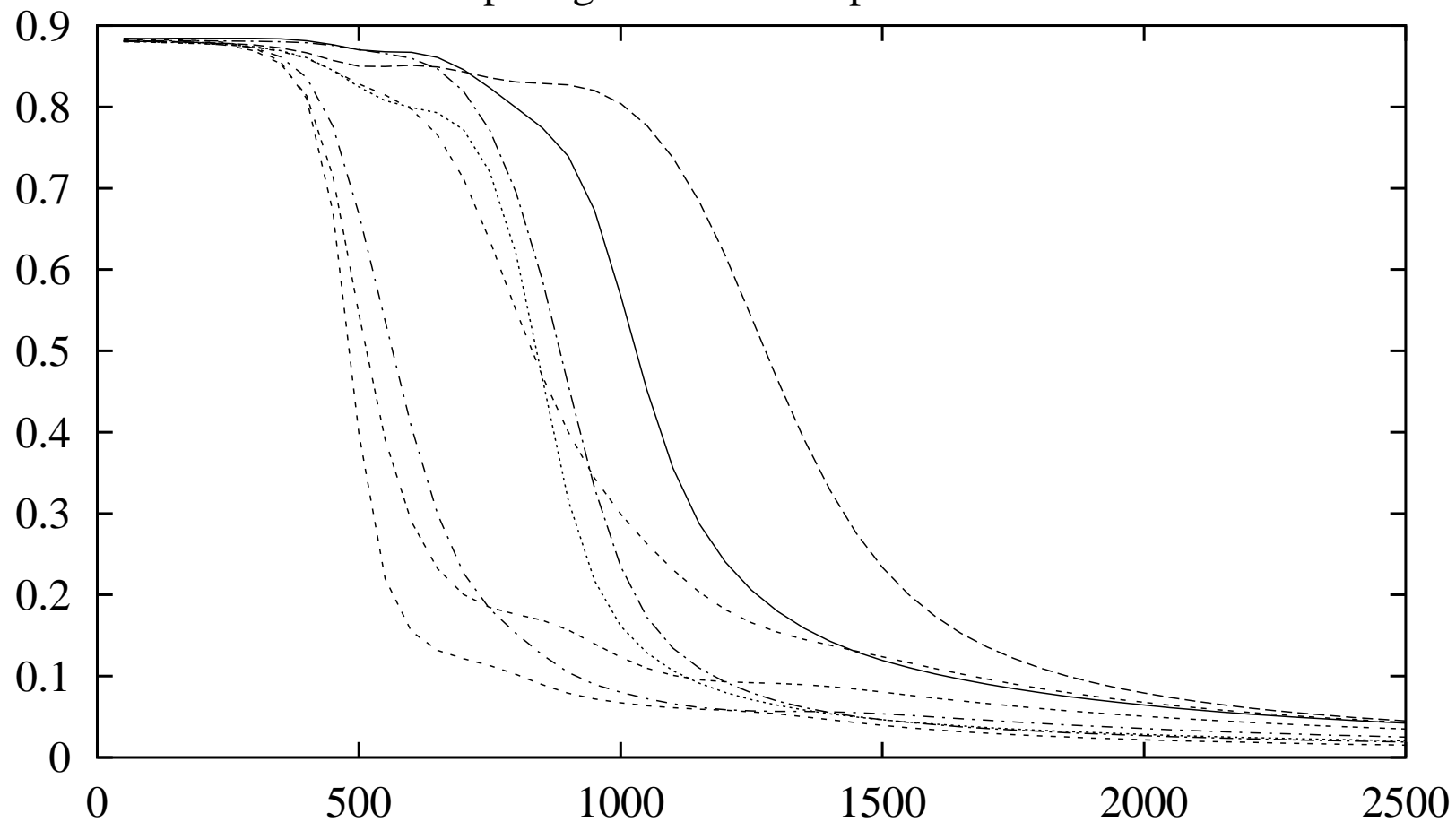
### Compressione di Dati



Input		Valori Nascosti		Output
10000000	→	0.89 0.04 0.08	→	10000000
01000000	→	0.01 0.11 0.88	→	01000000
00100000	→	0.01 0.97 0.27	→	00100000
00010000	→	0.99 0.97 0.71	→	00010000
00001000	→	0.03 0.05 0.02	→	00001000
00000100	→	0.22 0.99 0.99	→	00000100
00000010	→	0.80 0.01 0.98	→	00000010
00000001	→	0.60 0.94 0.01	→	00000001

