Analisi comparativa di reti neurali autoorganizzanti

Maria Grazia Di Bono

Introduzione

Questa particolare classe di reti neurali artificiali, conosciuta come Self-Organizing Maps (SOMs), ossia come delle mappe auto-organizzanti, si basano sull'apprendimento competitivo.

In questo tipo di apprendimento, i neuroni di output della rete competono tra di loro per essere attivati, con il risultato che solo un neurone di output, o un neurone per gruppo, è attivato per ogni competizione. Un neurone di output che vince la competizione è chiamato *"winner-takes-all neuron"* o semplicemente *"winning neuron"*.

Un modo per indurre una competizione *winner-takes-all* tra i neuroni di output è quello di usare connessioni laterali inibitorie tra di loro.

In una self-organizing map, i neuroni sono posizionati come nodi di un lattice generalmente mono o bi-dimensionale, sono inoltre possibili, ma non comuni, mappe di livelli dimensionali superiori.

I neuroni si sintonizzano in maniera selettiva su vari pattern di input (stimoli) o su classi di pattern di input, nel corso di un processo di apprendimento competitivo.

Una self-organizing map è inoltre caratterizzata dalla formazione di mappe topografiche dei pattern di input, nelle quali le locazioni spaziali, ossia le coordinate dei neuroni nel lattice, sono indicative delle caratteristiche statistiche intrinseche dei pattern di input. Da ciò il nome di "self-organizing map".

Lo sviluppo come modello neurale delle SOM, è motivato da una precisa caratteristica del cervello umano: il cervello è organizzano in zone in cui differenti segnali sensoriali di input sono rappresentati da mappe computazionali ordinate topologicamente. La corteccia cerebrale del cervello umano è il più complesso di tutti i sistemi biologici. Ad un livello microscopico, questa è organizzata in diversi strati di neuroni di diversa densità e tipologia, mentre ad un livello macroscopico questa è organizzata in regioni spaziali, in base alle funzioni del corpo.

Ciascuna regione consiste in un gran numero di neuroni simili che cooperano nell'eseguire quelle funzioni per le quali sono specializzati. Esiste una corrispondenza di caratteristiche tra i neuroni sensoriali e le regioni della corteccia cerebrale ad essi associate. In particolare, input sensoriali tattili, visivi, acustici sono inviati verso differenti aree della corteccia cerebrale in modo topologicamente ordinato, quindi la mappa topologica rappresenta un blocco base nel processo di elaborazione dell'informazione del nostro sistema nervoso.

Risulta quindi interessante, la costruzione di mappe topografiche artificiali, che apprendono, attraverso un'auto-organizzazione, similmente a quanto studiato in neurobiologia.

In questo contesto , una self-organizing feature map (SOFM ANN), rappresenta un modello semplificato della corrispondenza tra caratteristiche di segnali sensoriali e particolari regioni del cervello.

Essa può essere definita come una rete neurale auto-organizzante, competitiva, che impara dall'ambiente senza l'aiuto di un insegnante esterno, ossia in modo non supervisionato.

Un importante punto che emerge dalle discussioni sulle mappe computazionali nel cervello, è il *principio di formazione delle mappe topografiche*, che può essere precisato nel seguente modo (Kohonen, 1990) :

"La locazione spaziale di un neurone di output in una mappa topografica, corrisponde ad un particolare dominio o proprietà dei dati estratti dallo spazio degli input".

Ci sono essenzialmente due modelli di base delle SOM [Haykin99], in entrambi i casi i neuroni di output sono organizzati su un lattice bidimensionale. Questo tipo di organizzazione assicura che ciascun neurone abbia un insieme di nodi vicini.

I due modelli differiscono l'uno dall'altro per il modo in cui i pattern di input vengono specificati.

Il primo modello, fu originariamente proposto da Willshaw e von der Malsburg (1976) in campo biologico per spiegare il problema della corrispondenza dei segnali provenienti dalla retina, alla mappa visiva della corteccia cerebrale. In particolare, ci sono due distinti lattici bidimensionali di neuroni connessi tra loro. Un lattice rappresenta i neuroni presinaptici (input), mentre l'altro rappresenta i neuroni postsinaptici (output). I due lattici sono interconnessi da sinapsi modificabili di tipo Hebbiano. I neuroni postsinaptici non sono di tipo winner takes all, ma è usata una soglia per assicurare che solo pochi neuroni siano attivi ad ogni competizione. Tuttavia, per prevenire una crescita dei pesi sinaptici che può portare ad un'instabilità della rete, il peso totale associato a ciascun neurone postsinaptico è limitato da un limite superiore. Per ciascun neurone alcuni pesi sinaptici crescono mentre altri decrescono. L'idea base di questo modello è quella di codificare l'informazione di vicinanza geometrica dei neuroni presinaptici nella forma di correlazioni nella loro attività elettrica, per poi usare tali correlazioni nel lattice postsinaptico al fine di connettere i neuroni postsinaptici vicini allo stesso modo dei neuroni presinaptici vicini. Generalmente questo modello è utilizzato per i mapping in cui la dimensione dell'input è la stessa di quella dell'output.

Il secondo modello, fu introdotto da Kohonen nel 1982 e non ha l'obiettivo di spiegare dettagli neurobiologici.

Il modello di Kohonen è più generale di quello di Willshaw e von der Malsburg, nel senso che è in grado di effettuare una compressione dei dati, come ad esempio una riduzione dimensionale dei dati di input. In realtà il modello di Kohonen appartiene alla classe degli algoritmi di *vector- coding*. Esso offre un mapping topologico che colloca un numero fissato di vettori (*code words*) in uno spazio di input sovradimensionale, per cui facilita la compressione dei dati.

Il modello di Kohonen ha ricevuto in letteratura molta più attenzione rispetto al modello di Willshaw e von der Malsburg, perché possiede delle

proprietà che lo rendono particolarmente interessante per capire e modellare le mappe corticali del cervello.

Self-Organizing Maps, aspetti teorici

L'obiettivo principale di una self-organizing map (SOM) è quello di trasformare segnali di input di arbitraria dimensione in una mappa discreta di dimensione uno, due, tre o maggiore e di realizzare tale trasformazione in modo adattivo, rispettando l'ordine topologico.

Nel caso di una mappa bidimensionale, ciascun neurone del lattice è completamente connesso a tutti i nodi sorgenti nello strato di input. Questa rete rappresenta una struttura feedforward con un singolo strato computazionale composto da neuroni organizzati in righe e colonne.

La presentazione di ciascun pattern di input alla rete consiste nella localizzazione di precise regioni di attività che variano da un pattern all'altro, quindi tutti i nodi della rete devono essere esposti ad un numero sufficiente di pattern di input, per garantire che il processo di auto-organizzazione possa svilupparsi in modo appropriato.

A seconda della dimensione della mappa, si può avere una delle situazioni illustrate nelle seguenti figure, che si riferiscono rispettivamente al caso di SOM mono-dimensionale lineare, SOM mono-dimensionale anellare ed infine SOM bi-dimensionale.

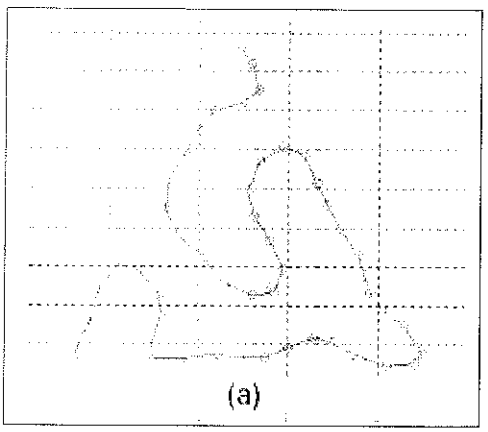

Figura 1.1. SOM con lattice mono-dimensionale lineare.

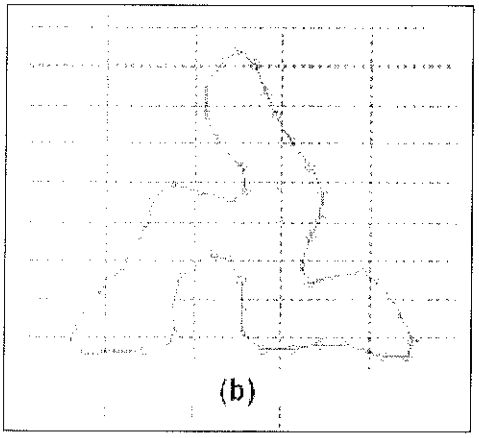

Figura 1.2. SOM con lattice mono-dimensionale anellare.

