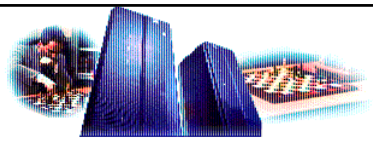
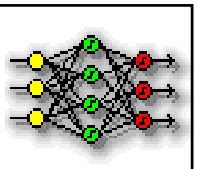
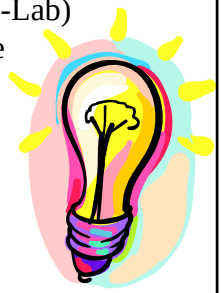




Le reti neurali

Alberto Borghese

Università degli Studi di Milano
Laboratory of Applied Intelligent Systems (AIS-Lab)
Dipartimento di Scienze dell'Informazione
borghese@dsi.unimi.it



Sommario

Modelli semplici di neuroni

Il problema dell'XOR

L'apprendimento in reti di perceptroni con funzioni di attivazione lineari.


Brains cause minds (J. Searle)


Le reti neurali

Se il neurone biologico consente l'intelligenza, perché non dovrebbe consentire l'intelligenza artificiale un neurone sintetico?

".. a neural network is a system composed of ***many simple processing elements*** operating in ***parallel*** whose function is determined by ***network structure***, ***connection strengths***, and the ***processing performed at computing elements*** or nodes. … Neural network architectures are inspired by the architecture of biological nervous systems, which use many simple processing elements operating in parallel to obtain high computation rates". (DARPA, 1988)….

A cosa servono?

Le reti neurali offrono i seguenti specifici vantaggi nell'elaborazione dell'informazione:

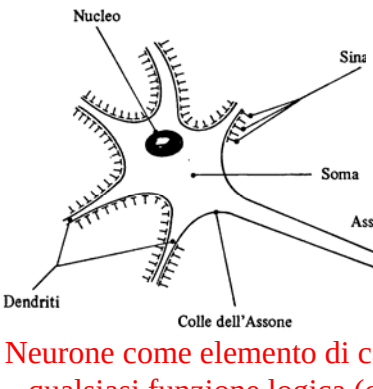
- Apprendimento basato su esempi (non è richiesta l'elaborazione di un modello aderente alla realtà)
- Autoorganizzazione dell'informazione nella rete
- Robustezza ai guasti (codifica ridondante dell'informazione)
- Funzionamento in tempo reale (realizzazione HW)
- Basso consumo (0.5nW ÷ 4nW per neurone, 20W per il SN).

Cosa sono le reti neurali artificiali?

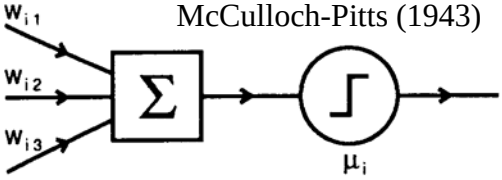
- Le reti neurali sono algoritmi non lineari per l'approssimazione di soluzioni di problemi dei quali non esiste un modello preciso (o se esiste è troppo oneroso computazionalmente), mediante l'utilizzo di esempi (dati e uscite) oppure per classificazioni. Connessioni con il dominio della statistica.
- Sono un capitolo importante negli argomenti di intelligenza artificiale.
- Da un altro punto di vista possono essere utilizzate per lo studio delle reti neurali naturali, ovvero dei processi cognitivi.

Il neurone artificiale

- *Potenziale di azione (tutto o nulla).*
- *Integrazione nel soma.*
- *Soglia di attivazione.*



McCulloch-Pitts (1943)



$$y_i(t+1) = \Theta(w_{ij}u_j(t)-\mu_i)$$

$$\Theta(x)= \begin{cases} 1 & se\ x \geq 0 \\ 0 & altrimenti \end{cases}$$

$-1 < x < +1$

Neurone come elemento di calcolo universale: in grado di calcolare qualsiasi funzione logica (cioè implementabile in un computer).

Costituenti delle reti neurali

Un neurone artificiale è costituito da:

- Un insieme di input (provenienti da altri neuroni)
- Un peso che rappresenta l'efficacia ed il segno della sinapsi.
- Una funzione di attivazione che trasforma gli input nell'output del neurone.

Una rete neurale è costituita da:

- Un insieme di neuroni artificiali.
- La connettività tra neuroni.

Critica al modello di McCulloch-Pitts

• I neuroni reali non possono essere ridotti ad un dipositivo a soglia. Lo spike ha la sua forma continua che ha una durata di qualche millisecondo.

• Il tempo di propagazione lungo i dendriti non viene considerato.

• La variazione delle forma d'onda del potenziale di membrana lungo il dendrita non viene considerata.

