MATERIALE DIDATTICO TRATTO DAL CORSO DI:

MODELLI DI SISTEMI FISIOLOGICI

ANNO 2009/2010

ING. GIOVANI PIOGGIA

ARGOMENTI:

-Introduzione alle reti neurali artificiali

-Apprendimento supervisionato

-Apprendimento non supervisionato

-Applicazioni delle reti multilivello

RETI NEURALI ARTIFICIALI

INTRODUZIONE

Osservando la natura risultano evidenti le difficoltà che dobbiamo affrontare nel tentativo di imitarla. Inoltre le tecniche a nostra disposizione risultano inadeguate rispetto a ciò che la natura è in grado di fare. La possibilità di riconoscere un eventuale predatore è essenziale alla sopravvivenza della preda e deve avvenire in un tempo sufficientemente piccolo per rendere efficace la fuga. Se si tiene conto del ritmo biologico e dei tempi di elaborazione a disposizione, si vede che l'algoritmo naturale messo in gioco dal sistema nervoso raggiunge lo scopo del riconoscimento in poche decine di passi.

D'altra parte, risulta altrettanto evidente che non è la capacità di elaborazione a nostra disposizione la ragione del nostro insuccesso. E' sufficiente un semplice calcolo, anche se grossolano, per renderci conto di ciò. Infatti se consideriamo una valutazione comparativa fra la capacità di elaborazione in bit/sec del sistema nervoso centrale (SNC) di un animale di piccola taglia ed un odierno personal computer si ottiene la tabella 1.

	SNC	**PC**
Elementi di base	10^9 [neuroni]	10^7 [transistori]
Ritmo di elaborazione	10 [operazioni/sec]	$5 \cdot 10^9$ [operazioni/sec]
Numero medio di collegamenti	10^3 [sinapsi per neurone]	2 [per transistore]
Informazione transitante per collegamento	4 [bit]	0.5 [bit]
Capacità di elaborazione	$4 \cdot 10^{13}$ [bit/sec]	$5 \cdot 10^{15}$ [bit/sec]

Tabella 1 – Valutazione comparativa delle capacità di calcolo fra il SNC di un animale di piccola taglia ed un odierno personal computer

Quindi apparentemente la capacità di elaborazione che abbiamo a disposizione, considerando un normale PC, è maggiore di quella biologica; evidentemente non è questa la caratteristica del SNC che dobbiamo imitare.

Dal punto di vista dell'imitazione fisica (hardware) è stato sperimentato, in alcuni casi particolari quali la retina e la coclea artificiali, l'impiego di meccanismi vicini a quelli biologici a costo energetico basso o addirittura nullo. E' interessante, tuttavia, avere presenti i limiti fisici che le tecnologie impongono, soprattutto in termini di richiesta di energia e potenza ed in confronto al contesto biologico. Gli elementi da tenere presenti nei riguardi di quest'ultimo sono l'energia necessaria per il cambiamento di stato di un neurone, che è dell'ordine di 10^{-16} Joule e la potenza richiesta mediamente dal cervello umano per effettuare un numero di operazioni stimato nell'ordine di 10^{16} op./sec è quindi dell'ordine di 1 Watt; intendendo per operazione il semplice cambiamento di stato. Queste cifre vanno confrontate con la richiesta di energia di un microprocessore attuale per effettuare un'operazione aritmetica (cambio di stato di circa 10^4 transistori) valutata in 10^{-8} Joule. Quindi la potenza richiesta nel caso più sfavorevole per implementare la capacità di elaborazione del cervello umano (10^{16} op./sec) ammonterebbe a 100 MWatt. Ciò è ottenibile sia con la tecnologia elettronica in via di sviluppo fino al limite fisico prevedibile sia con la tecnologia elettronica detta "neuromorfica" orientata allo sfruttamento di meccanismi fisici a basso o nullo consumo energetico. Quest'ultima potrebbe permettere al limite fisico un consumo di potenza di 10 MWatt.
Da queste considerazioni nasce subito una domanda: cosa bisogna propriamente imitare? A questo proposito svolge un ruolo determinante la sensibilità e quindi la specializzazione di chi imita, per la scelta dell'aspetto da privilegiare tra i molti presenti quali:

- il funzionamento parallelo di molte semplici e simili unità
- il comportamento non lineare degli elementi di base
- la distribuzione della memoria sulle connessioni
- la plasticità
- l'adattamento

L'aspetto, tuttavia, che caratterizza nettamente il sistema nervoso è la capacità di acquisire esperienza da esempi. Questa è la caratteristica peculiare da imitare, tenendo conto dell'esistenza di due situazioni diverse nell'apprendimento: presenza o assenza di un supervisore. Tale fattore ha un'influenza determinante ed ha radici biologiche: in assenza di supervisione esplicita, esiste sempre un'azione di guida con obiettivi pratici importanti quali ad esempio la sopravvivenza.
Dall'imitazione del sistema nervoso ne scaturiscono quindi algoritmi, usualmente implementati su calcolatori, noti come "reti neurali artificiali". Una rete neurale può essere definita come segue:

"Rete di elementi semplici interconnessi in parallelo in larga misura e la cui organizzazione gerarchica è progettata in modo da interagire con gli oggetti del mondo reale in modo simile al sistema nervoso biologico".

II campo tipico di applicazione delle reti neurali è quello dell'individuazione in sistemi complessi di legami ingresso-uscita, spesso nei casi in cui i dati a disposizione sono molto numerosi e per lo più contenenti poca informazione e di cui non si conosce la struttura ma solo esempi. L'esempio caratteristico è quello del riconoscimento. Per dare un'idea dei campi di utilizzazione si riporta successivamente un elenco non esaustivo di applicazioni note. E' importante tenere presente che una rete neurale non è certamente adatta a risolvere qualsiasi problema ed in molti casi è preferibile usare algoritmi già disponibili e che operano in modo soddisfacente.

Applicazioni:

- Sistemi intelligenti basati su ingressi da segnali di origine sensoriale (ad es. visiva, acustica, olfattiva, tattile): rivelazione, verifica e identificazione di caratteri e oggetti (codici a barre, caratteri stampati o scritti a mano, impronte digitali, firme, facce, difetti di prodotti, oggetti meccanici, minerali, etc.); riconoscimento di odori e composti (sistema olfattivo artificiale, lingua elettronica); movimento di oggetti e guida (sicurezza, automazione della fabbrica, sistemi di navigazione, robot mobili, localizzazione passiva di sorgenti sonore,

localizzazione attiva e guida di oggetti, sistemi di navigazione); elaborazione delle immagini (compressione, eliminazione di difetti di immagini in tempo reale); riconoscimento della voce (selezione telefonica, elaborazione di parole dettate, interfaccia intelligente uomo-calcolatore, verifica e identificazione di persone per via vocale, riconoscimento di oggetti dai suoni emessi, sistemi di diagnostica); elaborazione del suono (compressione e miglioramento in tempo reale della voce e della musica, aiuti audio e protesi); applicazioni basate su ingresso tattile (interfaccia intelligente uomo-macchina); altri sistemi sensoriali (sensori ad infrarossi, accelerometri, termocoppie, etc. per misure, controllo e sicurezza); sistemi sensoriali misti (dal testo al parlato, dalle immagini al suono, dalle immagini al tattile, riconoscimento di configurazioni tramite fusioni sensoriale);

- Sistemi di controllo adattativi: sistemi di integrazione sensoriale-motore; sistemi intelligenti di controllo dei processi; sistemi intelligenti di controllo di apparecchiature;
- Elaborazione adattativa dei segnati: compressione dati in un PC; comunicazione e archivio (voce, musica, immagini, video); miglioramento e sintesi dati (fax, voce, TV ad alta risoluzione, grafica, cancellazione di rumore); memorie associative;
- Sistemi esperti;
- Robot autonomi.

PROBLEMI FONDAMENTALI DELLE RETI NEURALI

I problemi che occorre affrontare nella progettazione di una rete neurale efficiente sono i seguenti:

Architettura della rete neurale

La struttura della rete neurale con i parametri che mette a disposizione è per sua natura adatta a realizzare compiti particolari. Quindi, quando si tratta di svolgere un compito assegnato, occorrerebbe fare riferimento ad un'architettura avente in sè la capacità di risolverlo. II problema in questione non è risolto, esistono solo risultati parziali, affetti inoltre dal pericolo di un'eccessiva dimensionalità, sicuramente ridondante;

Insieme di apprendimento

L'insieme di apprendimento contiene l'informazione su ciò che la rete neurale dovrebbe fare, cioè la rete neurale dovrebbe acquisire esperienza dagli esempi facenti parte dell'insieme di apprendimento. A questo proposito, quindi, la rete neurale può essere considerala come un meccanismo di generazione di un sistema esperto basato sull'imitazione della microstruttura del cervello umano. Ciò va inteso come contrapposizione al sistema esperto cosi come è inteso nell'ambito dell'intelligenza artificiale, dove l'imitazione del cervello dell'esperto umano avviene a livello di macrostruttura. II problema di valutare se l'insieme di apprendimento contenga tutta l'informazione necessaria al compilo che deve essere realizzato è ancora aperto. Tale problema è inoltre collegato a quello della riduzione della ridondanza dei dati;

Algoritmo di apprendimento

L'algoritmo di apprendimento serve a travasare l'informazione contenuta nell'insieme di apprendimento nell'architettura della rete neurale, fissandone i parametri. In particolare l'apprendimento si articola in una sequenza di operazioni iterative, che nel generico passo prevedono l'azione dell'ambiente e la modifica convergente dei parametri della rete mediante un algoritmo opportuno. Anche in questo caso è aperto il problema di valutare l'efficienza dell'algoritmo circa la capacità di estrarre la suddetta informazione. Sono disponibili soltanto risultati parziali.

LE BASI DELL'APPRENDIMENTO: IL PRINCIPIO DI OCCAM

La capacità di acquisire esperienza da esempi quindi è la caratteristica peculiare del sistema nervoso che più ci interessa imitare nelle reti neurali artificiali. L'imitazione può essere utilmente guidata dal buon senso, che distingue nettamente tra il semplice immagazzinamento degli esempi e l'acquisizione delle regole astratte che generano gli esempi. Occorre distinguere fra la MEMORIZZAZIONE delle regole esistenti fra gli esempi e la loro GENERALIZZAZIONE o ASTRAZIONE.

Il corretto funzionamento di una neurale sull'insieme di apprendimento non offre, ovviamente, garanzia di un altrettanto soddisfacente funzionamento su altri dati relativi allo stesso concetto, ma non utilizzati nella fase di apprendimento (insieme che può essere utilizzato per effettuare un test). Inoltre, è evidente che l'architettura della rete neurale gioca un ruolo fondamentale per l'efficienza della fase di apprendimento. Si consideri, ad esempio, il caso delle reti feedforward, che si vedranno più avanti. Verrà enunciata la loro capacità universale di approssimazione, che le porta ad apprendere qualsiasi funzione, si potrebbe dire che sono in grado di imparare ogni complesso legame ingresso-uscita. Quando il numero delle unità cresce aumenta il potere computazionale, ma la capacità di generalizzare su nuovi esempi tende a diminuire dato che il *fitting,* in questo caso può essere definito come *over-fitting*, sull'insieme di apprendimento ha luogo in un enorme spazio di parametri vincolati solo da pochi esempi. Questo origina una sorta di principio di indeterminazione dell'apprendimento secondo il quale non è possibile al variare dei pesi della rete neurale ottenere un'adeguata generalizzazione per nuovi esempi. Come si vedrà la limitazione del numero degli ingressi risulta particolarmente importante per limitare l'*over-fitting*. La memorizzazione e la generalizzazione possono anche essere espresse in termini di dispendio di risorse di memoria richiesto nel primo caso rispetto al secondo. Quindi l'efficienza nell'uso delle risorse disponibili, principio di base per un ingegnere, è anche il principio a cui si adegua il sistema nervoso nell'acquisizione di esperienza.

