Introduzione al Soft Computing

Le Reti Neurali

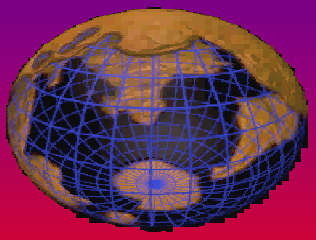
Cristian Randieri

Intellisystem TECHNOLOGIES

www.intellisystem.it

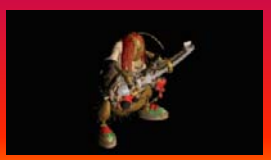
Reti Neurali: la linea di pensiero...

Esistono classi di problemi
che la mente umana
risolve molto meglio dei
tradizionali sistemi di calcolo

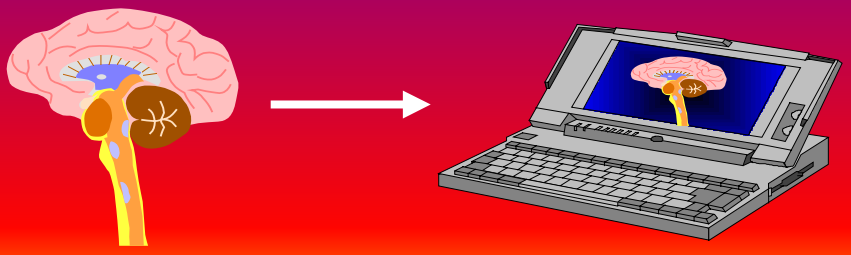


(es. ricostruzione di immagini e suoni
partendo da loro versioni degradate

La capacità di elaborazione del cervello umano
non è dovuta tanto alla velocità di trasmissione
ed elaborazione delle informazioni, quanto
all'elevato grado di parallelismo della struttura



Se si "copia" la struttura del cervello umano nella realizzazione di un sistema di calcolo, le capacità proprie della mente (apprendere da esempi, capacità di generalizzare etc) si potrebbero risolvere problemi più complessi...



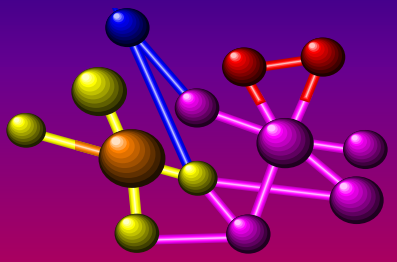
Tappe fondamentali del "connessionismo"

- 1943 - McCulloch e Pitts definiscono il primo modello formale del neurone
- 1949 D. Hebb propone la Hebbian learning rule
- 1954 Minsky presenta un neurocomputer
- 1958 Rosemblatt introduce il perceptrone
- 1960 widrow e Hoff introducono la ADALINE (ADAptive LINEar combiner)

... (continua) Tappe del connessionismo

- 1962 - Widrow e Hoff introducono la Widrow-Hoff learning rule
- 1969 - Minsky e Papert pubblicano "Perceptrons"
- 1982 - Hopfield introduce una struttura in grado di funzionare come memoria associativa
- 1986 McClelland e Rumelhart introducono l'algoritmo di "back-propagation"

Classificazione delle reti neurali



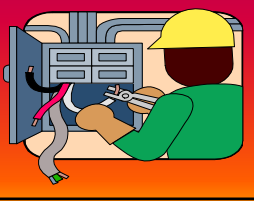
Feedforward
Ricorrenti
Tempo continue
Tempo discrete

Cosa sono i sistemi intelligenti ?

