

Introduzione alle Reti Neurali

Marco Botta

Dipartimento di Informatica, Università di
Torino

Corso Svizzera 185 - 10149 Torino

Giornata sul Data Mining

20 Febbraio 2002

Botta - Dip. di Informatica, Univ. Torino

Reti Neurali 1

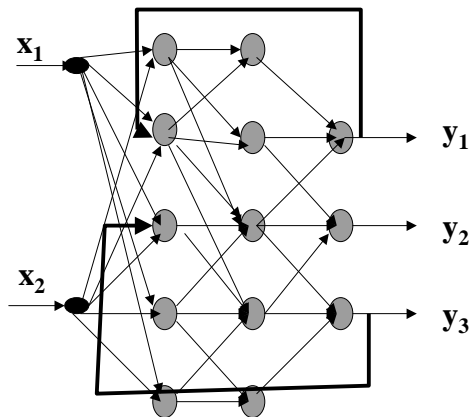
Reti Neurali

- Sono approssimatori di funzioni
- L'idea è ricostruire una funzione tramite la **composizione di unità elementari**, ciascuna delle quali è in grado di eseguire poche e semplici computazioni
- Le unità sono dette **neuroni** (per motivi storici)

Botta - Dip. di Informatica, Univ. Torino

Reti Neurali 2

Le Reti Neurali



Reti di funzioni elementari
dove le uscite dell'una sono
in ingresso all'altra

Reti Feed-forward

Reti ricorrenti

Reti auto-organizzanti

Molte altre sottofamiglie....

Botta - Dip. di Informatica, Univ. Torino

Reti Neurali 3

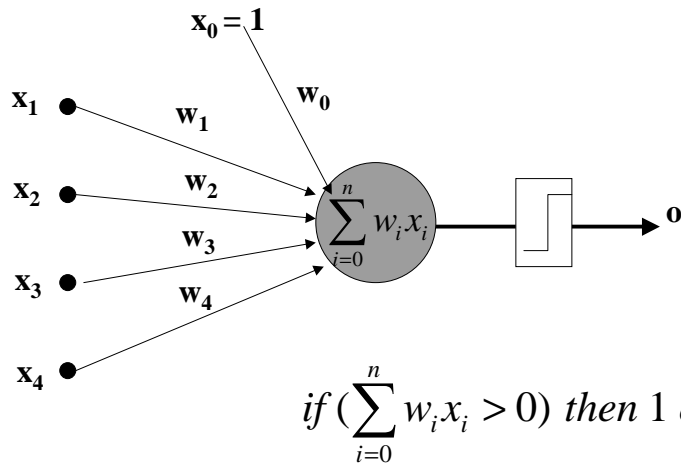
Caratteristiche

- Le reti neurali non eseguono istruzioni programmate ma rispondono a degli input tramite un procedimento parallelo
- Non sono sequenziali (né deterministiche)
- Danno risposte anche in corrispondenza di input mai visti (capacità di generalizzazione)
- Incapacità di spiegare i risultati

Botta - Dip. di Informatica, Univ. Torino

Reti Neurali 4

La rete più semplice: 1 solo neurone (Perceptron)

