



**UNIVERSITÀ  
DEGLI STUDI  
DI FOGGIA**



HR EXCELLENCE IN RESEARCH

# **Reti Neurali Artificiali**

**Master in Gestione dei Dati e Bioimmagini**

**a.a. 2012/2013**

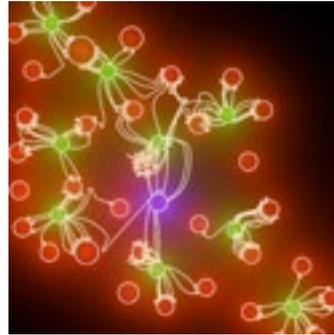
**Prof. Crescenzo Gallo**

**Dipartimento di Medicina Clinica e Sperimentale**

**C.GALLO@UNIFG.IT**



# Le Reti Neurali



Una rete neurale è una rappresentazione artificiale del cervello umano che cerca di simulare il processo di apprendimento.

Il termine "artificiale" significa che le reti neurali sono implementate in programmi per computer che sono in grado di gestire il gran numero di calcoli necessari durante il processo di apprendimento.

Per mostrare dove le reti neurali hanno la loro origine, diamo uno sguardo al modello biologico: il cervello umano.

# Le Reti Neurali

## Il modello biologico: il cervello umano



Il cervello umano è composto da un numero elevato ( $\sim 100$  miliardi) di cellule neurali che elaborano le informazioni.

Ogni cellula funziona come un processore semplice e solo la massiccia interazione tra tutte le cellule<sup>(\*)</sup> e la loro elaborazione in parallelo rende possibili le capacità cerebrali.

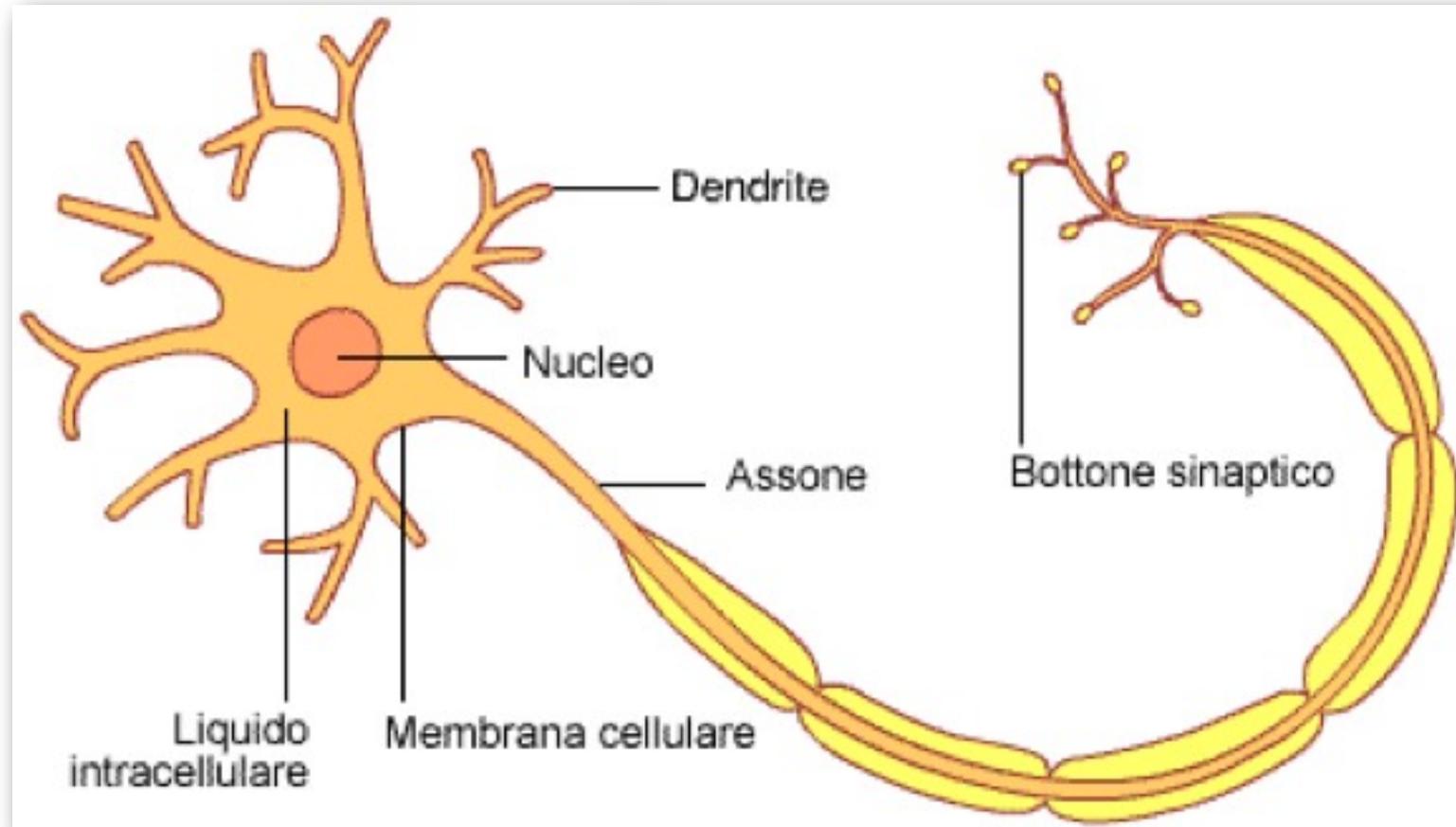
Ecco uno schema di tale cellula neurale, chiamata **neurone**:

---

<sup>(\*)</sup> *Ogni cellula sviluppa in media circa 10 000 connessioni con le cellule vicine.*

# Le Reti Neurali

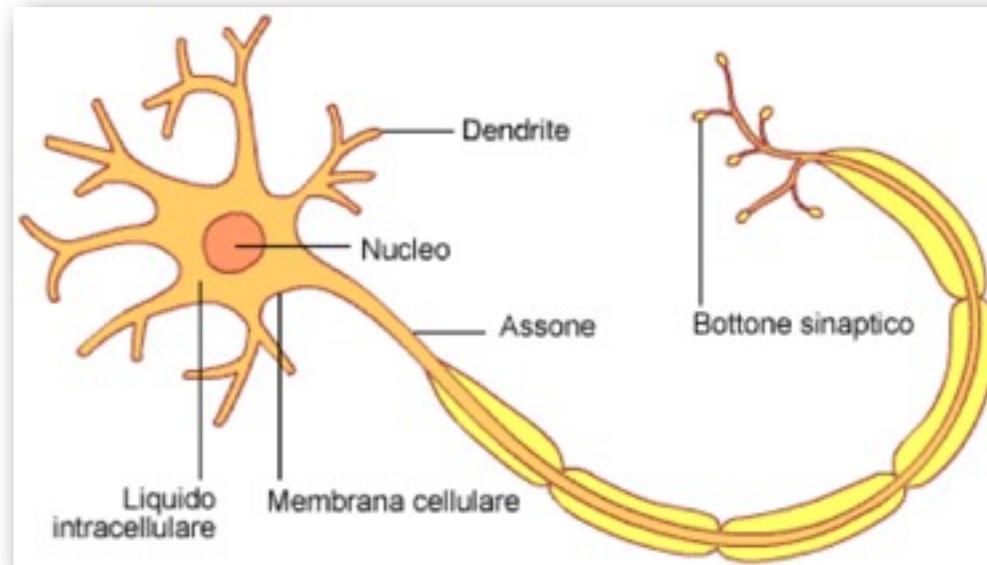
## Il modello biologico: il cervello umano



Come indica la figura, un neurone è costituito da un **nucleo**, da **dendriti** per le informazioni in entrata e un **assone** con dendriti per le informazioni in uscita che vengono passate ai neuroni connessi (**sinapsi**).