- **I sistemi intelligenti sono dei sistemi (o degli algoritmi) progettati per avere qualche qualità di tipo umano quale:**
  - **la possibilità di processare in parallelo le informazioni,**
  - **l'adattamento all'ambiente,**
  - **la memoria associativa,**
  - **l'apprendimento,**
  - **la capacità di generalizzazione,**
  - **l'auto-organizzazione.**

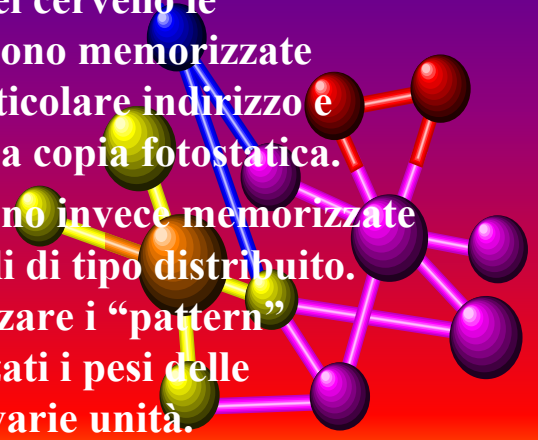
Processamento parallelo delle informazioni

- **Quando un operatore umano controlla un impianto (es. un impianto chimico) riceve informazioni da più sensori. Egli processa contemporaneamente le informazioni ed aggiusta i set-points dei vari loop di controllo allo scopo di ottenere dall'impianto le performances desiderate.**



Memorie associative ed apprendimento

- Se esaminassimo la memoria umana troveremmo che nel cervello le informazioni non sono memorizzate utilizzando un particolare indirizzo e nemmeno come una copia fotostatica.
- Le informazioni sono invece memorizzate secondo dei modelli di tipo distribuito. Invece di memorizzare i "pattern" vengono memorizzati i pesi delle connessioni tra le varie unità.



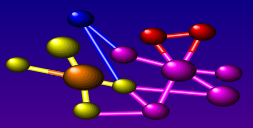
Quali sono i motivi percui le persone sono più brave delle macchine ?

L'apprendimento

Tecnicamente parlando si possono distinguere tra tipi di apprendimento:

- Supervised Learning
- Associative Reinforcement Learning
- Unsupervised Learning

Struttura delle reti neurali



Pur essendo un neurone reale molto complesso, una sua schematizzazione che si è rivelata di grande utilità (neurone artificiale) è la seguente. Ogni neurone riceve in ingresso segnali $x_1,x_2,...,x_i,...,x_n$ *da* **n** *altri neuroni, tramite connessioni di intensità* w_i *(pesi sinattici) fornendo in uscita un solo segnale* **y**. *I segnali di ingresso vengono consolidati in un potenziale post-sinattico*

$$P = \sum_{i=1}^{n} w_i x_i$$

media pesata degli ingressi, mentre l'uscita è fornita da una opportuna funzione di trasferimento $y = f(P-\theta)$ *dove* θ *è una soglia caratteristica del neurone in questione.*

Funzioni di attivazione

Funzione a gradino: ***y=f(x)=sign(x)***, **cioè** ***f(x)=1***, **per** $x \geq 0$ ($P \geq \theta$) **e** ***f(X)=0***

Funzione sigmoide o curva logistica:

$$y = f(x) = \frac{1}{1+e^{-Ax}}$$

Le caratteristiche della funzione sigmoide sono che essa vale 0.5 per ***x=0*** ($P = \theta$) **ed inoltre tende a zero per** ***x*** **grandemente negativo e tende a uno per** ***x*** **grandemente positivo. Inoltre la funzione può essere più o meno ripida in funzione del parametro** ***A*** **ed in particolare tende a diventare la funzione a gradino per** ***A*** **molto grande e la funzione lineare per** ***A*** **molto piccolo. In alcuni casi si usa la grandezza** ***T=1/A*** **che viene denominata temperatura per analogia con la termodinamica in quanto che, in alcuni casi viene diminuita gradualmente in una sorta di processo di raffreddamento.**