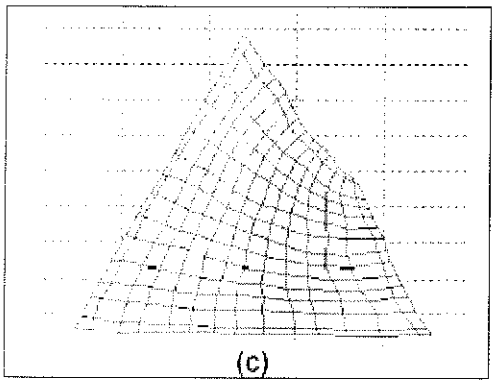

Figura 1.3. SOM con lattice bi-dimensionale.

L'algoritmo procede con un primo passo di inizializzazione dei pesi sinaptici della rete, in generale valori molto piccoli scelti in maniera casuale, senza nessun ordine preciso imposto alla mappa.

Una volta che la rete è stata inizializzata in modo appropriato, ci sono essenzialmente tre processi coinvolti nella formazione della mappa [Haykin99] :

- **Competizione** : per ciascun pattern di input, i neuroni della rete calcolano il loro rispettivo valore di una funzione discriminante che getta le basi per la competizione tra i neuroni. Il particolare neurone con il più grande valore di tale funzione vince la competizione ed è dichiarato il nodo vincente (winner).

- **Cooperazione** : il neurone vincente, il winner, determina la locazione spaziale dei neuroni con una certa vicinanza topologica, ponendo le basi per una cooperazione tra alcuni neuroni vicini.

- **Adattamento sinaptico** : quest'ultimo meccanismo, abilita i neuroni eccitati dal particolare stimolo di input, ad aumentare il valore individuale della funzione discriminante, attraverso la modifica applicata ai loro pesi sinaptici. Tali modifiche sono effettuate in modo tale che l'attivazione del nodo vincente, in seguito alla presentazione di un successivo pattern di input simile, sia accresciuta.

Processo di competizione.

Supponiamo di avere uno spazio dei dati di input di dimensione m. Sia

$$\mathbf{x} = [\, x_1, x_2, \ldots, x_m]^T$$

un pattern di input selezionato in modo casuale dallo spazio degli stimoli.

Il vettore peso sinaptico di ciascun neurone della rete ha la stessa dimensione dello spazio degli input, ad esempio per il neurone j, il vettore peso può essere denotato nel seguente modo :

$$\mathbf{w}_j = [w_{j1}, w_{j2}, \ldots, w_{jm}]^T, \quad j = 1, 2, \ldots, l$$

dove l è il numero totale di neuroni nella rete.

Per trovare il vettore peso sinaptico $\mathbf{w}_j$ con caratteristiche più simili al vettore di input $\mathbf{x}$, si confrontano i prodotti interni $\mathbf{w}_j^T \mathbf{x}$ per $j = 1, 2, \ldots, l$ e se ne seleziona il più grande. Si fa quindi l'assunzione che la stessa soglia, per la precisione , il bias con segno negativo, sia applicata a tutti i neuroni. Selezionando il neurone con il prodotto interno $\mathbf{w}_j^T \mathbf{x}$ più grande,

avremo in effetti determinato la collocazione del centro della vicinanza topologica dei neuroni eccitati.

Il criterio di scelta basato sulla massimizzazione del prodotto interno $\mathbf{w}_j^T \mathbf{x}$ è matematicamente equivalente a minimizzare la distanza Euclidea tra i vettori $\mathbf{x}$ e $\mathbf{w}_j$. Infatti si ha che :

$$\|x - w_j\|^2 = \|x\|^2 + \|w_j\|^2 - 2\left(w_j^T x\right)$$

Pertanto, se si indica con i(x) il neurone che meglio approssima il vettore di input x, si può determinare i(x) applicando la seguente condizione :

$$i(\mathbf{x}) = \arg\min_j \|x - w_j\| , \qquad j = 1, 2, ..., l \qquad (1.1)$$

Questa condizione è l'essenza del processo di competizione tra i neuroni.

In base al diverso tipo di applicazione di interesse, la risposta della rete potrà essere o l'indice del neurone vincente, ad esempio la sua posizione nel lattice, oppure il vettore peso sinaptico con distanza Euclidea minima rispetto al vettore di input **x**.

Processo di Cooperazione

Il neurone vincente (winner) nella competizione, rappresenta il centro di una zona di vicinanza topologica di neuroni cooperanti. La questione è trovare un modo per definire una zona di vicinanza topologica che sia neurobiologicamente corretta. In particolare l'osservazione che in natura un neurone eccitato tende ad eccitare i neuroni nelle sue immediate vicinanze molto di più rispetto a quelli più lontani, permette di definire una zona di vicinanza topologica intorno al neurone vincente i , che decresce uniformemente con la distanza laterale.

Nello specifico, siano :

- $h_{j,i}$, la funzione di vicinanza topologica centrata nel neurone vincente **i**, che comprende un insieme di neuroni eccitati cooperanti

- j , l'indice di uno dei neuroni cooperanti

- $d_{i,j}$, la distanza laterale tra il neurone vincente **i** ed il neurone eccitato **j**

Si definisce $h_{j,i}$, come una funzione unimodale della distanza laterale $d_{i,j}$ che soddisfa le due seguenti proprietà :

- La funzione $h_{j,i}$ è simmetrica rispetto al punto di massimo definito da $d_{i,j} = 0$, ossia $h_{j,i}$ ha il suo valore di massimo per il neurone vincente **i** , per il quale la distanza $d_{i,j}$ è zero
- L'ampiezza della funzione $h_{j,i}$ decresce in modo monotono con l'aumentare della distanza laterale $d_{i,j}$, tendendo a zero per $d_{i,j}$ che tende all'infinito. E' questa una condizione necessaria per garantire la convergenza.

Una scelta tipica di $h_{j,i}$ che soddisfa le precedenti proprietà è la *funzione Gaussiana* :

$$h_{j,i(x)} = \exp\left(-\frac{d^2_{j,i}}{2\sigma^2}\right) \qquad (1.2)$$

che è una funzione invariante per traslazione, ad esempio è indipendente dalla posizione del neurone vincente nella mappa.

Il parametro σ è l'ampiezza effettiva della funzione $h_{j,i}$. Esso definisce in quale misura, i neuroni eccitati nelle vicinanze del neurone vincente, partecipano al processo di apprendimento.

Per avere una cooperazione tra i neuroni vicini, è necessario che la funzione di vicinanza topologica $h_{j,i}$ dipenda dalla distanza laterale $d_{i,j}$ tra il neurone vincente **i** e quello eccitato **j** nello spazio degli output anziché da qualche distanza misurata nell'originale spazio degli input. Ad esempio nel caso di lattice mono-dimensionale, $d_{j,i}$ è un intero uguale a $|j-i|$, mentre nel caso di lattice bidimensionale o di dimensioni superiori, $d_{j,i}$ è definito nel seguente modo :

$$d_{j,i}^2 = \|r_j - r_i\|^2 \tag{1.3}$$

dove il vettore discreto r_j definisce la posizione del neurone eccitato j mentre r_i definisce la posizione discreta del neurone vincente i, entrambi i quali misurati nello spazio discreto di output.

Un'altra caratteristica dell'algoritmo delle SOM è il restringimento nel tempo della dimensione delle zone di vicinanza topologica. Questa proprietà si soddisfa facendo decrescere nel tempo l'ampiezza σ della funzione $h_{j,i}$. Una scelta comune per la dipendenza di σ dal tempo discreto n è la seguente :

$$\sigma(n) = \sigma_0 \exp\left(-\frac{n}{\tau_1}\right) \qquad n= 0,1,2,\ldots, \tag{1.4}$$

dove σ_0 è il valore iniziale di σ al momento dell'inizializzazione dell'algoritmo e τ_1 è una costante temporale.

Si può quindi riscrivere la funzione di vicinanza topologica nel seguente modo :

$$h_{j,i(x)}(n) = \exp\left(-\frac{d^2_{j,i}}{2\sigma^2(n)}\right) \qquad n=0,1,2,\ldots, \tag{1.5}$$

dove $\sigma(n)$ è definita dall'equazione (1.4).