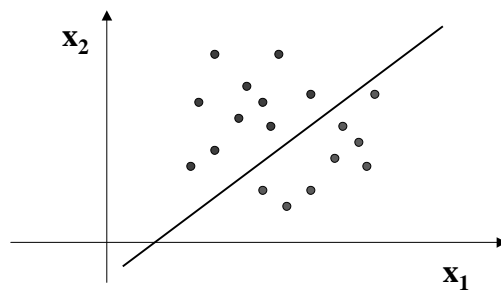


Botta - Dip. di Informatica, Univ. Torino

Reti Neurali 5

Qual'è la semantica

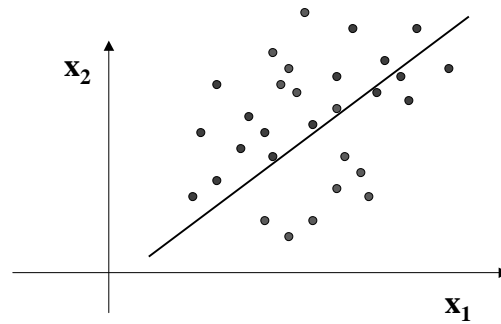
$$\sum_{i=0}^n w_i x_i = 0$$



Botta - Dip. di Informatica, Univ. Torino

Reti Neurali 6

Che cosa non può descrivere

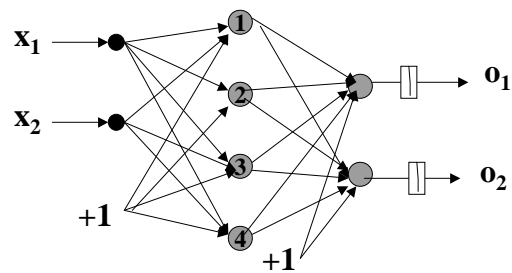


Si può comunque cercare di posizionare la retta in modo da minimizzare l'errore.

Botta - Dip. di Informatica, Univ. Torino

Reti Neurali 7

Reti a più strati

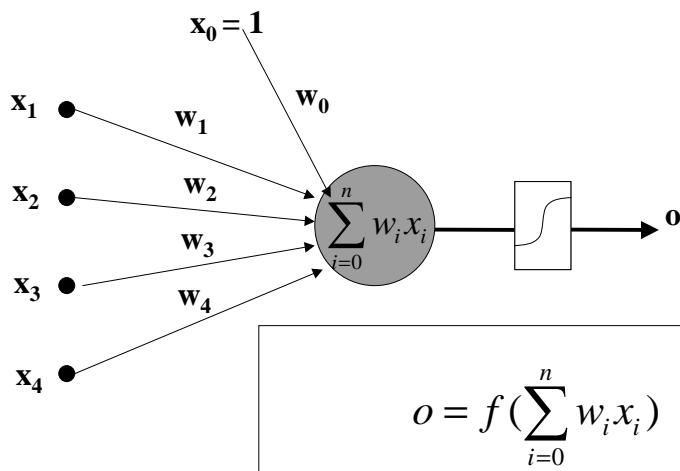


Se gli strati sono puramente lineari non si acquista potenza

Botta - Dip. di Informatica, Univ. Torino

Reti Neurali 8

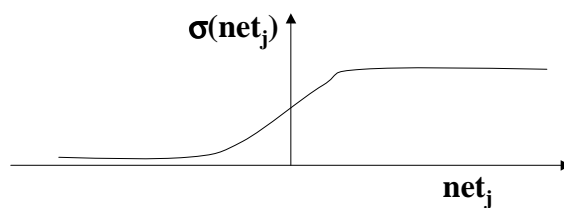
Funzioni non-lineari per disaccoppiare gli stadi



Botta - Dip. di Informatica, Univ. Torino

Reti Neurali 9

Squashing Functions



$$\text{net}_j = \sum_i w_{ji} x_i \quad \sigma(\text{net}_j) = \frac{1}{1 + e^{-\lambda \text{net}_j}}$$

