Apprendimento di Reti di Perceptron

Abbiamo visto che un singolo Perceptron non riesce ad apprendere tutte le funzioni booleane (es. XOR)

Però una rete di Perceptron può implementare una qualunque funzione booleana (tramite AND, OR, NOT).

Problema: come effettuare l'apprendimento di una rete di Perceptron ?

Non si sa come assegnare "credito" o "colpe" alle unità nascoste:

PROBLEMA DELL'ASSEGNAMENTO DEL CREDITO

Una possibile soluzione è quella di rendere il singolo neurone derivabile e sfruttare la tecnica di Discesa del Gradiente per apprendere i pesi "giusti".

Vediamo, quindi, come la Discesa del Gradiente si applica ad un Perceptron "semplificato".

Discesa di Gradiente

Consideriamo un Perceptron SENZA la hard-threshold:

$$out(\vec{x}) = \sum_{i=0}^{n} w_i x_i = \vec{w} \cdot \vec{x}$$

e definiamo una misura dell'errore commesso da un particolare vettore dei pesi:

Funzione Errore: $E[\vec{w}] = \frac{1}{2N_{Tr}} \sum_{(\vec{x}^{(i)}, t^{(i)}) \in Tr} \left(t^{(i)} - out(\vec{x}^{(i)}) \right)^2$

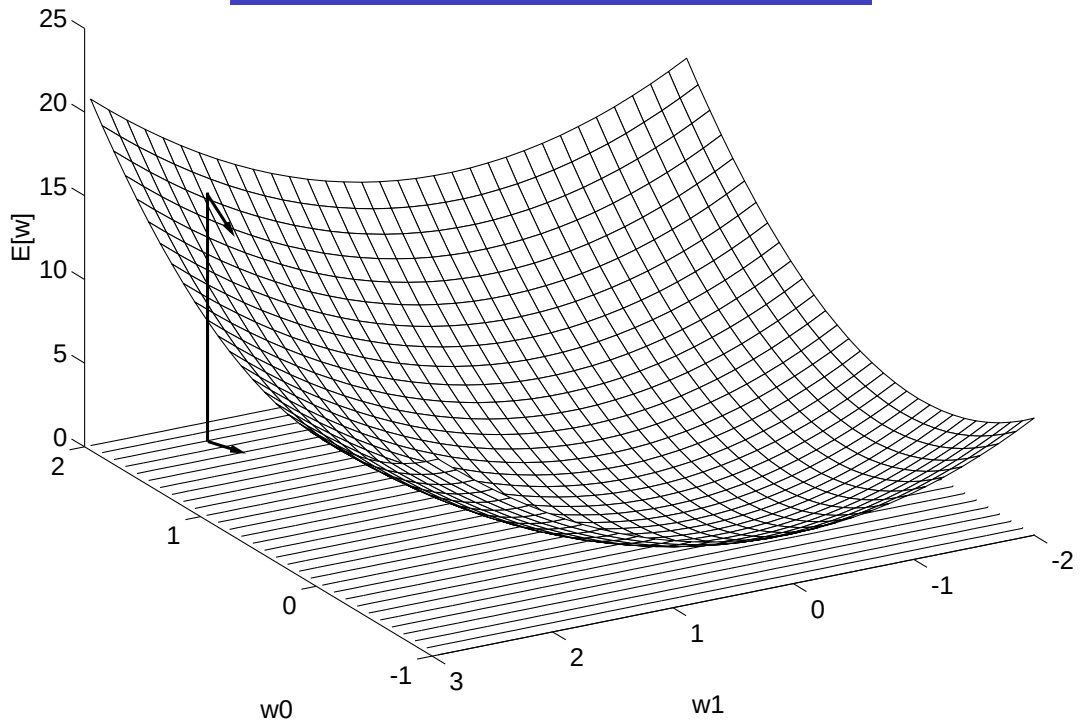
dove N_{Tr} è la cardinalità dell'insieme di apprendimento Tr.

La funzione errore di sopra misura lo scarto quadratico medio (diviso 2) del valore target da quello predetto dal neurone (out).

Ovviamente, se $\forall (\vec{x}^{(i)}, t^{(i)}) \in Tr$ si ha $out(\vec{x}^{(i)}) = t^{(i)} \rightarrow \ E[\vec{w}] = 0$

Quindi bisogna MINIMIZZARE $E[\vec{w}]$ rispetto a $\vec{w}$

Discesa di Gradiente



Idea base: partire da un $\vec{w}$ random e modificarlo nella direzione contraria al gradiente (che indica la direzione di crescita di $E[\vec{w}]$)

$$\underbrace{\nabla E[\vec{w}]}_{\text{gradiente}} \equiv \left[\frac{\partial E}{\partial w_0}, \frac{\partial E}{\partial w_1}, \ldots, \frac{\partial E}{\partial w_n}\right], \quad \Delta\vec{w} = -\eta \nabla E[\vec{w}], \quad \Delta w_i = -\eta \frac{\partial E}{\partial w_i}$$

Calcolo del gradiente

$$\frac{\partial E}{\partial w_i} = \frac{\partial}{\partial w_i} \frac{1}{2N_{Tr}} \sum_{d \in Tr} (t^{(d)} - out^{(d)})^2$$

Calcolo del gradiente

$$\frac{\partial E}{\partial w_i} = \frac{\partial}{\partial w_i}\frac{1}{2N_{Tr}}\sum_{d\in Tr}(t^{(d)} - out^{(d)})^2$$

$$= \frac{1}{2N_{Tr}}\sum_{d\in Tr}\frac{\partial}{\partial w_i}(t^{(d)} - out^{(d)})^2$$

Calcolo del gradiente

$$\begin{aligned}
\frac{\partial E}{\partial w_i} &= \frac{\partial}{\partial w_i}\frac{1}{2N_{Tr}}\sum_{d\in Tr}(t^{(d)}-out^{(d)})^2 \\
&= \frac{1}{2N_{Tr}}\sum_{d\in Tr}\frac{\partial}{\partial w_i}(t^{(d)}-out^{(d)})^2 \\
&= \frac{1}{2N_{Tr}}\sum_{d\in Tr}2(t^{(d)}-out^{(d)})\frac{\partial}{\partial w_i}(t^{(d)}-out^{(d)})
\end{aligned}$$

Calcolo del gradiente

$$
\begin{aligned}
\frac{\partial E}{\partial w_i} &= \frac{\partial}{\partial w_i}\frac{1}{2N_{Tr}}\sum_{d\in Tr}(t^{(d)}-out^{(d)})^2 \\
&= \frac{1}{2N_{Tr}}\sum_{d\in Tr}\frac{\partial}{\partial w_i}(t^{(d)}-out^{(d)})^2 \\
&= \frac{1}{2N_{Tr}}\sum_{d\in Tr}2(t^{(d)}-out^{(d)})\frac{\partial}{\partial w_i}(t^{(d)}-out^{(d)}) \\
&= \frac{1}{N_{Tr}}\sum_{d\in Tr}(t^{(d)}-out^{(d)})\frac{\partial}{\partial w_i}(t^{(d)}-\vec{w}\cdot\vec{x}^{(d)})
\end{aligned}
$$