Tale principio è ben noto in letteratura tecnica sotto il nome di "Principio di Occam" e può essere enunciato nel seguente modo in relazione alle reti neurali:

"date due reti che soddisfano l'insieme di apprendimento, la rete di minore complessità è quella che si comporta meglio su esempi non visti, cioè ha la migliore capacità di generalizzazione"

II Principio di Occam è la guida più adatta alla determinazione della struttura della rete neurale ottimale dal punto di vista della capacità di generalizzazione. Infatti, esso suggerisce, nel caso delle reti neurali supervisionate, di utilizzare una funzione obiettivo da minimizzare costituita da due termini: l'uno riguardante l'adeguatezza della rete rispetto all'insieme di apprendimento; l'altro riguardante la complessità della rete. Indicando con Es il primo, con Ec il secondo e con *m* un valore che misura il numero dei parametri della rete, si ha per la funzione obiettivo:

$$E(m)=Es(m)+Ec(m)$$

L'andamento di $E(m)$ nelle sue due componenti è mostrato qualitativamente in figura 1, in cui è anche evidenziata la fase di memorizzazione.

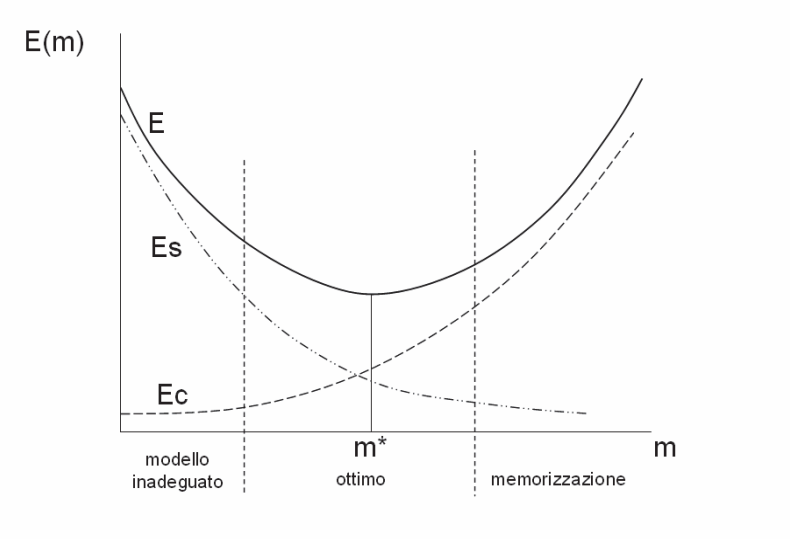


Figura 1 – Il principio di Occam applicato alle reti neurali

IL NEURONE FORMALE

Il neurone formale è un modello matematico del neurone biologico. E' l'elemento base necessario per la costruzione di una rete neurale. Il modello matematico non cattura appieno le proprietà biologiche e funzionali dei neuroni reali; costituisce una forte semplificazione della realtà mostrando tuttavia delle analogie che ne permettono l'utilizzo. Ci sono molti modi con cui semplificare le nostre conoscenze sui neuroni biologici allo scopo di caratterizzare matematicamente dei "neuroni artificiali" astratti (unità) che possano potenzialmente incorporare il calcolo computazione che avviene nel cervello. Le strategie più comuni partono dalle seguenti semplificazioni:

- Ignorare le non linearità delle relazioni fra gli spikes presinaptici e le attività del canale postsinaptico.
- Ingnorare le non linearità nelle relazioni fra le attività dei canali ed il potenziale postsinaptico.
- Incorporare gli effetti della struttura dendridica nei "pesi" sinaptici.
- Permettere ai pesi sinaptici di cambiare segno.

Alcune eventi hanno segnato la nascita del neurone formale e l'evoluzione delle reti.

•1943: McCulloch e Pitts presentano un lavoro sulle reti "neurologiche" introducendo un modello di neurone formale. Le reti formate con questi neuroni sono in grado di computare funzioni della logica del I ordine.

•1949: Donald Hebb ipotizzò l'apprendimento biologico come fenomeno sinaptico. Propose una formula matematica per simulare l'apprendimento nelle reti conosciuta come legge di Hebb

•1957: Rosenblatt presentò il *perceptron:* una rete in grado di riconoscere immagini. Viene dimostrato un teorema per la convergenza dell'apprendimento del perceptron. Il perceptron riusciva a riconoscere correttamente anche immagini mai viste prima (generalizzazione e capacità d'astrazione). Si ebbe un forte impulso in avanti nell'utilizzo e ricerca in questo settore.

•1969: Minsky e Papert analizzarono criticamente il perceptron evidenziandone i limiti e l'incapacità di risolvere problemi banali come lo XOR. Le ricerche subirono una forte battuta d'arresto

•anni '80: Rumelhart introdusse il "terzo strato" superando i limiti del perceptron. Si dimostrò formalmente come architetture di reti con almeno 3 strati (input, strato interno/i, output) siano in grado di computare qualsiasi funzione.

La j-esima unità (o processing element) riceve gli input da strati precedenti o direttamente dall'esterno (stimoli). Sulle connessioni sono presenti pesi sinaptici che denotano la "forza" della connessione (sinapsi). Questi segnali sono convogliati (dendridi), insieme alla soglia interna del neurone (bias), nel net_j (soma) del neurone artificiale. L'uscita del neurone (assone) è generata dalla funzione di attivazione $g(net_j)$. La figura 2 illustra il modello di neurone che adotteremo nel corso delle lezioni.

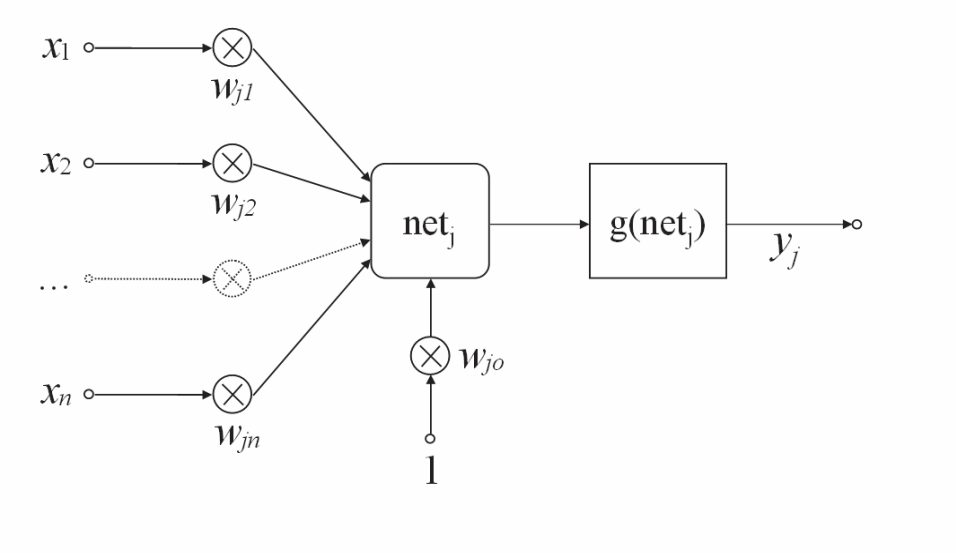


Figura 2 - Modello di un neurone

Quindi la variazione del potenziale di membrana postsinaptico si riflette nel net input dell'unità della rete neurale artificiale, che è tipicamente funzione degli ingressi:

$$net_j = w_{jo} + \sum_{i=1}^{n} w_{ji} x_i$$

ARCHITETTURE E LEGGI DI APPRENDIMENTO

E' possibile distinguere tra reti *completamente connesse*, in cui ogni neurone è connesso a tutti gli altri, e reti *stratificate* (multilivello) in cui i neuroni sono organizzati in strati. Una rete a tre strati è mostrata nella figura 3.

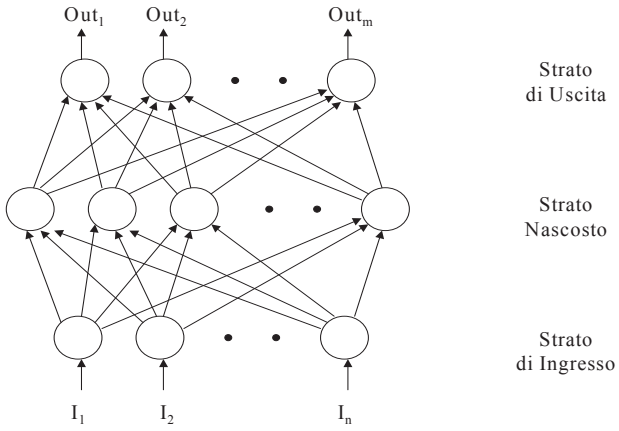


Figura 3 - Rete neurale a tre strati

Nelle reti multilivello tutti i neuroni di uno strato sono connessi con tutti i neuroni dello strato successivo, nel verso che va dall'ingresso all'uscita. Non esistono connessioni né tra neuroni di uno stesso strato né tra neuroni di strati non adiacenti. Come si vedrà, una rete con due soli strati non è in grado di discriminare un insieme di pattern se questo non è linearmente separabile[1]. Per superare questo problema si possono usare particolari funzioni di uscita non lineari oppure ricorrere ad architetture di rete a tre strati. Grazie ai neuroni nascosti si formano delle rappresentazioni interne degli ingressi che, opportunamente combinate, consentono il riconoscimento di forme più complesse. Si può dimostrare che l'aggiunta di ulteriori strati

[1] Una rete a due strati con *n* ingressi e *p* uscite è in grado di discriminare al massimo *p* classi diverse a condizione che ciascuna classe C_i sia *linearmente separabile*, ovvero esista un iperpiano in R^n che separi i punti che appartengono a C_i da quelli che non appartengono a C_i. Tipici problemi non separabili sono lo XOR, cioè riconoscere (1,0) e (0,1) da (0,0) e (1,1), la parità, cioè riconoscere vettori binari con numero pari di bit, e la verticalità, cioè riconoscere segmenti verticali da orizzontali.

nascosti non migliora le capacità discriminatorie della rete. La formazione di regioni decisionali complesse è possibile solo se la funzione di uscita dei neuroni è non lineare; infatti, una rete multistrato a neuroni lineari è sempre riconducibile ad una rete equivalente con due soli strati. L'attivazione dei neuroni può avvenire in modo *sincrono*, in cui tutti i neuroni cambiano stato simultaneamente, o in modo *asincrono*, in cui viene scelto un neurone per volta in modo sequenziale o casuale.