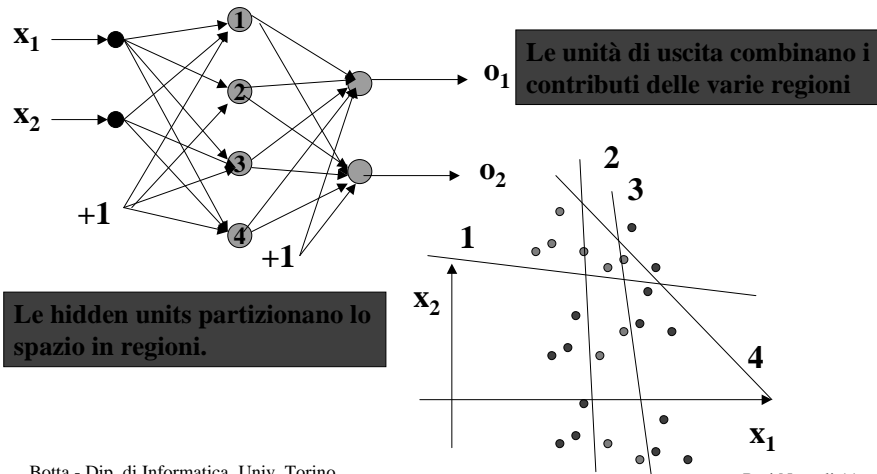
Altre forme: Tanh

**Poniamo per ora la richiesta che la funzione sia
monotona crescente**

Botta - Dip. di Informatica, Univ. Torino

Reti Neurali 10

Il Multilayer Perceptron



Botta - Dip. di Informatica, Univ. Torino

Reti Neurali 11

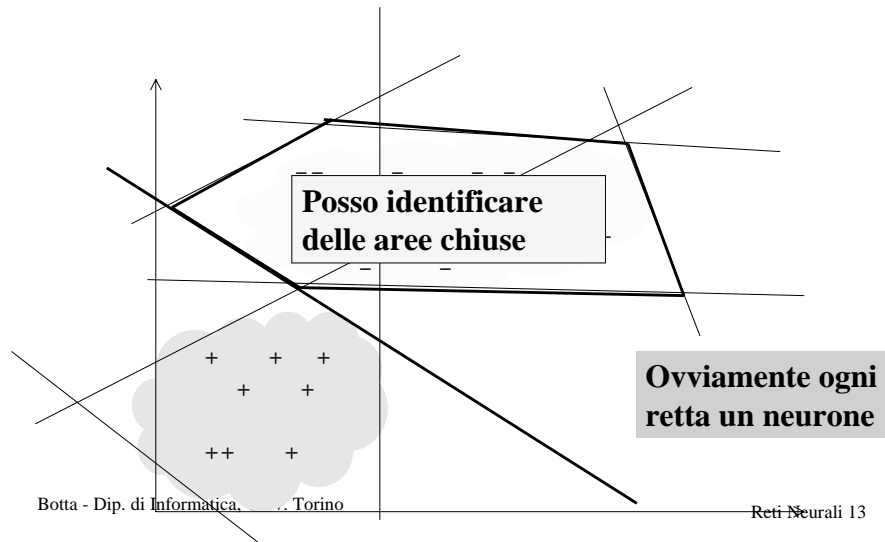
Quale è la potenza?

- Una rete a due livelli con sigmoidi (o simili) su tutte le unità, può rappresentare qualunque funzione booleana finita.
- Ogni funzione reale $\mathbb{R}^n \rightarrow \mathbb{R}$ limitata e continua, può essere approssimata con errore arbitrariamente piccolo da una rete a due livelli con sigmoidi sulle unità nascoste e unità lineari sull'uscita.
- Una qualunque funzione può essere approssimata da una rete a tre livelli in cui l'ultimo livello è costituito da funzioni lineari.

Botta - Dip. di Informatica, Univ. Torino

Reti Neurali 12

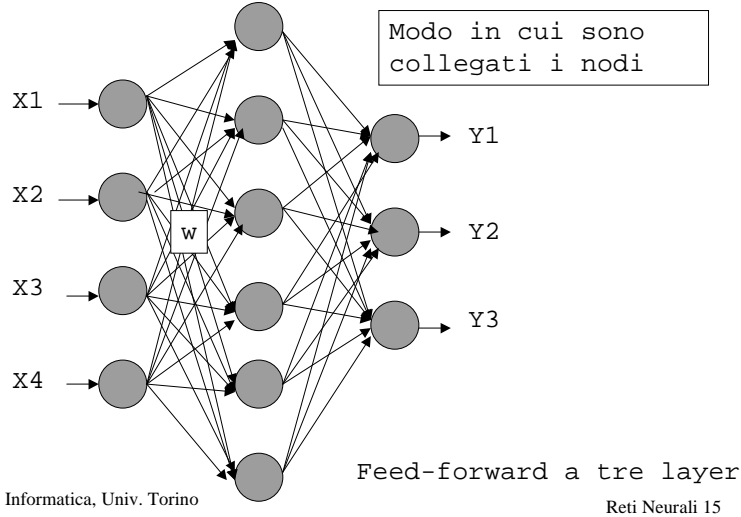
Uso tante rette



Come posizionare le rette?