# Le Reti Neurali

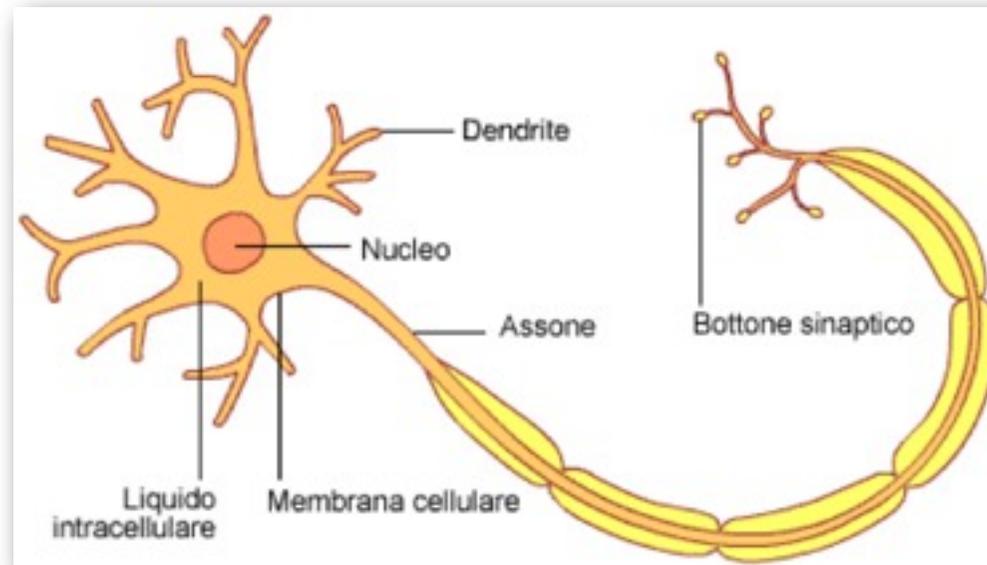
## Il modello biologico: il cervello umano



- L'informazione è trasportata tra i neuroni in forma di stimoli elettrici lungo i dendriti.
- Le informazioni in entrata che raggiungono i dendriti del neurone vengono sommate e poi inviate lungo l'assone ai dendriti alla sua estremità (sinapsi), dove l'informazione in uscita passa ai neuroni collegati, se supera una certa soglia. In questo caso, il neurone si dice essersi “**attivato**”.

# Le Reti Neurali

## Il modello biologico: il cervello umano

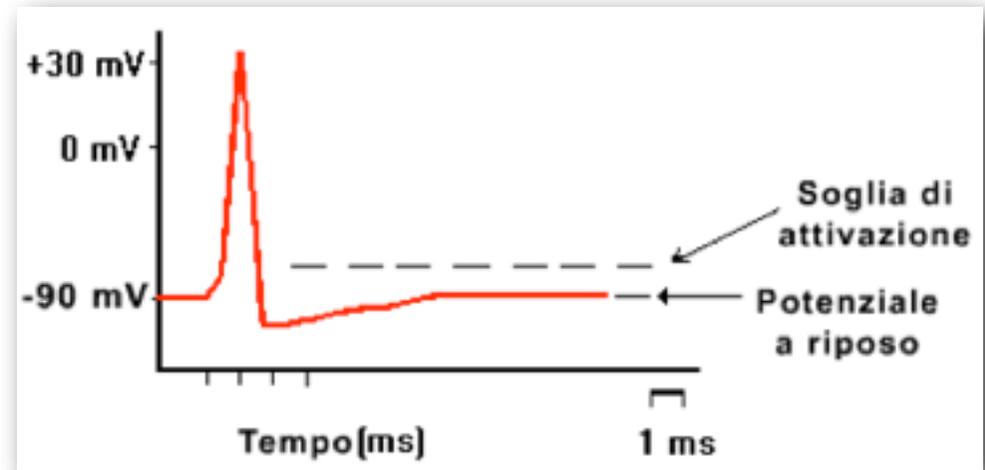


- Se la stimolazione in arrivo non è sufficiente, le informazioni non vengono trasportate oltre. In questo caso, il neurone si dice che è **“inibito”**.
- Le connessioni tra i neuroni sono adattative, che cosa significa che la struttura delle connessioni cambia dinamicamente.
- È comunemente riconosciuto che la capacità di apprendimento del cervello umano si basa su questo adattamento.

# Le Reti Neurali

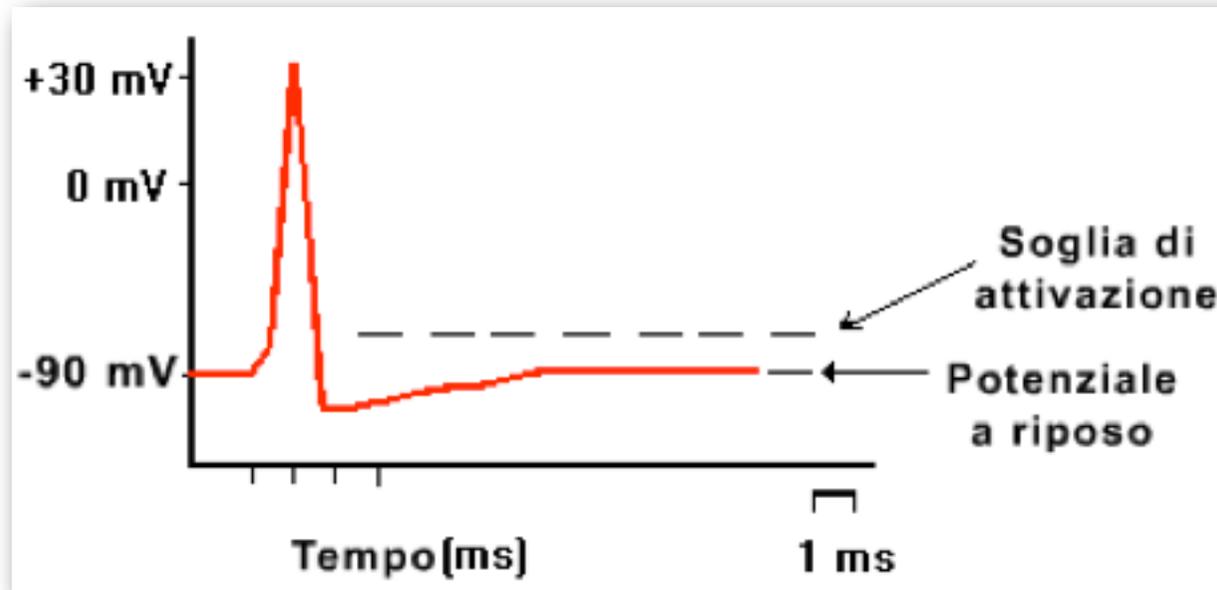
## Il modello biologico: il cervello umano

- Il neurone, dato che può emettere o meno un segnale elettrico avrà anche una **soglia di attivazione**.
- Fin quando la **membrana** del neurone resta indisturbata non si origina alcun potenziale d'azione, ma se un qualsiasi evento provoca un sufficiente aumento del potenziale dal livello di  $-90\text{mV}$  verso il livello zero, è lo stesso voltaggio in aumento che fa sì che molti canali del sodio comincino ad aprirsi.
- Ciò permette un rapido ingresso di **ioni sodio**, che provoca ancora un nuovo aumento del potenziale di membrana, che fa aprire un numero ancora maggiore di canali del sodio accrescendo il flusso di ioni sodio che entrano nella cellula.
- Il processo si autoalimenta con un circolo di feedback positivo fino a che tutti i canali del sodio non risultano totalmente aperti.