## Curve di Apprendimento

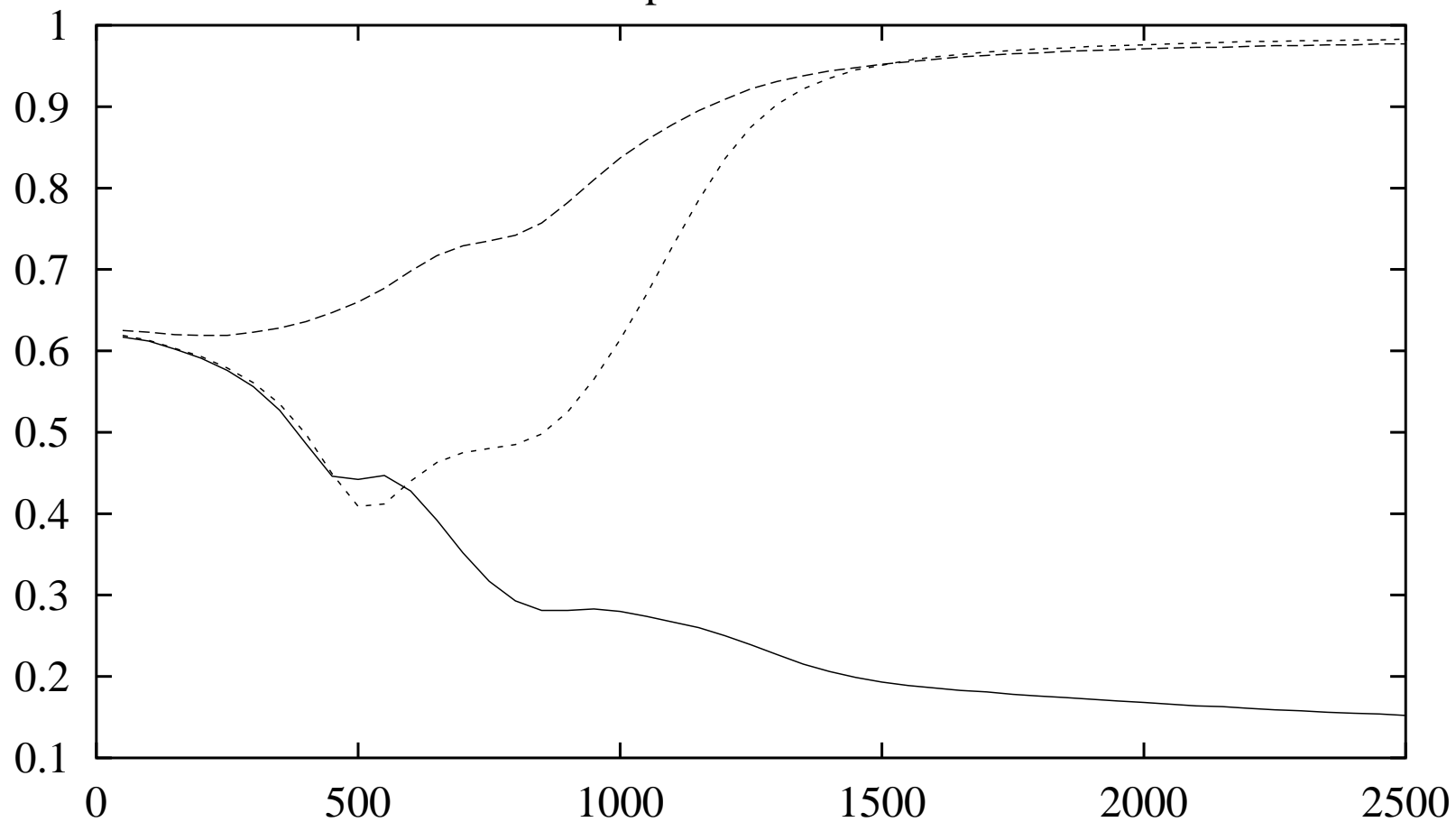
Errore per ogni unita' di output





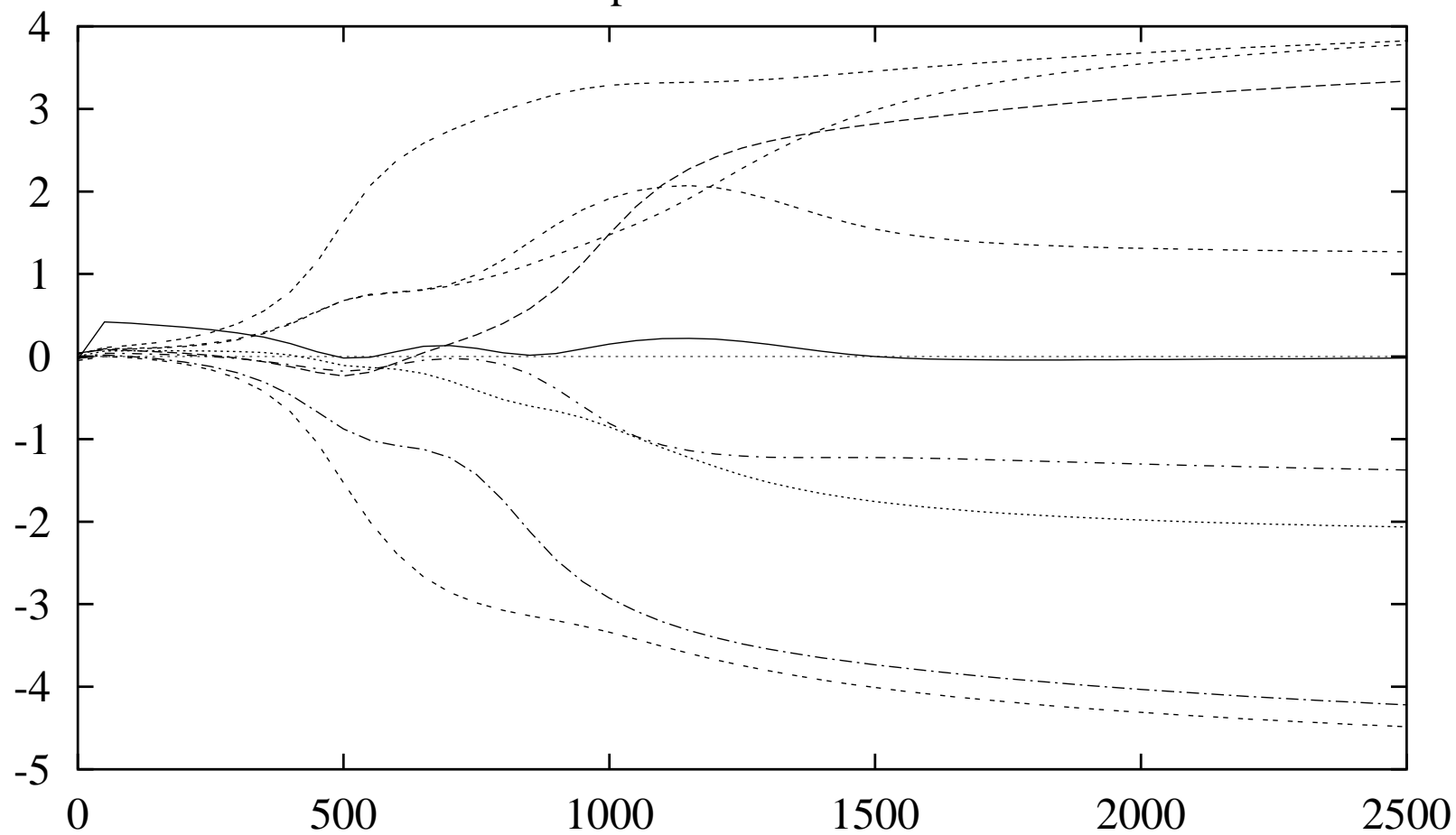
## Curve di Apprendimento

Codifica dell'input 01000000 a livello delle unita' nascoste



## Curve di Apprendimento

Pesi dall'input ad una unita' nascosta



## Potere Computazionale Reti Neurali

Il seguente teorema stabilisce l'universalità di reti feed-forward come approssimatori di funzioni continue.