- Cominciamo col posizionarne un certo numero a caso
- Utilizziamo un insieme di esempi noti (per i quali conosco il valore della funzione da approssimare) calcolando per ciascuno la differenza fra output ottenuto e output atteso ed utilizzando tale errore per modificare posizione e orientamento dei neuroni
- È possibile compiere questa operazione in modo automatico

Topologia



Botta - Dip. di Informatica, Univ. Torino

Reti Neurali 15

Structure	Regions	XOR	Meshed regions
single layer 	Half plane bounded by hyper-plane		
two layer 	Convex open or closed regions		
three layer 	Arbitrary (limited by # of nodes)		

Botta

Neurali 16

Gradi di libertà

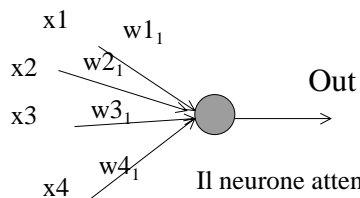
- Per approssimare bene una funzione occorre un numero sufficiente di neuroni hidden. Quanti? Non esiste modo di prevederlo
- I pesi vengono modificati durante la fase di apprendimento, ma le funzioni calcolate da ogni singolo neurone possono avere parametri che l'utente deve regolare

Apprendimento

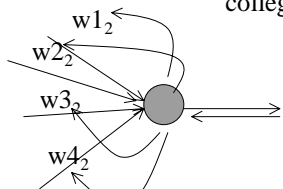
- I pesi di una rete neurale sono inizializzati a caso
- La rete sarà un pessimo approssimatore della funzione di interesse
- Modificando i pesi modifico la funzione realizzata dalla rete
- Posso usare gli esempi per modificare i pesi dimodoché la rete divenga un approssimatore più preciso

Apprendimento

(rete costituita da un solo neurone)



Il neurone attende che giunga un valore su ogni collegamento in input poi calcola il **Net** e un **valore di attivazione** che viene sparato su tutti i collegamenti in uscita



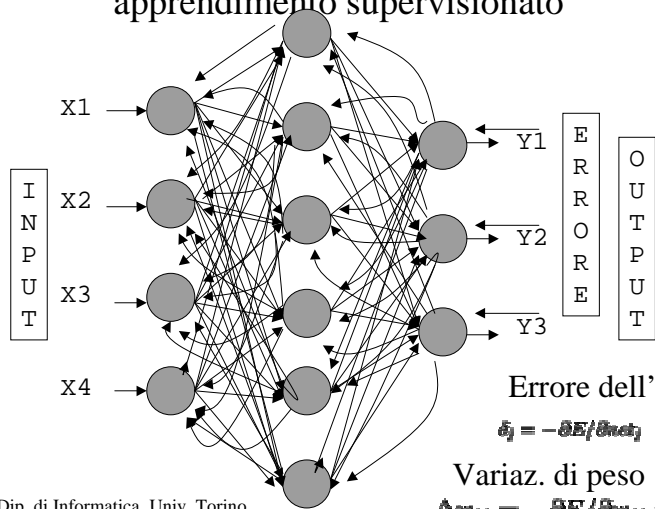
Errore = Target - Out

$w_{i2} = w_{i1} + \alpha * \text{Errore}$

$0 < \alpha \leq 1$

Back-propagation

apprendimento supervisionato



Errore dell'unità j-ma

$\delta_j = -\partial E / \partial net_j$

Variation of weight

$\Delta w_{ij} = -\partial E / \partial w_{ij}$

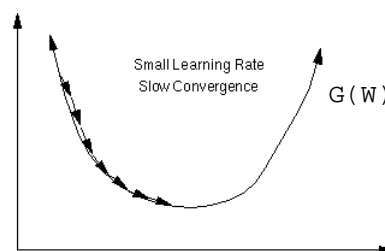
Errore = $G(W)$

- Il metodo descritto si basa sulla constatazione che se l'errore cambia al cambiare dei pesi allora l'errore può essere inteso come una funzione dei pesi medesimi $G(W)$
- $G(W)$ avrà dei punti in cui il suo valore è minimo (corrispondenti a configurazioni di pesi)
- Tali punti possono essere cercati con le tecniche di ricerca del minimo di una funzione, che si basano sullo studio della sua derivata

Botta - Dip. di Informatica, Univ. Torino

Reti Neurali 21

Discesa del gradiente



- Ad ogni iterazione i pesi vengono modificati di una percentuale μ (learning rate)
- μ piccolo comporta un apprendimento più lento ma spesso più accurato

Botta - Dip. di Informatica, Univ. Torino

Reti Neurali 22

Discesa del Gradiente

Il fatto di muoversi nella direzione indicata dal gradiente non significa necessariamente raggiungere il minimo della funzione!