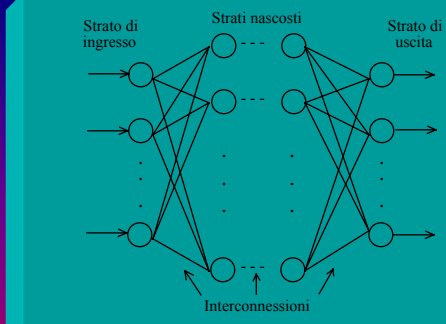
Caratteristica della sigmoide

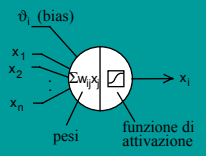
Una proprietà interessante della sigmoide, che sarà sfruttata in fase di apprendimento, è che la sua derivata è data da

$$y' = \frac{dy}{dx} = Ay(1-y)$$

In alternativa alla sigmoide si usa in alcuni casi la tangente iperbolica che ha la caratteristica di fornire valori compresi tra -1 e 1. La sua espressione è

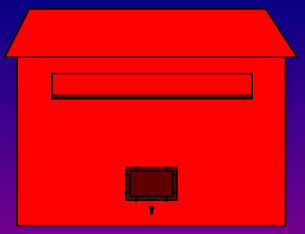
$$y = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}}$$





L'uscita y di ogni neurone, in una rete, è a sua volta uno degli input di molti altri neuroni. In generale una rete neurale artificiale è costituita da *n* neuroni di ingresso (corrispondenti ai neuroni sensori come quelli della retina nell'occhio), da *m* neuroni di uscita (corrispondenti ai neuroni motori come quelli che attivano i muscoli) e da *k* neuroni intermedi (interneuroni o neuroni nascosti). Questi ultimi sono collegati con altri neuroni intermedi e/o con neuroni sensori a monte e/o con neuroni motori a valle.

Black Box

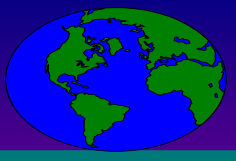


Se in un determinato momento, gli input della rete sono $x_1,x_2,...,x_i,...,x_n$ *allora si otterranno gli output* $y_1,y_2,...,y_m$ *con*

$$y_j = f_j(x_1,x_2,...,x_n)$$

In sostanza una rete può essere concepita come una scatola nera che trasforma determinati ingressi in corrispondenti uscite. Se gli input sono i dati di un problema e gli output le relative soluzioni, la rete risolve quel problema.

Interazioni col mondo esterno



L'apprendimento della rete è dovuto alle sue interazioni con l'ambiente esterno, tipicamente nel processo di apprendimento, i pesi sinattici tra ogni coppia di neuroni vengono modificati in funzione delle prestazioni. Nelle reti artificiali si hanno due tipi principali di apprendimento detti **supervisionato** *e non supervisionato. Nel primo caso vengono presentati alla rete, in ogni istante* **t**, *opportuni campioni degli ingressi* x_i *e dei corrispondenti output desiderati* d_j. *Le variazioni dei pesi sinattici necessarie per l'aggiornamento*

$$w(t+1) = w(t) + \Delta w$$

sono un'opportuna funzione di una metrica dell'errore e quindi delle differenze (y_j-d_j) tra output ottenuti y_j e output desiderati d_j. In generale gli algoritmi di apprendimento assumono come metrica l'errore quadratico medio che viene minimizzato.

Apprendimento supervisionato e non supervisionato

L'apprendimento supervisionato richiede quindi la conoscenza preliminare non solo degli input x_i ma anche dei corrispondenti output voluti d_j. Ogni campione

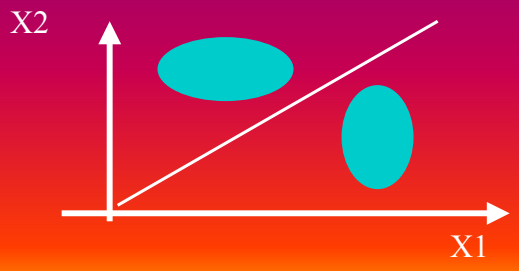
$$x_1, x_2, \ldots, x_n \qquad d_1, d_2, \ldots, d_m$$

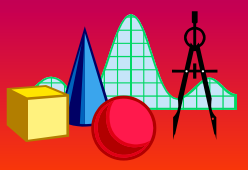
è un elemento di un insieme di *p* campioni denominato "training set".