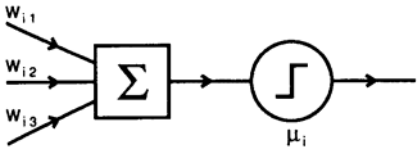
• Gli input non sono sincroni.

• Le interazioni tra input non sono lineari.

• I pesi sono supposti costanti.

Funzione di attivazione del neurone di McCullochPitts

$$y_i(t+1) = \Theta\left(\sum_{j=1}\left(w_{ij}u_j(t) - \mu_i\right)\right) \qquad y_i(t+1) = \mathrm{sgn}\left(\sum_{j=1}\left(w_{ij}u_j(t) - \mu_i\right)\right)$$



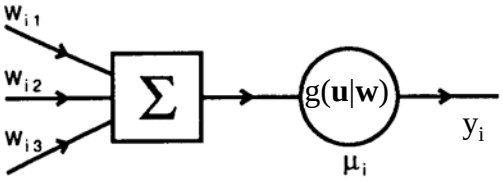
$$y_i(t+1) = \mathrm{sgn}\left(\sum_{j=0}\left(w_{ij}u_j(t)\right)\right)$$

$w_{i0} = -\mu_i$

$u_o \equiv 1$

$$y(t+1) = \mathrm{sgn}(\mathbf{w} \bullet \mathbf{u}(t))$$

Il perceptrone (Rosemblatt, 1962)

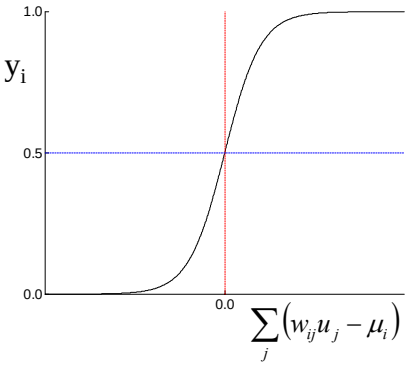


Uscita: singolo spike o frequenza di scarica.

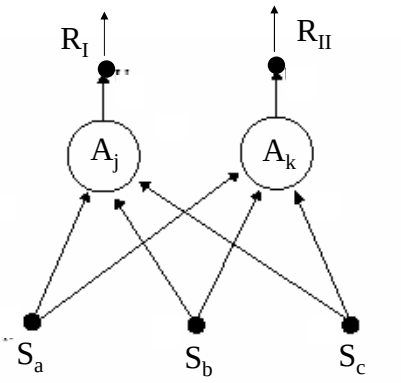
Neurone asincrono.
- Soglia μ_i -> traslazione.
- Pesi $\{w_{ij}\}$ -> pendenza.

$$y_i = g\left(\sum_j \left(w_{ij}u_j - \mu_i\right)\right)$$

Funzione logistica $g(.) = \dfrac{1}{1+e^{-\left(\sum_{j=0}^{N} w_{ij}u_j - \mu_i\right)}}$



Le reti di perceptroni



Apprendimento è la modifica dei parametri in funzione dei parametri di input/output.

$$y_i = g\left(\sum_j \left(w_{ij}u_j - \mu_i\right)\right)$$

Questa rete con neuroni a soglia, ($g(.) \equiv \Theta(.)$), non riesce ad apprendere però funzioni non linearmente separabili quali l'XOR (Minski & Papert, 1968).

Sommario

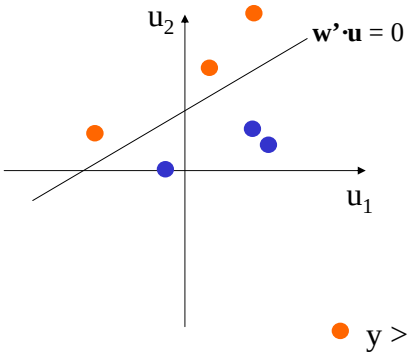
Modelli semplici di neuroni

Il problema dell'XOR

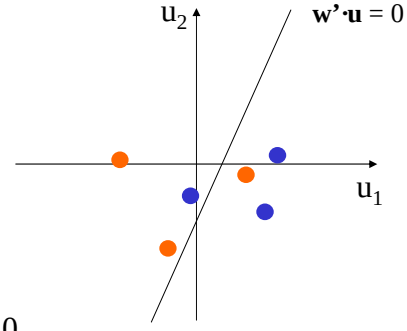
L'apprendimento in reti di perceptroni con funzioni di attivazione lineari.