- Se il passo è troppo lungo potrei saltare il minimo
- Oltre al minimo globale potrebbero esistere dei minimi locali

Discesa del Gradiente (Algoritmo)

```
1.Initialize all weights to small random values.
2.REPEAT until done
  1.For each weight  $w_{ij}$  set  $\Delta w_{ij} := 0$ 
  2.For each data point  $(x, t)^p$ 
    a.set input units to  $x$ 
    b.compute value  $y_j$  of output units
    c.For each weight  $w_{ij}$  set  $\Delta w_{ij} := \Delta w_{ij} + (t_j - y_j) z_j$ 
  3.For each weight  $w_{ij}$  set  $w_{ij} := w_{ij} + \mu \Delta w_{ij}$ 
```

In questo caso μ è il learning rate

Epoche di apprendimento

- La rete neurale viene addestrata su di un **learning set**
- Ogni esempio viene passato attraverso la rete calcolando l'errore che verrà usato per modificare i pesi
- Il learning set viene passato varie volte ogni volta si chiama **epoca di apprendimento**
- L'**errore medio** dovrebbe diminuire man mano
- Troppe epoche possono generare **over-fitting** (eccessiva specializzazione sulle istanze contenute nel learning set)

Problemi legati al dominio

- Maggiore il numero di dimensioni (spazio degli input) maggiore la possibile confusione
- **Rilevanza delle feature e loro dominio: x_1 può variare fra 0.05 e 0.02 mentre x_2 può variare fra 100 e 10000, numericamente x_2 pesa più di x_1 ma x_1 potrebbe essere più rilevante nel definire il comportamento della funzione**

Problemi legati ai dati

Per eseguire l'apprendimento occorrono dei dati

- I dati possono essere **costosi** da ottenere
- Possono essere **poco rappresentativi** in quanto concentrati in un'area particolare del dominio
- Possono essere **affetti da errore**

Quante Hidden Units?

Usando un numero arbitrariamente grande di unità nascoste si può approssimare qualunque funzione finita in un dominio finito.

Non c'è regola a priori per determinarne il numero

Il numero di unità nascoste definisce il numero di partizionamenti fattibili sul dominio. Aumentandone il numero si ottengono suddivisioni molto fini (si può arrivare ad un campione per regione!!!!)

Quante Hidden Units?

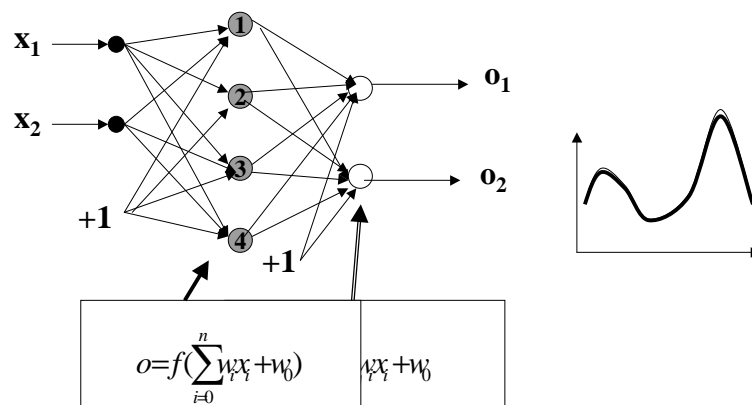
Se si ha un'idea del problema si può stimare il loro numero (stati da codificare) se no si va a tentativi

Un suggerimento è di cominciare provare con numeri piccoli ($\log_2(N_{in}+N_{out})$) e crescere progressivamente