# Le Reti Neurali

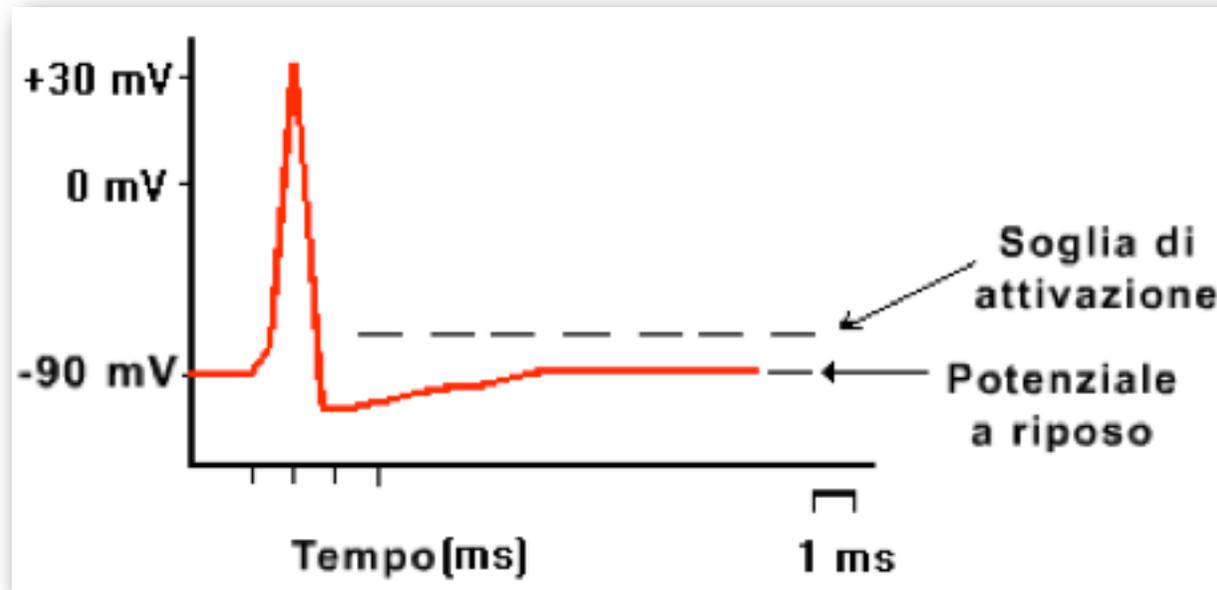
## Il modello biologico: il cervello umano



- Ma a questo punto in una successiva frazione di millisecondo il potenziale di membrana in aumento provoca una chiusura dei canali del sodio e una apertura di quelli del potassio, dopodiché il potenziale d'azione termina.
- Perché si inneschi il potenziale d'azione è necessario che il potenziale di membrana aumenti di 15/30mV, portando quest'ultimo a circa -65mV (soglia di attivazione).
- Quindi non sempre è detto che la soglia di attivazione venga raggiunta, dipende proprio dalla velocità del processo suddetto.

# Le Reti Neurali

## Il modello biologico: il cervello umano

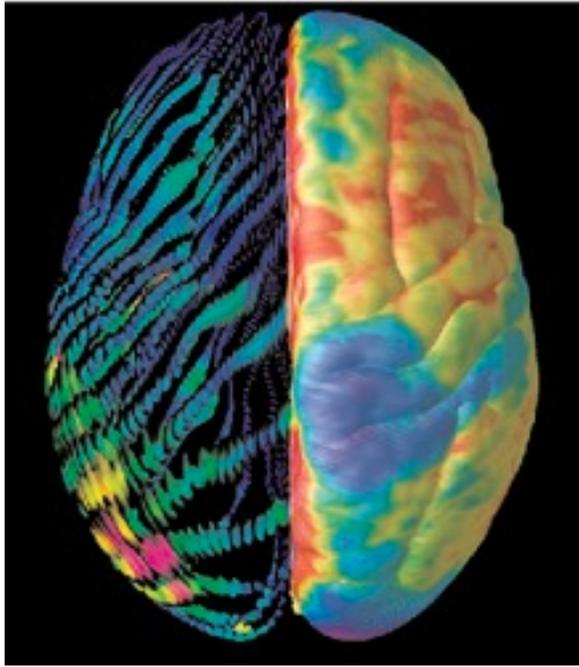


Questo **potenziale** è diffuso lungo tutta la struttura neuronale, anche alle estremità delle sinapsi là dove il segnale elettrico diventa chimico per passare al neurone interconnesso.

Se questo si verifica la sinapsi sarà quindi di **eccitazione**, altrimenti di **inibizione**.

# Le Reti Neurali

## Il modello biologico: il cervello umano



Il cervello, è l'oggetto più complesso e misterioso che si conosca: 1.300-1.500 grammi di tessuto gelatinoso composto da 100 miliardi di cellule (i **neuroni**), ognuna delle quali sviluppa in media 10 mila **connessioni** con le cellule vicine.

- Durante la vita fetale l'organismo produce non meno di 250.000 neuroni al minuto.
- Ma 15-30 giorni prima della nascita, la produzione si blocca e per il cervello comincia una seconda fase che durerà per tutta la vita: la creazione di connessioni tra le cellule.

# Le Reti Neurali

## Il modello biologico: il cervello umano



- In questo processo, le cellule che falliscono le connessioni vengono eliminate, tanto che al momento della nascita sono già dimezzate.
- Il cervello umano (più correttamente “**encefalo**”) è il risultato della sovrapposizione dei tre tipi di cervello apparsi nel corso dell’evoluzione dei vertebrati.

- Dal basso (alla base del cranio), il cervello più antico (**romboencefalo**), specializzato nel controllo di funzioni involontarie come vigilanza, respirazione, circolazione e tono muscolare. Comprende il cervelletto e le parti del midollo spinale che si allungano nel cervello.
- Salendo, c’è il **mesencefalo**: una piccola porzione di tessuto nervoso costituita dai cosiddetti peduncoli cerebrali e dalla lamina quadrigemina.
- Infine c’è il **prosencefalo**, la parte più “moderna”, suddiviso in diencefalo e telencefalo. Il primo, chiamato anche “sistema limbico”, contiene strutture come talamo, ipotalamo, ipofisi e ippocampo, da cui provengono sensazioni come fame, sete o desiderio sessuale. Infine, la parte più recente in assoluto: la corteccia, dove hanno sede le funzioni intelligenza e linguaggio.

# Le Reti Neurali

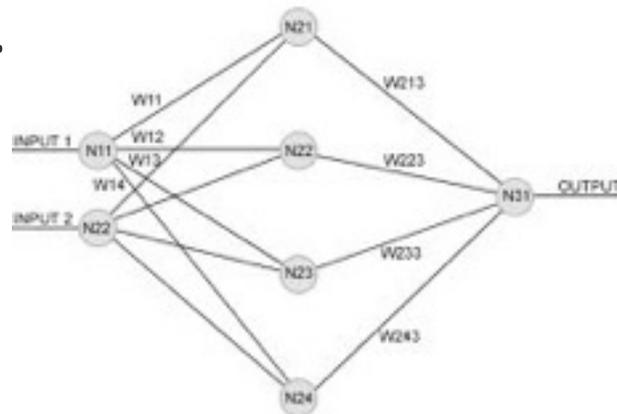
## Le componenti di una rete neurale

- Dal punto di vista generale ci sono **molti diversi tipi di reti neurali**, ma tutte hanno quasi gli stessi componenti.
- Se si vuole simulare il cervello umano utilizzando una rete neurale, è ovvio che alcune drastiche semplificazioni devono essere fatte.
- Prima di tutto, non è possibile “replicare” il funzionamento parallelo di tutte le cellule neurali. Anche se ci sono computer che hanno capacità di elaborazione parallela, il gran numero di processori che sarebbe necessario non può essere implementato dall'hardware attualmente disponibile.
- Un altro limite è che la struttura interna di un computer non può essere modificata durante l'esecuzione di qualsiasi compito.
- **E come implementare le stimolazioni elettriche in un programma per computer?**