Per paradigma di apprendimento si intende quella metodologia con cui si addestra la rete di neuroni. Operata questa scelta si passa a decidere con quali regole si debbano variare i singoli pesi, cioè la legge di apprendimento. Possiamo suddividere i *paradigmi di apprendimento* in:

- Apprendimento supervisionato.
- Apprendimento non supervisionato.

L'apprendimento supervisionato, che segue lo schema della figura 4, prevede la presentazione di un set di esempi (*training set*), composto dalle coppie X_k e Y_{dk}, rispettivamente il k-esimo ingresso e la k-esima uscita desiderata. L'uscita reale, Y_k, viene confrontata con quella desiderata e si aggiustano i pesi in modo tale da minimizzare la loro differenza. Il training set viene ciclicamente presentato finché $Y_k \approx Y_{dk}$.

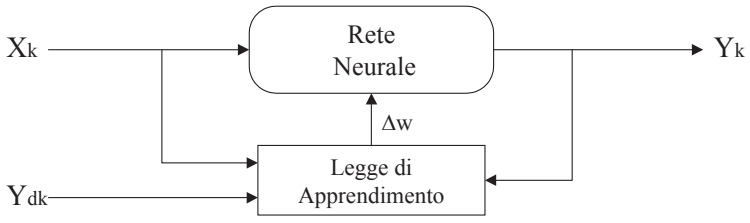


Figura 4 – Apprendimento supervisionato

Nell'apprendimento non supervisionato (schema generale nella figura 5) la rete aggiusta i pesi in modo autonomo. Vengono presentati dati da un *validation set* che include il training set.

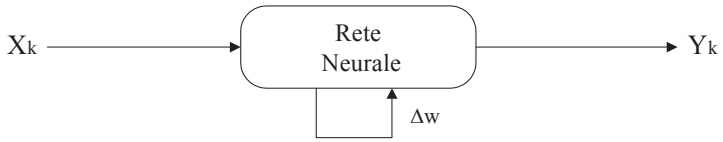


Figura 5 – Apprendimento non supervisionato

APPRENDIMENTO SUPERVISIONATO

Allo scopo di ottenere un modello in grado di classificare pattern in classi differenti è possibile adottare una strategia di apprendimento supervisionato. Nell'apprendimento supervisionato, che segue lo schema della figura 1, si addestrano i pesi cercando di insegnare alla rete come associare le coppie $(x_k;t_k)$, che indicano rispettivamente il k-esimo pattern in ingresso e la k-esima uscita desiderata, di un determinato insieme (*training set*). L'uscita reale, y_k, viene confrontata con quella desiderata e si aggiustano i pesi in funzione della legge di apprendimento scelta. Una volta presentate tutte le coppie del training set si considera che sia trascorsa un'epoca. Il training set viene ciclicamente presentato per epoche successive finché per ogni pattern si ha che $y_k \approx t_k$.

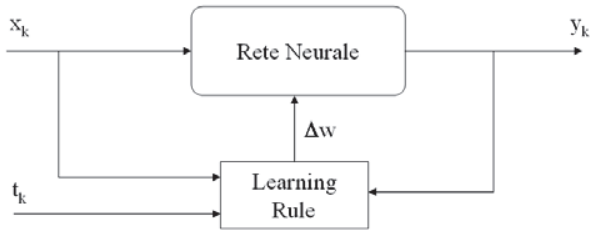


Figura 1 – Apprendimento supervisionato

Si opera in due fasi distinte:

- Fase di addestramento (anche nota come learning o training)
- Fase di generalizzazione (anche nota come recall)

Durante la fase di addestramento si cerca di far imparare alla rete le informazioni contenute nel training set; durante la fase di generalizzazione si usa il modello ottenuto per analizzare nuovi ingressi.

IL PERCEPTRON

Il Perceptron elementare coincide con la struttura del neurone formale; la funzione di attivazione è definita dalla funzione di Heaviside, nota come funzione a gradino (figura 2).

$$g(net_j) = \begin{cases} 1 & net_j \geq 0 \\ 0 & net_j < 0 \end{cases}$$

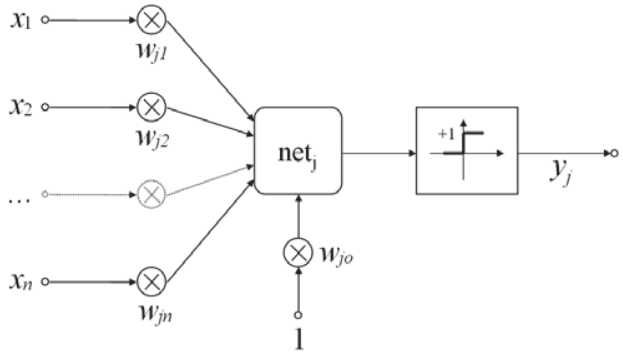


Figura 2 – Il Perceptron

Proprietà caratteristica del Perceptron è quella di dividere lo spazio di ingresso in due regioni mediante l'iperpiano in $\Re^n$ di equazione:

$$w_o + \sum_{i=1}^{n} w_i x_i = 0$$

Quindi è in grado solamente di risolvere problemi che sono *linearmente separabili*.
Nel caso di un Perceptron a due ingressi (n=2) l'iperpiano è una retta (figura 3).

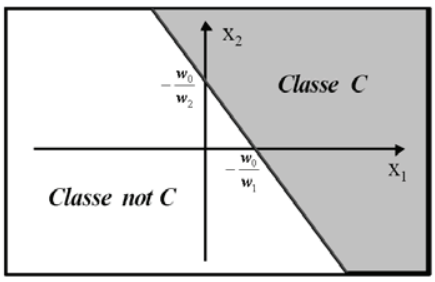


Figura 3 – Separazione lineare dello spazio di ingresso di un Perceptron

Il Perceptron quindi, avendo un'uscita binaria, può essere addestrato a riconoscere se un pattern in ingresso appartenga o no ad una determinata classe. Rosenblatt nel '57 riuscì ad addestrare un Perceptron a riconoscere se un'immagine fosse concava o convessa (figura 4).

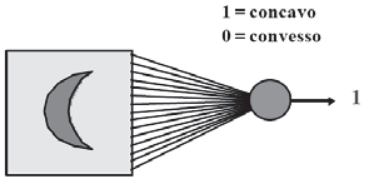


Figura 4 – Esperimento di Rosenblatt

Lo schema di addestramento adottato da Rosenblatt può essere schematizzato nella figura 5.

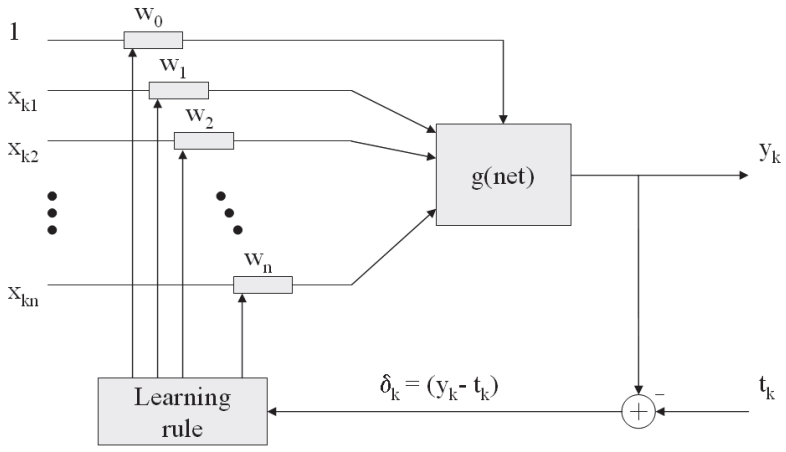


Figura 5 – Schema di addestramento supervisionato del Perceptron

Come possiamo trovare i valori appropriati dei pesi?, in altre parole la posizione dell'iperpiano ottimo? Attraverso un algoritmo di apprendimento supervisionato. La legge di apprendimento (Learning rule), cioè la regola con cui si variano i singoli pesi, è definita, in forma generale, dalla:

$$\overline{W}_{k+1} = \overline{W}_k + \Delta\overline{W}_k$$

Teorema di convergenza del Perceptron

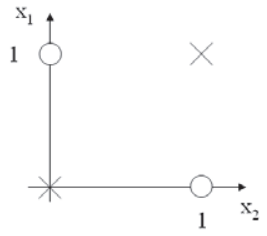
Se un Perceptron può risolvere il problema la procedura di addestramento convergerà ad una soluzione in un numero finito di passi.

In generale l'algoritmo di apprendimento è:

1. Inizializzazione casuale dei pesi
2. Presentazione di una coppia $(x_k;t_k)$
3. Calcolo della risposta y_k del Perceptron
4. Aggiornamento dei pesi con la legge di apprendimento prescelta
5. Si ripete il ciclo dal passo 2 finché tutte le risposte risultano giuste (con un errore tollerato)

LIMITI ED APPLICABILITA' DEL PERCEPTRON

L'algoritmo di apprendimento converge solo se il problema che gli esempi rappresentano è linearmente separabile. Nel caso che ciò non sia vero, l'algoritmo non converge. E' facile presentare esempi, anche semplici, di tale situazione. Il più noto è quello che si riferisce al problema booleano dello XOR (figura 6).



x_1	x_2	y
0	0	0
1	0	1
0	1	1
1	1	0

Figura 6 – Spazio degli ingressi e tabella di definizione della funzione XOR

La non separabilità la si può facilmente dedurre sia dalla figura 6 che cercando di risolvere il sistema che deriva dalla tabella di definizione:

$$w_o=0; \qquad w_1=1; \qquad w_2=1; \qquad w_1+w_2=0$$

l'ultima equazione è in contraddizione. Questo problema è facilmente risolubile aggiungendo un neurone intermedio tra l'ingresso ed il Perceptron (figura 7).

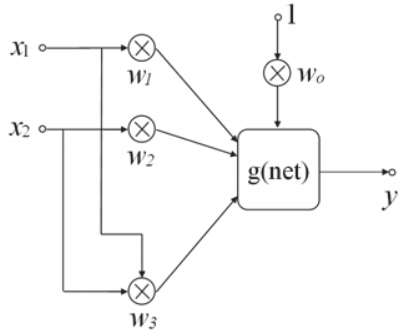


Figura 7 – Risoluzione del problema dello XOR

Con questa soluzione le equazioni diventano:

$$w_o=0; \qquad w_1=1; \qquad w_2=1; \qquad w_1+w_2+ w_3=0$$

e ne segue che w_3 dovrà essere pari a -2. Nello spazio la soluzione è rappresentata da un piano che separa i punti corrispondenti alle due classi (figura 8).

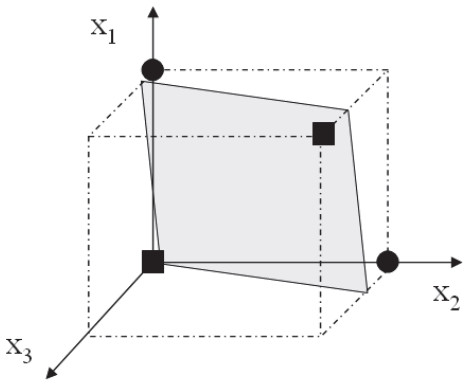


Figura 8 – Spazio degli ingressi nella risoluzione del problema dello XOR

L'uso di neuroni disposti tra l'ingresso e l'uscita in più strati porta all'architettura detta multilivello, che come si vedrà è in grado di risolvere qualsiasi problema di classificazione.