Nel caso invece dell'apprendimento non supervisionato, il training set è tipicamente costituito da numerosi campioni di ingresso $X_i = (x_{i1}, x_{i2}, \ldots, x_{in})$ che si vogliono classificare in un numero limitato *m* di classi $C_1, C_2, \ldots, C_m$.. Nel presentare i campioni non viene però dichiarata la classe di appartenenza. E' la stessa rete ad auto organizzarsi, cambiando progressivamente i suoi pesi sinattici, in modo tale da eseguire, dopo l'apprendimento, classificazioni corrette.

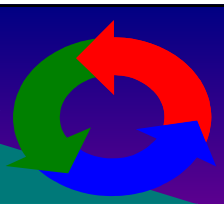
Reti a separazione lineare

La rete neurale più semplice che possiamo costruire è quella formata da un solo neurone con n ingressi ed una sola uscità y. Adottiamo, per fissare le idee, una funzione di trasferimento a gradino e studiamone le prestazioni. Una tale rete, che viene indicata come perceptrone, può riconoscere tutti gli esemplari di una determinata forma F , distinguendoli da quelli relativi a forme non-F. Nel caso bidimensionale basterà che una retta basterà che una retta separi i punti (x_1, x_2) corrispondenti ad una forma F da quelli relativi a forme non F.





Formulazione matematica..



Se w_1 e w_2 sono i pesi sinattici e θ è la soglia del neurone, l'equazione della retta deve essere

$$w_1 x_1 + w_2 x_2 - \theta = 0$$

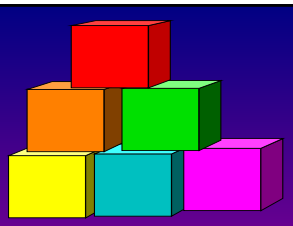
In tal caso la retta divide il piano in due semipiani per i quali si ha rispettivamente

$w_1 x_1 + w_2 x_2 - \theta > 0$, **cioè** $\sum_{i=1}^{n} w_i x_i = P > \theta$, **quindi** ***y=1***

$w_1 x_1 + w_2 x_2 - \theta < 0$, **cioè** $\sum_{i=1}^{n} w_i x_i = P < \theta$, **quindi** ***y=0***

dove P è già stato definito potenziale postsinattico. Fornendo come ingressi le coordinate di un punto che giace nel semipiano di F la rete darà uscita unitaria, altrimenti l'uscita sarà nulla.

Iperpiani di separazione..



Nel caso di tre coordinate possiamo parlare di piano di separazione fra due semispazi e, nel caso più generale di n dimensioni, si parlerà di iperpiani di separazione. Una variazione dei valori dei pesi sinattici w_i determina una variazione dell' inclinazione dell'iperpiano, mentre una variazione della soglia determina uno spostamento dell'iperpiano parallelamente a se stesso. L'apprendimento dei pesi e della soglia si risolve allora in una serie di piccoli movimenti dell'iperpiano sino a realizzare la separazione lineare, partendo da un iperpiano iniziale generico che non la realizza.

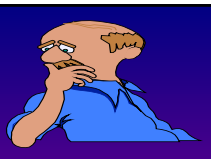
Velocità d'apprendimento

In generale si usa parlare di apprendimento dei pesi e si suole considerare la soglia come il peso di un ingresso sempre unitario. Questo è possibile ridefinendo P come

$$P = \sum_{i=1}^{n} w_i x_i - \theta$$

Il modo più semplice per addestrare una rete di questo tipo è quello di far variare i pesi (aumentarli o diminuirli) in modo proporzionale agli ingressi cioè secondo la formula $\Delta w = \eta \cdot x$ dove η è un coefficiente detto "learning rate" o "velocità di apprendimento" .

Limitazioni di tale approccio



E' ovvio che la rete descritta funziona solo se esiste un iperpiano che divide due classi e che ciò non avviene sempre.