Man mano che n, quindi il numero di iterazioni, aumenta, l'ampiezza $\sigma(n)$ decresce in modo esponenziale e le zone di vicinanza topologica si restringono in modo corrispondente.

Un altro modo di vedere la variazione della funzione di vicinanza intorno al nodo vincitore, consiste nel mantenere costante l'ampiezza della funzione $h_{j,i(x)}(n)$ ma incrementare gradualmente il numero totale di neuroni della mappa, inserendoli in mezzo ai vecchi nodi [Luttrell, 1989].

Processo di Adattamento sinaptico

L'ultimo processo per la formazione della mappa di caratteristiche auto-organizzante, è quello dell'adattamento sinaptico, ossia il vettore peso sinaptico $\mathbf{w}_j$ deve essere modificato in relazione al vettore di input $\mathbf{x}$.

Il problema è come effettuare tale aggiornamento. Nella regola di apprendimento di Hebb, il peso sinaptico è incrementato con termini che contengono entrambe le attività pre e post-sinaptiche, il ché va bene in caso di apprendimento associativo.

Nel caso di apprendimento non supervisionato, i cambiamenti delle connessioni si attuano in una direzione sola, pertanto si modifica la regola Hebbiana, includendo un termine di dimenticanza $g(y_j)w_j$, dove w_j è il vettore peso sinaptico del neurone j e $g(y_j)$ è una funzione scalare positiva della risposta y_j del neurone.

Si impone che il termine costante dello sviluppo in serie di Taylor della funzione $g(y_j)$ sia zero, ossia :

$$g(y_j) = 0 \qquad \text{per} \quad y_j = 0 \tag{1.6}$$

Si può esprimere l'aggiornamento del vettore peso del neurone j nel lattice come segue :

$$\Delta w_j = \eta y_j x - g(y_j) w_j \tag{1.7}$$

dove η è il parametro di apprendimento (*learning-rate*) dell'algoritmo.

Il primo termine a destra dell'equazione (1.7) è il *termine Hebbiano*, mentre il secondo è il *termine di dimenticanza*.

Affinché possa valere la condizione (1.6) si può scegliere per $g(y_j)$ la seguente funzione lineare :

$$g(y_j) = \eta y_j \tag{1.8}$$

Ponendo

$$y_j = h_{j,i(x)} \tag{1.9}$$

si può semplificare la (1.7) , ottenendo l'equazione (1.10)

$$\Delta w_j = \eta h_{j,i(x)}(x - w_j) \tag{1.10}$$

Infine, dato il vettore peso sinaptico $\mathbf{w_j(n)}$ del neurone **j** al tempo **n**, il vettore peso aggiornato $\mathbf{w_j(n+1)}$ al tempo **n+1** è definito come segue :

$$w_j(n+1) = w_j(n) + \eta(n) h_{j,i(x)}(n)(x - w_j) \tag{1.11}$$

Tale aggiornamento è applicato a tutti i neuroni del lattice giacenti nella zona di vicinanza topologica del neurone vincente **i** ed ha l'effetto di spostare il vettore peso $\mathbf{w_i}$ verso il vettore di input **x**. Dopo la presentazione ripetuta dei dati di allenamento, i vettori peso tendono a seguire la distribuzione dei vettori di input, grazie all'aggiornamento dei vicini.

L'algoritmo porta ad un ordinamento topologico della mappa nello spazio degli input, nel senso che vettori adiacenti nel lattice tendono ad avere vettore peso simile.

Resta da definire un'euristica per la determinazione del parametro di learning-rate $\eta(n)$, che può essere inizializzato con un valore η_0 di

partenza per poi decrescere gradualmente con il crescere del tempo **n**, attraverso l'uso della seguente funzione di decadimento esponenziale :

$$\eta(n)=\eta_0 \exp\left(-\frac{n}{\tau_2}\right) \qquad n=0,1,2,..., \qquad (1.12)$$

dove τ_2 è un'altra costante temporale dell'algoritmo.

Il processo di adattamento sinaptico dei pesi, può essere decomposto nelle seguenti due fasi :

1. **Fase di Organizzazione** : questa fase copre un numero di iterazioni dell'algoritmo di circa n = 1000. Riguardo le scelte del parametro di apprendimento e della funzione di vicinanza, quelle più comunemente effettuate sono le seguenti:

   - Il parametro learning-rate $\eta(n)$ deve inizialmente avere un valore vicino a 0.1, successivamente deve decrescere gradualmente ma rimanere intorno al valore 0.01. Questi valori desiderabili si ottengono ponendo $\eta_0 = 0.1$ e $\tau_2 = 1000$ nell'equazione (1.12).

   - La funzione di vicinanza $h_{j,i(x)}(n)$ deve inizialmente includere il maggior numero di neuroni con centro il neurone vincente e successivamente restringersi lentamente nel tempo. Ad esempio, supponendo di usare un lattice bidimensionale di neuroni, si potrebbe inizializzare l'ampiezza σ_0 della funzione di vicinanza con il raggio del lattice e la costante temporale τ_1 nel seguente modo :

$$\tau_1 = \frac{1000}{\log \sigma_0}$$

2. **Fase di Convergenza** : questa fase è necessaria per meglio sintonizzare la mappa ottenuta con lo spazio degli input di cui fornire un'accurata quantificazione statistica e può richiedere centinaia o migliaia di iterazioni. Per una buona accuratezza statistica, il learning-rate $\eta(n)$ deve essere mantenuto su piccoli valori dell'ordine di 0.01 e non deve scendere a zero, altrimenti è possibile per la rete trovarsi in particolari stati detti *"metastable state"*, a cui appartengono configurazioni di mappe con difetti topologici. La funzione di vicinanza $h_{j,i(x)}(n)$ deve contenere solo i vicini immediati del neurone vincente, eventualmente ridotta ad uno o zero neuroni vicini.

Proprietà delle SOM

Dopo la convergenza dell'algoritmo, la *feature map* calcolata, mostra importanti caratteristiche statistiche dello spazio degli input [Haykin99].

Sia X lo spazio continuo dei dati di input, la cui topologia è definita da relazioni metriche tra i vettori $\mathbf{x} \in X$.

Sia A lo spazio discreto di output, la cui topologia è data dalla risistemazione di un insieme di neuroni di un lattice.

Sia Φ una trasformazione non lineare, chiamata *Feature Map*, che effettua la corrispondenza tra lo spazio di input X e lo spazio di output A, ossia :

$$\Phi : X \rightarrow A$$

Dato un vettore di input $\mathbf{x}$, l'algoritmo procede prima identificando il neurone vincente $\mathbf{i(x)}$ nello spazio di output A in accordo con la feature map Φ.

Il vettore peso $\mathbf{w}_i$ del neurone $\mathbf{i(x)}$ può essere visto come un puntatore per quel neurone nello spazio di input X, ossia gli elementi del vettore $\mathbf{w}_i$ possono essere visti come le coordinate dell'immagine del neurone $\mathbf{i(x)}$ proiettato nello spazio degli input.

La trasformazione Φ ha alcune importanti proprietà :

1.Approssimazione dello spazio di input. *La trasformazione* Φ, *rappresentata dall'insieme dei vettori peso sinaptico* $\{w_j\}$ *nello spazio di output* A, *offre una buona rappresentazione dello spazio di input* X.

2.Ordinamento Topologico. *La trasformazione* Φ *effettuata dall'algoritmo è ordinata topologicamente, nel senso che la posizione di un neurone nel lattice corrisponde ad un particolare dominio o caratteristica dei patterns di input.*

3.Corrispondenza delle densità. *La trasformazione* Φ *riflette le variazioni statistiche della distribuzione degli input, ossia regioni nello spazio degli input* X, *dalle quali si sono scelti vettori esempio* $\boldsymbol{x}$ *con un'alta probabilità di occorrenza, corrispondono ad ampi domini dello spazio di output* A, *rispetto a quelle regioni in* X, *da cui sono stati scelti vettori con una bassa probabilità di occorrenza.*

4.Selezione delle caratteristiche. *Presi dei dati da uno spazio di input con una distribuzione non lineare, una self-organizing map è in grado di selezionare un insieme delle migliori caratteristiche per l'approssimazione di tale distribuzione.*

Varianti delle Self-Organizing Maps

Andando avanti nella ricerca sono emerse alcune proposte di varianti delle SOM bidimensionali, sia per quel che riguarda essenzialmente il modo di vincere nella competizione, proponendo delle funzioni *Errore* diverse e più complesse per adattarsi alla complessità del problema analizzato, che per quel che riguarda la struttura architettonica della rete [Koho95].