Calcolo del gradiente

$$
\begin{aligned}
\frac{\partial E}{\partial w_i} &= \frac{\partial}{\partial w_i}\frac{1}{2N_{Tr}}\sum_{d\in Tr}(t^{(d)}-out^{(d)})^2 \\
&= \frac{1}{2N_{Tr}}\sum_{d\in Tr}\frac{\partial}{\partial w_i}(t^{(d)}-out^{(d)})^2 \\
&= \frac{1}{2N_{Tr}}\sum_{d\in Tr}2(t^{(d)}-out^{(d)})\frac{\partial}{\partial w_i}(t^{(d)}-out^{(d)}) \\
&= \frac{1}{N_{Tr}}\sum_{d\in Tr}(t^{(d)}-out^{(d)})\frac{\partial}{\partial w_i}(t^{(d)}-\vec{w}\cdot\vec{x}^{(d)}) \\
\frac{\partial E}{\partial w_i} &= \frac{1}{N_{Tr}}\sum_{d\in Tr}(t^{(d)}-out^{(d)})(-x_i^{(d)}) \\
&= -\frac{1}{N_{Tr}}\sum_{d\in Tr}(t^{(d)}-out^{(d)})(x_i^{(d)})
\end{aligned}
$$

Discesa di Gradiente

Gradient-Descent(Tr, η)

ogni esempio di apprendimento è una coppia $(\vec{x}, t)$*, dove* $\vec{x}$ *è il vettore di valori in input, e* t *è il valore desiderato in output (target).* η *è il coefficiente di apprendimento (che ingloba* $\frac{1}{N_{Tr}}$*).*

- Assegna a w_i valori piccoli random
- Finché la condizione di terminazione non è verificata, fai
  - $\Delta w_i \leftarrow 0$
  - Per ogni $(\vec{x}, t)$ in Tr, fai
    - Presenta $\vec{x}$ al neurone e calcola l'output out
    - Per ogni w_i, fai

$$\Delta w_i \leftarrow \Delta w_i + \eta(t - out)x_i$$

  - Per ogni w_i, fai

$$w_i \leftarrow w_i + \Delta w_i$$

Discesa di Gradiente con Sigmoide

Consideriamo un Perceptron con funzione sigmoidale:

$$out(\vec{x}) = \sigma(\sum_{i=0}^{n} w_i x_i) = \sigma(\vec{w} \cdot \vec{x})$$

dove si ricorda che $\sigma(net) = \frac{1}{1+e^{-net}}$

Si noti che per $\sigma()$ vale la seguente relazione

$$\sigma'(net) = \frac{d\,\sigma(net)}{d\,net} = \sigma(net)(1 - \sigma(net))$$

e ricordiamo che (derivata di funzioni composte)

$$\frac{d\,f(g(x))}{d\,x} = \frac{d\,f(g(x))}{d\,g(x)} \frac{d\,g(x)}{d\,x}$$

Calcolo del gradiente con sigmoide

$$\begin{aligned}
\frac{\partial E}{\partial w_i} &= \frac{\partial}{\partial w_i}\frac{1}{2N_{Tr}}\sum_{d\in Tr}(t^{(d)}-out^{(d)})^2 \\
&= \frac{1}{2N_{Tr}}\sum_{d\in Tr}\frac{\partial}{\partial w_i}(t^{(d)}-out^{(d)})^2 \\
&= \frac{1}{2N_{Tr}}\sum_{d\in Tr}2(t^{(d)}-out^{(d)})\frac{\partial}{\partial w_i}(t^{(d)}-out^{(d)})
\end{aligned}$$

Calcolo del gradiente con sigmoide

$$
\begin{aligned}
\frac{\partial E}{\partial w_i} &= \frac{\partial}{\partial w_i}\frac{1}{2N_{Tr}}\sum_{d\in Tr}(t^{(d)}-out^{(d)})^2 \\
&= \frac{1}{2N_{Tr}}\sum_{d\in Tr}\frac{\partial}{\partial w_i}(t^{(d)}-out^{(d)})^2 \\
&= \frac{1}{2N_{Tr}}\sum_{d\in Tr}2(t^{(d)}-out^{(d)})\frac{\partial}{\partial w_i}(t^{(d)}-out^{(d)}) \\
&= \frac{1}{N_{Tr}}\sum_{d\in Tr}(t^{(d)}-out^{(d)})\frac{\partial}{\partial w_i}(t^{(d)}-\sigma(\vec{w}\cdot\vec{x}^{(d)}))
\end{aligned}
$$