Funzioni linearmente separabili

Linearmente separabile



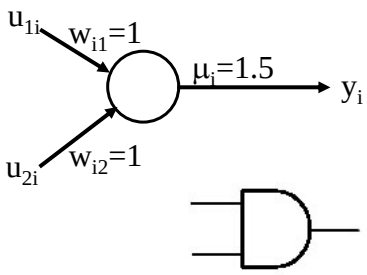
Non linearmente separabile



y > 0
y < 0

$y = f(u_1,u_2)$

Esempio - AND



u_1	u_2	y
-1	-1	-1
-1	1	-1
1	-1	-1
1	1	1

$$y_i(t+1) = \mathrm{sgn}\left(\sum_{j=0}^{2}\left(w_{ij}u_j(t)\right)\right)$$

Iperpiano di separazione (u_0=1, $w_o = -\mu_0$):

L'equazione generale della retta di separazione è:

$w_0u_0 + w_1u_1 + w_2u_2 = 0$ ovverosia: $u_2 + (w_1/w_2)u_1 + w_0/w_2 = 0$

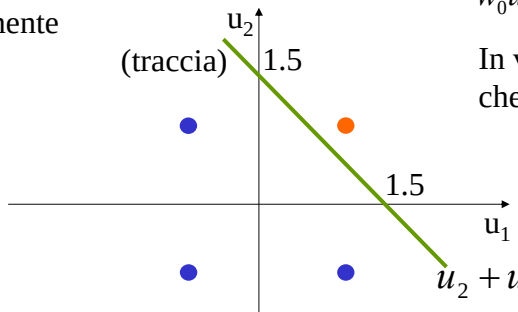
$y - mx - q = 0$ $\qquad w_1/w_2 = -m \quad w_0/w_2 = -q$

Esempio - AND (grafica)

Troviamo la soluzione graficamente

$w_0u_0 + w_1u_1 + w_2u_2 = 0$

In verde la retta **w'·u** = 0 che taglia il piano u_1u_2.



$u_2 + u_1 - 1.5 = 0$

$u_2 + (w_1/w_2)u_1 + w_0/w_2 = 0$

● y(u_1, u_2, 1)=1

● y(u_1, u_2, 1)=-1

$u_2 + u_1 - 1.5 = 0$

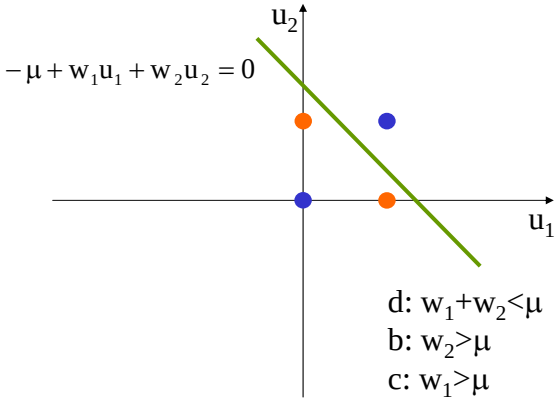
⇓

Esistono più soluzioni

$w_1/w_2 = 1 \quad w_0/w_2 = -1.5 \quad \Rightarrow \quad w_2 = k \quad w_1 = k \quad w_0 = -1.5 * k$

Esempio - XOR

$w_0 u_0 + w_1 u_1 + w_2 u_2 = 0$ $\mathbf{w'} \cdot \mathbf{u} = 0$ $u_0 = 1$



u_1	u_2	y	
0	0	-1	a
0	1	1	b
1	0	1	c
1	1	-1	d

d: $w_1+w_2<\mu$
b: $w_2>\mu$
c: $w_1>\mu$
a: $\mu > 0$

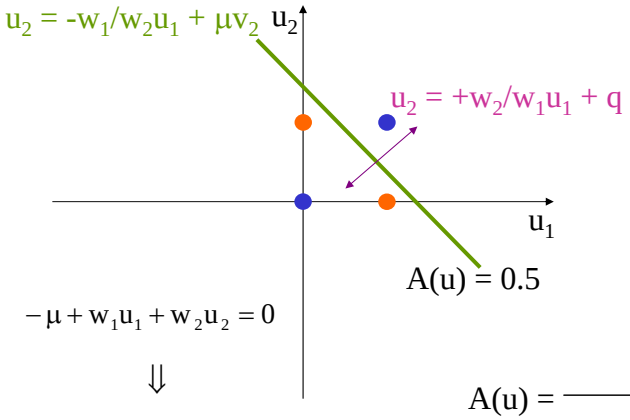
Il sistema di 4 equazioni non è risolubile.

● $y(u_1, u_2, 1)=1$
● $y(u_1, u_2, 1)=-1$

$w_1, w_2 > \mu$ e $w_1+w_2 < \mu$ Impossibile!!