Le tre principali motivazioni per suggerire l'uso delle SOM strutturate sono le seguenti:

- Una ricerca più veloce dell'unità vincitrice nella competizione
- Una convergenza più rapida
- Una migliore generalizzazione ed astrazione

Per quanto riguarda la prima motivazione, sono state messe a punto delle SOM strutturate in un modo particolare per velocizzare la ricerca del nodo vincitore. Tra queste un esempio interessante è costituito dalle *Tree-Stuctured SOM* [Koho95]. La ricerca sequenziale del nodo vincitore nelle implementazioni software delle SOM può comportare un eccessivo dispendio di tempo, perciò per velocizzare tale ricerca è stato introdotto uno schema gerarchico di ricerca che coinvolge diverse SOM organizzate come una struttura piramidale . L'idea di base, è partire dal livello radice dell'albero e formare una SOM di un solo neurone con il suo vettore peso fissato. In seguito le SOM più larghe e di più basso livello sono allenate, un livello per volta ed i loro vettori peso sono fissati. Nella localizzazione del nodo vincitore di una SOM che si sta formando, i livelli precedenti già fissati sono usati come alberi di ricerca. In questo caso, la ricerca nei livelli più bassi è effettuata in tutte quelle unità collegate al nodo vincitore ed ai suoi vicini del livello immediatamente più alto. Per determinare il nodo vincitore di un certo livello la ricerca inizia al livello radice, scendendo ai successivi livelli più bassi sui quali è determinato il nodo vincitore di quel livello. Nel successivo livello ancora più basso, si avrà bisogno di considerare solo quel sottoinsieme di unità subordinato al nodo vincitore.

Per quanto riguarda i punti 2 e 3, sono state studiate altre varianti di SOM che possono essere fatte rientrare in due distinte categorie [Koho95] :

- *Varianti Supervisionate*
- *Varianti non supervisionate*

Per quanto riguarda le prime, queste sono state usate nei sistemi di riconoscimento del linguaggio parlato. Nelle *SOM supervisionate*, applicate a tale problema, i vettori di input sono formati da due parti, x_ν ed x_μ. Il primo rappresenta il vettore di componenti dello spettro acustico. Il secondo rappresenta il vettore, le cui componenti sono assegnate ad una delle classi fonemiche. Come input per la SOM quindi, è usato il vettore concatenazione

$$x = [x_\nu^T, x_\mu^T]^T$$

Poiché x_μ è lo stesso per i vettori della stessa classe e diverso per classi diverse, come risultato finale si ha che nel clustering si migliora la separazione delle classi.

Durante il riconoscimento di un vettore x non noto, solo la sua parte x_ν è confrontata con la parte corrispondente dei vettori peso. Quindi, questo tipo di SOM, discrimina meglio tra più classi di esempi. I vettori peso di celle etichettate diversamente, nella loro parte x_μ, definiscono i contorni di decisione tra le varie classi.

Tra le varianti non supervisionate ce ne sono diverse, tutte basate sugli stessi principi e fondamenti teorici. Esse sono una estensione delle SOM di base e sono note con il nome di *Multilayer-SOMs o Hierarchical SOMs*.

Le Hierarchical Self-Organizing Maps (*HSOMs*) rientrano nella classe dei metodi non supervisionati di clustering [Lampinem92].

In analogia con le reti multilayer-feedforward, si è mostrato che le HSOMs riescono a formare dei cluster arbitrariamente complessi.

In esperimenti con dati reali ed artificiali è stato dimostrato che le SOM multistrato formano dei cluster che corrispondono meglio alle classi desiderate, rispetto agli algoritmi come il classico *K-means* o le SOM monostrato.

Molte delle ricerche sulle reti neurali nel campo del *Pattern Recognition, Image Processing* ed *Image Vision*, si sono basate su apprendimenti supervisionati. Ad esempio, reti neurali supervisionate come il *Multilayer-Perceptron (MLP)*, offrono metodi molto efficienti per costruire un classificatore di esempi, non lineare, arbitrariamente complesso.

Il risultato teorico centrale, che ha rappresentato l'impeto del crescente interesse per le reti neurali, è che il MLP con un solo strato nascosto può approssimare qualsiasi funzione continua su un dominio compatto, con un'arbitraria precisione.

Tuttavia, esistono dei problemi con domini che non possono essere spiegati semplicemente con un potente classificatore. Gli esempi più significativi di questo tipo di problemi si hanno nel campo del *Machine Vision* e dell' *Image Understanding*. In questi campi, gli obiettivi essenziali consistono nel localizzare e riconoscere oggetti individuali e nel dare un'interpretazione delle loro relazioni.

Per trattare l'ampia variabilità delle scene, i sistemi di analisi delle immagini devono avere un ampio numero di parametri liberi, la stima dei quali richiederebbe una sostanziosa quantità di dati. Quindi, l'uso di una rete neurale allenata in modo supervisionato, per un sistema di analisi delle immagini, richiederebbe una rete di enormi dimensioni con un altrettanto enorme numero di esempi classificati manualmente. Questo dilemma si risolve usando tecniche di apprendimento non supervisionato per ridurre il numero di gradi di libertà nei dati. Successivamente si possono usare dei classificatori supervisionati che richiederanno un numero più basso di parametri liberi e quindi un numero altrettanto basso di esempi di allenamento preclassificati. A questo scopo sono state usate le SOM come "cluster preprocessor" nei sistemi di analisi di immagini.

Mentre le SOM monostrato di base, dividono lo spazio degli input in regioni convesse analogamente a quanto avviene per le reti feed-forward monostrato, nelle HSOM, gli output della prima SOM sono

mandati come input all'altra SOM ed il risultato è che la HSOM forma dei cluster di arbitraria complessità, in analogia alle reti feed-foward multistrato.

Le SOM monostrato, come descritto nel paragrafo 1.1, sono guidate da un algoritmo di apprendimento che consiste, approssimativamente, nei seguenti passi :

1. Scegli i valori iniziali dei vettori peso m_i delle unità i della rete, in maniera random;
2. Ripeti i passi 3 e 4 finché l'algoritmo non converge;
3. Scegli un esempio x dalla distribuzione di probabilità dell'insieme degli esempi di input e cerca l'unità b che più rappresenta l'esempio scelto, in accordo con la seguente:

$$\| x - m_b \| = \min \| x - m_j \| \quad \text{con j} = 1, \ldots\ldots, \text{M} \qquad (1.4.1.1)$$

4. Aggiusta i pesi di tutte le unità della rete nel seguente modo:

$$m_i := m_i + \gamma * h_{b,i} * (x - m_i) \qquad (1.4.1.2)$$

dove:

- γ è la funzione di apprendimento;
- $h_{b,i}$ è la cosiddetta funzione di neighborood

La SOM gerarchica è definita come una SOM bi-dimensionale, il cui principio operativo si può riassumere nei seguenti passi:

- Per ogni vettore di input x, l'unità con il migliore grado di similarità è scelta sulla mappa del primo strato ed il suo indice b ,o il suo peso, diventa l'input per il secondo strato della rete

- L'unità con il miglior grado di similarità per b è scelta nella mappa del secondo strato della rete ed il suo indice, o il suo peso, rappresenta l'output dell'intera rete

Poiché ciascun'unità i , della mappa del primo strato, ha una regione convessa e poliedrica V_i definita dall'equazione (1.4.1.1) e ciascun'unità j, del secondo strato, rappresenta l'unità con il miglior grado di similarità per un sottoinsieme $\{i_1, i_2, \ldots\ldots, i_k\}$ di indici del primo strato, l'unità j del secondo strato è difatti quella con il miglior grado di similarità per qualche vettore $x \in \bigcup_{k=1}^{K} V_{ik}$. Questa regione risultante è un'unione arbitraria di regioni convesse poliedriche non sovrapposte. Una regione in R^n può essere approssimata con una certa accuratezza da una tale unione, quando il numero delle regioni componenti V_{ik} è arbitrariamente grande, quindi, cluster di forme arbitrarie possono essere rappresentati da una mappa a due strati. Non esiste comunque nessuna garanzia sul fatto che una certa predeterminata forma di cluster possa essere appresa dalla mappa, poiché, per definizione di apprendimento non supervisionato, non possono esistere cluster usati come target.