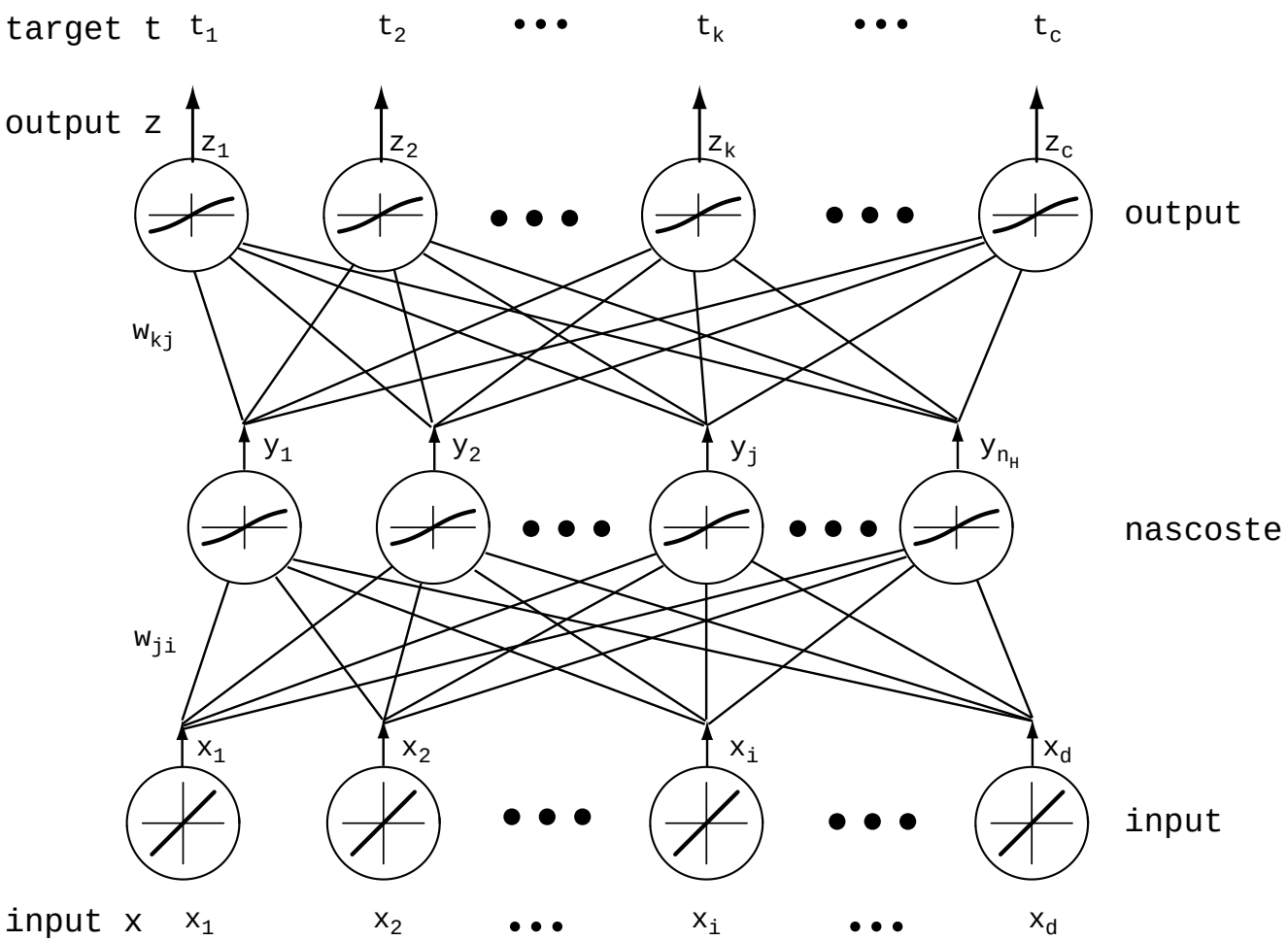
Calcolo del gradiente con sigmoide

$$
\begin{aligned}
\frac{\partial E}{\partial w_i} &= \frac{\partial}{\partial w_i}\frac{1}{2N_{Tr}}\sum_{d\in Tr}(t^{(d)}-out^{(d)})^2 \\
&= \frac{1}{2N_{Tr}}\sum_{d\in Tr}\frac{\partial}{\partial w_i}(t^{(d)}-out^{(d)})^2 \\
&= \frac{1}{2N_{Tr}}\sum_{d\in Tr}2(t^{(d)}-out^{(d)})\frac{\partial}{\partial w_i}(t^{(d)}-out^{(d)}) \\
&= \frac{1}{N_{Tr}}\sum_{d\in Tr}(t^{(d)}-out^{(d)})\frac{\partial}{\partial w_i}(t^{(d)}-\sigma(\vec{w}\cdot\vec{x}^{(d)})) \\
\frac{\partial E}{\partial w_i} &= \frac{1}{N_{Tr}}\sum_{d\in Tr}(t^{(d)}-out^{(d)})(-\frac{\partial\sigma(\vec{w}\cdot\vec{x}^{(d)})}{\partial\vec{w}\cdot\vec{x}^{(d)}}\frac{\partial\vec{w}\cdot\vec{x}^{(d)}}{\partial w_i})
\end{aligned}
$$

Calcolo del gradiente con sigmoide

$$
\begin{aligned}
\frac{\partial E}{\partial w_i} &= \frac{\partial}{\partial w_i}\frac{1}{2N_{Tr}}\sum_{d\in Tr}(t^{(d)}-out^{(d)})^2 \\
&= \frac{1}{2N_{Tr}}\sum_{d\in Tr}\frac{\partial}{\partial w_i}(t^{(d)}-out^{(d)})^2 \\
&= \frac{1}{2N_{Tr}}\sum_{d\in Tr}2(t^{(d)}-out^{(d)})\frac{\partial}{\partial w_i}(t^{(d)}-out^{(d)}) \\
&= \frac{1}{N_{Tr}}\sum_{d\in Tr}(t^{(d)}-out^{(d)})\frac{\partial}{\partial w_i}(t^{(d)}-\sigma(\vec{w}\cdot\vec{x}^{(d)})) \\
\frac{\partial E}{\partial w_i} &= \frac{1}{N_{Tr}}\sum_{d\in Tr}(t^{(d)}-out^{(d)})(-\frac{\partial\sigma(\vec{w}\cdot\vec{x}^{(d)})}{\partial\vec{w}\cdot\vec{x}^{(d)}}\frac{\partial\vec{w}\cdot\vec{x}^{(d)}}{\partial w_i}) \\
&= -\frac{1}{N_{Tr}}\sum_{d\in Tr}(t^{(d)}-out^{(d)})\sigma(\vec{w}\cdot\vec{x}^{(d)})(1-\sigma(\vec{w}\cdot\vec{x}^{(d)}))x_i^{(d)}
\end{aligned}
$$

Reti Neurali Feed-forward: notazione



Reti Neurali Feed-forward: notazione

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

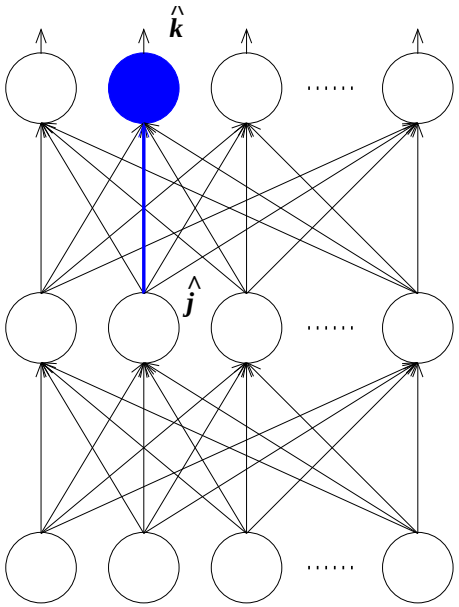
La funzione errore, considerando che si hanno c unità di output, diventa

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

Calcolo del gradiente per i pesi di una unità di output

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

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

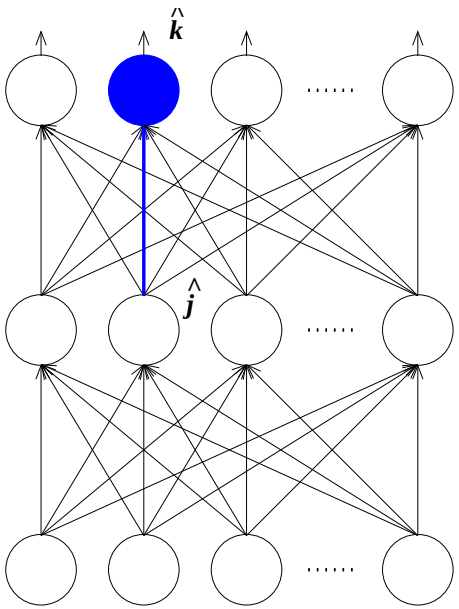


Calcolo del gradiente per i pesi di una unità di output

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

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

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



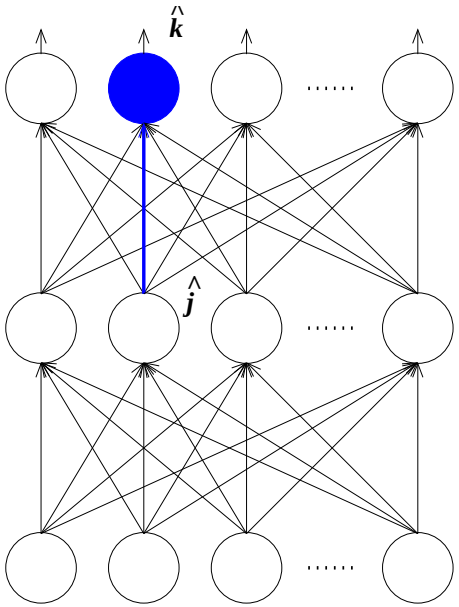
Calcolo del gradiente per i pesi di una unità di output

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

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

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

$$= \frac{1}{2cN_{Tr}} \sum_{p \in Tr} 2(t_{\hat{k}}^{(p)} - z_{\hat{k}}^{(p)}) \frac{\partial}{\partial w_{\hat{k}\hat{j}}} (t_{\hat{k}}^{(p)} - z_{\hat{k}}^{(p)})$$



Calcolo del gradiente per i pesi di una unità di output

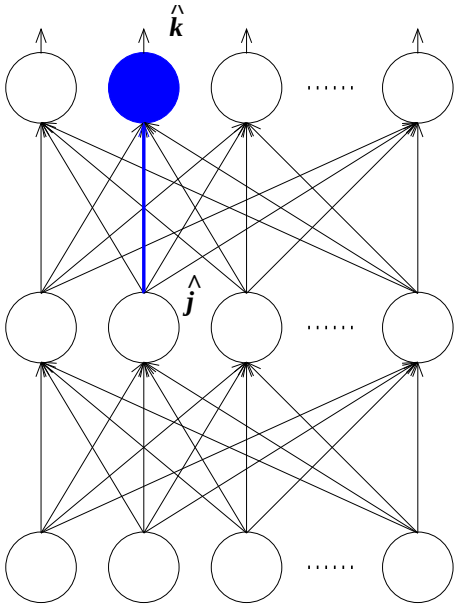
Fissiamo gli indici $\hat{k}$ e $\hat{j}$:

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

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

$$= \frac{1}{2cN_{Tr}} \sum_{p \in Tr} 2(t_{\hat{k}}^{(p)} - z_{\hat{k}}^{(p)}) \frac{\partial}{\partial w_{\hat{k}\hat{j}}} (t_{\hat{k}}^{(p)} - z_{\hat{k}}^{(p)})$$

$$= \frac{1}{cN_{Tr}} \sum_{p \in Tr} (t_{\hat{k}}^{(p)} - z_{\hat{k}}^{(p)}) \frac{\partial}{\partial w_{\hat{k}\hat{j}}} (t_{\hat{k}}^{(p)} - \sigma(\vec{w}_{\hat{k}} \cdot \vec{y}^{(p)}))$$



Calcolo del gradiente per i pesi di una unità di output

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

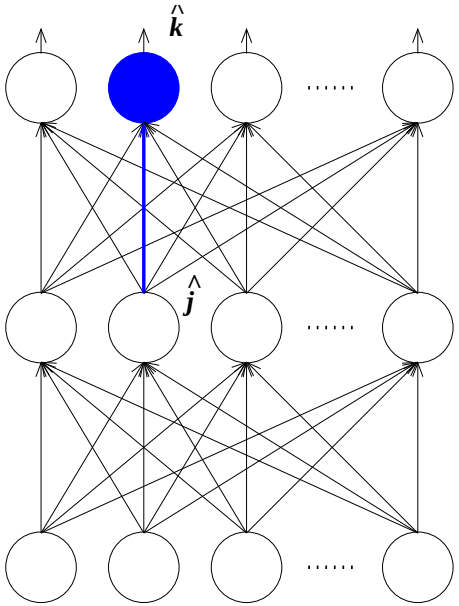
$$\frac{\partial E}{\partial w_{\hat{k}\hat{j}}} = \frac{\partial}{\partial w_{\hat{k}\hat{j}}} \frac{1}{2cN_{Tr}} \sum_{p \in Tr} \sum_{k=1}^{c} (t_k^{(p)} - z_k^{(p)})^2$$

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

$$= \frac{1}{2cN_{Tr}} \sum_{p \in Tr} 2(t_{\hat{k}}^{(p)} - z_{\hat{k}}^{(p)}) \frac{\partial}{\partial w_{\hat{k}\hat{j}}} (t_{\hat{k}}^{(p)} - z_{\hat{k}}^{(p)})$$

$$= \frac{1}{cN_{Tr}} \sum_{p \in Tr} (t_{\hat{k}}^{(p)} - z_{\hat{k}}^{(p)}) \frac{\partial}{\partial w_{\hat{k}\hat{j}}} (t_{\hat{k}}^{(p)} - \sigma(\vec{w}_{\hat{k}} \cdot \vec{y}^{(p)}))$$

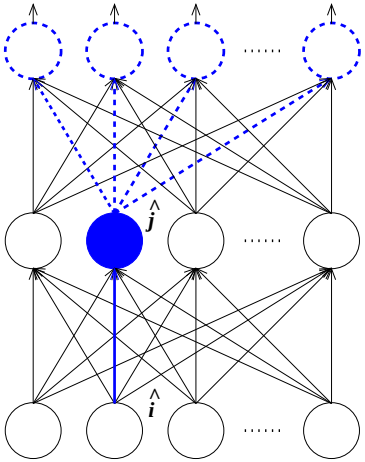
$$= -\frac{1}{cN_{Tr}} \sum_{p \in Tr} (t_{\hat{k}}^{(p)} - z_{\hat{k}}^{(p)}) \sigma'(\vec{w}_{\hat{k}} \cdot \vec{y}^{(p)}) y_{\hat{j}}^{(p)}$$



Calcolo del gradiente per i pesi di una unità nascosta

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

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

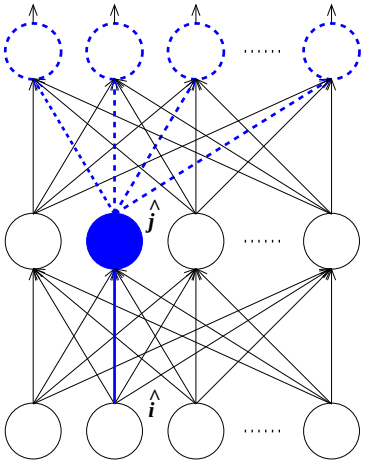


Calcolo del gradiente per i pesi di una unità nascosta

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

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

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



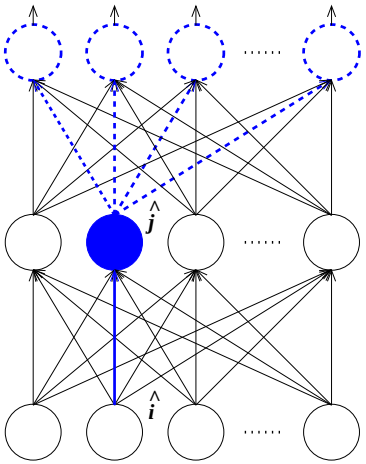
Calcolo del gradiente per i pesi di una unità nascosta

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

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

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

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



Calcolo del gradiente per i pesi di una unità nascosta

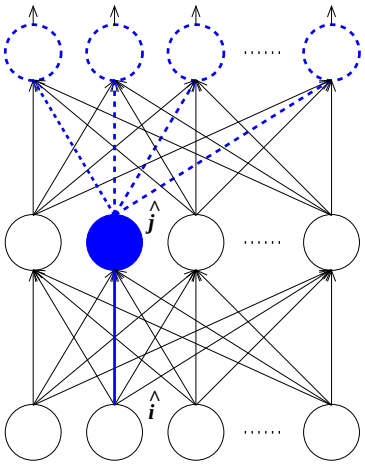
Fissiamo gli indici $\hat{j}$ e $\hat{i}$:

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

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

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

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

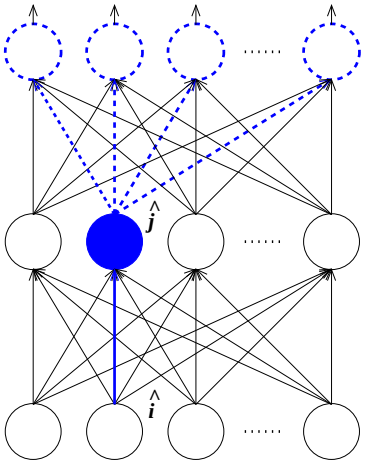


Calcolo del gradiente per i pesi di una unità nascosta

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

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

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



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

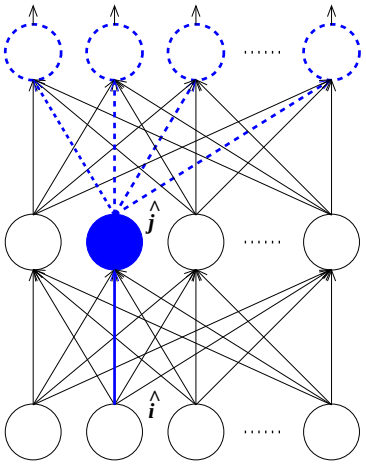
$$= -\frac{1}{cN_{Tr}} \sum_{p \in Tr} \sum_{k=1}^{c} (t_k^{(p)} - z_k^{(p)}) \sigma'(\vec{w}_k \cdot \vec{y}^{(p)}) \frac{\partial}{\partial w_{\hat{j}\hat{i}}} \sum_{j=1}^{N_H} w_{kj} y_j^{(p)}$$