APPRENDIMENTO HEBBIANO

Nell'apprendimento di tipo hebbiano viene modificato ogni valore del peso di una connessione in proporzione al prodotto dei livelli di attivazione sull'altra parte della connessione stessa. Definendo:

$\overline{W}_k = [w_1, w_2, ..., w_n]^T$ vettore dei pesi al k-esimo ingresso

$\overline{x_k} = [x_1, x_2, ..., x_n]^T$ vettore degli k-esimo ingresso

si ha che:

$$\Delta \overline{W}_k = \eta \overline{x}_k t_k$$

Dove η è uno scalare noto come Learning rate.

L'algoritmo di addestramento è il seguente:

- se $y_k = t_k$ allora non cambiare i pesi
- se $y_k > t_k$ allora $\Delta\overline{W}_k = -\eta\overline{x}_k t_k$
- se $y_k < t_k$ allora $\Delta\overline{W}_k = \eta\overline{x}_k t_k$

REGOLA DELTA

Con regola delta si intende un apprendimento con semplice correzione dell'errore. Infatti definendo

$\overline{W}_k = [w_1, w_2, ..., w_n]^T$ vettore dei pesi al k-esimo ingresso

$\overline{x_k} = [x_1, x_2, ..., x_n]^T$ vettore degli k-esimo ingresso

$\delta_k = y_k - t_k$ DELTA

si ha che:

$$\Delta\overline{W}_k = \eta\overline{x}_k \delta_k$$

L'algoritmo di addestramento è il seguente:

- se $y_k = t_k$ allora non cambiare i pesi
- se $y_k \neq t_k$ allora $\Delta\overline{W}_k = \eta\overline{x}_k \delta_k$

METODO DI DISCESA DEL GRADIENTE

L'apprendimento in questo caso viene visto come una ricerca dei pesi delle connessioni che minimizzano una data funzione errore (figura 9).

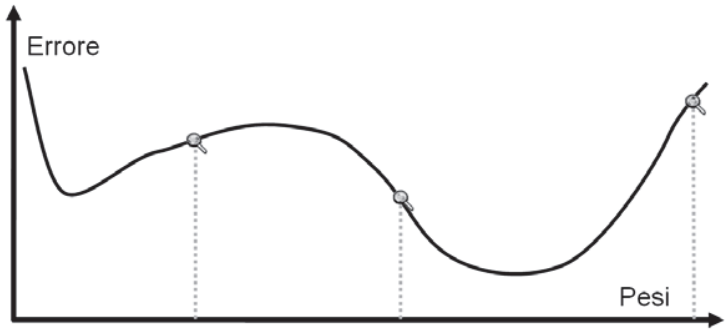


Figura 9 – Ipotetica funzione errore al variare dei pesi

La ricerca dei valori dei pesi migliori può essere guidata in modo proporzionale dall'informazione fornita dal gradiente locale della funzione errore (figura 10).

$$\Delta\overline{W} \quad \propto \quad \nabla_w E(\overline{W})$$

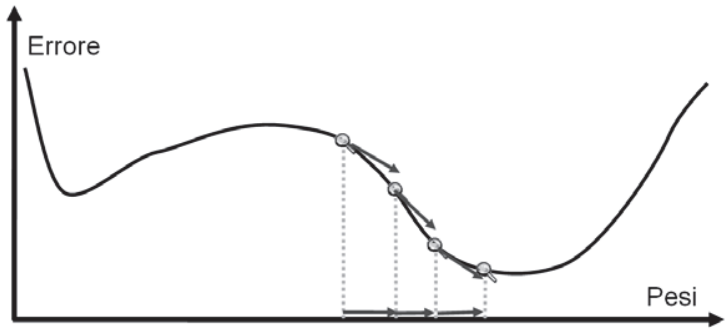


Figura 10 – Metodo della discesa del gradiente

Considerato che

$$\frac{dE(\overline{w})}{d\overline{w}} = \nabla_w E(\overline{w}) = \left(\frac{\partial E(\overline{w})}{\partial w_1}, \frac{\partial E(\overline{w})}{\partial w_2}, \ldots, \frac{\partial E(\overline{w})}{\partial w_n} \right)^T$$

Si avrà nella sua forma generale:

$$\Delta \overline{W}_k = \eta \frac{dE(\overline{w})}{d\overline{w}}$$

Si definisce *errore quadratico medio* di risposta al pattern k-esimo la quantità:

$$\mathrm{E}_k = \frac{1}{2}(y_k - t_k)^2$$

Per la k-esima coppia presentata si ha:

$$\begin{cases} y_k = g(net_k) \\ \delta_k = y_k - t_k \end{cases}$$

Consideriamo l'i-esimo peso al k-esimo pattern, si avrà che:

$$\Delta w_i = \eta \frac{\partial E_k}{dw_i}$$

Sviluppando questa quantità:

$$\frac{\partial E_k}{\partial w_i} = \frac{\partial}{\partial w_i}\left[\frac{1}{2}(y_k - t_k)^2\right] = (y_k - t_k)\frac{\partial (y_k - t_k)}{\partial w_i} =$$

$$= (y_k - t_k)\frac{\partial y_k}{\partial w_i} = (y_k - t_k)\frac{\partial g(net_k)}{\partial w_i} =$$

$$= (y_k - t_k)\frac{\partial g(net_k)}{\partial net_k}\frac{\partial net_k}{\partial w_i} = \delta_k g'(net_k)x_i$$

Quindi:

$$\Delta w_i = \eta \delta_k g'(net_k)x_i$$

L'algoritmo di addestramento è quello generale del Perceptron.

IL PERCEPTRON COME APPROSSIMATORE DI UNA FUNZIONE NO

Il Perceptron può essere utilizzato con successo nell'individuazione di una funzione a partire da esempi, se si ricorre ad un'espansione dello spazio di ingresso tramite un'opportuna pre-elaborazione. E' facile rendersi conto di tale possibilità considerando lo sviluppo di una funzione sia con il metodo di Kolmogorov-Gabor sia semplicemente facendo riferimento ad uno sviluppo ortogonale.

Nel caso del metodo di Kolmogorov-Gabor, una f(X) si può rappresentare tramite:

$$f(X) = a_o + \sum_{i=1}^{N} a_1(i)x_i + \sum_{i_1=1}^{N}\sum_{i_2=1}^{N} a_2(i_1,i_2)x_{i1}x_{i2} + \ldots + \sum_{i_1=1}^{N}\sum_{i_2=1}^{N}\ldots\sum_{i_k=1}^{N} a_k(i_1,i_2,\ldots i_k)x_{i1}x_{i2}\ldots x_{ik} + \ldots$$

Tale formula viene utilizzata, troncandola ad un opportuno ordine, con la conseguenza di ottenere come modello per l'implementazione della f(X) lo schema di figura 11, in cui i blocchi presenti hanno la funzione di effettuare la pre-elaborazione. Risulta evidente l'espansione notevole dello spazio di ingresso al Perceptron, in quanto accanto agli ingressi compaiono anche le loro combinazioni due a due, tre a tre e così via fino all'ordine previsto.

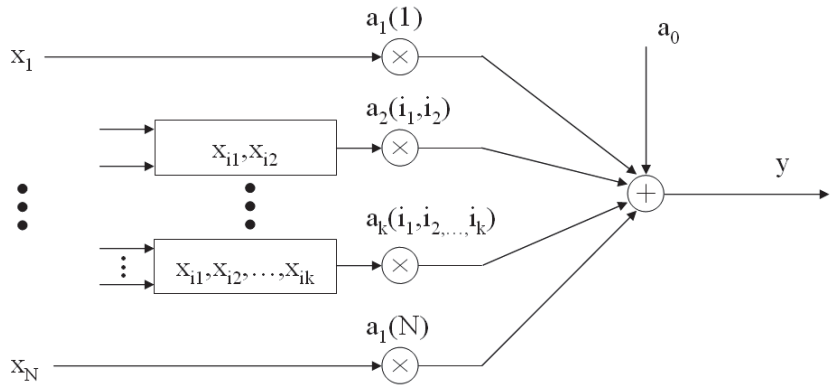


Figura 11 – Schema del Perceptron con espansione dello spazio di ingresso

L'espansione dello spazio di ingresso può essere ottenuta ricorrendo ad un qualsiasi sviluppo troncato di funzioni ortogonali. Si ottiene così lo schema di figura 12, in cui si indica con $g_i(X)$ la generica funzione base prescelta. Nella letteratura tecnica lo schema in questione viene indicato come Perceptron con "connessioni funzionali". E' evidente la possibilità di ottenere prestazioni notevoli in casi particolari rispetto al Perceptron iniziale.

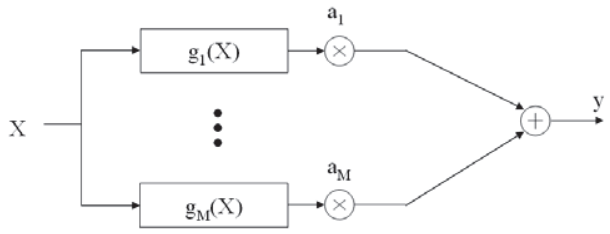


Figura 12 – Perceptron con connessioni funzionali

APPRENDIMENTO NON SUPERVISIONATO

Per poter imitare in modo efficiente il sistema nervoso è necessario avere un'idea della natura dei processi che hanno luogo in esso. Una ipotesi ragionevole che può essere fatta è che essi siano guidati da meccanismi tali da ottimizzare l'obiettivo sconosciuto che perseguono. Ogni processo evolve da una situazione iniziale associata ad uno stimolo fino ad una terminale in cui si ha una risposta, che costituisce il risultato del processo stesso. E' intuitivo che in tale evoluzione è presente un trasferimento di informazione. Infatti, lo stimolo fornisce l'informazione necessaria ad ottenere la risposta voluta. Perciò è importante che tale informazione si trasmetta il più fedelmente possibile fino alla conclusione del processo. Un criterio ragionevole per interpretare i processi che hanno luogo nel sistema nervoso è quindi quello di considerarli come trasferimenti di informazione con massima preservazione della stessa.

L'interpretazione precedente è suscettibile di applicazione diretta per la derivazione di algoritmi di apprendimento non-supervisionati. Si tratta cioè di utilizzare le tecniche della teoria dell'apprendimento per misurare la perdita dì informazione che si è avuta nel trasferimento. Si considera il processo in esame come la trasmissione di un segnale attraverso un canale rumoroso, utilizzando le tecniche ben note sviluppate nel campo delle comunicazioni. E' possibile, tuttavia, seguire un approccio diverso basato su una rappresentazione geometrica del processo. Infatti, sia lo stimolo che la risposta sono caratterizzati da un opportuno numero di componenti, che riportate in uno spazio corrispondono ad un punto. Quindi il processo può essere interpretato come una trasformazione geometrica dello spazio di ingresso a quello di uscita. Lo spazio di uscita ha una dimensione minore di quella dell'ingresso, in quanto lo stimolo contiene l'informazione necessaria ad attivare molti processi simultanei. Rispetto ad uno solo esso è ridondante. Ciò significa che nella trasformazione in esame è sempre presente un'operazione di riduzione di ridondanza.