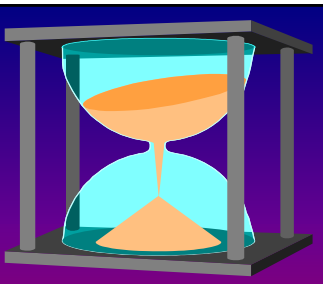
Supponiamo adesso di considerare lo spazio bidimensionale binario. Gli elementi dello spazio sono le quattro coppie 00,01,10 e 11. E' facile separare, ad esempio, la forma logica AND = 11 dalle altre, o la forma logica OR = 10,01,11 da 00 ed anzi è facile verificare, graficamente o analiticamente, che esistono infinite rette che operano questa separazione.

Se però consideriamo la forma XOR = 01,10 i suoi due elementi non sono separabili dagli altri due punti con una retta. In tal caso la separazione è effettuabile mediante linee curve cioè mediante l'introduzione di reti più complicate con almeno un livello di neuroni nascosti.

Algoritmo di Widrow-Hoff

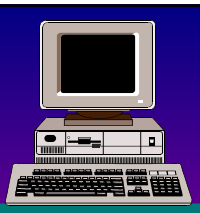
*Se esistono m forme diverse F_j, ciascuna delle quali costituisce una configurazione di n punti x ed è separabile linearmente dalle altre, allora ogni neurone può riconoscere una configurazione particolare e la rete, globalmente, può riconoscere m configurazioni. Per ottenere questo occorre assegnare agli m*n pesi sinattici. L'algoritmo di Widrow-Hoff calcola i pesi necessari, partendo da pesi casuali e apportando ad essi piccole variazione, graduali e progressive, in un processo che converge alla soluzione finale.*

.....(continua)



Supponendo che la rete abbia n ingressi x, ed m neuroni (uscite). Gli m neuroni abbiano tutti la stessa funzione di trasferimento associata f e che tale funzione sia differenziabile. Sia inoltre disponibile un training set di p esempi. Ciò premesso l'algoritmo consiste nel ripetersi ciclico di alcuni passi fino al raggiungimento dei risultati desiderati.

Dentro l'algoritmo...



1) Presentazione di un input generico $X=(x_1,x_2,...,x_i,...x_n)$

2) Calcolo degli output corrispondenti $y_j=f(P_j)$ **con** $P_j=\sum_{i=1}^{n} w_{ji}x_i$ **;**

3) Calcolo dell'errore quadratico medio $E=\frac{1}{2}\sum_{j=1}^{m}(y_j-d_j)^2$

4) Calcolo delle variazioni $w_{ji}(t+1)-w_{ji}(t)=\Delta w_{ji}=-\eta\frac{dE}{dw_{ji}}$ **dove** η **è un coefficiente compreso tra zero ed uno detto "learning rate".**

Il processo si interrompe quando E raggiunge un valore piccolo prefissato.

Formule di aggiornamento..

Per giustificare la formula dell'aggiornamento dei pesi sinattici rileviamo che occorre minimizzare* E *quindi se* E *aumenta con w (*dE/dw>0*) i pesi devono diminuire (*$\Delta w<0$*) mentre se E diminuisce all'aumentare di w (*dE/dw<0*) i pesi devono essere aumentati (*$\Delta w>0$*).

Vediamo come si possono scrivere le formule di aggiornamento scritte al punto quattro. Essendo

$$\frac{dE}{dw_{ji}}=(y_i-d_j)\cdot\frac{dy_j}{dw_{ji}}=(y_i-d_j)\cdot\frac{dy_j}{dP_j}\cdot\frac{dP_j}{dw_{ji}}$$

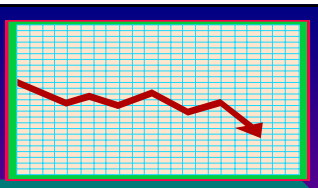
dove $\frac{dy_j}{dP_j}=f'(P_j)$ $\frac{dP_j}{dw_{ji}}=x_i$

si ha allora $\Delta w_{ji}=-\eta(y_j-d_j)f'(P_j)x_i=-\eta D_j x_i$

avendo posto $D_j=(y_j-d_j)f'(P_j)$**.**

Se la funzione di trasferimento considerata è la sigmoide si ha $f'(P_j)=y_j(1-y_j)$ ***e dunque*** $\Delta w_{ji}=-\eta(y_j-d_j)y_j(1-y_j)x_i$