**Teorema** Sia  $\varphi(\cdot)$  una funzione continua monotona crescente, limitata e noncostante. Si indichi con  $I_n$  l'ipercubo n-dimensionale  $[0, 1]^n$  e lo spazio delle funzioni continue su esso definite sia  $C(I_n)$ . Data una qualunque funzione  $f \in C(I_n)$  e  $\varepsilon > 0$ , allora esiste un intero  $M$  e insiemi di costanti reali  $\alpha_i$ ,  $\theta_i$ , e  $w_{ij}$ , dove  $i = 1, \dots, M$  e  $j = 1, \dots, n$  tale che  $f(\cdot)$  possa essere approssimata da

$$F(x_1, \dots, x_n) = \sum_{i=1}^M \alpha_i \varphi\left(\sum_{j=1}^n w_{ij} x_j - \theta_i\right) \quad (1)$$

in modo tale che

$$|F(x_1, \dots, x_n) - f(x_1, \dots, x_n)| < \varepsilon \quad (2)$$

per tutti i punti  $[x_1, \dots, x_n] \in I_n$ .

## Potere Computazionale Reti Neurali

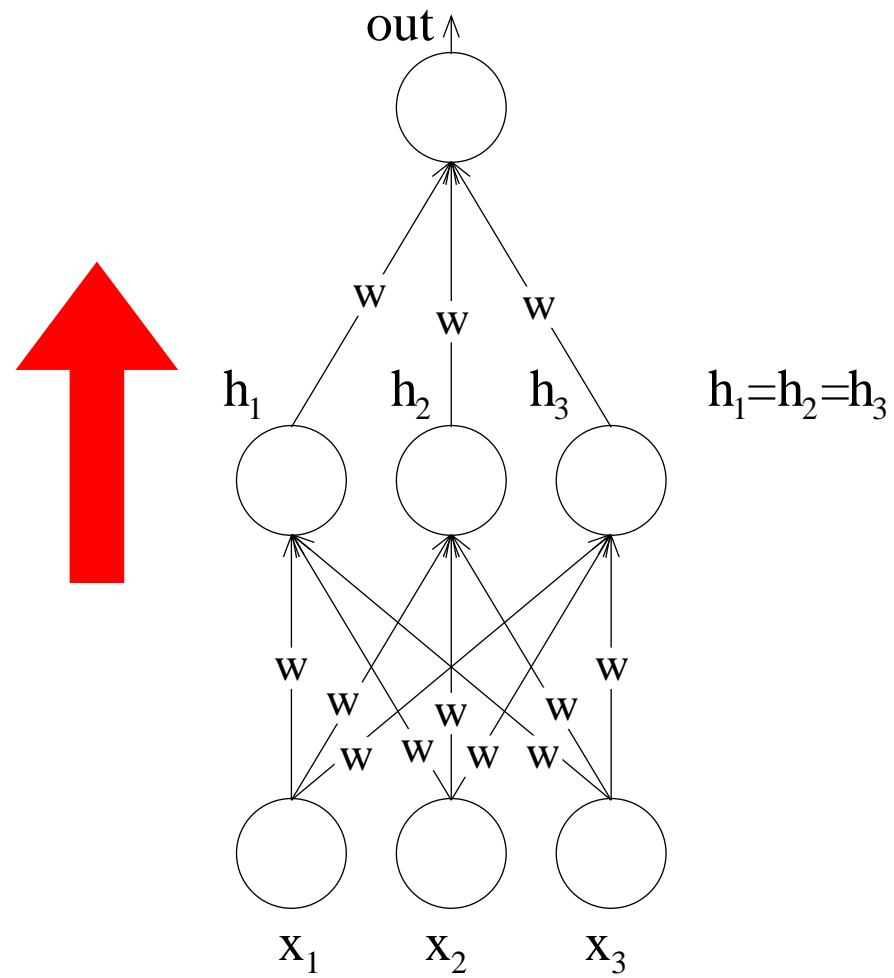
Notare che qualunque funzione sigmoideale soddisfa le condizioni imposte nel teorema su  $\varphi(\cdot)$ . Inoltre, l'equazione (1) rappresenta l'output di una rete multistrato descritta come segue