# Le Reti Neurali

## Le componenti di una rete neurale

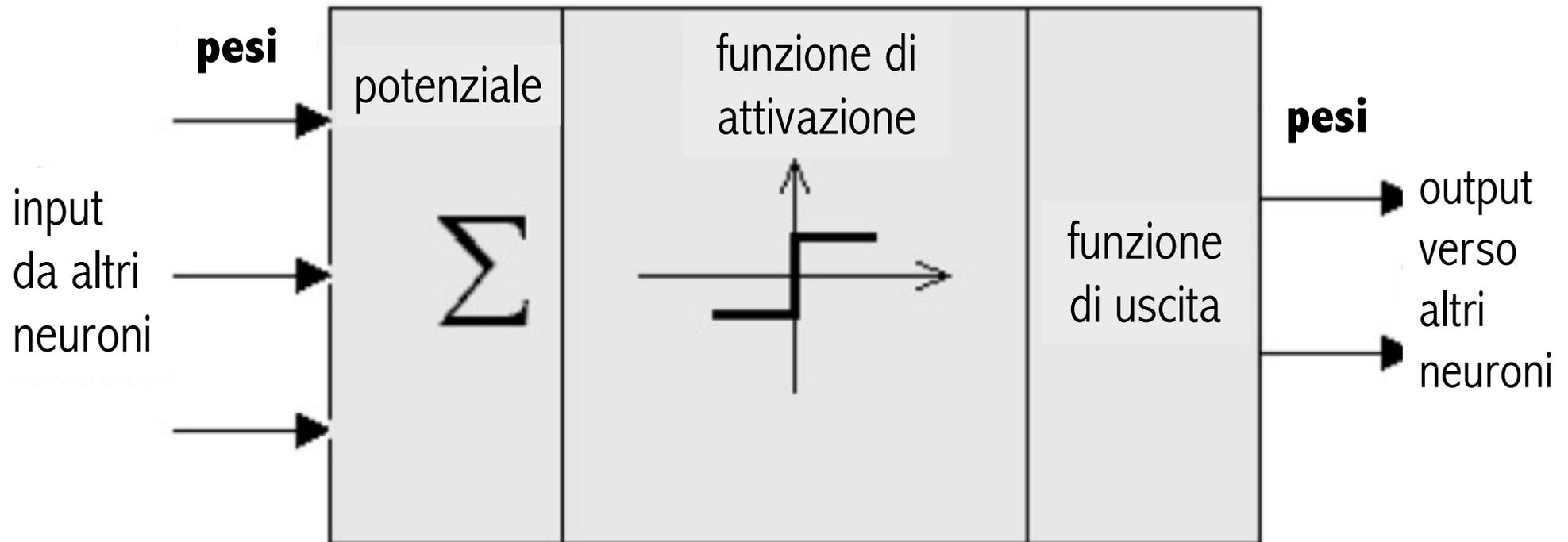
- ▶ Da ciò si ricava un modello idealizzato per scopi di simulazione.
- ▶ Come il cervello umano, una **rete neurale artificiale** consiste di neuroni e delle connessioni tra di loro.
- ▶ I neuroni trasportano le informazioni in entrata sulle loro connessioni in uscita verso altri neuroni. Nella terminologia delle reti neurali queste connessioni sono chiamate **pesi**.
- ▶ Le informazioni “elettriche” sono simulate con valori specifici memorizzati nei pesi.
- ▶ Il cambiamento della struttura di connessione può essere simulata semplicemente cambiando i valori dei pesi.



# Le Reti Neurali

## Le componenti di una rete neurale

La figura seguente mostra un **neurone** di una rete neurale artificiale:

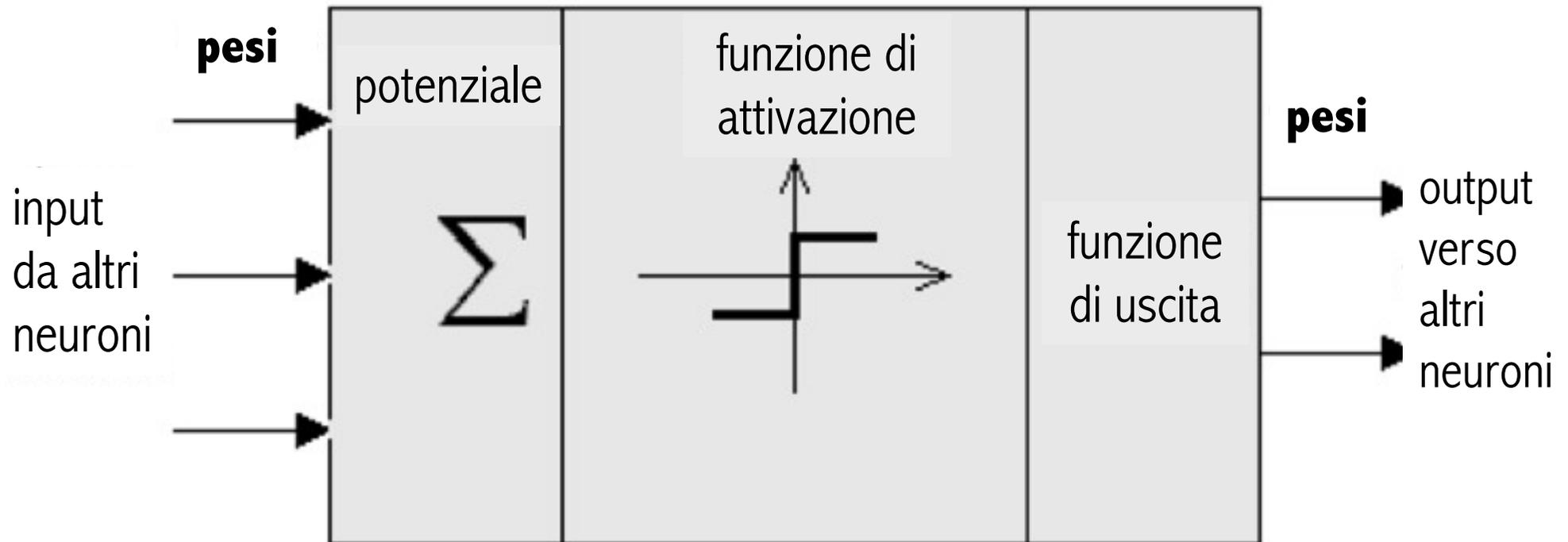


Un **neurone artificiale** è simile a una cellula neurale biologica, e funziona nello stesso modo. Le informazioni (valori di **input**) arrivano al neurone e vengono combinate (moltiplicandole) con i relativi **pesi**.

# Le Reti Neurali

## Le componenti di una rete neurale

La figura seguente mostra un **neurone** di una rete neurale artificiale:

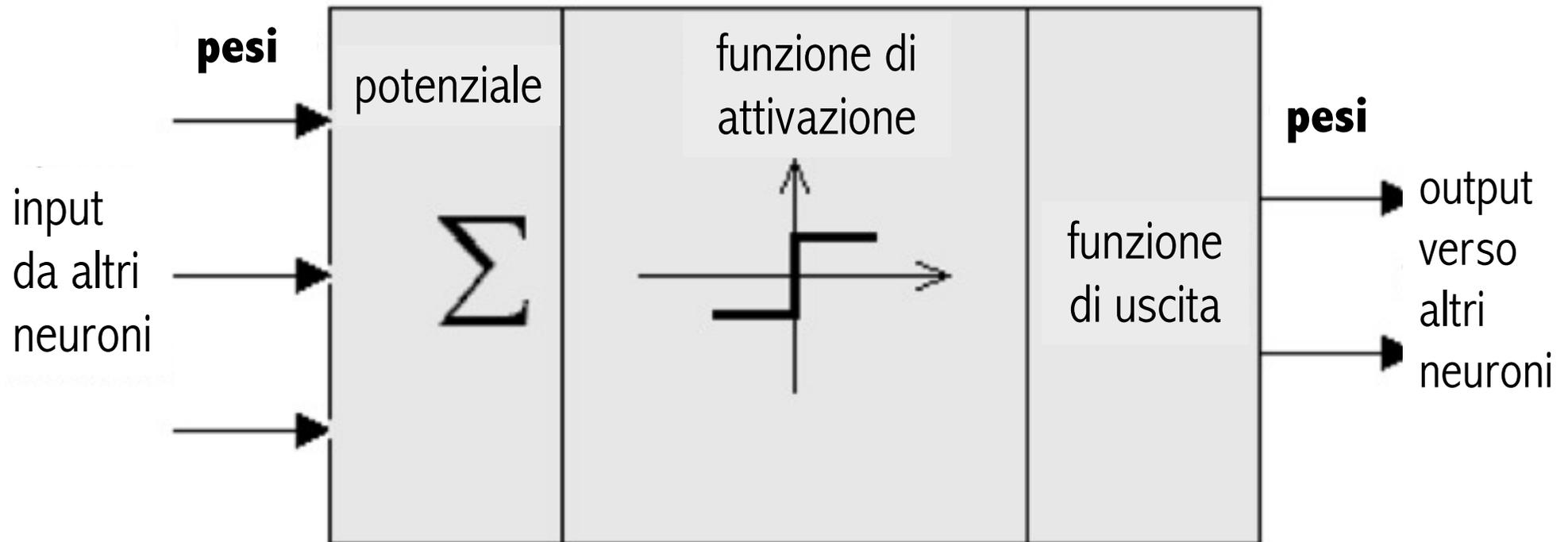


Una funzione di propagazione somma tali valori, fornendo il **potenziale** del neurone. Il valore risultante viene confrontato con un dato **valore di soglia** in base alla funzione di **attivazione** del neurone.