Botta - Dip. di Informatica, Univ. Torino

Reti Neurali 29

Regressione



Botta - Dip. di Informatica, Univ. Torino

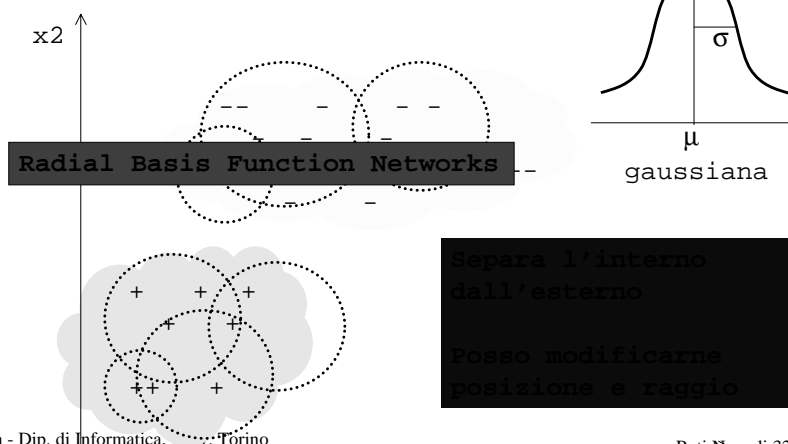
Reti Neurali 30

Altri modelli

Botta - Dip. di Informatica, Univ. Torino

Reti Neurali 31

Non solo rette!



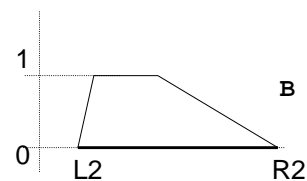
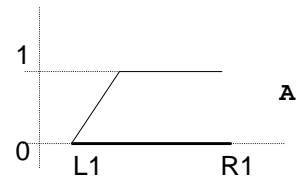
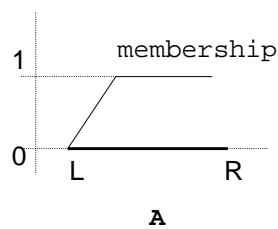
Botta - Dip. di Informatica, Univ. Torino

Reti Neurali 32

Regole fuzzy e RBF

If (A and B) then C

Vecchio caldo rigido ...



Botta - Dip. di Informatica, Univ. Torino

Reti Neurali 33

Apprendimento non supervisionato

Es. Mappe di Kohonen

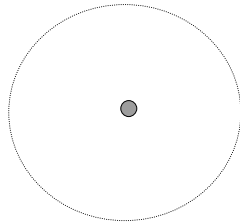
Eseguono un **clustering**

- si basano sulla somiglianza fra i dati
- creano raggruppamenti di dati simili
- determinano l'appartenenza di un input ad un gruppo
- tipicamente sono usate per ridurre il numero delle feature nello spazio di rappresentazione

Botta - Dip. di Informatica, Univ. Torino

Reti Neurali 34

Somiglianza = Vicinanza



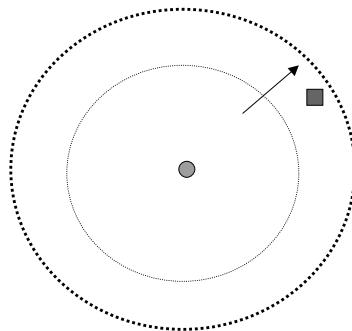
Neurone: ha una **sfera di influenza**

Quando la rete riceve un input in fase di apprendimento viene **selezionato il neurone più simile** (più vicino nello spazio degli input) al dato in questione e si opera secondo uno dei seguenti casi