1. la rete ha  $n$  nodi di input ed un singolo strato di unità nascoste con  $M$  unità; gli input sono denotati da  $x_1, \dots, x_n$ .
2. l' $i$ -esima unità ha associati i pesi  $w_{i1}, \dots, w_{in}$  e soglia  $\theta_i$ .
3. l'output della rete è una combinazione lineare degli output delle unità nascoste, dove i coefficienti della combinazione sono dati da  $\alpha_1, \dots, \alpha_M$ .

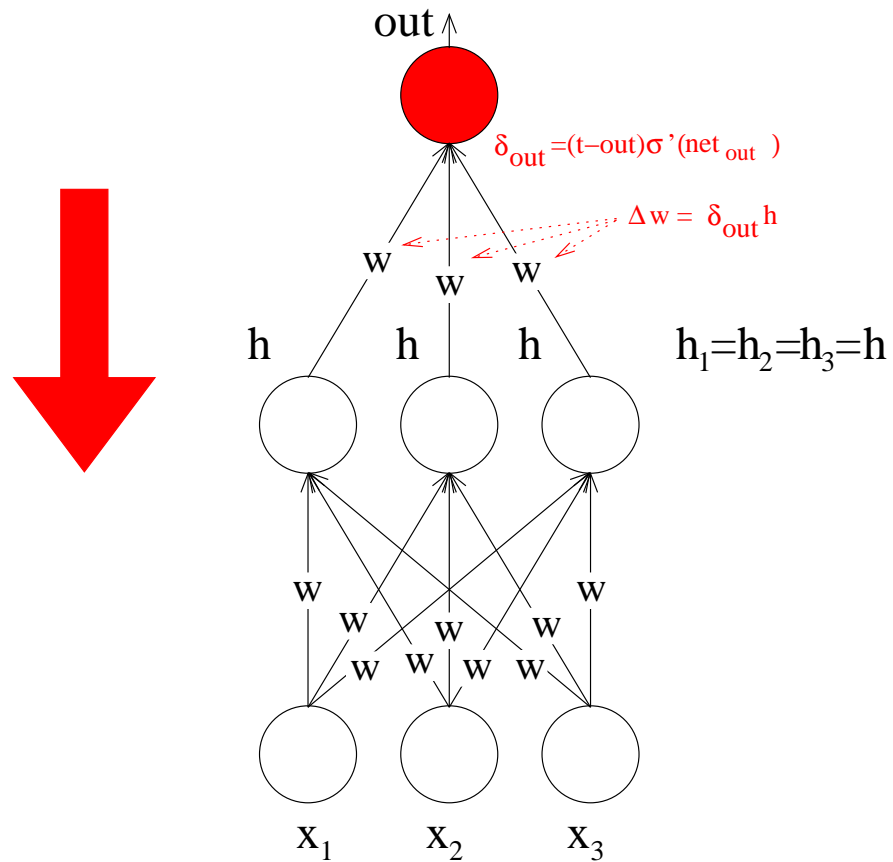
Quindi, data una tolleranza  $\varepsilon$ , una rete con un unico strato nascosto può approssimare una qualsiasi funzione in  $C(I_n)$ .

Si noti che il teorema afferma solo l'esistenza di una rete e non fornisce alcuna formula per il calcolo del numero  $M$  di unità nascoste necessarie per approssimare la funzione target con la tolleranza desiderata.