# Le Reti Neurali

## Le componenti di una rete neurale

La figura seguente mostra un **neurone** di una rete neurale artificiale:

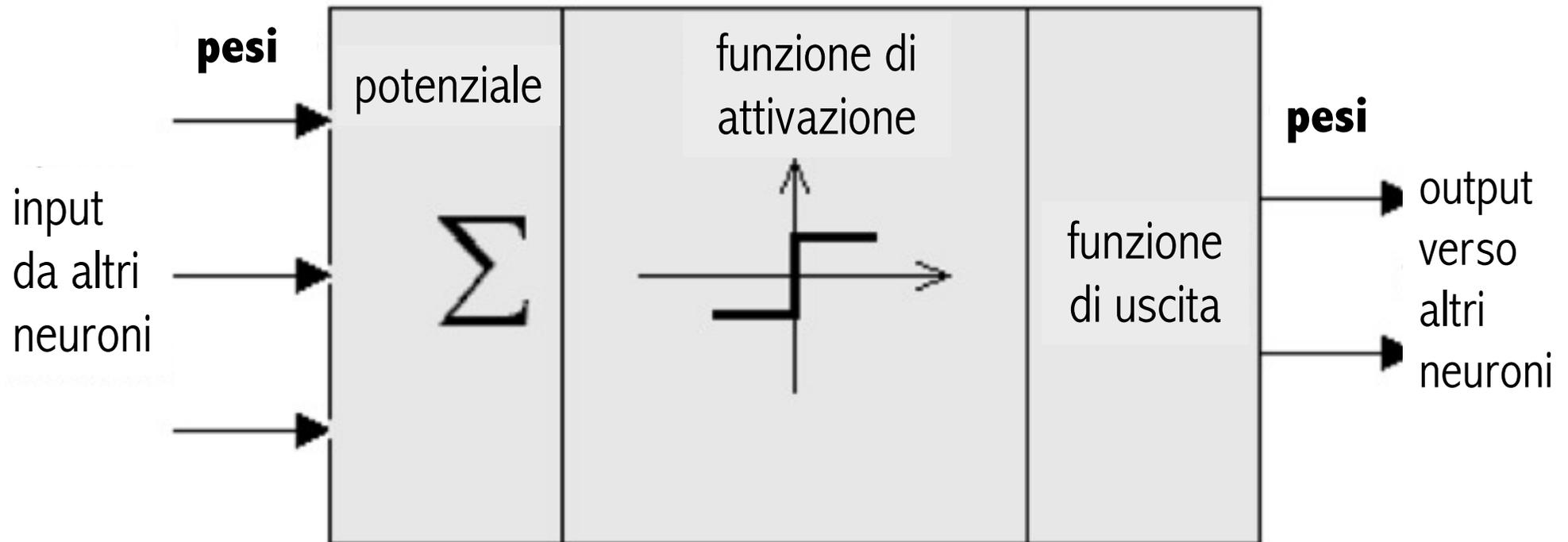


Se la somma supera il valore di soglia, il neurone si **attiva**, altrimenti sarà **inibito**.

# Le Reti Neurali

## Le componenti di una rete neurale

La figura seguente mostra un **neurone** di una rete neurale artificiale:



Se attivato, il neurone invia un output sulle **connessioni pesate in uscita** a tutti i neuroni collegati, e così via.

# Le Reti Neurali

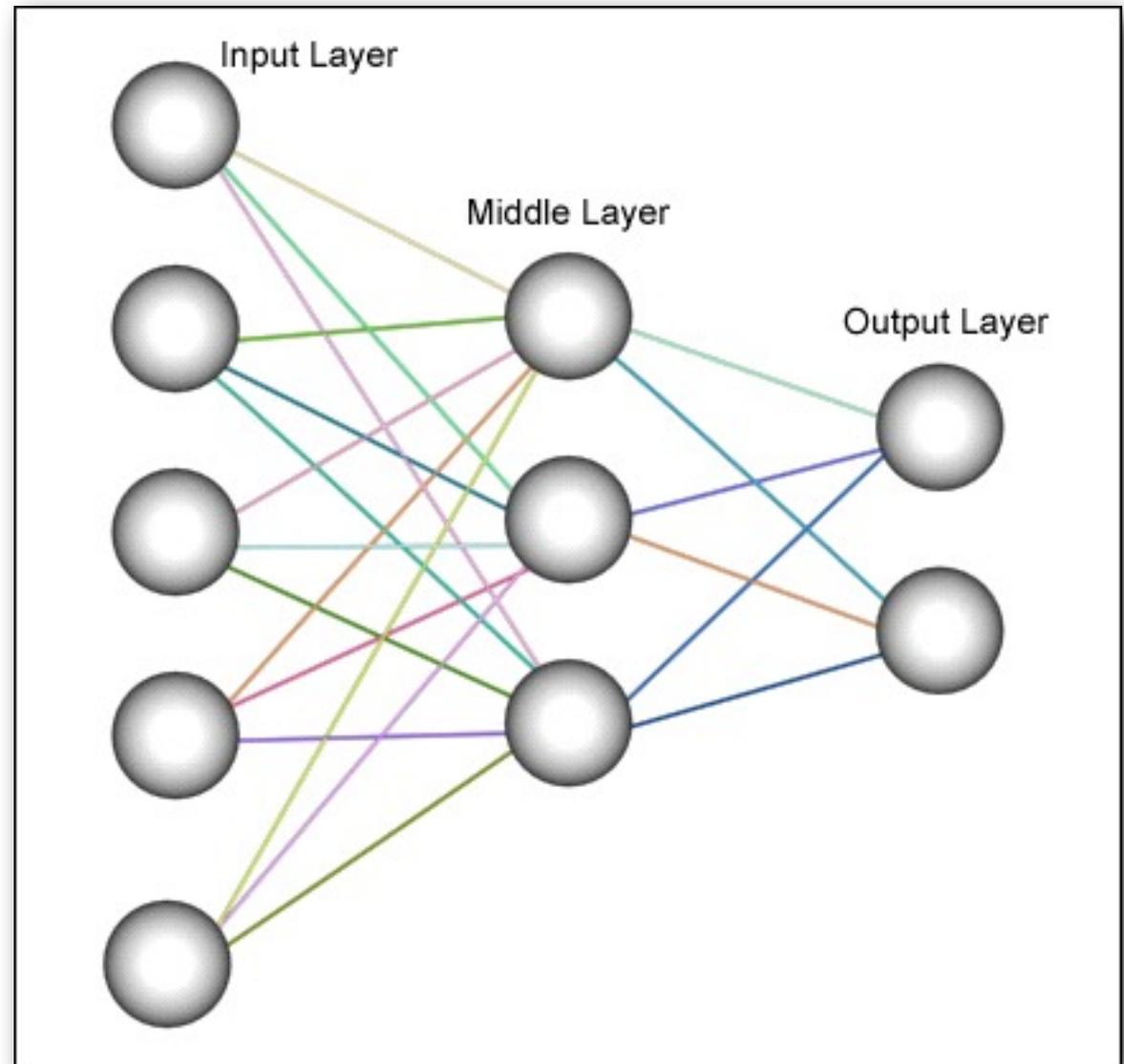
## Le componenti di una rete neurale

In una rete neurale, i neuroni sono raggruppati in strati (*layer*), denominati **strati di neuroni** (*neuron layers*).