Botta - Dip. di Informatica, Univ. Torino

Reti Neurali 35

Caso 1 input simile

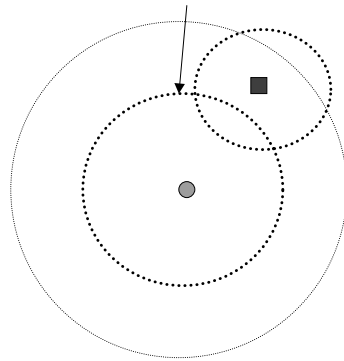


Botta - Dip. di Informatica, Univ. Torino

Reti Neurali 36

Caso 2

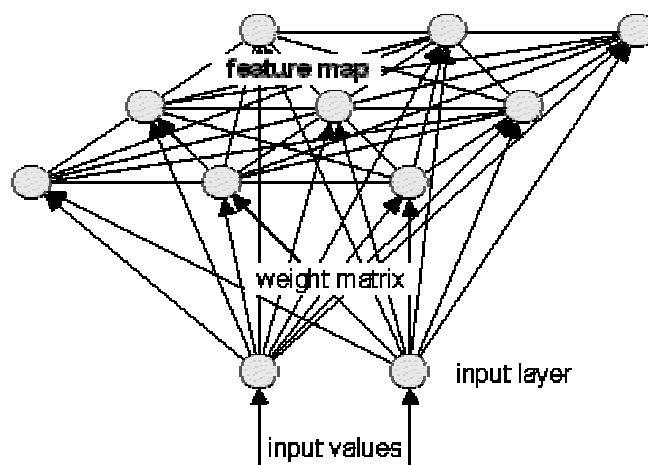
input diverso



Botta - Dip. di Informatica, Univ. Torino

Reti Neurali 37

struttura della rete



Botta - Dip. di Informatica, Univ. Torino

Reti Neurali 38

Self-growing networks

apprendimento competitivo

Sogno: avere una rete che si costruisce da sola
introducendo nuovi neuroni se necessario ...

Tali reti esistono - alcuni modelli:

- **growing cell structures (Fritzke)**
- **neural gas**
- **competitive Hebbian learning**
- ...

<http://www.ki.inf.tu-dresden.de/~fritzke/research/incremental.html>

Botta - Dip. di Informatica, Univ. Torino

Reti Neurali 39

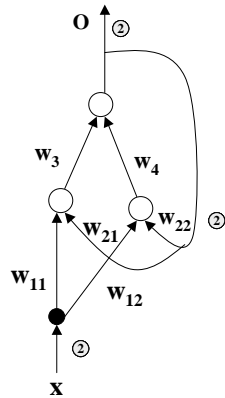
Idea base

- **Hard-competitive Learning**
- Dato un input un solo neurone verrà modificato - intuitivamente quello che codifica il caso più simile all'input in questione
- Spesso i neuroni sono organizzati in una struttura di **vicinato**
- In molti casi anche i vicini del vincitore vengono modificati (in misura minore)
- Di tanto in tanto vengono aggiunti/eliminati neuroni

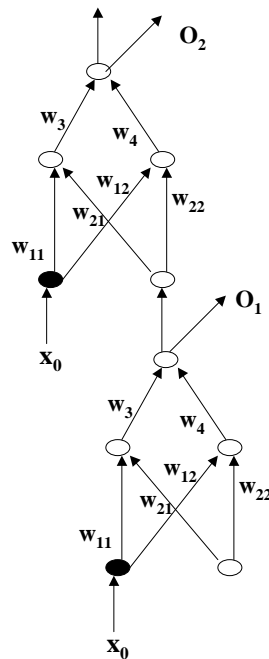
Botta - Dip. di Informatica, Univ. Torino

Reti Neurali 40

Reti Ricorrenti



Botta - Dip. di Informatica, Univ. Torino



Reti Neurali 41

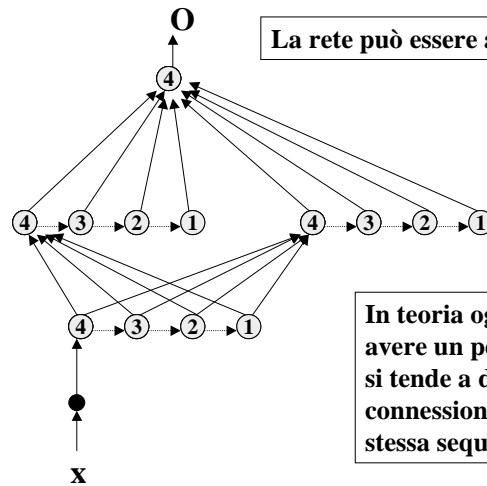
Come si addestra una rete ricorrente

- **Il comportamento di una rete ricorrente in una sequenza di istanti può essere riprodotto da una rete feedforward costruita come nella slide precedente.**
- **Note le uscite che devono essere prodotte, la rete può essere addestrata usando la Back-propagation**
- **Le $[W]$ nei vari stadi sono le stesse che quindi ricevono l'accumulo delle varie correzioni riportate negli stadi concatenati.**
- **l'uscita può non essere nota a tutti i passi.....**