Il principale vantaggio del clustering con le *HSOM* rispetto ai classici metodi di clustering, come ad esempio l'algoritmo *K-means*, è la misura adattabile della distanza [Lampinem92]. Infatti, nell'algoritmo *K-means* può succedere che ampi cluster siano divisi in cluster più piccoli ed anche che cluster più piccoli siano uniti per formarne uno più ampio, finché tutti i cluster non raggiungono le dimensioni desiderate. Tuttavia, in pratica, risulta molto difficile determinare la dimensione appropriata dei cluster.

Infine, ulteriori varianti delle *SOM*, sono le *Temporal Kohonen Maps* (*TKMs*) e le *Recurrent Self-Organizing Maps* (*RSOMs*), che offrono la possibilità di organizzare spazialmente e temporalmente i dati di input [Euliano96].

La SOM di base è indifferente rispetto all'ordine degli esempi di input. Tuttavia in natura spesso i dati sono sequenziali, perciò il contesto temporale degli esempi può significativamente influenzarne la corretta interpretazione. Una delle metodologie usate per la messa a punto delle *mappe spazio-temporali* , consiste nell'utilizzo di una rete, che usa l'accoppiata spazio-tempo che crea onde di attività temporale nella rete.

Queste iniziano nel nodo vincente e fluiscono attraverso la mappa, in una predefinita direzione del tempo. L'attività spazio-temporale è usata per aumentare la probabilità che un nodo ha di vincere la competizione, in modo tale che i nodi vicini dell'ultimo nodo vincitore, nella direzione del tempo, hanno maggiore probabilità di vincere la prossima competizione.

Questa metodologia tiene conto dell'organizzazione spaziale e temporale e crea delle zone di vicinato, nella mappa, che sono locali in entrambi il tempo e le caratteristiche spaziali.

Uno degli aspetti più vantaggiosi di questa tecnica è che l'allenamento della SOM spazio-temporale è semplice e molto simile a quello della SOM standard. Nel caso mono-dimensionale, l'algoritmo d'apprendimento richiede solo due parametri aggiuntivi. Il primo e più importante è il parametro di proporzione Tempo/spazio (*TSP*), che descrive l'importanza relativa dell'attività temporale verso la distanza spaziale, nella scelte del nodo vincente. L'algoritmo di allenamento consiste in due passi:

1. Scelta del nodo vincitore;
2. Aggiornamento dei pesi vicini.

Il secondo passo è esattamente quello della normale rete di Kohonen, mentre il primo è stato modificato aggiungendo un aumento temporale pesato, prima della determinazione del vincitore.

Il vincitore è selezionato applicando la formula seguente:

$$win(t) = \arg_j \min(\|x - w_j\| - \beta * temp_j)$$

dove $temp_j(t)$ è così definito :

$$temp_j(t) = \lambda * [temp_j(t-1) + \delta_{j,win(t-1)}] + (1-\lambda) * [temp_{j-1}(t-1) + \delta_{j-1,win(t-1)}]$$

con j = 1, 2,....., N.

dove

β è il parametro TSP

λ definisce il decadimento del peso

δ è il Kronecker Delta

I fattori λ e $(1-\lambda)$ sono stati inseriti per normalizzare l'attività totale nella rete.

Incrementando il TSP, il sistema forza gli esempi di input ad essere sequenziali nella mappa di output, invece riducendo tale parametro il sistema opera più similmente ad una normale SOM ed usa solo lo spazio delle distanze per determinare il vincitore.

Molte sono state le applicazioni che hanno fatto uso di tali reti, tra le quali interessante risulta essere quella per la localizzazione di oggetti in immagini bi-dimensionali con diversi livelli di grigio [Lakany]. L'idea di base di questo approccio è quella di rappresentare piccoli blocchi di un'immagine come una sequenza di pixel con diversi valori di intensità come se si stesse facendo la scannerizzazione dell'immagine , tenendo conto del cambiamento del valore dei pixel nel tempo.

Lo stesso approccio è stato utilizzato per il problema della verifica dell'appartenenza di una scrittura ad un insieme di scritture di riferimento [Chappelier]. A tal proposito, la scrittura non viene considerata solamente come dei segni sulla carta, ma come una sequenza temporale dei movimenti della penna sul foglio. L'input per la rete consiste in una funzione temporale nelle due coordinate planari della

penna ed una funzione binaria che, ad ogni passo, specifica se la penna è sopra o sotto.

Self-organizing map : applicazioni

Le SOM più comunemente usate nelle applicazioni pratiche sono quelle mono o bi-dimensionali, poiché la visualizzazione nello spazio 2D è più semplice ed immediata, inoltre la maggior parte delle implementazioni software o degli ambienti di sviluppo per applicazioni con reti di questo tipo, supportano solo mappe mono o bi-dimensionali.

Le applicazioni di questo particolare tipo di reti non supervisionate sono molteplici e spaziano dal campo del *"Vector Qantization"* a quelli di *"Data Compression"* , *"Image Processing"* ed *"Image Undestanding"*e trattano problemi che vanno da quelli di ottimizzazione, controllo di robot, a problemi di riconoscimento di caratteri, riconoscimento del linguaggio parlato, a problemi di classificazione e comparazione di immagini.

Le implementazioni software per la risoluzione di tali problemi, grazie anche alle tecniche di signal-processing e quelle standard dell'intelligenza artificiale, si sono rivelati dei validi strumenti di supporto. Come illustrato nella sezione 1.2, le SOM sono particolarmente utili perché sono in grado di rappresentare relazioni non lineari tra dati di input multidimensionali con semplici relazioni geometriche su una griglia, generalmente bidimensionale. Questo processo di *"mapping"* è in grado di preservare le più importanti relazioni topologiche e metriche degli elementi dei dati originali. Questo

è un valido motivo per preferire le SOM ad altre possibili architetture di reti neurali, in quei problemi dove tali proprietà necessitano.

Un utilizzo comune delle SOM , riguarda i problemi di clustering (raggruppamento). A titolo d'esempio si può riportare un lavoro di tesi

[RS99], in cui sono state usate tali SOM per il clustering di punti uniformemente distribuiti internamente a sette volumi, che rappresentano grossolanamente una sagoma umana, come illustrato nelle figure riportate di seguito.

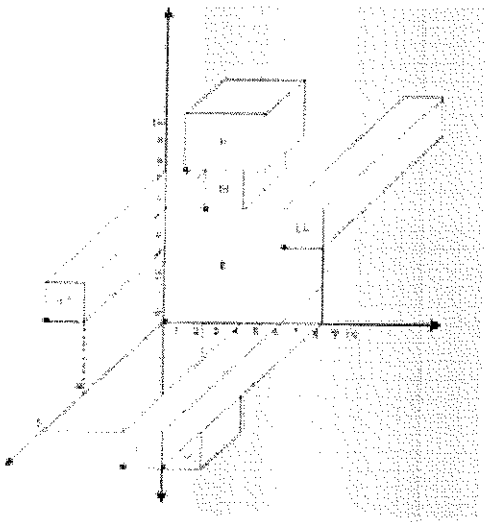

Figura1.4. Figura umana grossolana rappresentata da sette volumi.

Ogni volume della figura umana, rappresenta una zona caratteristica del corpo, essenzialmente il busto, il collo, la testa e gli arti superiori ed inferiori, etichettati diversamente.

Alla rete vengono dati in input dei punti scelti nello spazio tridimensionale. La rete è in grado di organizzare spazialmente tali punti in modo che punti vicini nello spazio risultino appartenenti allo stesso volume e quindi alla stessa zona del corpo.

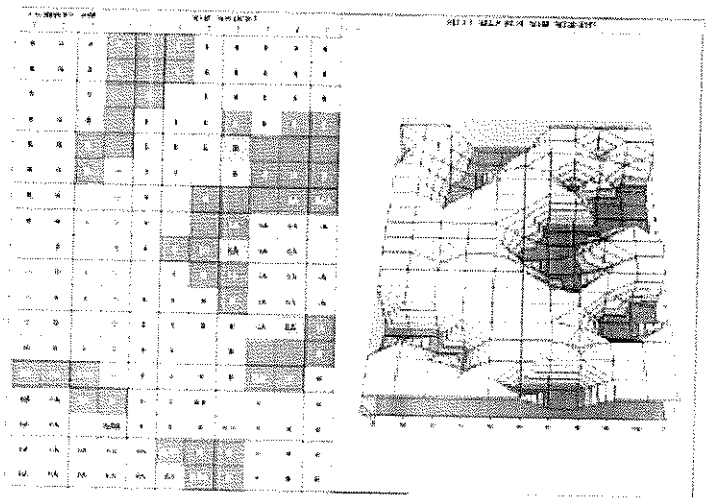

Figura1.5. Organizzazione dell'informazione sul lattice bidimensionale.