Di solito ogni neurone di uno strato è collegato a tutti i neuroni dello strato precedente e successivo (eccetto il livello di input e lo strato di uscita della rete).

Le informazioni fornite da una rete neurale sono propagate layer-by-layer dallo strato di input a quello di output attraverso nessuno, uno o più strati nascosti (*hidden*).

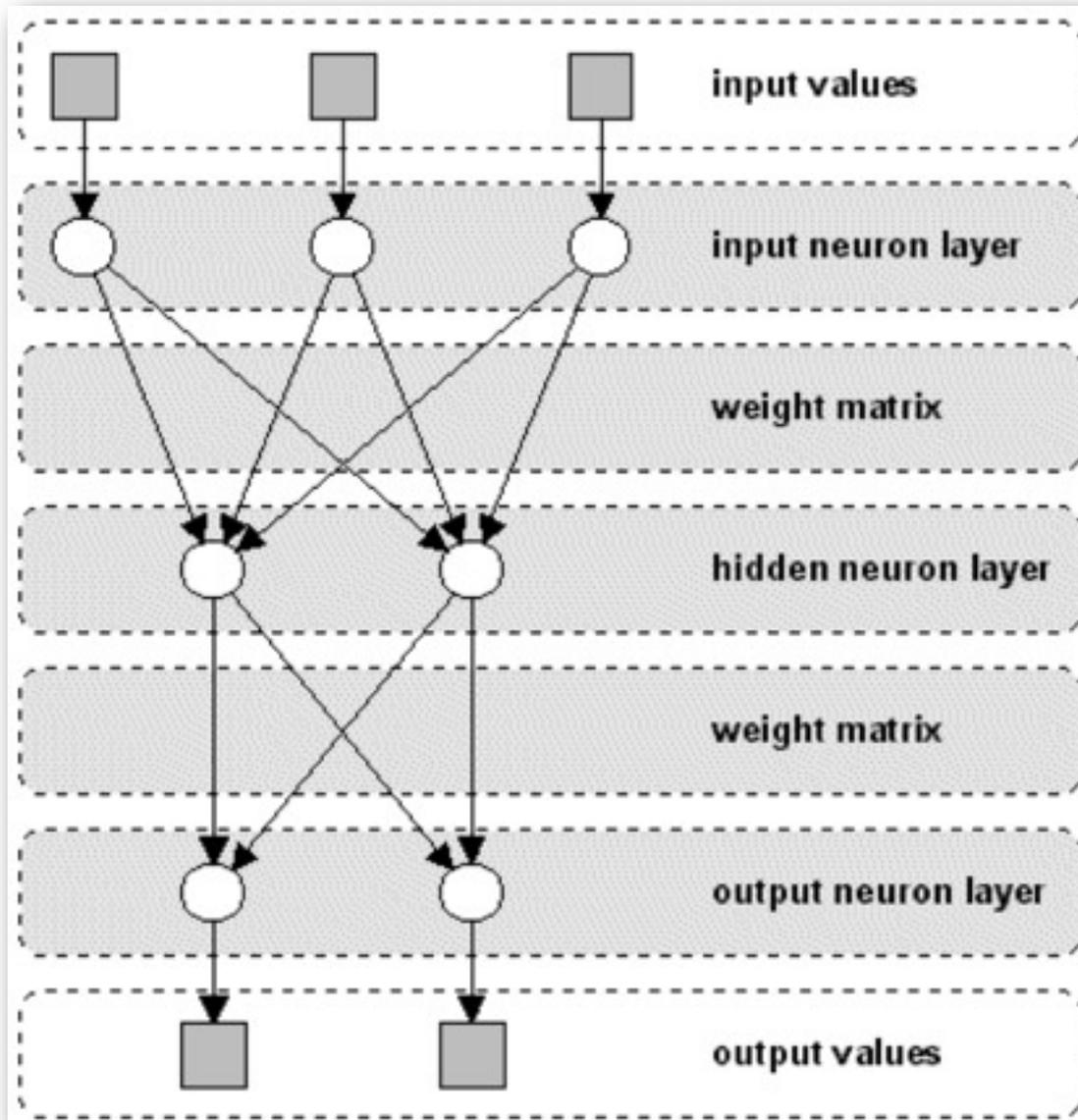
A seconda dell'**algoritmo di apprendimento**, è anche possibile che l'informazione si propaghi all'indietro attraverso la rete (*backpropagation*).



# Le Reti Neurali

## Le componenti di una rete neurale

La seguente figura mostra una rete neurale con tre strati:



Si noti che questa non è la struttura generale di una rete neurale.

Ad esempio, alcuni tipi di reti neurali non hanno livelli nascosti o i neuroni in uno strato sono disposti come una matrice.

Ciò che è **comune** a tutti i tipi di reti neurali è la presenza di almeno una **matrice dei pesi**, che individua le connessioni tra due strati di neuroni.

# Le Reti Neurali

## Possibilità e limiti

Le reti neurali vengono costruite per risolvere problemi che non possono essere risolti mediante algoritmi tradizionali. Tali problemi sono solitamente problemi di **ottimizzazione** o di **classificazione**.

I diversi domini in cui le reti neurali possono essere utilizzati sono:

- riconoscimento/classificazione di modelli (*pattern association/classification*)
- rilevamento di regolarità (*regularity detection*)
- elaborazione delle immagini (*image processing*)
- analisi del parlato (*speech analysis*)
- problemi di ottimizzazione (*optimization problems*)
- controllo di robot (*robot steering*)
- elaborazione di input inesatti o incompleti
- garanzia della qualità (*quality assurance*)
- previsioni di mercato (*stock market forecasting*)
- simulazione
- ...

# Le Reti Neurali

## Possibilità e limiti

- Ci sono molti **differenti tipi di reti neurali**, ciascuna con proprietà particolari, per cui ogni dominio applicativo ha il proprio tipo rete.
- In generale si può dire che le reti neurali sono **sistemi molto flessibili** per scopi di “problem solving”.
- Una caratteristica particolare delle reti neurali è la **tolleranza agli errori**. Ciò significa che, se una rete neurale viene addestrata per un problema specifico, sarà in grado di produrre risultati corretti anche se il problema da risolvere non è esattamente lo stesso di quello già appreso.
- Per esempio, supponiamo che una rete neurale sia stata addestrata a riconoscere il linguaggio umano. Durante il processo di apprendimento, una persona pronuncia alcune parole, che vengono apprese dalla rete. Poi, se addestrata correttamente, la rete neurale dovrebbe essere in grado di riconoscere le parole pronunciate anche da un'altra persona.

# Le Reti Neurali

## Possibilità e limiti

- Sebbene le reti neurali siano in grado di trovare soluzioni a problemi difficili, **i risultati non possono essere garantiti.**
- Essi sono solo **approssimazioni della soluzione** desiderata e un certo errore è sempre presente.
- Inoltre, **esistono problemi che non possono essere correttamente risolti** con reti neurali.

# Le Reti Neurali

## Possibilità e limiti

*Un esempio sul “pattern recognition” può chiarire meglio il concetto.*

*Se si incontra una persona vista in precedenza, di solito la si riconosce una seconda volta, anche se non sembra la stessa del primo incontro.*

*Supponiamo ora di aver addestrato una rete neurale con una fotografia di quella persona: l'immagine sarà sicuramente riconosciuta dalla rete.*

*Ma se si “disturba” l'immagine o ad es. la si ruota di un certo angolo, il riconoscimento probabilmente fallirà.*

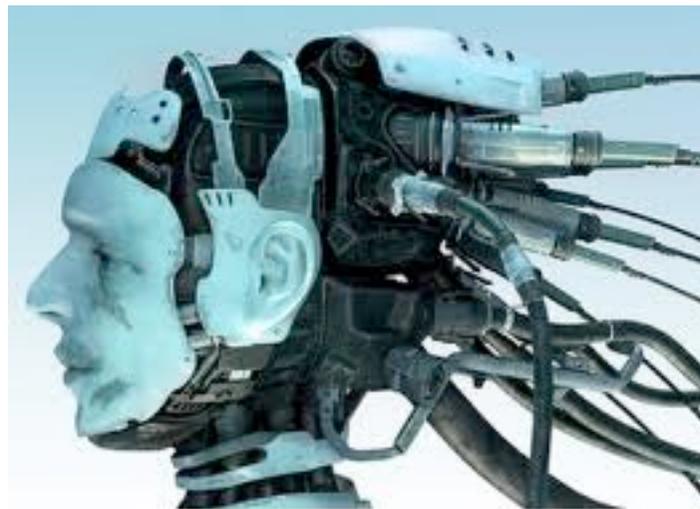


# Le Reti Neurali

## Possibilità e limiti

Sicuramente, nessuno utilizzerebbe mai una rete neurale in un algoritmo di ordinamento, poiché esistono algoritmi molto migliori e più veloci.