Calcolo del gradiente per i pesi di una unità nascosta

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

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

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



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

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

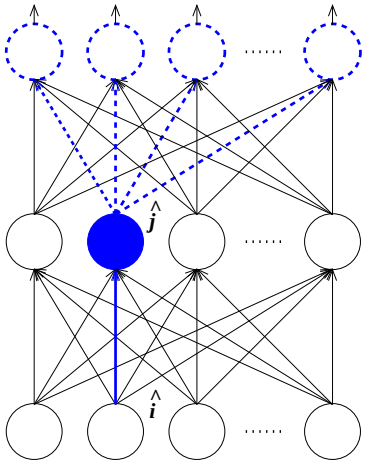
$$= -\frac{1}{cN_{Tr}} \sum_{p \in Tr} \sum_{k=1}^{c} (t_k^{(p)} - z_k^{(p)}) \sigma'(\vec{w}_k \cdot \vec{y}^{(p)}) w_{k\hat{j}} \frac{\partial}{\partial w_{\hat{j}\hat{i}}} y_{\hat{j}}^{(p)}$$

Calcolo del gradiente per i pesi di una unità nascosta

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

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

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



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

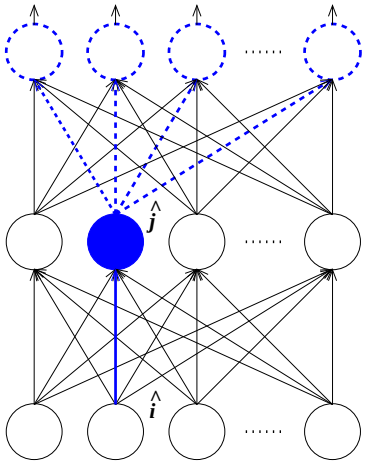
$$= -\frac{1}{cN_{Tr}} \sum_{p \in Tr} \sum_{k=1}^{c} (t_k^{(p)} - z_k^{(p)}) \sigma'(\vec{w}_k \cdot \vec{y}^{(p)}) w_{k\hat{j}} \sigma'(\vec{w}_{\hat{j}} \cdot \vec{x}^{(p)}) \frac{\partial}{\partial w_{\hat{j}\hat{i}}} (\vec{w}_{\hat{j}} \cdot \vec{x}^{(p)})$$

Calcolo del gradiente per i pesi di una unità nascosta

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

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

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

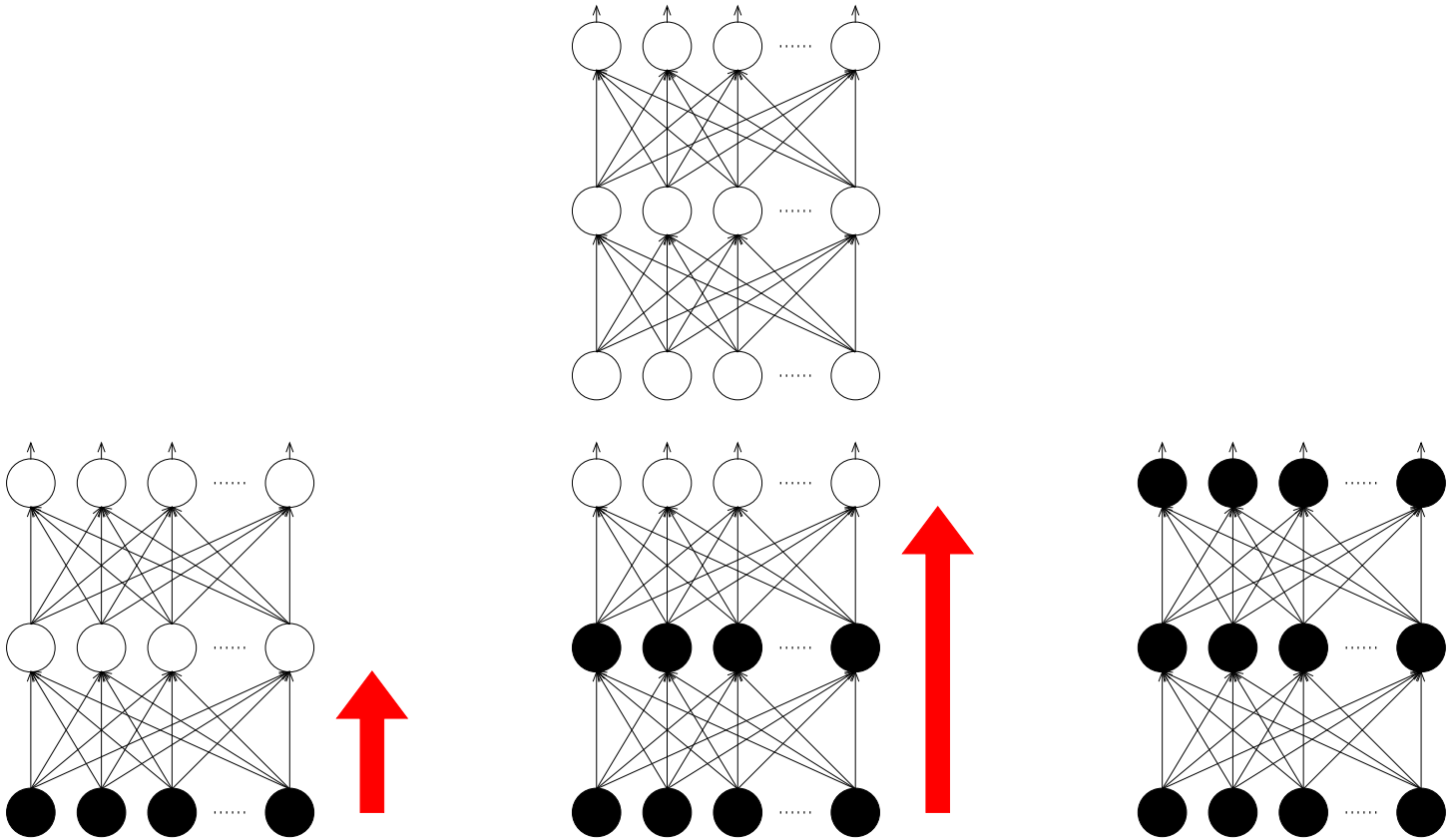


$$= -\frac{1}{cN_{Tr}} \sum_{p \in Tr} \sum_{k=1}^{c} (t_k^{(p)} - z_k^{(p)}) \sigma'(\vec{w}_k \cdot \vec{y}^{(p)}) w_{k\hat{j}} \frac{\partial}{\partial w_{\hat{j}\hat{i}}} y_{\hat{j}}^{(p)}$$