Esempio – XOR – funzione di attivazione logistica

$u_2 = -w_1/w_2 u_1 + \mu v_2$



$-\mu + w_1 u_1 + w_2 u_2 = 0$

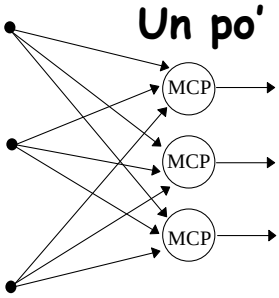
$\Downarrow$

A(u) = 0.5

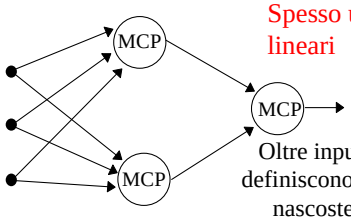
u_1	u_2	y	
0	0	-1	a
0	1	1	b
1	0	1	c
1	1	-1	d

$$A(u) = \frac{1}{1 + e^{-\left(\sum_{i=0}^{N} w_i u_i - \mu\right)}}$$

Un po' di tassonomia



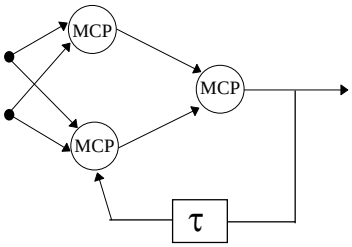
Perceptrone semplice



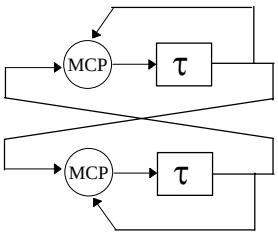
Spesso unità lineari

Oltre input/output si definiscono anche unità nascoste **(hidden units)**

Perceptrone multistrato



Ricorrente

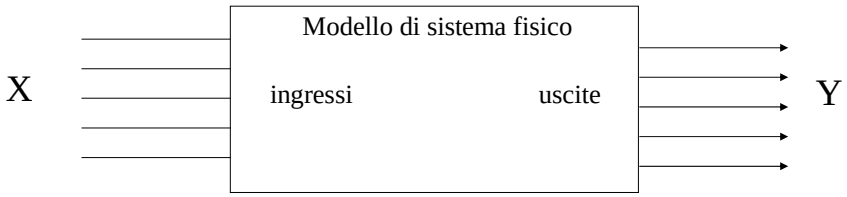


Ricorrente completamente connessa: autoassociativa (ingresso=stato)

Complessità della funzione realizzabile

Quanti più neuroni artificiali vengono connessi tanto più la funzione complessiva approssimabile diviene più complessa

$$Y=|y_1, y_2, y_3, \ldots y_n)|^T$$
$$y_i=g(X)$$
$$X=|x_1, x_2, x_3, \ldots x_m|^T$$



Reti neurali = approssimatori universali.

Riassunto

I neuroni connessionisti sono basati su:
•Ricevere una somma pesata degli ingressi.
•Trasformarla secondo una funzione non-lineare (scalino o logistica)
•Inviare il risultato di questa funzione all'uscita o ad altre unita'.

Le reti neurali sono topologie ottenute connettendo tra loro i neuroni in modo opportuno e riescono a calcolare funzioni molto complesse.

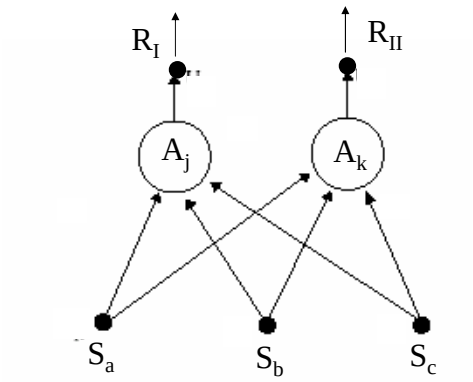
Sommario

Modelli semplici di neuroni

Il problema dell'XOR

L'apprendimento in reti di perceptroni con funzioni di attivazione logistica

Apprendimento



Apprendimento è la modifica dei parametri $\{w_{ij}\}$ e $\{\mu_j\}$ in modo tale che la rete neurale approssimi la trasformazione tra i pattern di input e di output.

$$y = g\left(\sum_j \left(w_{ij} u_j - \mu_i\right)\right) = \frac{1}{1 + e^{-\left(\sum_{i=0}^{N} w_i u_i - \mu\right)}}$$

I vari tipi di apprendimento

Supervisionato (learning with a teacher). Viene specificato per ogni pattern di input, il pattern desiderato in input.

Non-supervisionato (learning without a teacher). I neuroni verranno associati a pattern di ingresso contigui. Clustering. Mappe neurali.

Apprendimento con rinforzo (reinforcement learning, learning with a distal teacher). L'ambiente fornisce un'informazione del tipo success or fail.