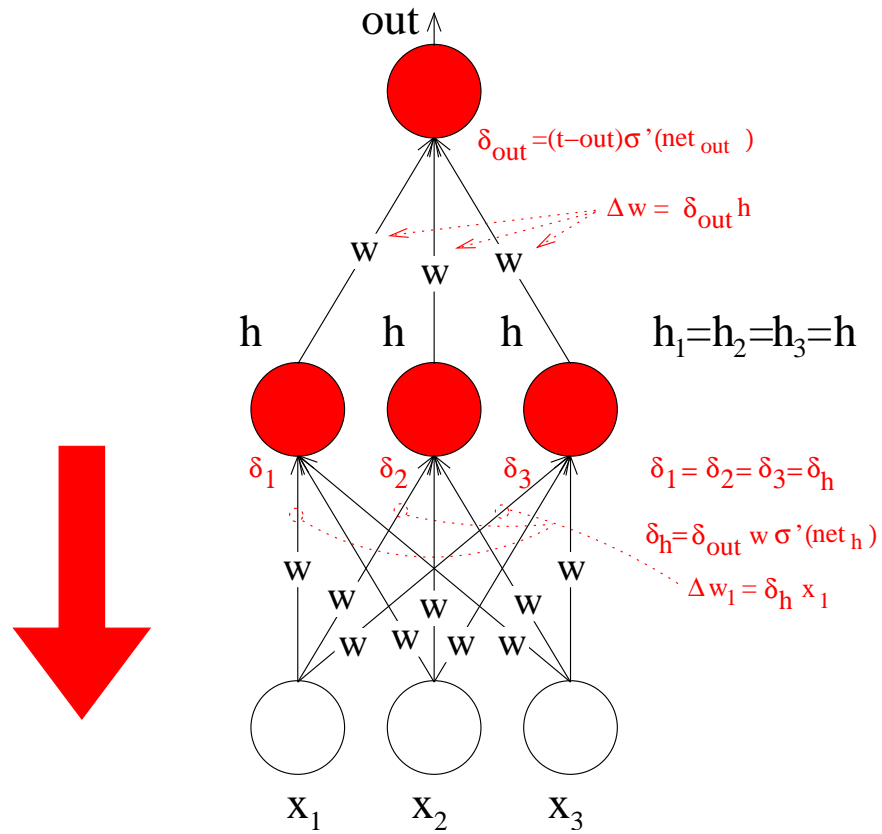
# Simmetrie



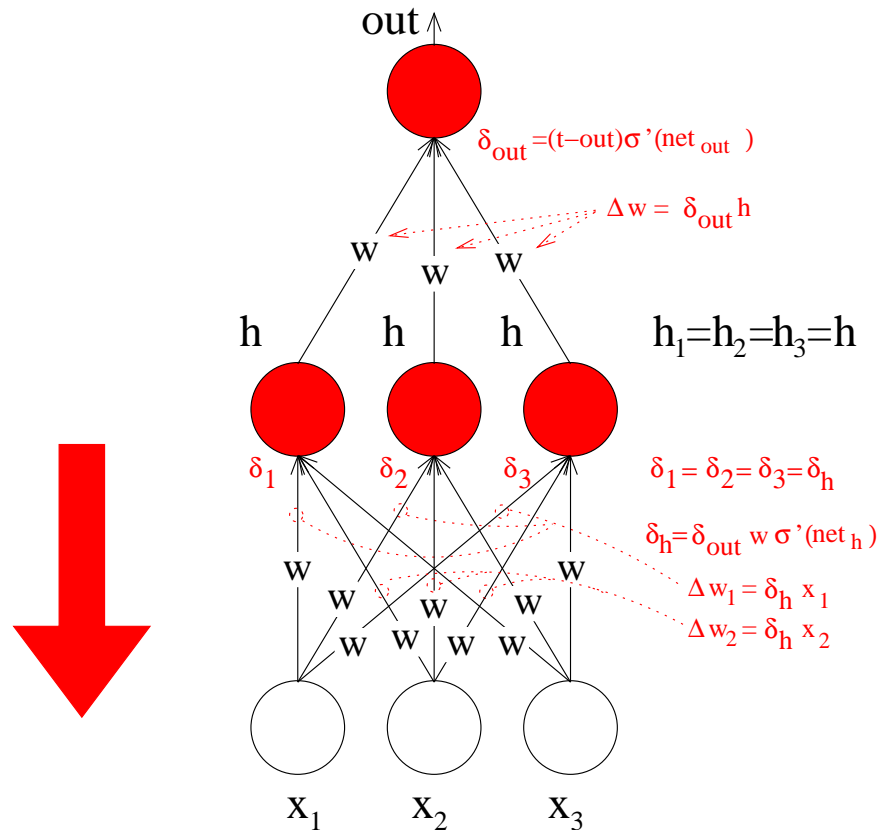
# Simmetrie



# Simmetrie

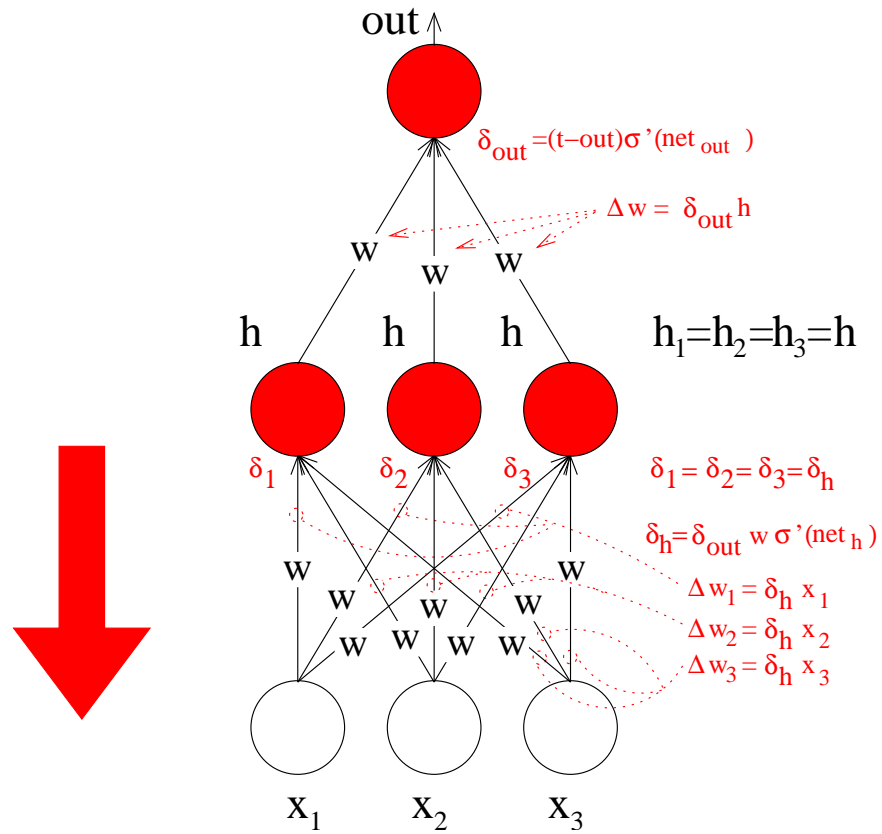


# Simmetrie





# Simmetrie



## Apprendimento Perceptron: prova di convergenza

Sia dato un insieme di esempi  $Tr = \{(\vec{x}^{(i)}, t^{(i)})\}$ , dove  $t \in \{-1, +1\}$   
(classificazione binaria)

Se  $\exists \vec{w}^*$  t.c. (lineare separabilità)

$$\forall i, t^{(i)}(\vec{w}^* \cdot \vec{x}^{(i)}) \geq \delta = \min_i t^{(i)}(\vec{w}^* \cdot \vec{x}^{(i)}) > 0$$

allora

$$\forall i, \vec{w}^* \cdot (t^{(i)} \vec{x}^{(i)}) \geq \delta$$

cioè  $\vec{w}^*$  è soluzione anche del problema di apprendimento definito dall'insieme di apprendimento  $Tr' = \{(t^{(i)} \vec{x}^{(i)}, +1)\}$  e viceversa