Botta - Dip. di Informatica, Univ. Torino

Reti Neurali 42

Reti a Time Delay



La rete può essere addestrata usando la BP

In teoria ogni connessione può avere un peso diverso. In pratica si tende a dare lo stesso peso alle connessioni che appartengono alla stessa sequenza

Botta - Dip. di Informatica, Univ. Torino

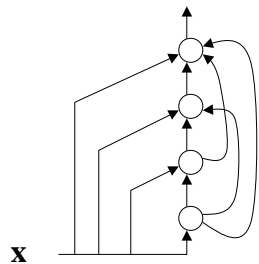
Reti Neurali 43

Altri Tipi di Reti Basate sul Perceptron

Reti costruite incrementalmente

Cascade Correlation:
si aggiunge uno strato per correggere l'errore degli strati precedenti

Perceptron trees:
come decision tree ma le condizioni sono definite da un perceptron



Oblique trees:
le condizioni sono definite da un iperpiano obliquo

Botta - Dip. di Informatica, Univ. Torino

Reti Neurali 44

Ancora.....

Tecniche di potatura:

- Si costruisce una rete sovradimensionata che viene ridotta tagliando le connessioni che sono inutili (quelle che sono poco influenti per l'errore)

Tecniche ibride Simboliche Numeriche:

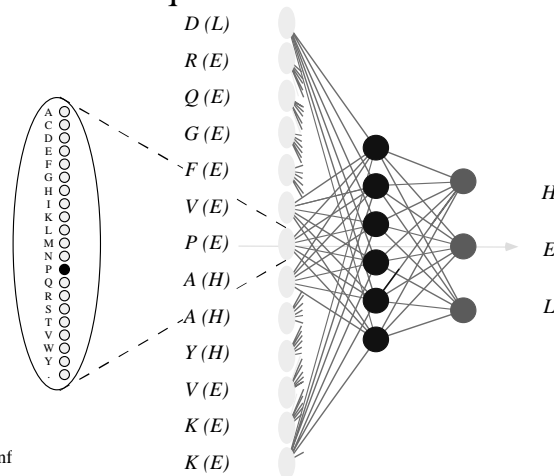
- **KBANN**: si usa una rete di regole proposizionali per costruire il grafo di connessione
- **Rapture**: si usa un albero di decisione per aggiungere neuroni ad una rete che deve essere potenziata

Botta - Dip. di Informatica, Univ. Torino

Reti Neurali 45

Un esempio 'biologico' (Qian e Sejnowsky)

- Predizione della struttura secondaria delle proteine a partire dalla sequenza di amminoacidi



Botta - Dip. di Inf

Reti Neurali 46

Effetti del dimensionamento della finestra

	Window Size	Q ₃ (%)	C _α	C _β	C _{coil}
Accuratezza della predizione in dipendenza dalla dimensione della finestra	1	53.90	0.11	0.14	0.17
	3	57.70	0.22	0.20	0.30
	5	60.50	0.28	0.26	0.37
	7	61.90	0.32	0.28	0.39
	9	62.30	0.33	0.28	0.38
	11	62.10	0.36	0.29	0.38
	13	62.70	0.35	0.29	0.38
	15	62.20	0.35	0.31	0.38
	17	61.50	0.33	0.27	0.37
	21	61.60	0.33	0.27	0.32

Botta - Dip. di Informatica, Univ. Torino

Reti Neurali 47

Simulazioni per imparare

<http://home.cc.umanitoba.ca/~umcorbe9/anns.html#Applets>
Contiene simulazioni (applet) + file sorgenti in java

Botta - Dip. di Informatica, Univ. Torino

Reti Neurali 48

Alcune risorse

<http://www.kcl.ac.uk/neuronet/>
Rete di eccellenza NEuroNet

<http://www-ra.informatik.uni-tuebingen.de/SNNS/>
Stuttgart neural network simulator

FINE