Hebbian learning rule (1949)

…"When the axon of a cell A is near enough to excite a cell B and repeatedly or persistently takes part in firing it, some growth process or metabolic change takes place in one or both cells such that A's efficiency, as one of the cells firing B, is increased…." ($\Delta w_{ij} = f(y_i, x_j)$).

La forza di una sinapsi aumenta con l'utilizzo => Memoria?

Memoria a breve termine. Circuiti elettrici.
Memoria a lungo termine. Modificazioni chimiche.

In termini biologici si chiama **potenziamento**. LTP.

Apprendimento supervisionato

$$\min_{\{w\}} J(.) \qquad J = \left\| Y^D - g(W^{nuovo} U) \right\| \leq \left\| Y^D - g(W^{vecchio} U) \right\|$$

Y^D è l'uscita desiderata nota.

•Si tratta di un problema di minimizzazione di una cifra di merito (J) sullo spazio di parametri W.

Soluzione iterativa:
Obbiettivo: se esiste una soluzione, trovare ΔW in modo iterativo tale che l'insieme dei pesi W^{nuovo} ottenuto come:

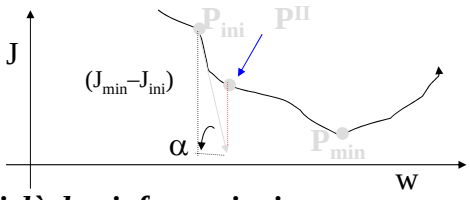
$$W^{nuovo} = W^{vecchio} + \Delta W$$

dia luogo a un errore sulle uscite di norma minore che con $W^{vecchio}$

Minimizzazione tramite gradiente

Minimizzo J(.) rispetto ai parametri.

Tecnica del gradiente applicata alla minimizzazione di funzioni non-lineari di **una variabile**: J= J(w|....).



La derivata, mi dà due informazioni:

1) In quale direzione di w, la funzione decresce.
2) Quanto rapidamente decresce.

Definisco uno spostamento arbitrario lungo la pendenza: maggiore la pendenza maggiore lo spostamento.Mi muovo lungo la direzione della pendenza, arrivo in P^{II}. Calcolo $J(w^{II})$.

Da qui riparto fino a quando non arrivo in P_{min} : $J(w^{k+1}) > J(w^k)$.

Minimizzazione di funzioni di più variabili

min(J{**w**}| …..) funzione costo od errore

Gradiente: $$\frac{\partial J(\{w\}|....)}{\partial w_j}\frac{\boldsymbol{w_1}}{|\boldsymbol{w_1}|}+\frac{\partial J(\{w\}|....)}{\partial w_2}\frac{\boldsymbol{w_2}}{|\boldsymbol{w_2}|}+\frac{\partial J(\{w\}|....)}{\partial w_3}\frac{\boldsymbol{w_3}}{|\boldsymbol{w_3}|}+\frac{\partial J(\{w\}|....)}{\partial w_4}\frac{\boldsymbol{w_4}}{|\boldsymbol{w_4}|}+.....$$

Modifico il valore dei pesi di una quantità proporzionale alla pendenza della funzione costo rispetto a quel parametro. Estensione della tecnica del gradiente a più variabili.

Serve un'**approssimazione iniziale** per i pesi $W_{ini} = \{w_j\}_{ini}$.

Apprendimento supervisionato tramite gradiente

Coppie input/output note.

Definizione di una funzione costo che misuri l'errore sull'uscita.

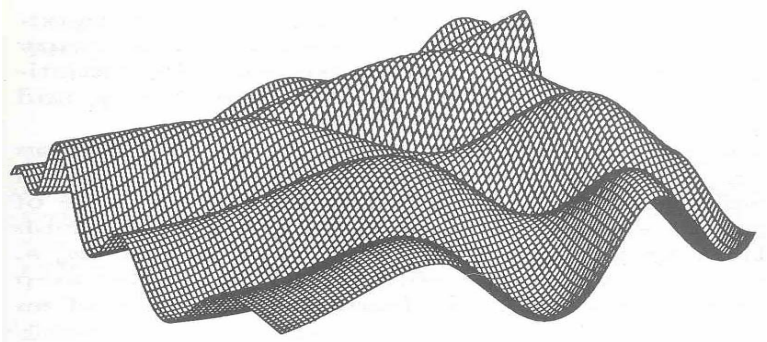
Modifica dei valori dei pesi in modo tale che la funzione costo sia minimizzata.

Reti multi-strato hanno elevata capacità computazionale, ma anche elevata complessita'.

Problemi nell'apprendimento supervisionato tramite gradiente

•Nota: W_{ini} è generalmente casuale e può condizionare la convergenza degli algoritmi iterativi.

•I problemi di convergenza sono legati all'esistenza di minimi locali del funzionale J(w | ….)