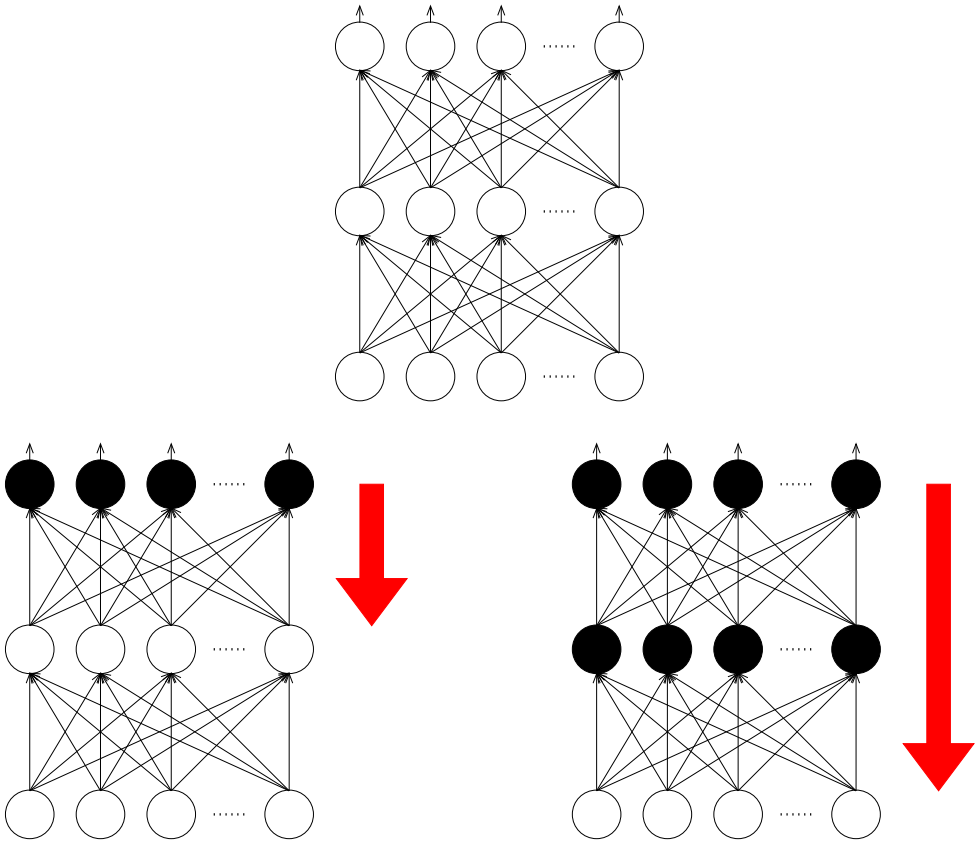
$$= -\frac{1}{cN_{Tr}} \sum_{p \in Tr} \sum_{k=1}^{c} (t_k^{(p)} - z_k^{(p)}) \sigma'(\vec{w}_k \cdot \vec{y}^{(p)}) w_{k\hat{j}} \sigma'(\vec{w}_{\hat{j}} \cdot \vec{x}^{(p)}) \frac{\partial}{\partial w_{\hat{j}\hat{i}}} (\vec{w}_{\hat{j}} \cdot \vec{x}^{(p)})$$

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

Reti Neurali Feed-forward: Fase Forward



Reti Neurali Feed-forward: Fase Backward



Algoritmo Back-Propagation (uno strato nascosto, stocastico)

Back-Propagation-1hl-stocastico(Tr, η, *topologia rete*)

- Inizializza tutti i pesi a valori random piccoli
- Finché la condizione di terminazione non è verifi cata, 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

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

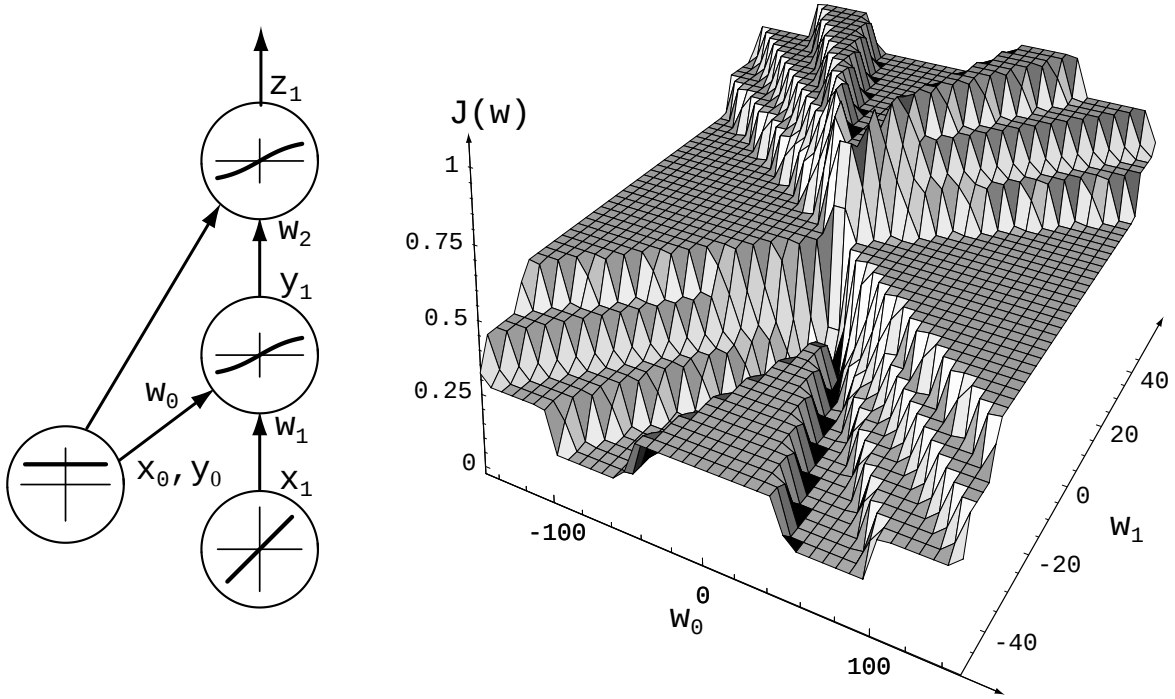
    3. Per ogni unità nascosta j

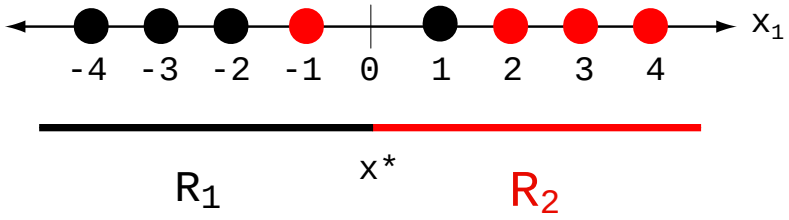
$$\delta_j \leftarrow o_j(1 - o_j) \sum_{k \in outputs} w_{k,j}\delta_k$$

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

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

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. $\vec{w} \leftarrow \vec{w} - \eta \nabla E_{p \in Tr}[\vec{w}]$

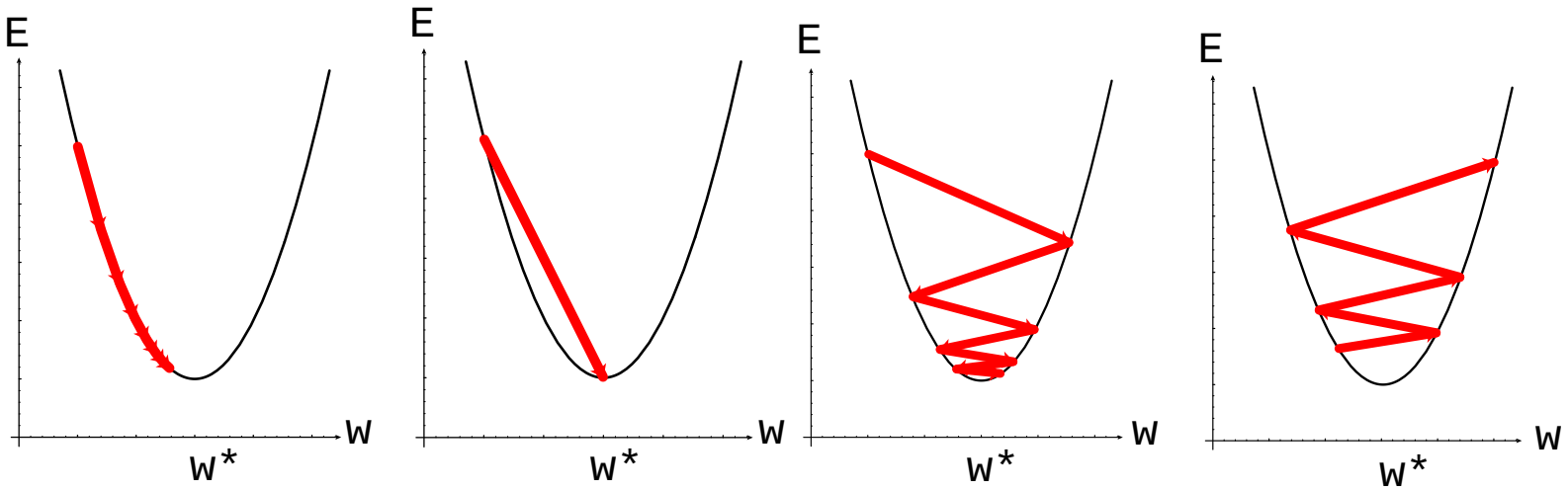
dove

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

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

Alcuni Problemi ...