Negli spazi di ingresso e di uscita si formano regioni tipiche a cui risulta associata l'informazione. Il meccanismo naturale che controlla il trasferimento di informazione deve quindi, in qualche modo, individuare queste regioni importanti per il processo considerato e fare in modo che si corrispondano nella trasformazione. E' perciò presente nel processo in esame un'operazione di raggruppamento dei dati; tale operazione può essere identificata con l'acquisizione di esperienza. Le due operazioni precedenti di raggruppamento e riduzione di ridondanza sono tipiche del trattamento ottimale dei segnali e si ha evidenza biologica della loro esistenza nel funzionamento del sistema nervoso. E' interessante inoltre notare che tali due operazioni conseguono automaticamente nel caso dell'apprendimento non-supervisionato basato su principi sperimentali, quali l'apprendimento competitivo. Come si vedrà la rete neurale di Kohonen, caratterizzata da tale comportamento, effettua automaticamente le operazioni di raggruppamento e riduzione di ridondanza.

BASI BIOLOGICHESPUNTI

Come abbiamo visto, l'attività della corteccia che si organizza sotto l'azione di uno stimolo può essere caratterizzata dalla presenza di attività di gruppi neurali.
Nel 1983, Teuvo Kohonen osservò questa attività in termini di emissione di numero di impulsi nell'unità di tempo superiore ad una certa soglia in gruppi di neuroni. Osservò come la demarcazione spaziale tra gruppi fosse netta, per cui parlò di formazione di "bolle" di attività. La formazione della bolla sotto l'azione dello stimolo avviene partendo da una situazione indifferenziata di bassa attività di tutti i neuroni presenti. A partire da questa situazione, in presenza di piccole fluttuazioni, comincia a formarsi una bolla che poi evolve rapidamente esaltando le piccole differenze iniziali. Una volta formata, la bolla persiste per un tempo lungo se comparato con quello richiesto per la sua formazione e quindi si và restringendo lentamente fino a scomparire. Per quanto riguarda le sinapsi, la loro variazione è lenta se confrontata con i tempi di formazione della bolla ed avviene in tempi analoghi a quelli della persistenza della bolla di attività. Osservò inoltre come l'evoluzione dei suddetti fenomeni può essere modellata in modo soddisfacente mediante equazioni differenziali non-lineari. In particolare dedusse come il fenomeno caratteristico della bolla sia dovuto ad un'azione collettiva dei neuroni,

considerando l'equazione dell'attività di un neurone in presenza ed assenza dell'azione degli altri neuroni.

Nel caso di assenza degli altri neuroni, l'attività del neurone j-esimo può essere descritta dalla:

$$\frac{d\eta_j}{dt} = \sum_{i=1}^{n} \mu_{ji}\xi_i - \gamma(\eta_j)$$

in cui:

η_j attività del neurone i-esimo;

ε_i componente i-esima dello stimolo, con n il numero complessivo di ingressi;

μ_{ji} è il peso della connessione tra il neurone j-esimo e l'ingresso i-esimo (sinapsi);

γ è un termine che tiene conto delle perdite nel processo di trasferimento dell'informazione.

Si può dimostrare dalla soluzione dell'equazione che soluzioni simili non presentano alcuna traccia del fenomeno della bolla di attività. Affinché sia presente tale fenomeno occorre considerare il loro comportamento collettivamente, cioè quando interagiscono tra loro. A questo scopo supponiamo che ciascun neurone sia collegato lateralmente con quelli vicini con sinapsi eccitatorie di ampiezza decrescente con la distanza e successivamente inibitorie. Queste sinapsi laterali sono fisse e uguali per tutti i neuroni. L'andamento delle sinapsi con la distanza dal neurone esaminato ha l'andamento tipico di figura 1 (a cappello messicano):

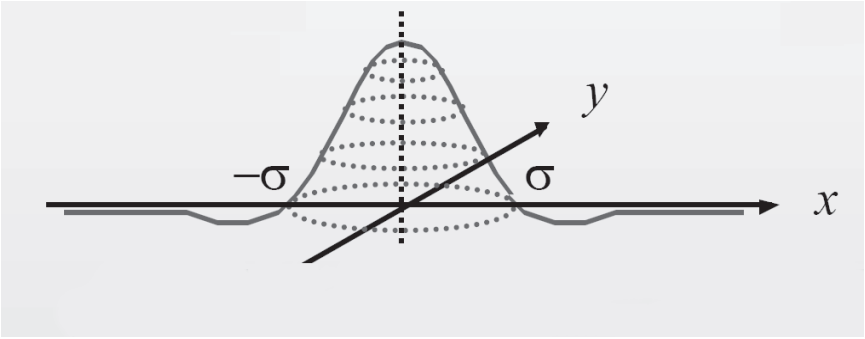


Figura 1 – Andamento delle sinapsi laterali in funzione della distanza dal neurone

L'andamento può essere descritto tramite il laplaciano di gaussiana:

$$\nabla^2 G = \frac{4}{\sigma^2}\left(\frac{x^2 + y^2}{\sigma^2} - 1\right) e^{\frac{x^2+y^2}{\sigma^2}}$$

Tenendo conto di esse, l'equazione dell'attività del neurone j-esimo si modifica come segue:

$$\frac{d\eta_j}{dt} = \sum_{i=1}^{n} \mu_{ji}\xi_i - \gamma(\eta_j) + \sum_{k \neq j} w_{kj}\eta_k$$

in cui w_{kj} è la sinapsi della connessione tra i neuroni *k* e *j*. Si può dimostrare che questo sistema di equazioni evidenzia il fenomeno della bolla. Si può quindi osservare che grazie all'inibizione laterale i neuroni competono per rispondere ad uno stimolo. Un particolare neurone, ad esempio quello con uscita maggiore, vince la competizione (apprendimento competitivo) e si specializza a riconoscere lo stimolo. Grazie alle connessioni eccitatorie quindi, i neuroni vicini al vincitore risultano così sensibili ad ingressi simili creando un isomorfismo tra spazio di ingresso e spazio di uscita.

L'informazione presente nello stimolo viene trasferita alle sinapsi la cui variazione è regolata da un'equazione differenziale di tipo hebbiano, temperata dalla presenza di un fattore di dimenticanza, cioè:

$$\boxed{\frac{d\mu_{ji}}{dt} = \alpha\,\eta_j\xi_i - \beta\,(\eta_j)\,\mu_{ji}}$$

in cui α controlla la velocità dell'apprendimento e $\beta(\eta_j)$ la dimenticanza. Si vede chiaramente che l'evoluzione temporale delle sinapsi dipende dall'attività dei neuroni della connessione relativa. Poiché questa attività è molto diversa a seconda che il neurone sia oppure non sia contenuto nella bolla di attività, occorre distinguere i due casi seguenti:

1) neurone entro la bolla: in questo caso il neurone ha attività massima, che può essere normalizzata ad uno, cioè $\eta_j \sim 1$. Normalizzando anche le altre grandezze presenti in modo che si abbia $\alpha\eta_j \sim \beta(\eta_j)$ risulta:

$$\frac{d\mu_{ji}}{dt} = \alpha\left(\xi_i - \mu_{ji}\right)$$

Da questa equazione si deduce che la sinapsi evolve cercando di uguagliare l'ingresso relativo.

2) neurone fuori della bolla: i neuroni non contenuti entro la bolla hanno un'attività trascurabile, cioè $\eta_j = 0$. Quindi le sinapsi di questi neuroni non vengono modificate:

$$\frac{d\mu_{ji}}{dt} = 0$$

RETI DI KOHONEN

Teuvo Kohonen riuscì a costruire un modello neurale in grado di replicare il processo di formazione delle bolle. In particolare, egli usò una rete stratificata con apprendimento senza supervisione basato sulla competizione fra neuroni. L'architettura della rete (figura 2) è una griglia rettangolare di unità, nel caso usuale bidimensionale, collegate a tutti gli ingressi.

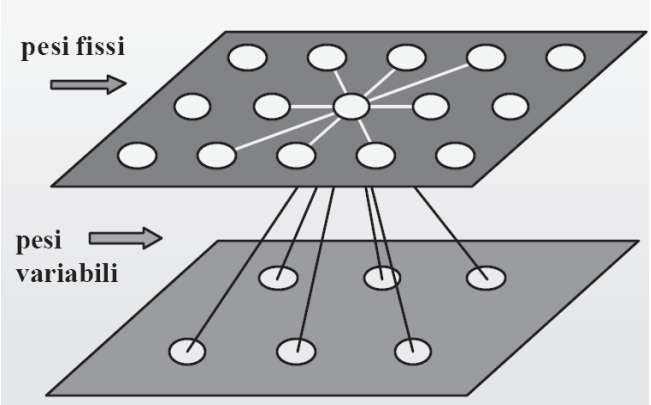


Figura 2 – Architettura della rete di Kohonen

Le unità della rete sono lineari (figura 3):

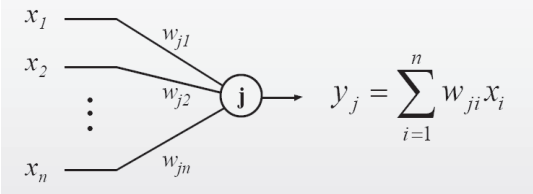
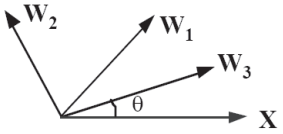


Figura 3 – Unità lineare della rete di Kohonen

Indichiamo con w_{ji} il peso della connessione tra il neurone j e l'ingresso i. Supponiamo che gli ingressi siano N ed i neuroni M. In totale le connessioni e quindi i pesi sono in numero di N^M. In realtà, per problemi di efficienza, i neuroni di uscita non vengono connessi fra di loro.

Il neurone che vince la competizione (neurone vincente) viene scelto con una strategia globale confrontando le uscite di tutti i neuroni. II fenomeno tipico della bolla, con la conseguente modifica solo dei neuroni contenuti in essa viene imitato mediante l'introduzione della nozione di "vicinato" di un neurone.

E' possibile selezionare il neurone con uscita massima oppure il neurone il cui vettore dei pesi è più simile all'ingresso presentato. Nel primo caso si ha che:

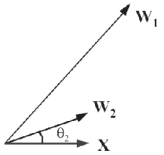


$$y_j = \sum_{i=1}^{n} w_{ji} x_i = W_j \bullet X = |W_j| \, |X| \cos\theta$$

Si osservi come sia necessario normalizzare i vettori:

$$|\mathbf{X}| = |\mathbf{W}| = 1$$

altrimenti può accadere che:

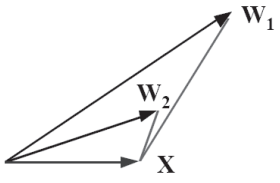


$$|W_1||X| \cos\theta_1 > |W_2||X| \cos\theta_2$$

nonostante:

$$\theta_2 < \theta_1$$

Nel secondo caso il neurone vincente sull'ingresso X è quello che ha il vettore dei pesi più simile ad X stesso. Indicando con j_o il neurone vincente si ha:



$$j_0 : \; DIS(W_{j0}, X) < DIS(W_j, X) \quad \forall j \neq j_0$$

in questo caso non occorre che i vettori siano normalizzati. Come distanza si può considerare la distanza euclidea; per il neurone j-esimo:

$$DIS(W_J, X) = \sqrt{\sum_{i=1}^{N} \left(x_i - w_{ji} \right)^2}$$

Cambiando il tipo di distanza si ha una variazione modesta del risultato.