Quindi ci possiamo concentrare su problemi dove **tutti gli esempi sono positivi**

## Apprendimento Perceptron: prova di convergenza

Supponiamo che

- inizialmente  $\vec{w} = \vec{w}(0) = 0$  (inizializzazione)
- $\eta = \frac{1}{2}$  e  $\forall i, \|\vec{x}^{(i)}\|^2 \leq K$

Dopo aver commesso  $q$  errori (tutti falsi negativi) si avrà

$$\vec{w}(q) = \sum_{j=1}^q \vec{x}^{(i_j)}$$

infatti all' errore  $j$ -esimo il vettore dei pesi è aggiornato sommandogli l'input  $\vec{x}^{(i_j)}$  classificato erroneamente:

$$\vec{w}(j) = \vec{w}(j-1) + \vec{x}^{(i_j)}$$

## Apprendimento Perceptron: prova di convergenza

Mostriamo adesso che il modulo di  $\vec{w}(q)$  non può crescere indefinitamente (succede se non si converge ad una soluzione in un numero finito di iterazioni)

Iniziamo definendo un lower bound sul modulo di  $\vec{w}(q)$ :

$$\vec{w}^* \cdot \vec{w}(q) = \vec{w}^* \cdot \sum_{j=1}^q \vec{x}^{(i_j)} \geq q\delta \quad (\text{ricordiamo che } \delta = \min_i \vec{w}^* \cdot \vec{x}^{(i)})$$

e per la disuguaglianza di Cauchy-Swartz ( $[\vec{x} \cdot \vec{y}]^2 \leq \|\vec{x}\|^2 \|\vec{y}\|^2$ ) abbiamo

$$\|\vec{w}^*\|^2 \|\vec{w}(q)\|^2 \geq [\vec{w}^* \cdot \vec{w}(q)]^2 \geq [q\delta]^2 \Rightarrow \|\vec{w}(q)\|^2 \geq \frac{[q\delta]^2}{\|\vec{w}^*\|^2}$$

## Apprendimento Perceptron: prova di convergenza

Definiamo adesso un upper bound sul modulo di  $\vec{w}(q)$ :

$$\|\vec{w}(q)\|^2 = \|\vec{w}(q-1) + \vec{x}^{(i_q)}\|^2 = \|\vec{w}(q-1)\|^2 + 2\vec{w}(q-1) \cdot \vec{x}^{(i_q)} + \|\vec{x}^{(i_q)}\|^2$$

e poiché  $\vec{w}(q-1) \cdot \vec{x}^{(i_q)} < 0$  ( $q$ -esimo errore)

$$\|\vec{w}(q)\|^2 \leq \|\vec{w}(q-1)\|^2 + \|\vec{x}^{(i_q)}\|^2$$

Applicando ricorsivamente questa disuguaglianza su tutti gli errori abbiamo

$$\|\vec{w}(q)\|^2 \leq \sum_{i=1}^q \|\vec{x}^{(i_q)}\|^2 \leq qK$$

## Apprendimento Perceptron: prova di convergenza

Mettendo insieme il lower bound con l'upper bound otteniamo:

$$\frac{[q\delta]^2}{\|\vec{w}^*\|^2} \leq \|\vec{w}(q)\|^2 \leq qK$$

Pertanto il numero massimo di errori  $q_{max}$  che si possono commettere mantenendo i due vincoli soddisfatti è ottenuto quando vale l'uguaglianza dei bound:

$$\frac{[q_{max}\delta]^2}{\|\vec{w}^*\|^2} = q_{max}K$$

da cui si ottiene

$$q_{max} = \frac{\|\vec{w}^*\|^2 K}{\delta^2}$$

Quindi, in caso di lineare separabilità, l'algoritmo di apprendimento del Perceptron commette un numero finito di errori, al più  $\frac{\|\vec{w}^*\|^2 K}{\delta^2}$