La pratica dell'apprendimento supervisionato

Fino a quando l'apprendimento non è stato completato:

1. Presentazione di un pattern di input / output.
2. Calcolo dell'output della rete con il pattern corrente.
3. Calcolo dell'incremento dei pesi.

Aggiornamento dei pesi.

Aggiornamento dei pesi:
- Per trial (ogni pattern)
- Per epoca (ogni insieme di pattern).

Perceptrone con unità di attivazione continue

Possiamo derivare una regola di apprendimento di spirito Hebbiano per una qualsiasi funzione di attivazione continua

$$y = g\left(\sum_{j=1}\left(w_{ij}u_j - \mu_i\right)\right) = g\left(\sum_{j=0}\left(w_{ij}u_j\right)\right)$$

Si tratta di un problema di minimizzazione di una cifra di merito, J, sullo spazio di parametri W:

$$J = \left\|y^D - g(W^{nuovo}U)\right\| \le \left\|y^D - g(W^{vecchio}U)\right\|$$

Errore

Devo trovare {w} : E(w) è minimo.

$$J = E(\mathbf{w}) = \frac{1}{2}\sum_p\left[\sum_i\left(y_{ip}^D - y_{ip}\right)^2\right] = \frac{1}{2}\sum_p\left[\sum_i\left(y_{ip}^D - g(\sum_j w_{ij}u_{jp})\right)\right]^2$$

Unità di attivazione lineari

$$y = g\left(\sum_{j=1}\left(w_{ij}u_j - \mu_i\right)\right) = g\left(\sum_{j=0}\left(w_{ij}u_j\right)\right)$$

Caso lineare (g = 1):

$$y_i = \sum_{j=1}\left(w_{ij}u_j - \mu_i\right) = \sum_{j=0}\left(w_{ij}u_j\right) \qquad \text{====>} \qquad \mathbf{Y} = \mathbf{W}\,\mathbf{U}$$

Soluzione di un sistema lineare nei pesi!!

Condizione di risolubilità: **W** di rango massimo → {w} sono linearmente indipendenti.

Unità lineari, soluzione iterativa

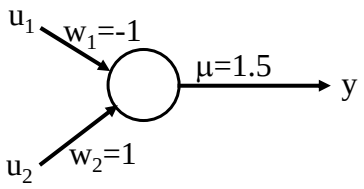
$$J = E(\mathbf{w}) = \frac{1}{2}\sum_p\left[\sum_i\left(y_{ip}^D - y_{ip}\right)^2 = \frac{1}{2}\sum_i\left(y_{ip}^D - \left(\sum_j w_{ij}u_{jp}\right)\right)^2\right]$$

$$\Delta w_{ij} = -\eta\frac{\partial}{\partial w_{ij}}\frac{1}{2}\sum_i\left(y_i^D - \left(\sum_j w_{ij}u_j\right)\right)^2$$

$$\Delta w_{ij} = +\eta\sum_i\left(y_i^D - \left(\sum_j w_{ij}u_j\right)\right)u_j = +\eta\left(y_i^D - y_i\right)u_j$$

δ rule (Hoff, 1960)

Esempio di delta rule - I



u_1	u_2	y^D
-1	-1	-1
-1	1	-1
1	-1	-1
1	1	1

U=|-1, 1| y^D = -1

$\eta = 0.2$

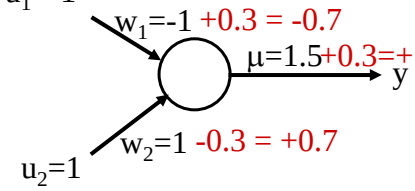
$$y = \sum_{j=1} \left(w_j u_j - \mu \right) = \sum_{j=0} \left(w_j u_j \right) \quad = (-1)(-1) + (1)(1) - 1.5 = 0.5 >> -1$$

$u_0 = 1 \quad w_0 = -\mu$

Esempio di delta rule - II

U=|-1, 1| y^D = -1

$\eta = 0.2$



$$y = \sum_{j=1} \left(w_j u_j - \mu \right) = \sum_{j=0} \left(w_j u_j \right) = \mathbf{0.5}$$

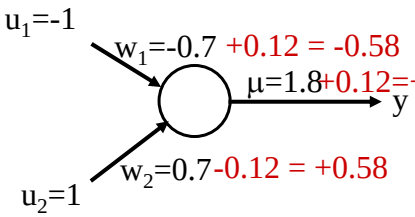
$$\Delta w_{ij} = +\eta \left(y_i^D - y_i \right) u_j$$

$$-\Delta\mu = \Delta w_0 = \eta \left(y_i^D - y_i \right) u_0 = \eta(-1-0.5)(1) = -0.30$$
$$\Delta w_1 = \eta \left(y_i^D - y_i \right) u_1 = \eta(-1-0.5)(-1) = +0.30$$
$$\Delta w_2 = \eta \left(y_i^D - y_i \right) u_2 = \eta(-1-0.5)(1) = -0.30$$