La legge di apprendimento, con la quale si aggiornano le sinapsi del neurone vincente e, come si vedrà, dei neuroni del suo "vicinato", è la seguente:

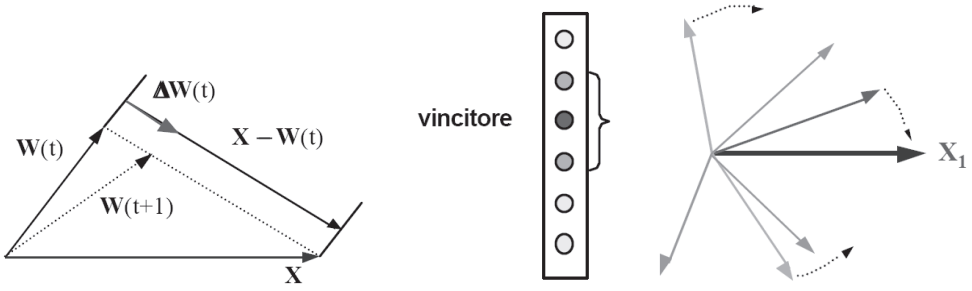
$$w_{ji}(k) = w_{ji}(k-1) + \alpha(k)\left[x_i(k) - w_{ji}(k-1)\right]$$

$$\forall\ j \subseteq V_{jo}(k) \quad \text{con} \quad i = 1,2,\ldots N$$

$V_{jo}(k)$ indica il vicinato del neurone vincente j_o in corrispondenza dell'iterazione corrente K.
Considerando la notazione vettoriale la legge di apprendimento può essere scritta:

$$\Delta W(t) = \alpha\,(X - W)$$

per cui:



Come illustrato in figura 4, il vicinato di un neurone può essere definito come l'insieme di neuroni aventi una distanza dal vincitore minore di un certo Raggio di Interazione (R).

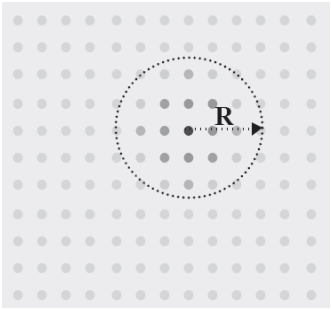


Figura 4 – Raggio di interazione

Il vicinato viene generalmente scelto in modo da imitare nel modo migliore la forma circolare caratteristica del contesto biologico. In particolare possiamo evidenziare alcune di queste scelte (figura 5).

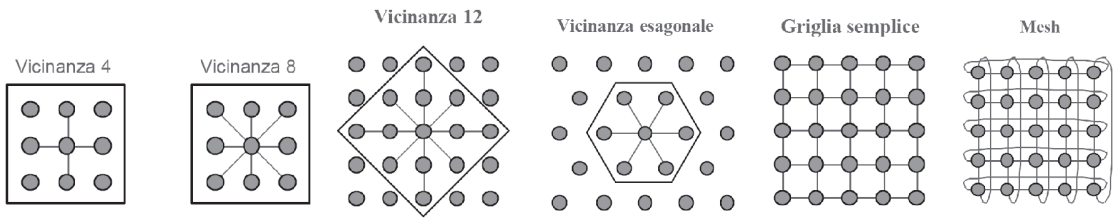


Figura 5 – Alcune tipologie di vicinato

La legge di apprendimento può essere applicata indistintamente a tutti i neuroni del vicinato, come si evince dalla formula, oppure in modo diverso in funzione della distanza che il neurone j-esimo ha rispetto al neurone vincitore. In questo caso possiamo considerare una funzione che pesa la distanza dei neuroni dal neurone vincente (figura 6).

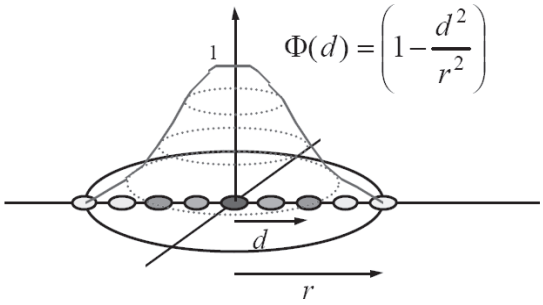


Figura 6 – Funzione che pesa la distanza dei neuroni dal neurone vincente

La legge di apprendimento in questo caso potrà essere espressa dalla:

$$w_{ji}(k) = w_{ji}(k-1) + \alpha(k)\Phi(d)\left[x_i(k) - w_{ji}(k-1)\right]$$

$$\forall\ j \subseteq V_{jo}(k) \quad \text{con} \quad i = 1,2,\ldots N \ \text{e}\ d = DIS(j, j_o)$$

La scelta del tipo di vicinato non ha molta influenza; ciò che è importante è farla variare nel tempo avendo cura di includere tutti i neuroni della rete nel vicinato ed alla fine ridurre il vicinato al solo neurone vincente. Ciò tiene conto della riduzione dell'estensione della bolla nel tessuto biologico.

Anche il fattore di convergenza deve essere variato nel tempo. Generalmente lo si mantiene costante (circa pari ad 1) per un certo numero di iterazioni, quindi lo si fa decrescere fino a circa 0.1 (a volte anche fino a zero).

La forma con cui avviene la loro riduzione progressiva al passare del tempo, cioè al crescere del numero delle iterazioni del processo iterativo di apprendimento, non ha molta influenza. In generale, nel caso usuale bidimensionale si ha che (figura 7):

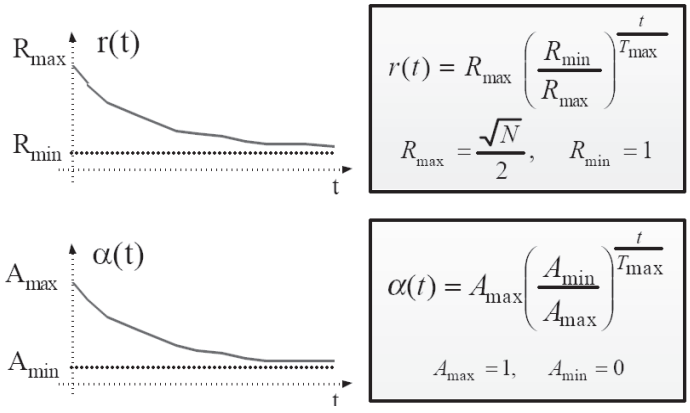


Figura 7 – Riduzione progressiva al passare del tempo di *r(t)* e di $\alpha(t)$

Nel caso di una rete avente dimensioni diverse da due, ciò che si modifica è solo la forma del vicinato che diviene un segmento nel caso monodimensionale ed un parallelepipedo nel caso tridimensionale. L'uso di un numero di dimensioni maggiore di due è tuttavia raro.

Queste considerazioni chiariscono l'algoritmo di apprendimento della rete di Kohonen, che riportiamo di seguito. L'algoritmo è iterativo ed il numero delle iterazioni dipende dal problema e dalla complessità della rete, misurata dal numero M di neuroni. Usualmente il numero di iterazioni da utilizzare viene scelto nell'intervallo (500-5000)M.

1. Si inizializzano i pesi in modo casuale;
2. Si inizializzano i parametri $\alpha=A_{max}$ e $r=R_{max}$;
3. DO {
4. FOR EACH x_k del TS {
5. Si calcolano tutte le uscite y_j;
6. Si determina il neurone vincitore j_o;
7. Si aggiornano i pesi del vicinato;
8. }
9. Si riducono α e r;
10. } WHILE ($\alpha > A_{min}$)

Alla fine di questo processo, nel caso in cui A_{min} sia diverso da zero, si può anche effettuare una rifinitura in cui vengono corrette solo le sinapsi del neurone vincente.
Alla fine del processo di apprendimento, dopo aver congelato le sinapsi, la rete di Kohonen viene utilizzata inviando in ingresso un vettore X e prendendo come risposta o la posizione o le sinapsi del vettore vincente. Nel primo caso essa può essere considerata come un dispositivo per effettuare una trasformazione da uno spazio N-dimensionale a quello di uscita, costituito dalla griglia dei neuroni. Lo spazio di uscita è discretizzato in quanto sono presenti solo M punti. Inoltre, esso ha un numero di dimensioni molto minore di quello di ingresso. Perciò viene attuata una riduzione di ridondanza. La trasformazione ingresso-uscita determinata dalla rete di Kohonen ha molte proprietà, la più importante è quella dell'ordinamento da cui si origina anche il suo nome inglese: SOFM (self-organizing feature mapping). Infine quando la rete di Kohonen viene usata facendo riferimento al vettore delle sinapsi del neurone vincente, essa implementa un codificatore vettoriale.

La rete neurale di Kohonen quindi, dopo apprendimento, è in grado di svolgere alcuni compiti strettamente legati alle sue proprietà. Di seguito accenniamo a tali proprietà, precisando il ruolo che esse svolgono nell'utilizzazione della rete.

Proprietà 1 "codifica dello spazio di ingresso"
Lo stimolo X, che viene inviato in ingresso alla rete neurale, è rappresentato da un punto nello spazio di ingresso N-dimensionale. In tale spazio esistono regioni caratteristiche del contesto in esame a cui è associata l'informazione del processo che si sta considerando. Tali regioni giocano un ruolo importante nell'elaborazione dei segnali ed esistono algoritmi che cercano di individuarle. Una delle applicazioni più importanti di questi algoritmi è la codifica, che consiste nell'associare alle suddette regioni un punto particolare detto "centroide" che costituisce il valore rappresentativo (prototipo) dell'intera regione a cui si riferisce. L'obiettivo di un codificatore è quello di ridurre al minimo l'errore che si commette sostituendo ad X il relativo prototipo, calcolato su tutto lo spazio e tenendo conto della densità di probabilità di X. La rete di Kohonen funziona come un codificatore. Infatti, ciascun neurone è collegato all'ingresso e le relative sinapsi rappresentano un punto dello spazio di ingresso. A tale neurone corrisponde

inoltre una regione dello spazio di ingresso, quella degli ingressi X che rendono vincente quel neurone. Perciò, la rete suddivide lo spazio in regioni, ognuna associata ad un neurone, con centroide coincidente con il vettore delle sue sinapsi. II funzionamento come codificatore della rete di Kohonen richiede le seguenti operazioni:

1) inviare l'ingresso X;
2) determinare il neurone vincente;
3) considerare come risultato della codifica il vettore delle sinapsi del neurone vincente.

Proprietà 2 "ordinamento"

Lo spazio discretizzato di uscita della rete di Kohonen è caratterizzato dal corrispondere ad eventuali variabili implicite da cui dipendono i dati di ingresso. Consideriamo dapprima il caso di dati dipendenti da due variabili implicite δ e ε e di considerare una rete bidimensionale. Si può dimostrare che dopo l'apprendimento i neuroni della griglia rettangolare corrispondono nelle due direzioni ortogonali alle due variabili ed in modo ordinato. Quindi è come se la griglia rappresentasse un piano coordinato avente come ascissa ed ordinata rispettivamente δ e ε. Usualmente, non è controllabile a priori il verso positivo di tali assi ne l'uniformità della scala. Tale uniformità dipende dall'importanza relativa delle due variabili e dalla rappresentatività dei dati del TS rispetto ai vari valori delle due variabili. Nel caso che le variabili siano in numero maggiore di due, sempre per una rete bidimensionale, l'ordinamento avviene rispetto alle due variabili implicite che influiscono di più. L'ordinamento finale risulta allora tanto più perturbato quanto maggiore è l'influenza delle altre variabili meno influenti.