Come evitare i minini locali

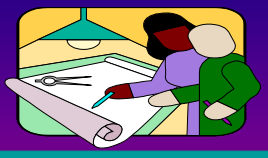


Se la funzione di trasferimento è lineare la sua derivata è costante e dunque

$$\Delta w_{ji} = -\eta\left(y_j - d_j\right)x_i$$

Dopo la presentazione di un congruo numero di esempi in altrettanti cicli consecutivi l'errore diventerà più piccolo di un minimo prefissato ed il processo di apprendimento sarà così concluso. Se invece, dopo una prima presentazione di tutti gli esempi del training set (prima epoca) l'errore non è ancora accettabile si procede a una seconda epoca a così via. Solo nel caso di funzione di trasferimento lineare l'errore avrà un minimo assoluto mentre negli altri casi esistono dei minimi locali. E' opportuno allora in alcuni momenti inserire delle perturbazioni casuali sui pesi in modo da evitare lo stallo in uno di questi minimi.

Apprendimento incrementale e/o cumulativo ?

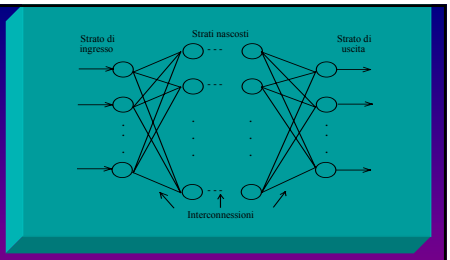


Precedentemente abbiamo considerato la versione on-line, "apprendimento proporzionale" dell'algoritmo cioè quella che prevede l'aggiornamento dei pesi alla presentazione di ogni singolo esempio. La versione batch dell'algoritmo "apprendimento cumulativo" prevede invece che l'aggiornamento sia effettuato ad ogni epoca. L'errore da minimizzare in questo caso risulta

$$E = \frac{1}{2}\sum_{k=1}^{p}\sum_{j=1}^{m}\left(y_{kj} - d_{kj}\right)^2$$

In generale la versione on-line è preferita alla versione batch in quanto fornisce normalmente risultati migliori cioè convergenza più veloce e minori oscillazioni.

Perceptrone multirstato



Una rete neurale con uno stadio di ingresso, uno o più strati intermedi ed uno stadio di uscita è denominato Multi-Layers Perceptron (perceptrone multistadio MLP). Tale rete è anche definita di tipo feed-forward perchè i segnali si propagano dagli ingressi alle uscite, attraverso i neuroni intermedi, non avendosi connessioni trasversali o connessioni feed-back. In conclusione una rete neurale MLP può, in teoria, eseguire qualunque separazione risolvendo così qualsiasi problema di classificazione, di riconoscimento, di scelta. Tuttavia, nel caso di problemi complessi, la realizzabilità teorica di una rete MLP che risolva il problema, può non essere di grande aiuto. Infatti il grande numero richiesto di neuroni intermedi, la notevole lunghezza del processo di addestramento e la non disponibilità di un training set adeguato potrebbero diventare ostacoli insormontabili per una realizzazione effettiva.

Back-Propagation (BP)

Purtroppo l'algoritmo di Widrow-Hoff (WH) non può essere applicato al caso di reti MLP. Sappiamo infatti calcolare l'errore $E_j = (y_j - d_j)$ per ogni neurone di uscita ma non sappiamo calcolarlo per i neuroni intermedi *k* non conoscendo i valori voluti d_k.