Ma in domini applicativi come quelli sopra menzionati, le reti neurali sono sempre una **buona alternativa agli algoritmi esistenti** e sicuramente vale la pena di provare.



# Le Reti Neurali

## Vantaggi applicativi

### ANN: vantaggi applicativi

- *Non-linearità* → migliore adesione ai dati.
- *Insensibilità al rumore* → previsioni accurate in presenza di incertezza nei dati ed errori di misurazione.
- *Elevato parallelismo* → velocità di elaborazione e tolleranza a “guasti hardware”.
- *Apprendimento e adattamento* → il sistema è in grado di “aggiornare” (update) la sua struttura interna in risposta a cambiamenti ambientali (input esterni).
- *Generalizzazione* → applicazione del modello risultante a dati nuovi (non appresi).

# Le Reti Neurali

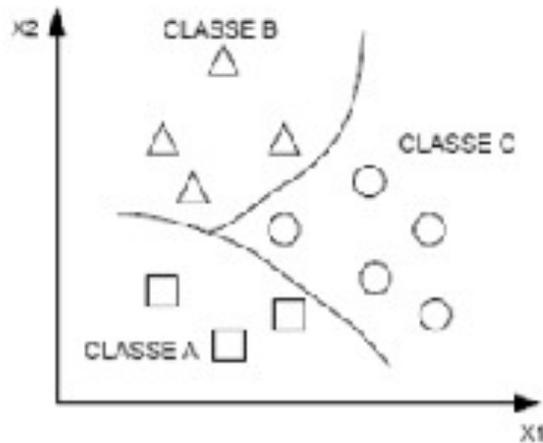
## Vantaggi applicativi

Vale la pena di affrontare un problema con le reti neurali quando:

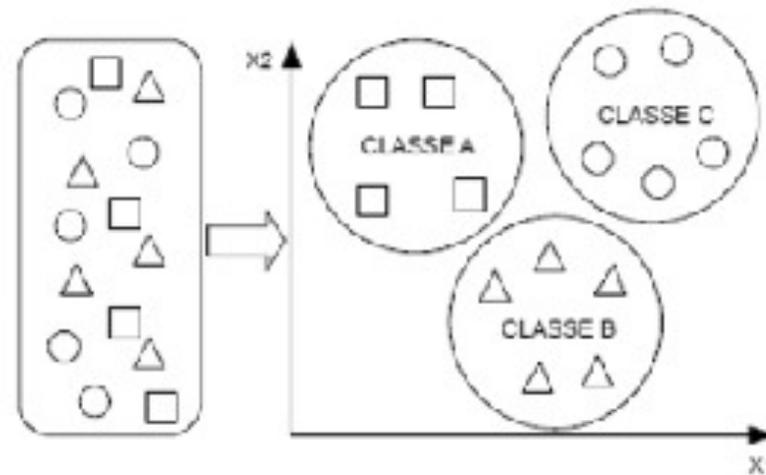
- **il problema è complesso** (funzione non lineare);
- **si hanno molti dati**, ma nessun modello fisico-matematico è disponibile per approssimare in modo utile la funzione che li connette;
- **le variabili in gioco sono molte** e di natura diversa;
- si hanno molti dati, ma l'**attendibilità** di alcuni è **incerta**;
- **altre metodologie** non hanno fornito risultati soddisfacenti;
- è utile una **soluzione anche approssimata** del problema, e non necessariamente una soluzione certa o la migliore in assoluto;
- una soluzione anche accettabile del problema produce **grandi risparmi** in termini umani e/o economici;
- si ha una notevole serie di casi che mostrano il comportamento “strano” del fenomeno che rappresenta il problema;
- *si desidera veramente risolvere il problema.*

# Le Reti Neurali

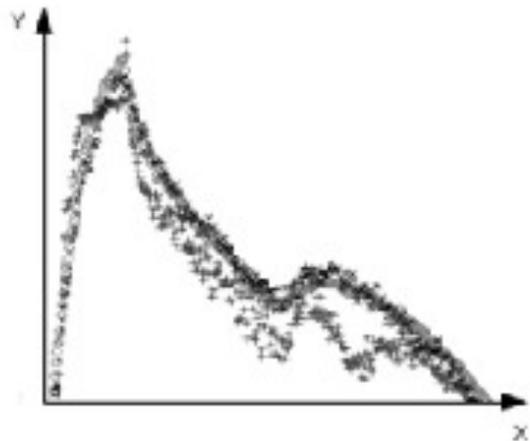
## Utilizzo



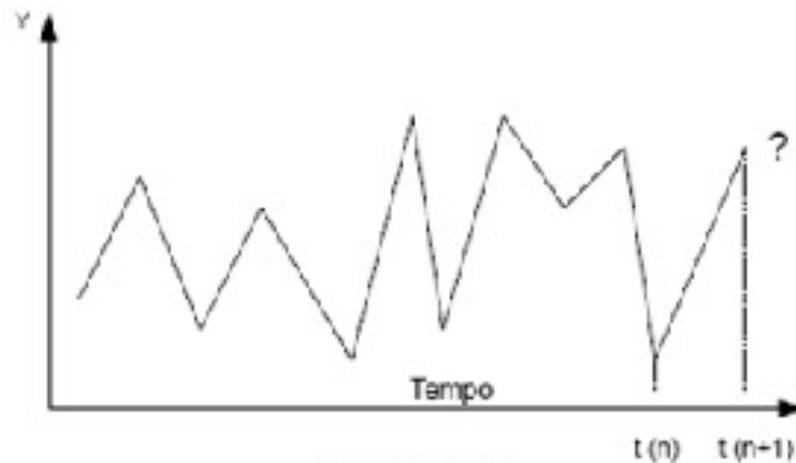
A) CLASSIFICAZIONE PATTERN



B) CLUSTERING



C) APPROSSIMAZIONE FUNZIONI



D) PREVISIONI

# Le Reti Neurali

## Utilizzo

*Nel campo particolare della finanza, le reti neurali artificiali vengono utilizzate per:*

- ▶ analisi del rischio di credito;
- ▶ identificazione delle perdite;
- ▶ distribuzione degli investimenti e analisi del portafoglio;
- ▶ previsione e simulazione degli scenari di cambio valutari;
- ▶ previsioni di titoli e azioni;
- ▶ simulazione di scenari economici;
- ▶ analisi dell'evasione e dell'elusione fiscale;
- ▶ ...