Uno dei risultati più entusiastici fu raggiunto già a partire dai primi anni del 1980 da Teuvo Kohonen e colleghi, alla *University of Technology* di Helsinki, con lo sviluppo del *"phonetic typewriter"*, un sistema di riconoscimento in grado di registrare e trascrivere un linguaggio parlato in un testo ortograficamente corretto [Patt]. Kohonen ed il suo gruppo ebbero un notevole successo per linguaggi fonetici come il Finlandese o il Giapponese. Non è molto chiaro, quanto facilmente i loro risultati possano essere stati estesi a linguaggi meno fonetici come l'Inglese, il Russo o il Cinese, dove un ruolo fondamentale è svolto anche dall'intonazione della voce. A tale scopo è stata usata una rete LVQ

(*Learning Vector Quantization*), per effettuare la corrispondenza tra i fonemi e le classi fonetiche di riferimento. I vettori pattern di input sono sati ottenuti dallo spettro del suono del linguaggio parlato e successivamente classificati in una delle 26 classi fonetiche del linguaggio Finlandese. Da tale classificazione risulta una sequenza di simboli che devono essere fusi in segmenti, ciascuno dei quali rappresenta un fonema specifico del linguaggio parlato.

Un'altra area di utilizzo delle SOM è quella dei problemi di ottimizzazione. Ne sono un esempio le applicazioni che, facendo uso di SOM mono-dimensionali, risolvono problemi di ottimizzazione come quello del *Commesso Viaggiatore (TSP)* [Patt]. Tale problema può essere descritto come la ricerca del percorso più breve, che attraversa una ed una sola volta, ogni città appartenente ad un insieme prefissato. La soluzione ottenuta utilizzando una SOM, cresce solo linearmente con il numero delle città e non richiede né il calcolo di parametri per l'allenamento della rete, né la presenza di un insegnante esterno per la supervisione. Tale soluzione, implementata da Angeniol et al. Nel 1988, fa uso di una SOM mono-dimensionale i cui neuroni sono disposti ad anello e sono liberi di muoversi nel piano durante il processo iterativo di adattamento sinaptico fintanto che non vengono "catturati" dalle città le cui posizioni sono fissate nel piano. A ciascun passo di iterazione viene presentata alla rete una città, viene scelto il nodo con il peso più vicino a tale città e quindi viene modificato il suo peso e quello dei nodi vicini, seguendo l'andamento di una funzione che decresce in base alla loro distanza dal nodo vincente. La modifica dei nodi vicini al winner tende a minimizzare la distanza tra due nodi vicini , riducendo quindi la lunghezza del giro. Successivamente viene scelta una nuova città e si ripete questo processo fintanto che non si siano presentate tutte le città. I

nodi della rete e le città sono caratterizzate dalle loro coordinate nel piano. Partendo dal nodo in posizione (0,0), il numero N di nodi della rete cresce con la duplicazione di un nodo se e solo se questo è stato scelto come winner di due diverse città. Il nuovo nodo viene inserito come vicino del nodo winner ed entrambi i nodi gemelli vengono inibiti per una iterazione. Tale operazione ha lo scopo di assicurare che i due nodi non siano influenzati dai vicini dei winner della successiva iterazione.

Un altro settore di problemi trattati con questo tipo di reti neurali è quello del *Pattern recognition*. Tra i problemi in questo campo, significativa risulta essere la questione dell'autenticazione dell'impronta digitale, che rimane uno dei metodi più usati per la certificazione dell'identità personale. Un'interessante applicazione della autenticazione automatizzata fa uso di un sistema ibrido di reti neurali artificiali [Patt]. Il sistema utilizza due SOM per una iniziale classificazione delle impronte e tre reti feedforward multistrato allenate con la backpropagation per l'autenticazione finale. Per la classificazione iniziale si sono usate le SOM poiché sarebbe molto difficile creare un insieme di dati per l' allenamento supervisionato, che riescano accuratamente a classificare le impronte in insiemi distinti. I pattern di input della SOM sono vettori di 29 componenti che rappresentano caratteristiche estratte dalle immagini delle impronte digitali, quali la dimensione, l'orientamento dei segni e i momenti di inerzia. L'ultimo stadio del sistema consiste in una rete feedforward monostrato allenata con la backpropagation, con input provenienti dagli stadi precedenti e con output dei valori binari che simboleggiano la risposta di accettazione o di rifiuto dell'impronta.

Un altro dei campi di utilizzo delle reti neurali è quello dell'elaborazione di immagini di varia natura, in particolar modo di immagini mediche, per conseguire diverse tipologie di risultati, quali ad esempio l'individuazione di zone tumorali per la pianificazione di interventi chirurgici e lo studio dell'evoluzione delle malattie oncologiche. Sono state usate le self-organizing map per la scoperta e la localizzazione di masse tumorali nelle immagini mediche di un paziente [Seiffert2001]. Questo passo rappresenta solo un punto di partenza per il trattamento medico al quale il paziente sarà sottoposto, come ad esempio la *radioterapia*, durante la quale le cellule tumorali sono irradiate e distrutte, evitando nel frattempo di danneggiare i tessuti sani vicini.

Un'altra possibile applicazione in campo medico è il *matching 2D* di immagini mediche [Ph.D.S] , che consiste nella comparazione di coppie di immagini omologhe che rappresentano la stessa scena ma in tempi diversi o in condizioni diverse del paziente. Generalmente queste immagini sono slice estratti da un data-set ottenuto in seguito ad un esame di TAC (*Axial Computed Tomography*) o NMR (*Nuclear Magentic Resonance*), la cui descrizione è riportata nell' appendice A. L'obiettivo è quello di trovare una regola che controlli in qualche modo la modifica delle caratteristiche globali della prima immagine, in modo tale da corrispondere alla seconda. L'approccio seguito utilizza una SOM, i cui nodi corrispondono ai pixel nell'immagine mentre le sinapsi corrispondono ad alcune caratteristiche di tali pixel, quali la posizione, il livello di grigio, i gradienti, gli angoli ecc. Tale maching viene effettuato in una prima fase considerando solo le caratteristiche morfometriche, per poi essere completato in una fase successiva considerando anche le informazioni densitometriche.

In generale, tra l'acquisizione di un'immagine e l'altra , ad esempio nel caso di MR o TAC del cervello, il paziente mantiene la sua posizione e quindi gli slice corrispondenti fanno riferimento realmente a sezioni anatomiche corrispondenti. Può comunque accadere, ad esempio nel caso di MR o TAC dei polmoni, che il paziente respirando, alteri tale corrispondenza, o comunque che il paziente si muova, variando la sua posizione, durante l'acquisizione degli slice successivi.

Inoltre, nel caso di immagini acquisite in tempi diversi, il paziente avrà assunto certamente una diversa posizione, perciò gli slice corrispondenti presenteranno delle differenze, che possono essere dovute anche alla presenza di una nuova massa tumorale o ad un suo aumento o regressione nel tempo.

Sono state sviluppate molte applicazioni che fanno uso di questo tipo di SOM, dalle più semplici ed accademiche, a quelle molto più complesse, con valenza sia pratica che teorica.

Una caratteristica delle SOM è quella di essere in grado di effettuare anche *mappings* non lineari.

Tale proprietà è evidenziata in una complessa applicazione per il funzionamento di un braccio meccanico di un robot [Patt]. Un braccio meccanico è in grado di muoversi nello spazio. L'obiettivo è rappresentare la posizione di un oggetto target, registrata da due telecamere, con gli output di tre motori guida, che posizionano il braccio meccanico nella posizione desiderata. Le due telecamere offrono alla rete un'informazione stereo, ossia ciascuna posizione, data in input alla rete, rappresenta un punto nel piano dell'immagine di ciascuna delle due telecamere, per cui il loro output è rappresentato da vettori bidimensionali di coordinate. Tale output fornisce informazioni sulla posizione tridimensionale dell'oggetto target. La SOM è caratterizzata da

un lattice tridimensionale di unità che devono apprendere la corrispondenza degli input a quattro dimensioni, provenienti dalle telecamere, con l'unità più appropriata. Tale unità manda come output ai tre motori guida un vettore tridimensionale (Ψ, Θ, Φ), per posizionare il braccio meccanico nella posizione target desiderata.