Quando viene sfruttata questa proprietà, la rete di Kohonen si presenta come una mappa divisa in regioni a ognuna delle quali è associata una particolare informazione. Tipiche sono le mappe fonotopiche che si ottengono quando il vettore di ingresso corrisponde ai dati estratti mediante un'opportuna pre-elaborazione da un segmento di segnale vocale. In questo caso la griglia della rete si suddivide in zone costituite da gruppi di neuroni che corrispondono ai fonemi (di qui il nome di mappe fonotopiche). E' interessante mettere in evidenza che tali mappe hanno un riscontro biologico: nel cervello umano sono presenti gruppi di neuroni

specializzati nel riconoscere i fonemi. Sono possibili anche tipi diversi dì mappe in grado di riconoscere caratteristiche implicite nascoste.

Per consentire la formazione di mappe occorre definire una topologia sullo strato di uscita. Ogni neurone deve avere una posizione identificata da un vettore di coordinate. La mappa di uscita viene di solito definita come uno spazio ad una o a due dimensioni.

Consideriamo un esempio in cui tentiamo di addestrare una rete bidimensionale mediante un training set (TS) costituito da vettori aventi come componenti le coordinate dei punti di un piano (figura 8).

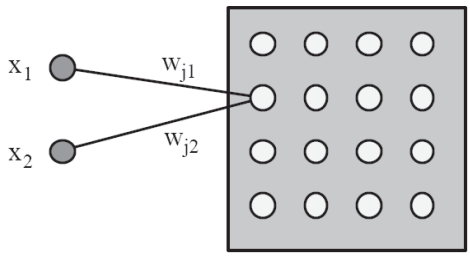


Figura 8 – Rete bidimensionale con due ingressi

Si può così descrivere su un piano coordinato i punti del TS ed i punti che hanno come coordinate le sinapsi di un neurone. Si congiungono quindi i punti che corrispondono a neuroni contigui. Si ottengono curve che godono della seguente proprietà: è minima la distanza tra un punto di essa ed il punto iniziale corrispondente. Curve di questo tipo sono note nella geometria sotto il nome di "curve di Peano". Una mappa di Kohonen genera automaticamente tali curve.

Considerando un esempio in cui il numero di neuroni sia pari al numero degli ingressi, nella figura 9a possiamo evidenziare un possibile stato iniziale casuale, mentre nella figura 9b il risultato dell'addestramento.

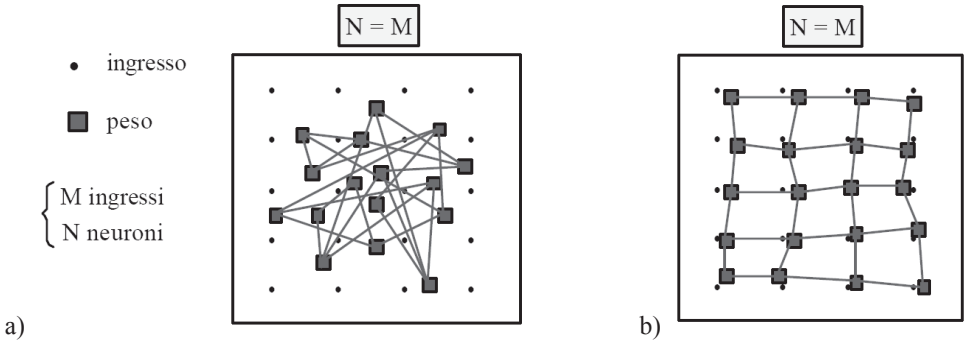


Figura 9 – Esempio di rete bidimensionale con neuroni pari al numero degli ingressi

Considerando il caso in cui i neuroni siano in numero maggiore rispetto agli ingressi, un possibile risultato dell'addestramento è illustrato in figura 10.

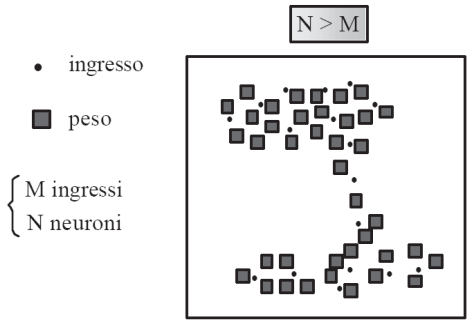


Figura 10 – Esempio di rete bidimensionale con N>M

Considerando invece il caso in cui i neuroni siano in numero minore rispetto agli ingressi, un possibile risultato dell'addestramento è illustrato in figura 11.

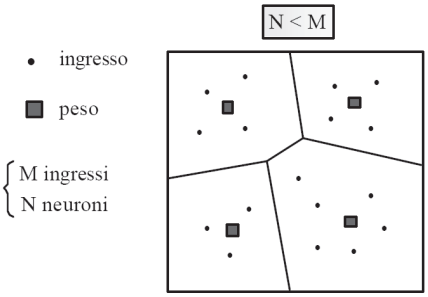


Figura 11 – Esempio di rete bidimensionale con N<M

In questo caso è possibile notare come ogni neurone possa essere considerato rappresentativo di una regione dello spazio, che risulta diviso secondo la "tassellazione di Voronoi". Quest'ultima è una suddivisione di uno spazio S in *n* sottospazi S_i tali che:

$$\mathbf{W_i} = \text{centroide di } S_i$$

Per cui lo spazio in figura 11 risulterà essere (figura 12):

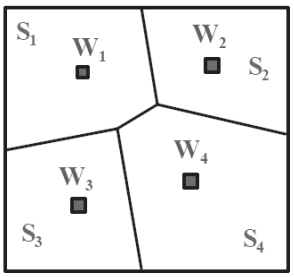


Figura 12 – Tassellazione di Voronoi

spazio per il quale varranno le seguenti proprietà:

$$\begin{cases} \bigcup_{i=1}^{n} S_i = S \\ S_i \cap S_j = \phi \quad \forall i \neq j \\ DIST(X_k, W_i) < DIST(X_k, W_j) \quad \forall i \neq j, X_k \in S_i \end{cases}$$

Le applicazioni di queste reti sono molteplici, fra le quali:

Classificazione

Esempio: classificare come originati da diversi classi di odoranti un insieme di segnali acquisiti attraverso un sistema olfattivo artificiale.

Clustering

Esempio: raggruppare un insieme enorme di dati in un numero limitato di sottoinsiemi.

Compressione

Esempio: convertire un'immagine con milioni di colori in un immagine compressa su 256 livelli

Ulteriori esempi sono la classificazione di fonemi (figura 13).

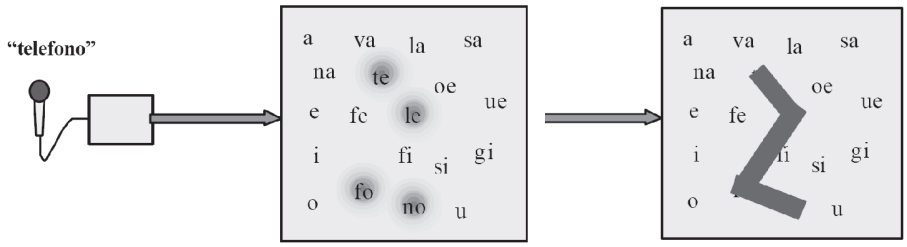


Figura 13 – Classificazione di fonemi con una mappa di Kohonen

Nel caso di uso combinato di una mappa di Kohonen e di una rete multilivello si può riuscire a riconoscere le parole di una conversazione (figura 14).

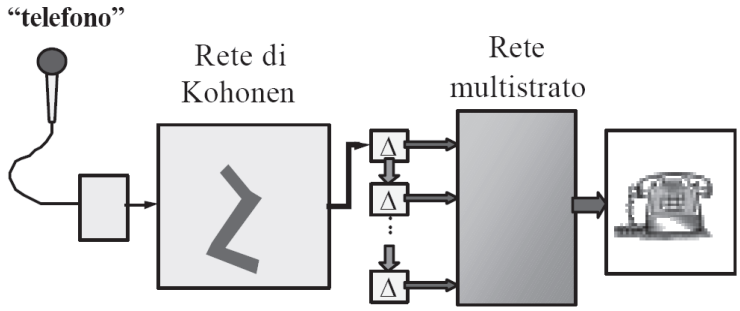


Figura 14 – Classificazione di parole con un uso combinato di mappa di Kohonen e rete multilivello

Presso il Centro Interdipartimentale di Ricerca "E. Piaggio" è stata sviluppata una rete gerarchica composta da quattro mappe di Kohonen ed un MLP in grado di riconoscere le espressioni del volto (figura 15).

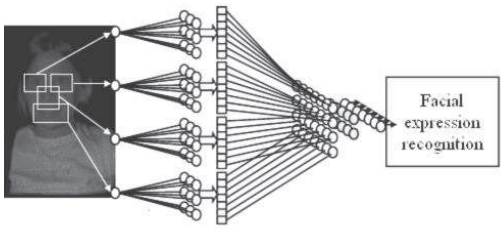


Figura 15 – Schema della rete neurale gerarchica in grado di riconoscere le espressioni facciali

APPLICAZIONI DELLE RETI MULTILIVELLO

GENERALIZZAZIONE

Con generalizzazione si intende la capacità della rete di riconoscere ingressi leggermente diversi da quelli con cui è stata addestrata. Per valutare le capacità della rete di generalizzare gli esempi del TS, si definisce un ulteriore insieme di esempi, il Validation Set (VS). Un primo metro di valutazione del risultato ottenuto si può ottenere, da un punto di vista generale, calcolando l'errore sul TS (E_{TS}) e successivamente l'errore sul VS (E_{VS}).
Per una rete multilivello si può affermare che il numero di parametri della rete dipende dal numero di neuroni nascosti della rete. Pochi neuroni nascosti potrebbero non essere sufficienti a ridurre l'errore globale (figura 1).

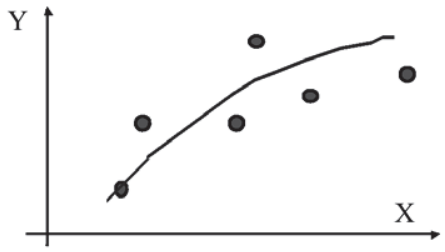


Figura 1 – Errore alto sul training set

Troppi neuroni nascosti potrebbero fossilizzare eccessivamente la rete sugli esempi specifici del TS. La rete risponderebbe bene sul TS, ma l'errore sarebbe elevato su altri esempi (). Questo fenomeno viene indicato come *overfitting* (figura 2).