Questo fatto ha impedito per molto tempo lo sfruttamento pratico delle notevoli possibilità delle reti MLP. Solo nel decennio scorso è stato proposto l'algoritmo back-propagation (BP) che supera tale problema. Sostanzialmente l'algoritmo BP è un estensione del WH. Il punto chiave è l'opportuna ripartizione dell'errore, noto al livello di uscita, tra i neuroni dello strato e quelli degli strati intermedi. In ogni ciclo dell'algoritmo si hanno due fasi; l'una (forward) per il calcolo degli errori che è virtualmente identica al caso del perceptrone e l'altra per la ripartizione degli errori negli strati intermedi.

...dentro l'algoritmo

Consideriamo una rete di tre strati; il primo con attività neurali x_i, il secondo (intermedio) con attività y_j e pesi sinattici w_{ji}, il terzo con attività z_k e pesi sinattici w_{kj}. L'aggiornamento dei pesi w_{kj} è identico a quello già visto per l'algoritmo WH $\Delta w_{kj} = -\eta \frac{dE}{dw_{kj}} = -\eta (z_k - d_k) f'(P_k) y_j$

ponendo $D_k = (z_k - d_k) f'(P_k)$ **si ha** $\Delta w_{kj} = -\eta D_k y_j$

Per quanto riguarda l'aggiornamento dei pesi sinattici w_{ji} si ha, similmente a quanto visto per l'algoritmo Widrow-Hoff

$$\Delta w_{ji} = -\eta \frac{dE}{dw_{ji}} \qquad \frac{dE}{dw_{ji}} = \frac{dE}{dy_j} \cdot \frac{dy_j}{dP_j} \cdot \frac{dP_j}{dw_{ji}} \qquad \frac{dy_j}{dP_j} = f'(P_j)$$

$$\frac{dP_j}{dw_{ji}} = x_i \qquad \frac{dE}{dw_{ji}} = \frac{dE}{dy_j} \cdot f'(P_j) \cdot x_i$$

...continua

Per ottenere dE/dw_{ji} occorre ora conoscere dE/dy_i , ma gli errori noti sono (z_k-d_k), differenza tra gli output ottenuti z_k e quelli desiderati d_k. Esprimendo dE/dy_i in funzione di questi errori ne deriva che $\frac{dE}{dw_{ji}} = f'(P_j) \cdot x_i \cdot \sum_k D_k \cdot w_{kj}$

Ponendo $D_j = f'(P_j) \cdot \sum_k D_k \cdot w_{kj}$ **si ottiene** $\Delta w_{ji} = -\eta \frac{dE}{dw_{ji}} = -\eta D_j x_i$

Quanto sopra può essere generalizzato dicendo che, dati due strati successivi con attività neurali x_i e pesi w_{ji}, l'aggiornamento Δw_{ji} deve essere effettuato con la formula $\Delta w_{ji} = -\eta D_j x_i$ **dove però**

$D_j = (y_j - d_j) f'(P_j)$ **se *j* è un neurone di uscita**

$D_j = f'(P_j) \cdot \sum_k D_k \cdot w_{kj}$ **se *j* è un neurone intermedio.**

Momentum

Generalmente l'aggiornamento dei pesi viene eseguito con una formula modificata nel seguente modo

$$\Delta w_{ji}(t+1) = -\eta D_j x_i + \beta \cdot \Delta w_{ji}(t)$$

dove β è un coefficiente positivo minore di uno denominato "momentum". In questo caso l'aggiornamento nel ciclo *t* tende a non differenziarsi molto da quello nel precedente ciclo *t-1* per una sorta di effetto di inerzia. Si può anche affermare che il termine di momentum ha l'effetto di filtrare le variazioni ad alta frequenza della superficie di errore *E*, nello spazio dei pesi *w*. In sostanza l'adozione di tale termine (tipicamente β=0.9) consente di assumere un coefficiente η più elevato (tipicamente η=0.7) senza incorrere in oscillazioni.