Esempio di delta rule - III

U=|-1, 1| y^D = -1

η = 0.2



$$y = \sum_{j=1} \left(w_j u_j - \mu \right) = \sum_{j=0} \left(w_j u_j \right) = \textbf{-0.4}$$

-0.76

$$\Delta w_{ij} = +\eta \left(y_i^D - y_i \right) u_j$$

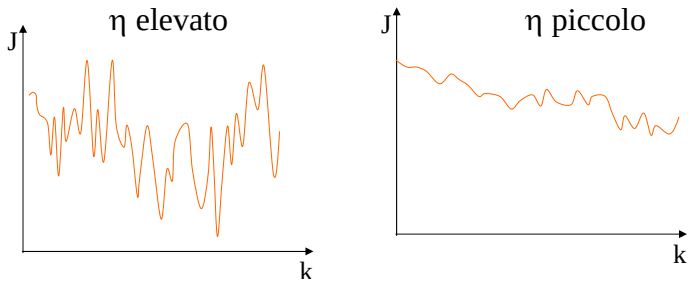
$$-\Delta\mu = \Delta w_0 = \eta \left(y_i^D - y_i \right) u_0 = \eta(-1-(-0.4))(1) = -0.12$$
$$\Delta w_1 = \eta \left(y_i^D - y_i \right) u_1 = \eta(-1-(-0.4))(-1) = +0.12$$
$$\Delta w_2 = \eta \left(y_i^D - y_i \right) u_2 = \eta(-1-(-0.4))(1) = -0.12$$

Ruolo di η – learning rate

$$\Delta w_{ij} = +\eta \left(y_i^D - y_i \right) u_j$$

Calmiera il Δw_{ij} per evitare che :

- Un peso sia specifico di un'unità ingresso-uscita.
- Oscillazioni durante l'apprendimento senza convergenza.

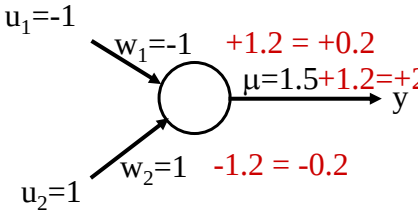


 □η può variare durante l'addestramento.

Esempio di delta rule – Cattiva scelta di η

U=|-1, 1| $y^D = -1$

η = 0.8



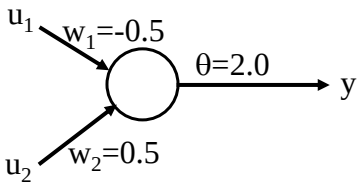
$$y = \sum_{j=1} \left(w_j u_j - \mu \right) = \sum_{j=0} \left(w_j u_j \right) = \mathbf{0.5}$$

$$\mathbf{-2.3}$$

$$\Delta w_{ij} = +\eta \left(y_i^D - y_i \right) u_j$$

$$-\Delta\mu = \Delta w_0 = \eta \left(y_i^D - y_i \right) u_0 = \eta(-1-0.5)(1) = -1.2$$
$$\Delta w_1 = \eta \left(y_i^D - y_i \right) u_1 = \eta(-1-0.5)(-1) = +1.2$$
$$\Delta w_2 = \eta \left(y_i^D - y_i \right) u_2 = \eta(-1-0.5)(1) = -1.2$$

Esempio di specializzazione sui pattern a, b, c



	u_1	u_2	y^D	
a	-1	-1	-1	a
b	-1	1	-1	b
c	1	-1	-1	c
d	1	1	1	d

a $y = \sum_{j=1} \left(w_j u_j - \theta \right) = \sum_{j=0} \left(w_j u_j \right)$ = (-0.5)(-1) + (-0.5)(1) – 2.0 = -2

b $y = \sum_{j=1} \left(w_j u_j - \theta \right) = \sum_{j=0} \left(w_j u_j \right)$ = (-0.5)(-1) + (0.5)(1) – 2.0 = -1

c $y = \sum_{j=1} \left(w_j u_j - \theta \right) = \sum_{j=0} \left(w_j u_j \right)$ = (-0.5)(1) + (0.5)(-1) – 2.0 = -3

d $y = \sum_{j=1} \left(w_j u_j - \theta \right) = \sum_{j=0} \left(w_j u_j \right)$ = (-0.5)(1) + (0.5)(1) – 2.0 = **-2**