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

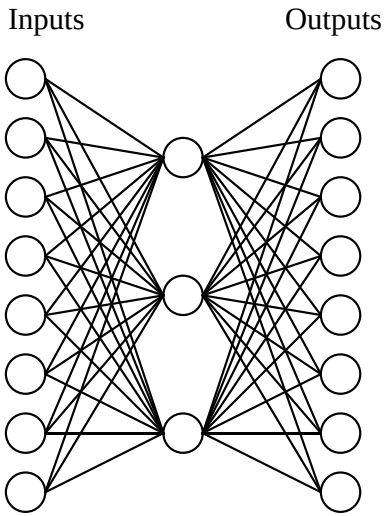


- apprendimento lento..., ma calcolo di output veloce
- MINIMI LOCALI !!

Bias Induttivo: sia nella rappresentazione che nella ricerca

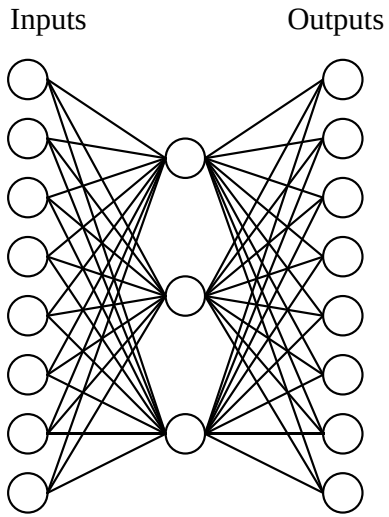
Esempio di Apprendimento per Rete Feed-forward

Compressione di Dati



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

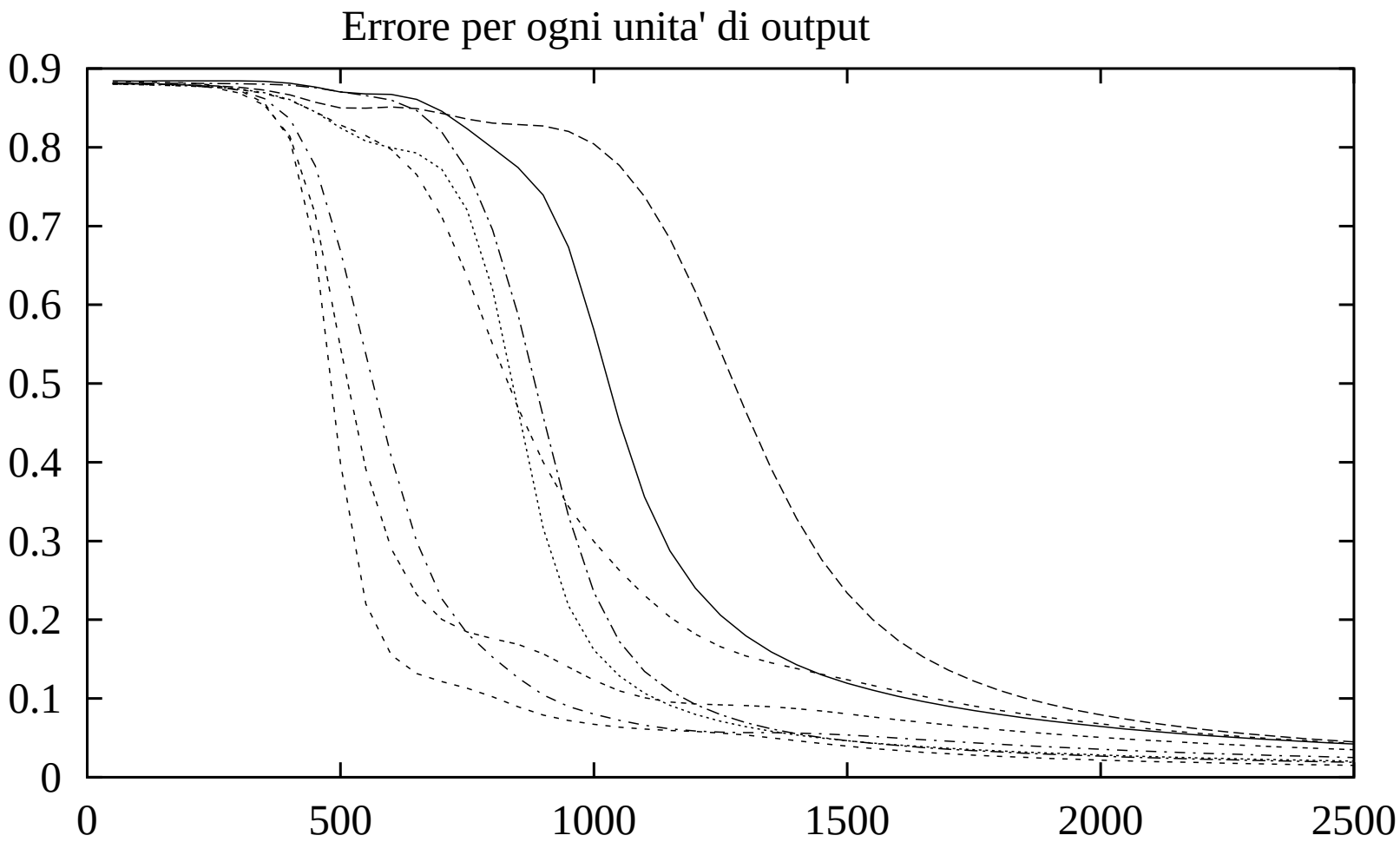
Esempio di Apprendimento per Rete Feed-forward