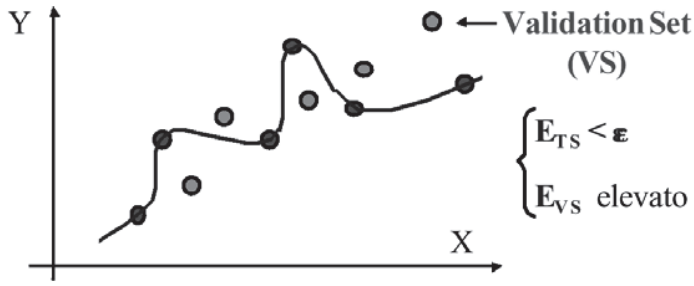


Figura 2 – Fenomeno dell'overfitting

Per migliorare la capacità di generalizzazione si può addestrare la rete sul TS, monitorare l'errore sul VS e fermare l'apprendimento quando $E_{VS} < \varepsilon_{VS}$ (figura 3).

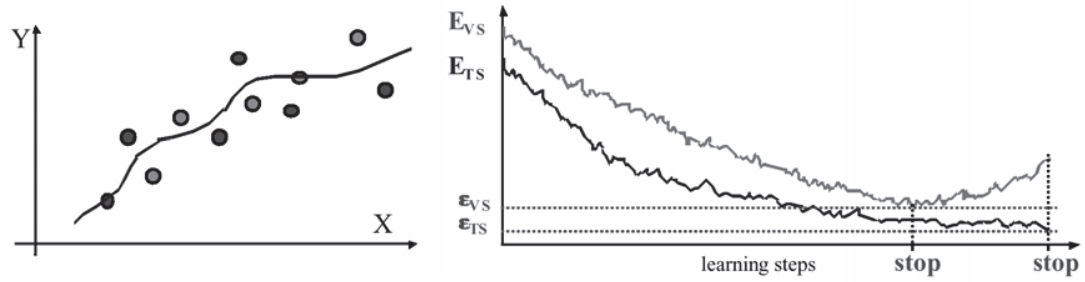


Figura 3 – Valutazione dell'errore sul VS durante l'addestramento del TS

APPLICAZIONI

-Una delle tipiche applicazioni per una rete multilivello è quello del riconoscimento ottico di caratteri, noto come Optical Character Recognition (OCR) (figura 4).

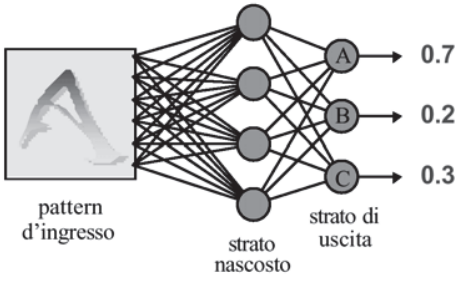


Figura 4 – Optical Character Recognition

Ad esempio, per insegnare ad una rete multilivello a riconoscere una cifre alfanumerica scritta a mano, si potrebbe costruire un training set di immagini. Ogni immagine potrebbe essere una mappa binaria 15x20 rappresentante una cifra scritta a mano (figura 5). Un insieme adeguato potrebbe essere costituito da 100 immagini diverse, 10 per ogni cifra.

Figura 5 – Esempio di pattern in ingresso per il problema dell'OCR

Il numero di neuroni di ingresso alla rete potrebbe essere pari al numero di pixel della mappa binaria di ingresso, cioè il valore di ogni pixel è un ingresso diretto alla rete, oppure si potrebbe ricorrere ad un preprocessing limitando così il numero di neuroni di ingresso. In quest'ultimo caso si potrebbe suddividere l'immagine in sottoaree il cui valore è l'ingresso per un singolo neurone. Il valore di una sottoarea si può ottenere con una semplice operazione quale il valore medio dei pixel. Le unità di uscita saranno 10, ognuna indicante un valore simbolico per la cifra riconosciuta.

- Un'altra applicazione è quella di imparare a prevedere il valore che un segnale (non casuale) avrà nel futuro sulla base del valore corrente (figura 6).

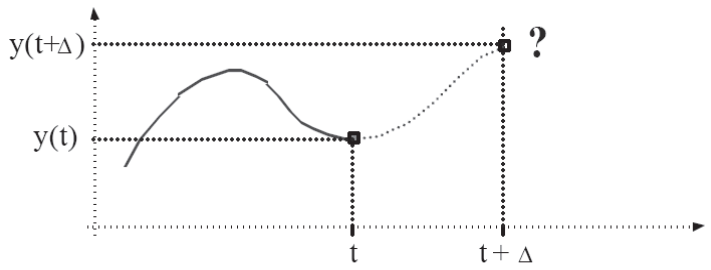


Figura 6 – Previsione di segnali

Si procede cercando di addestrare la rete sulla base dei valori passati (figura 7).

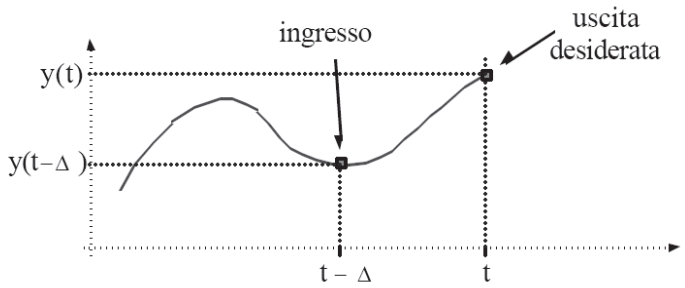


Figura 7 – Previsione di segnali sulla base dei valori passati

Durante la fase di addestramento, la rete impara ad associare al valore y(t-Δ) il valore y(t) (figura 8).

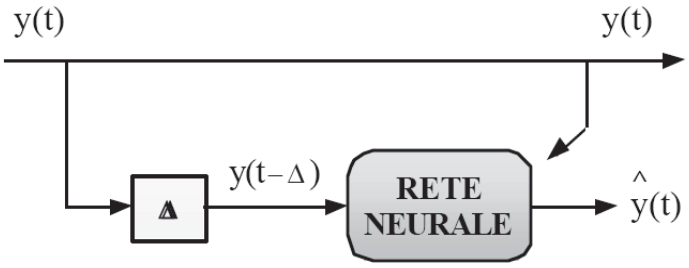


Figura 8 – Addestramento sui valori passati

Durante la fase di generalizzazione ogni volta che si presenta il valore y(t), la rete risponde con una stima di y(t+Δ) (figura 9).

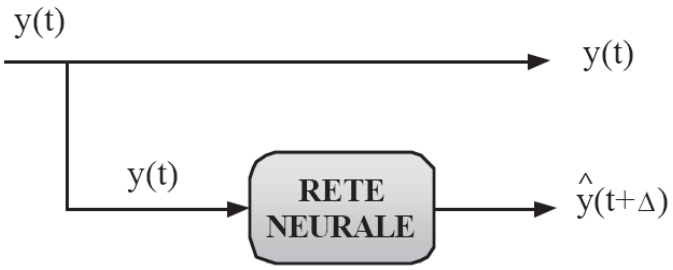


Figura 9 – Fase di generalizzazione

Per esempio in figura 10 è mostrato un segnale di esempio ed il corrispondente training set.

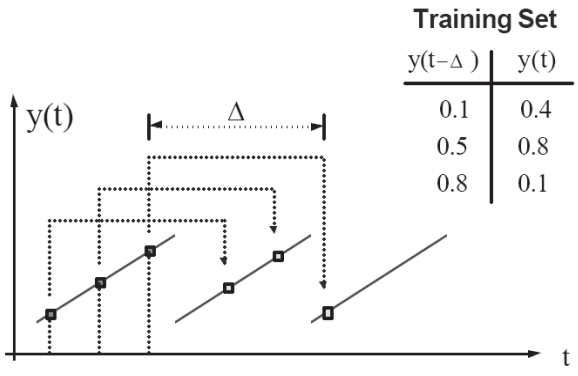


Figura 10 –Esempio di training set

Con alcuni segnali però si potrebbe verificare un training set inconsistente (figura 11).

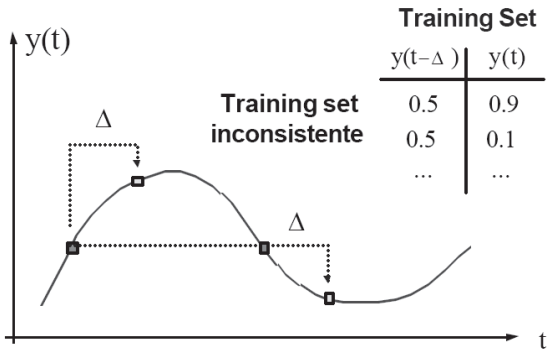


Figura 11 – Training set inconsistente

Usando però due intervalli diversi si potrebbe risolvere il problema (figura 12).

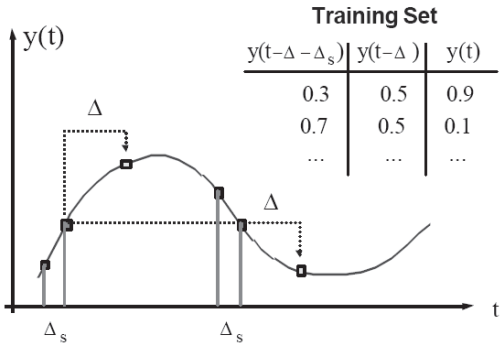


Figura 12 – Risoluzione del training set inconsistente

In questo caso lo schema di addestramento può essere schematizzato nella figura 13, in cui la rete viene addestrata con due ritardi distinti y(t-Δ-Δs) e y(t-Δ).

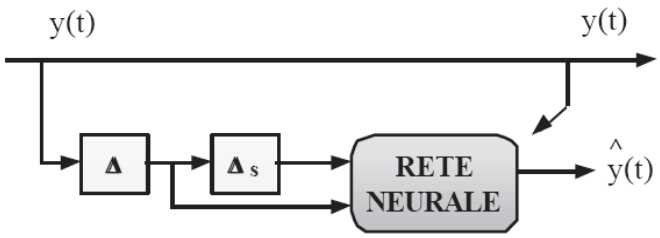


Figura 13 – Schema di addestramento con due ritardi distinti

In generale possiamo considerare la serie storica di figura 14.

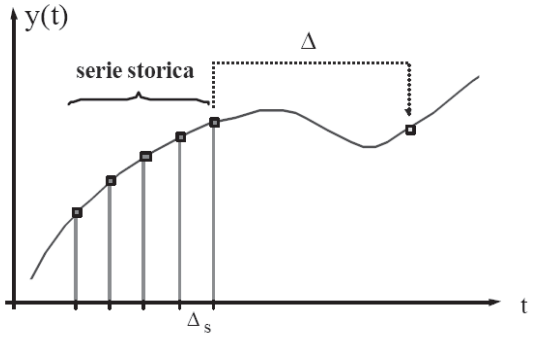


Figura 13 – Serie storica di un segnale

Il cui schema di addestramento è illustrato in figura 14.

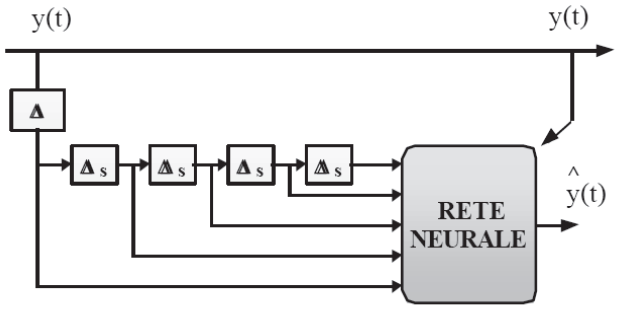


Figura 14 – Schema di addestramento con serie storica