Unità non-lineari, soluzione iterativa

$$J = E(\mathbf{w}) = \frac{1}{2}\sum_p \left[\sum_i \left(y_{ip}^D - y_{ip} \right)^2 \right] = \frac{1}{2}\sum_p \left[\sum_i \left(y_{ip}^D - g(\sum_j w_{ij} u_{jp}) \right)^2 \right]$$

$$\Delta w_{ijp} = -\eta \frac{\partial}{\partial w_{ij}} \frac{1}{2} \sum_i \left(y_i^D - g(\sum_j w_{ij} u_j) \right)^2 =$$

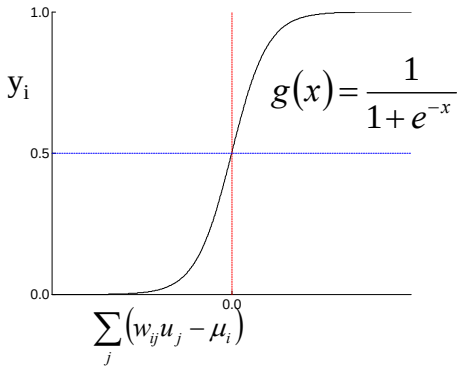
$$\eta \sum_i \left(y_i^D - g\left(\sum_j w_{ij} u_j \right) \right) g'\left(\sum_j w_{ij} u_j \right) u_j = +\eta \left(y_i^D - y_i \right) g'\left(\sum_j w_{ij} u_j \right) u_j$$

δ rule

Perceptrone con unità di attivazione logistiche

$$g'(x) = g(x) \cdot (1 - g(x)) \qquad y_i = g\left(\sum_j \left(w_{ij} u_j - \mu_i \right) \right)$$

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



Update dei pesi per funzione logistica

$$J = E(\mathbf{w}) = \frac{1}{2}\sum_p \left[\sum_i \left(y_{ip}^D - y_{ip}\right)^2 \right] = \frac{1}{2}\sum_i \left(y_{ip}^D - g(\sum_j w_{ij} u_{jp}) \right)^2 \Bigg]$$

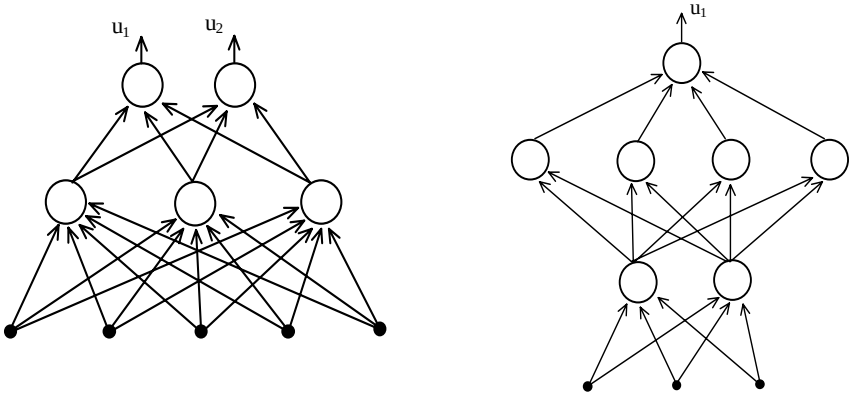
$$\Delta w_{ijp} = +\eta \sum_i \left(y_i^D - g(.)\right) g'(.) u_j = +\eta \left(y_i^D - y_i\right) y_i (1 - y_i) u_j$$

δ rule

NB yi =[0 1]. Per yi = 0 o yi = 1 non c'è apprendimento anche se l'uscita è sbagliata. Quando si verifica questa situazione?

Si cerca di mantenere le unità lontane della saturazione.

Perceptrone a più strati



Algoritmi di *apprendimento più sofisticati: back-propagation. Collegamento con la teoria statistica dell'apprendimento*
➔ *Corso di reti neurali.*

Riassunto - Apprendimento

Algoritmi iterativi per adattare il valore dei parametri (pesi).

Definizione di una funzione costo che misura la differenza tra valore fornito e quello desiderato.

Algoritmo (gradiente) che consente di aggiornare i pesi in modo da minimizzare la funzione costo.

Training per pattern (specializzazione) o per epoche.

Problemi

Quando si termina l'algoritmo di apprendimento?

Bootstrap – Vengono estratti pattern con ripetizioni.

Cross-Validation - Errore sull'insieme di training = Errore sull'insieme di test.

Utilizzare lo "structural risk" invece dell'"empirical risk".

Si vuole evitare che la rete si specializzi troppo sui pattern di training e non sia in grado di interpolare.

Sommario

Modelli semplici di neuroni

Il problema dell'XOR

L'apprendimento in reti di perceptroni con funzioni di attivazione lineare

Problemi

Qual è il problema principale dell'apprendimento supervisionato?

L'uscita delle funzioni logistiche è compresa tra 0 e 1. Come si possono approssimare funzioni con un range più ampio?