Sono stati inoltre sviluppati applet java che simulano l'apprendimento di una SOM, fornendo la visualizzazione di figure 3D non lineari, come mostra la figura 1.6. Partendo da una distribuzione casuale dei punti nello spazio 3D, si è arrivati ad ottenere la forma desiderata, dando come input alla SOM, i vettori posizione di 10 punti nello spazio.

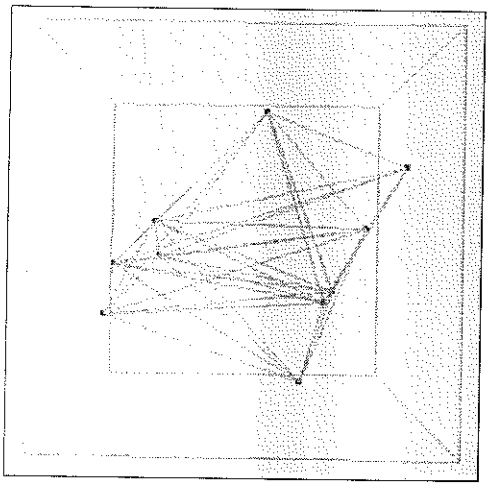

Figura1.6. Visualizzazione di una forma tridimensionale.

Sono state studiate SOM multilayer per la classificazione delle proprietà di un'immagine [Seiffert_multi] e [Seiffert_three]. Più precisamente, attraverso l'uso di reti neurali e di informazioni a-priori sui contenuti della storia di una sequenza di immagini si è riusciti a

migliorare l'accuratezza e la velocità, nella stima dei parametri di movimento, nel caso di oggetti distorti o sovrapposti . A tal proposito sono state usate delle SOM tridimensionali con input bidimensionali, come delle memorie associative.Nella figura 1.7, è rappresentata l'idea di base di questo processo. Dopo l'acquisizione dell'immagine, questa è suddivisa in diverse parti o blocchi (1) e successivamente è calcolata la correlazione (2) tra le parti corrispondenti nell'immagine precedente. Una rete neurale poi rappresenta un sistema di riconoscimento della

massima posizione della funzione di correlazione (3) ed una memoria associativa (SOM multilayer) precedentemente allenata con informazioni disponibili a priori realizza la classificazione prendendo come input la funzione di correlazione (4).

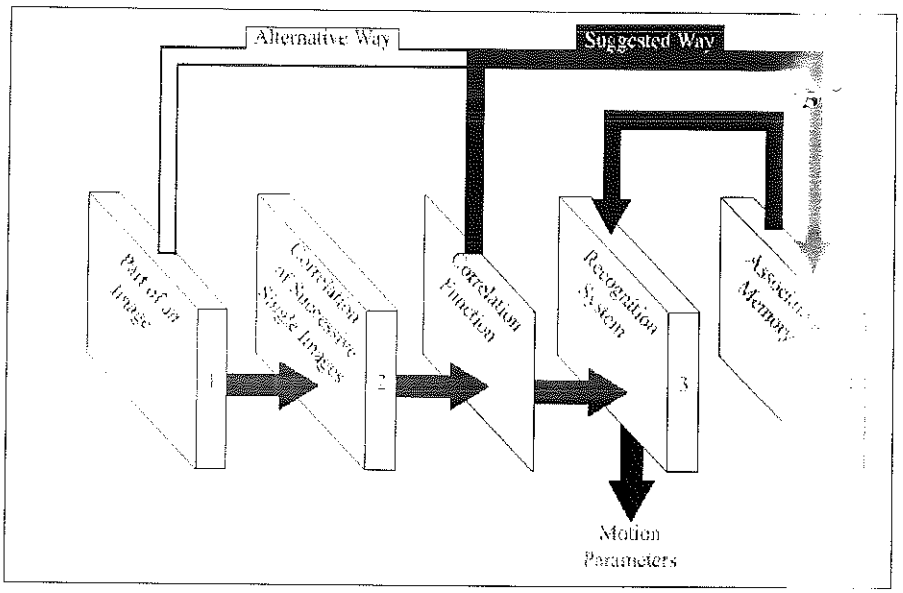

Figura 1.7. Schema del processo.

Ogni strato della *SOM* corrisponde ad una particolare classe

La topologia della SOM multilayer è mostrata nella figura 1.8, riportata di seguito.

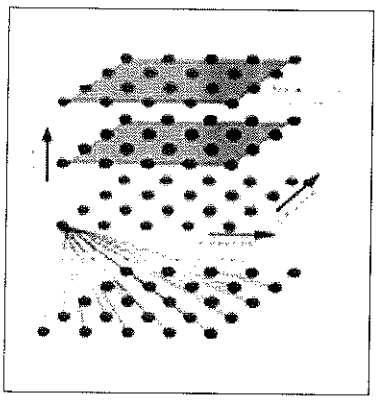

Figura 1.8. Struttura della SOM tridimensionale

Come detto in precedenza, la SOM multilayer è usata come *Memoria Associativa*, ossia, una volta allenata, la rete è richiamata con dei dati disturbati (creati generalmente aggiungendo del rumore ai dati dell'insieme di allenamento) ed è stato verificato che fino ad un ammontare del 25% di rumore, la rete riconosce la classe corretta.

Un'ulteriore variazione della SOM consiste nel non definire a priori il numero di strati, ma nel considerare la possibilità di aumentare tale numero al momento del bisogno.

Si parla quindi di Growing 3D-SOMs con input bidimensionali [Seiffert_three]. La figura 1.9, mostra il processo di crescita della rete.

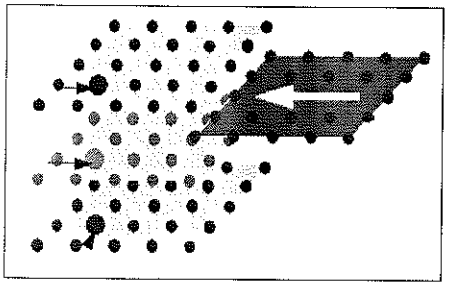

Figura1.9. Processo d'inserzione nuovo strato.

Per decidere dove inserire il nuovo strato della SOM, si va a cercare il neurone più frequentemente vincente,indicato con q ed il neurone vicino con il più grande valore d'errore,indicato con r. Il nuovo piano sarà inserito tra gli strati determinati dai neuroni q e r.

Altri usi di queste reti di Kohonen tridimensionali, hanno portato alla messa a punto del *Quasi-Four-Dimensional_Neurocube* (QFDN-Network) nell'ambito dei sistemi di rivelazione del movimento già accennato in precedenza [Seiffert_multi].

Questa nuova topologia di rete, la QFDN-net, rappresenta una ulteriore variazione della SOM multilayer vista precedentemente. Nella figura 1.10, è mostrata la topologia della QFDN-net, con una dimensione di 3*5*6*5 neuroni con 15 sottomappe 6*5 e con la definizione di due differenti aree di nodi vicini. Usando questa architettura di rete, è emerso che è possibile formare diverse classi consistenti in blocchi di funzioni di correlazione, o di parti dell'immagine con una certa gradazione di grigio, ordinate in base alle similarità geometriche.

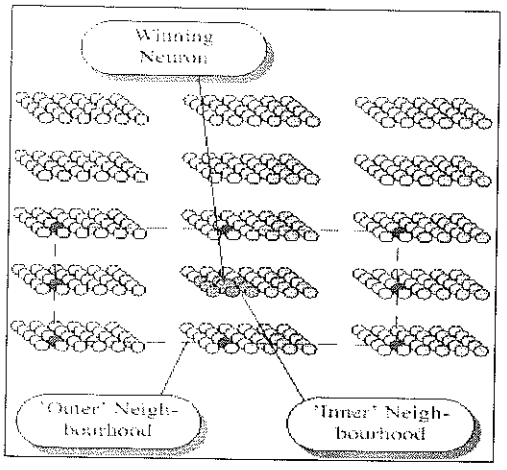

Figura 1.10. Topologia del QFDN.

A regime, l'allenamento alterna fasi di aggiornamento dei pesi a fasi d'inserzione di nuove sottomappe, come si può vedere nella figura 1.11. Il processo di crescita della dimensione della rete è arrestato quando si è ottenuta la qualità di allenamento desiderata.