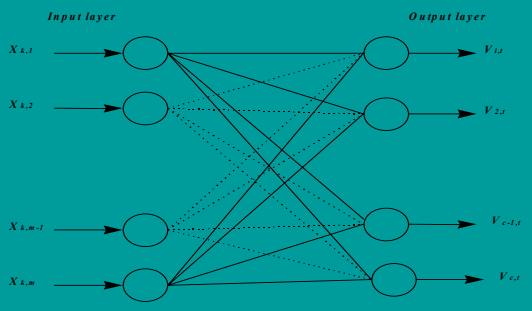
....anche la BP ha i propri limiti!!!

Non esiste a tutt'oggi una teoria che suggerisca, per risolvere un determinato problema, il numero di strati intermedi o il numero di neuroni necessari per ognuno di questi strati. Occorre quindi procedere per tentativi applicando criteri empirici. La scelta oculata del numero di neuroni intermedi è molto importante perchè da essa deriva la più o meno buona capacità di generalizzazione della rete. Se ci sono troppi neuroni intermedi la rete impara troppo il training set, non generalizza e richiede anche un processo di apprendimento più lungo. D'altra parte, se ci sono pochi neuroni intermedi la rete non riesce ad apprendere nemmeno il training set. La capacità di generalizzazione della rete dipende anche da altri fattori quali la lunghezza del processo di apprendimento o la presentazione degli esempi.

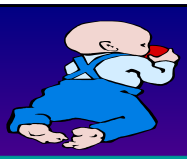
Apprendimento non supervisionato

Nell'apprendimento senza supervisione dove, come già è stato detto, il training set è costituito da numerosi campioni $X_i = (x_{i1}, x_{i2}, ..., x_{in})$ che si vogliono classificare in un numero limitato p di classi $C_1, C_2, ..., C_p$. Ad esempio i campioni sono una lunga sequenza di segnali vocali X_i e le classi sono i p=26 fonemi della lingua inglese. Nel presentare i campioni, non viene ora dichiarata la loro classe di appartenenza (a priori ignota), al contrario di quanto avviene nell'apprendimento supervisionato. E' la stessa rete ad auto-organizzarsi, cambiando progressivamente i suoi pesi sinattici, in modo tale da eseguire classificazioni corrette ad apprendimento completato.

....continua



La rete in questione ha n *ingressi e* m *(*m>p*) neuroni di uscita collegati con pesi sinattici* w_{ji}*. Ad ogni neurone* j *è associato quindi un vettore* $W_j = (w_{j1}, w_{j2}, ..., w_{jn})$ *cioè anche un punto nello spazio ad* n *dimensioni. Anche i vettori* $X_i = (x_{i1}, x_{i2}, ..., x_{in})$ *rappresentano dei punti nello stesso spazio. Se gli ingressi* X_i *sono raggruppati in* p *classi distinte si avranno* p *"nuvole" distinte di punti.*



....continua

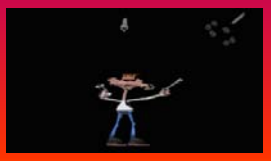
L'apprendimento consiste nella migrazione dei punti W_j verso i centroidi di queste nuvole partendo da una generica configurazione iniziale. Presentando dunque un certo input X_i tutti i neuroni competono per essere attivati ma vince uno solo di essi cioè quello per cui W_j è più vicino a X_i. Il premio di tale vittoria consiste in un ulteriore piccolo avvicinamento a X_i secondo la formula

$$\Delta w_j = \eta \cdot \left(X_i - W_j \right)$$

dove η è il solito coefficiente di apprendimento detto learning rate e compreso tra zero ed uno. Presentando in successione tutti i punti del training set si avrà come risultato l'avvicinamento progressivo dei punti W_j ai centroidi delle classi. Alla fine dell'apprendimento gli input che si riferiscono ad una certa classe attiveranno solo il neurone i cui pesi nello spazio rappresentano tale classe. Se questo avviene la rete ha imparato il problema di classificazione.

L'apprendimento che abbiamo ora descritto è denominato "competitive learning" per il fatto che i neuroni si competono la vittoria ad ogni presentazione di un ingresso.

Tale strategia è denominata "Winner Takes All" (WTA).



Intellisystem TECHNOLOGIES

Fine