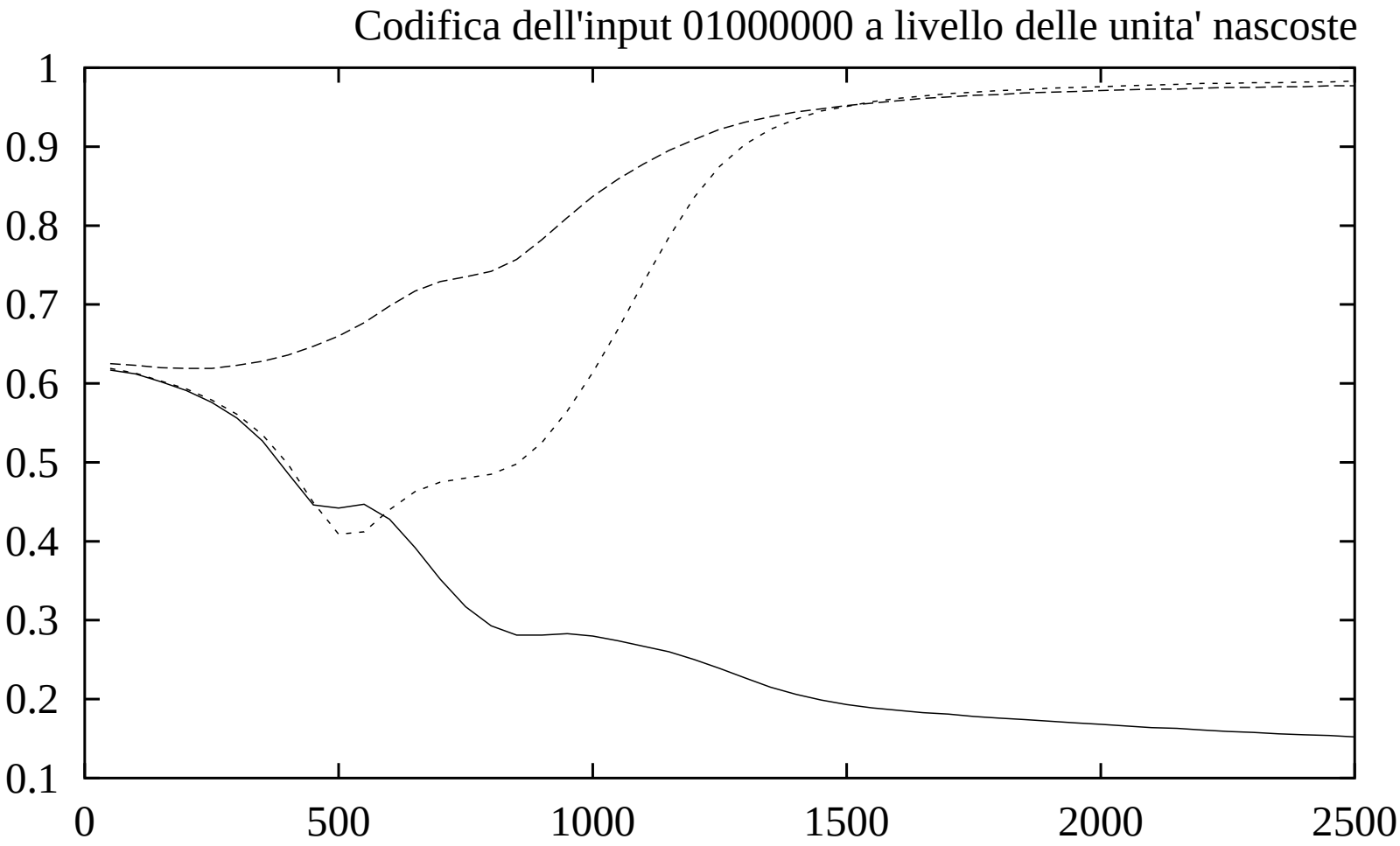
Compressione di Dati

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

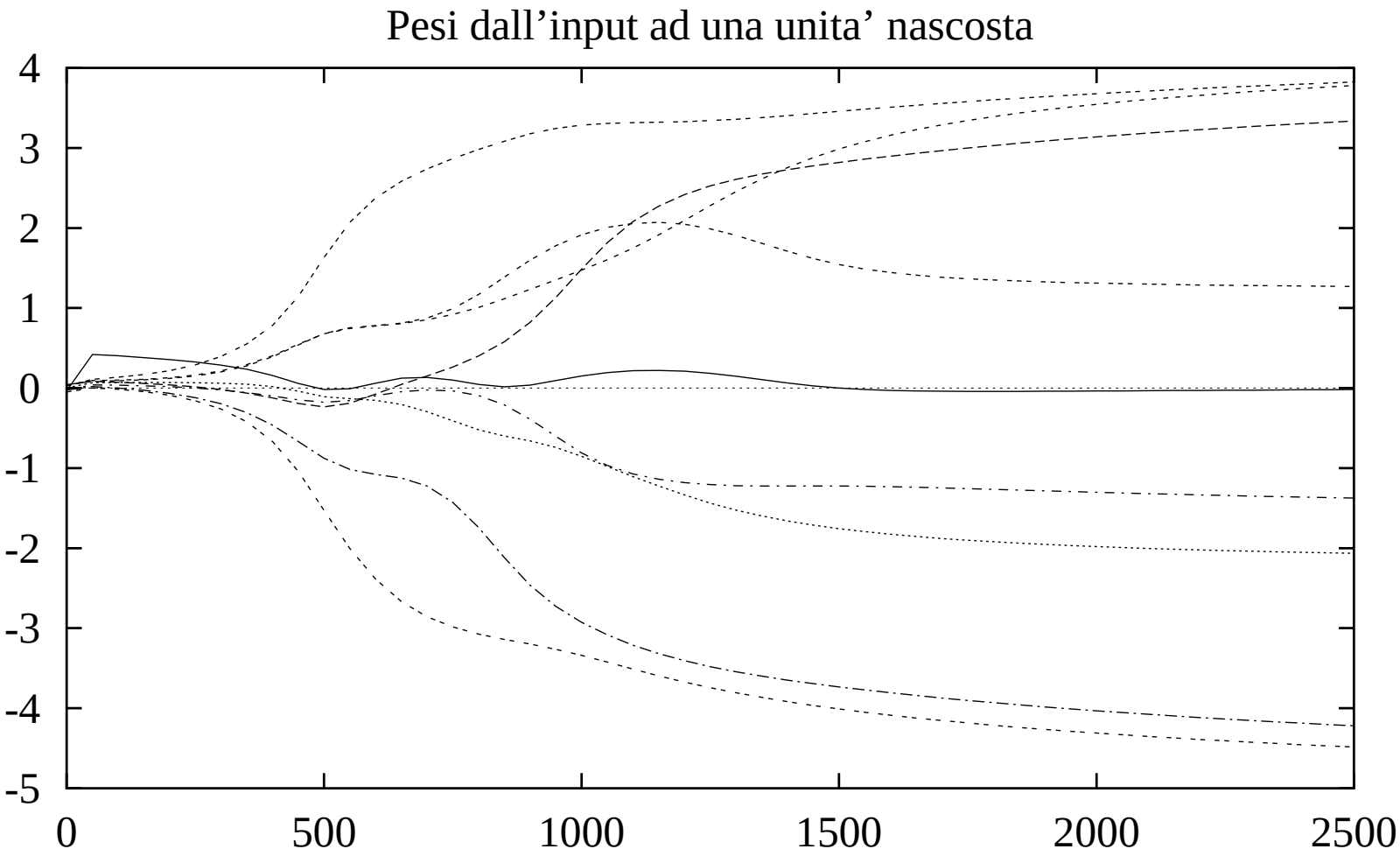
Curve di Apprendimento



Curve di Apprendimento



Curve di Apprendimento



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 α_i, θ_i, e w_{ij}, dove $i = 1, \ldots, M$ e $j = 1, \ldots, n$ tale che $f(\cdot)$ possa essere approssimata da

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

in modo tale che

$$|F(x_1, \ldots, x_n) - f(x_1, \ldots, x_n)| < \varepsilon \tag{2}$$

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

Potere Computazionale Reti Neurali

Notare che qualunque funzione sigmoidale 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 $x_1, \ldots, x_n$.
2. l'i-esima unità ha associati i pesi $w_{i1}, \ldots, w_{in}$ e soglia θ_i.
3. l'output della rete è una combinazione lineare degli output delle unità nascoste, dove i coefficienti della combinazione sono dati da $\alpha_1, \ldots, \alpha_M$.

Quindi, data una tolleranza ε, 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.