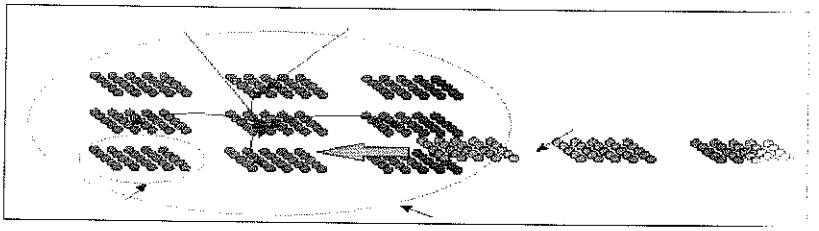

Figura 1.11. Processo di crescita del QFDN.

Un altro settore d'applicazione delle *SOM* è quello finanziario. A tal proposito, è stata fatta una comparazione tra le 2D e le 3D SOM, per la visualizzazione di dati finanziari e loro relazioni [Kiviluoto].

L'intento è quello di fornire una caratterizzazione delle differenze tra i rapporti finanziari di compagnie che falliscono e di quelle che non falliscono. Per raggiungere tale scopo è stata usata una SOM in grado di evidenziare quelle aree dove occorrevano maggiori situazioni di bancarotta. Nelle figure 1.17 e 1.18, sono riportati i vettori peso delle SOM 2D e 3D, mostrati nello spazio degli input. Nelle figure 1.19 ed 1.20, si mostrano i risultati ottenuti per la classificazione di due anni di dati, , usando entrambe le strutture di *SOM* per scoprire situazioni di possibile bancarotta per le società finanziare.

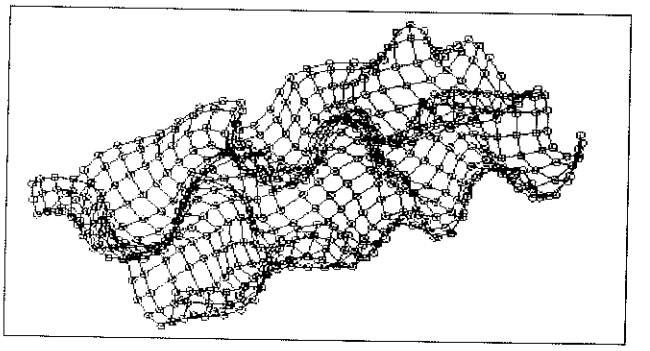

Figura 1.17. Vettori peso della SOM 2D.

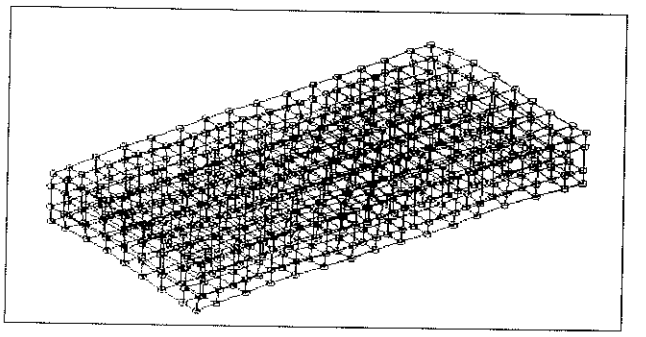

Figura1.18. Vettori peso della 3D SOM.

Dall'analisi dei risultati è emerso che, le SOM 3D si sono comportate meglio raggruppando dati simili in un unico cluster. Per le reti 2D, invece, è successo che una particolare parte dello spazio degli input è stata rappresentata in diverse aree non vicine della mappa. Se ne è dedotto che in quei casi in cui le dimensioni intrinseche dei dati in input sono più elevate di quelle della mappa della rete, i risultati della visualizzazione dei dati devono essere letti con più cautela.

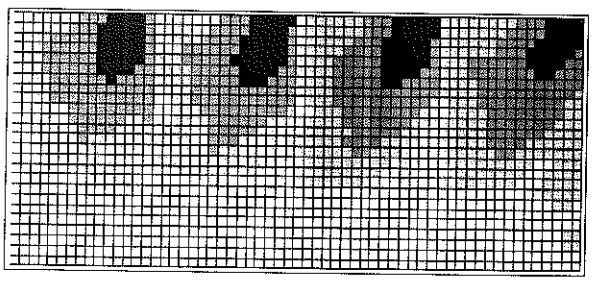

Figura1.19. Clustering della 3D SOM.

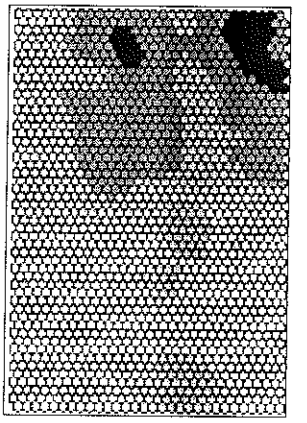

Figura 1.20. Clustering della 2D SOM

Bibliografia

[Patt] **Patterson, Dan W.** Artificial neural networks : theory and applications. *pages. 367-404.*

[Haykin99] **Simon Haykin.** Neural Networks: a comprrehensive foundation. *Prentice Hall, Inc, 1999. Pp 443-477.*

[Lakany] **H. M. Lakany, G.M. Hayes.** Object Localisation in 2D images using A Temporal Kohonen Network. *University of Edinburgh, Department of Artificial Intelligence.*

[Chappelier] **J. Chappelier, A. Grumbach.** A Kohonen Map fpr Temporal Sequences. *Dèpartement Informatique - Groupe MILC, Paris.*

[Freeman92] **James A. Freeman, David M. Skapura.** Neural Networks. Algorithms, Applications and Programming Techniques. *Addison-Wesley Publishing Company, july 1992.*

[Koho95] T. Kohohnen. Self-organizing maps. *Pubblication: Berlin, Springer, 1995.*

[Koho84] **T. Kohohnen.** Self-Organization and Associative Memory. *Springer-Verlag, New York, 1984.*

[RS99] **Raúl Saavedra.** Self-Organizing Map as a clustering tool in monitoring deterioration of three-phase induction motors. *A Thesis submitted the 24 of March 1999, to the Department of Electrical Engineering and Computer Science of the school of Engineering of Tulane University, in partial fulfillment of the requirements for degree of Master of Science in Computer Science.* Copyright 1999 by Raúl Saavedra

[Seiffert_three] **Udo Seiffert, Bernd Michaelis.** Three-dimensional Self-Organizing Maps for Classification of Image properties. *Institute for Measurement Technology and Electronics, Magdeburg, Germany. www: http://ipe.et.uni-magdeburg.de.*

[Varsta2000] **Markus Varsta, Jukka Heikkonen, Jouko Lampinen.** Analytical Comparison of the Temporal Kohonen Map and the Recurrent Self-Organizing Map. *ESANN'2000 proceedings. European Symposium on Artificial Neural Networks, Bruges(Belgium), 26-28 April 2000, pp. 273-280.*

[Seiffert_multi] **Udo Seiffert, Bernd Michaelis**. Growing Multi-dimensional Self-Organizing Maps. *Institute for Measurement Technology and Electronics, Magdeburg, Germany. www: http://ipe.et.uni-magdeburg.de.*

[Seiffert_ Growing] **Udo Seiffert, Bernd Michaelis**. Growing 3D-SOM's with 2D-Input Layer as a classification Tool in a Motion Detection System. *Institute for Measurement Technology and Electronics, Magdeburg, Germany. www: http://ipe.et.uni-magdeburg.de.*

[Shunemann] **Stefan Shunemann, Udo Seiffert, Bernd Michaelis**. Two More Modifications of SOMs to Handle Signals with Special Properties. *Institute for Measurement Technology and Electronics, Magdeburg, Germany. www: http://ipe.et.uni-magdeburg.de.*

[Kiviluoto] **Kimmo Kiviluoto**. Comparing 2D and 3D Self-Organizing Maps in Finantial Data Visualization. *Laboritory of Computer and Information Science, Helsinki University of Technology, Finland. E-mail: Kimmo.Kiviluoto@hut.fi.*

[Seiffert2001] **Angelo Zizzarri, Udo Seiffert, Bernd Michaelis, Guenther Gademann, Sebastian Swidreski**. Detection of Tumor in Digital images of the Brain. *Proceedings of the IASTED International Conference Signal Processing, Pattern recognition and Applications, July 3-6 2001, Rhodes, Greece.*