# Tipologie di Reti Neurali

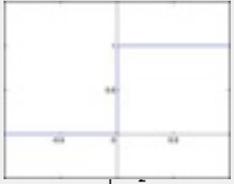


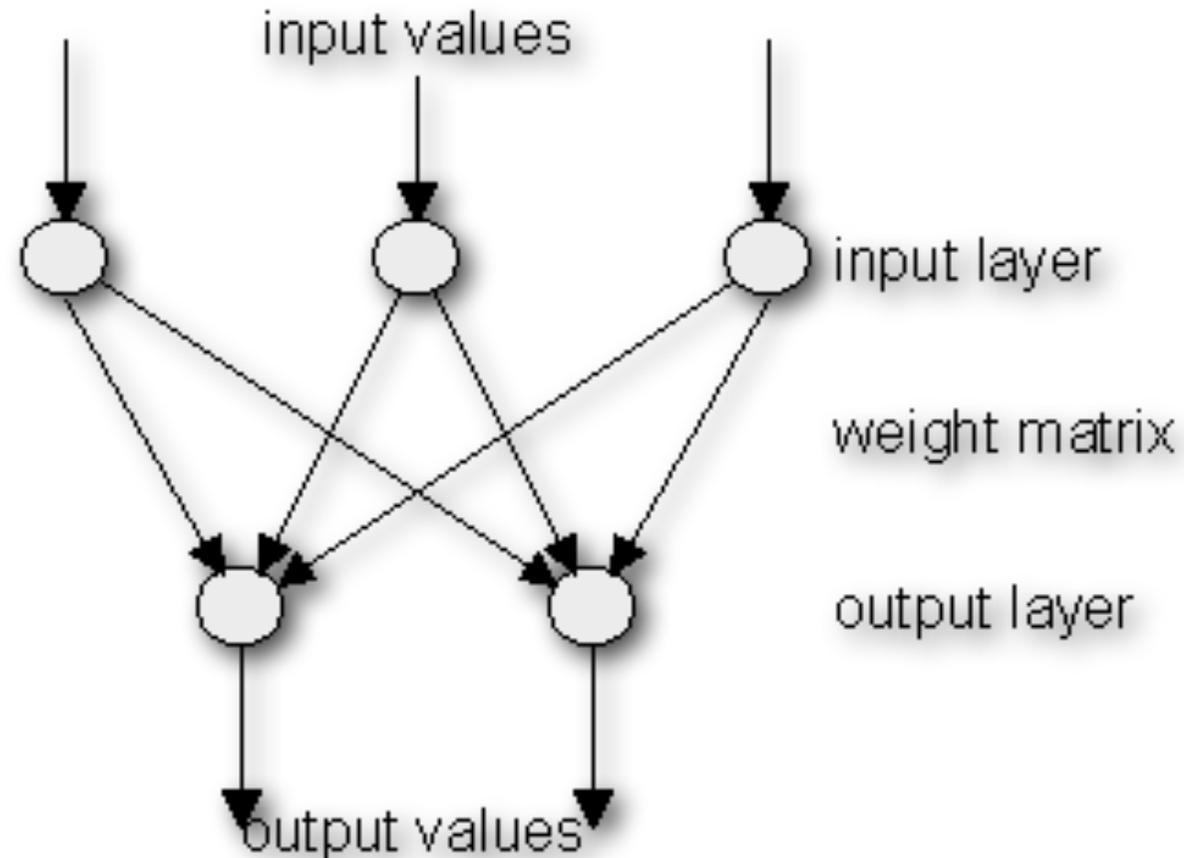
- [Esistono diverse tipologie di reti neurali, che possono essere distinte per **tipo** (*feedforward* o *feedback*), **struttura** e **algoritmo di apprendimento** utilizzato.
- [Il **tipo** di una rete neurale indica se i neuroni dei vari strati della rete possono essere collegati tra loro.
- [Le reti neurali di tipo **Feedforward** permettono solo collegamenti neuronali tra due diversi strati, mentre le reti del tipo di **Feedback** consentono anche connessioni tra i neuroni dello stesso strato.
- [Di seguito verrà esaminata una selezione di tipologie di reti neurali.

# Tipologie di Reti Neurali

## Perceptron

- Il Perceptron è stato introdotto da F. Rosenblatt nel 1958.
- Si tratta di un tipo molto semplice rete neurale con due strati di neuroni che accetta solo input ed output binari (0 e 1).
- Il processo di apprendimento è supervisionato e la rete è in grado di risolvere operazioni logiche di base come AND e OR.
- È utilizzato anche per la classificazione di modelli (pattern classification).
- Operazioni logiche più complesse (come XOR) non possono essere risolte da un Perceptron.

tipo	Feedforward
strati	1 strato di input, 1 strato di output
valori di input/output	binari
funzione di attivazione	hard limit 
metodo di apprendimento	supervisionato
algoritmo di apprendimento	regola di Hebb
utilizzo	semplici operazioni logiche, pattern classification

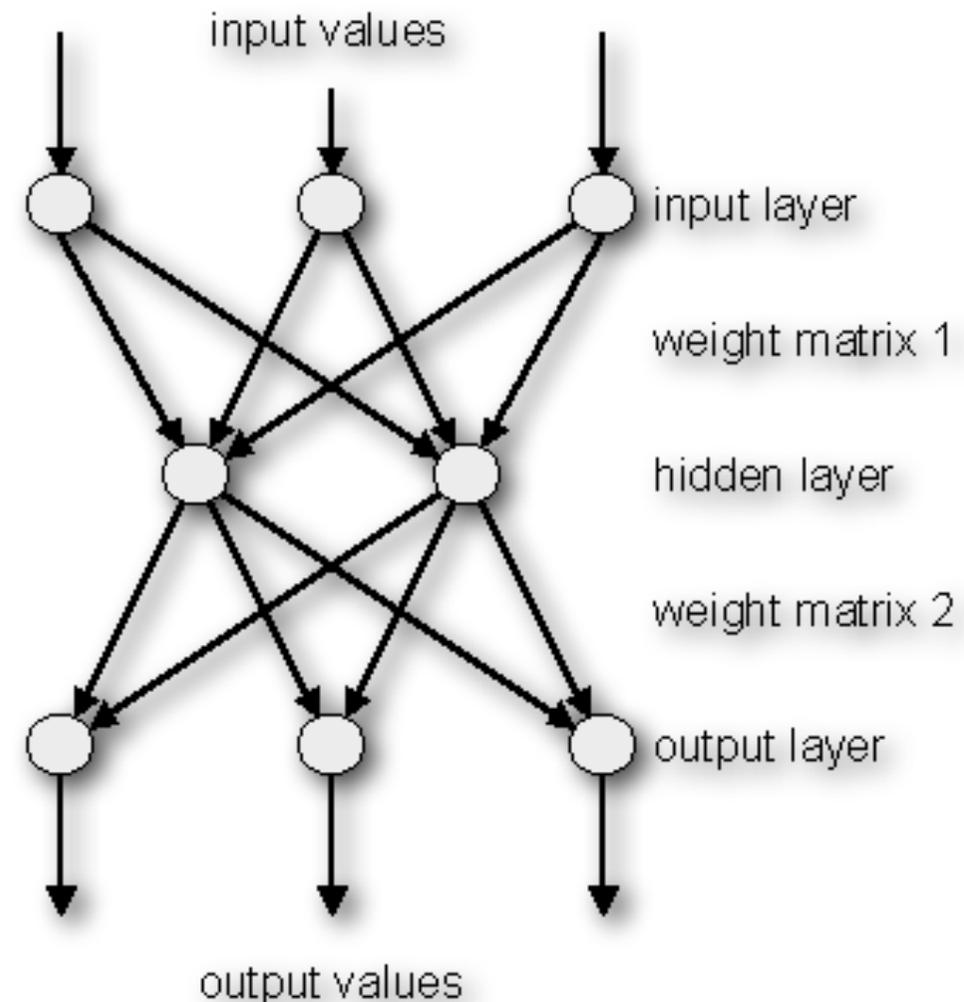


# Tipologie di Reti Neurali

## Multi-Layer Perceptron

- Il Multi-Layer Perceptron è stato introdotto da M. Minsky e S. Papert nel 1969.
- Si tratta di un Perceptron esteso, con uno o più livelli nascosti tra lo strato di input e quello di output.
- Grazie alla sua struttura estesa, un Multi-Layer Perceptron è in grado di risolvere ogni operazione logica, compreso il problema XOR.

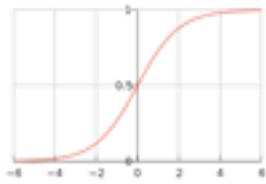
tipo	Feedforward
strati	1 strato di input, 1 o più strati nascosti, 1 strato di output
valori di input/output	binari
funzione di attivazione	hard limit, sigmoidea
metodo di apprendimento	supervisionato
algoritmo di apprendimento	delta rule backpropagation (più usata)
utilizzo	operazioni logiche complesse, pattern classification

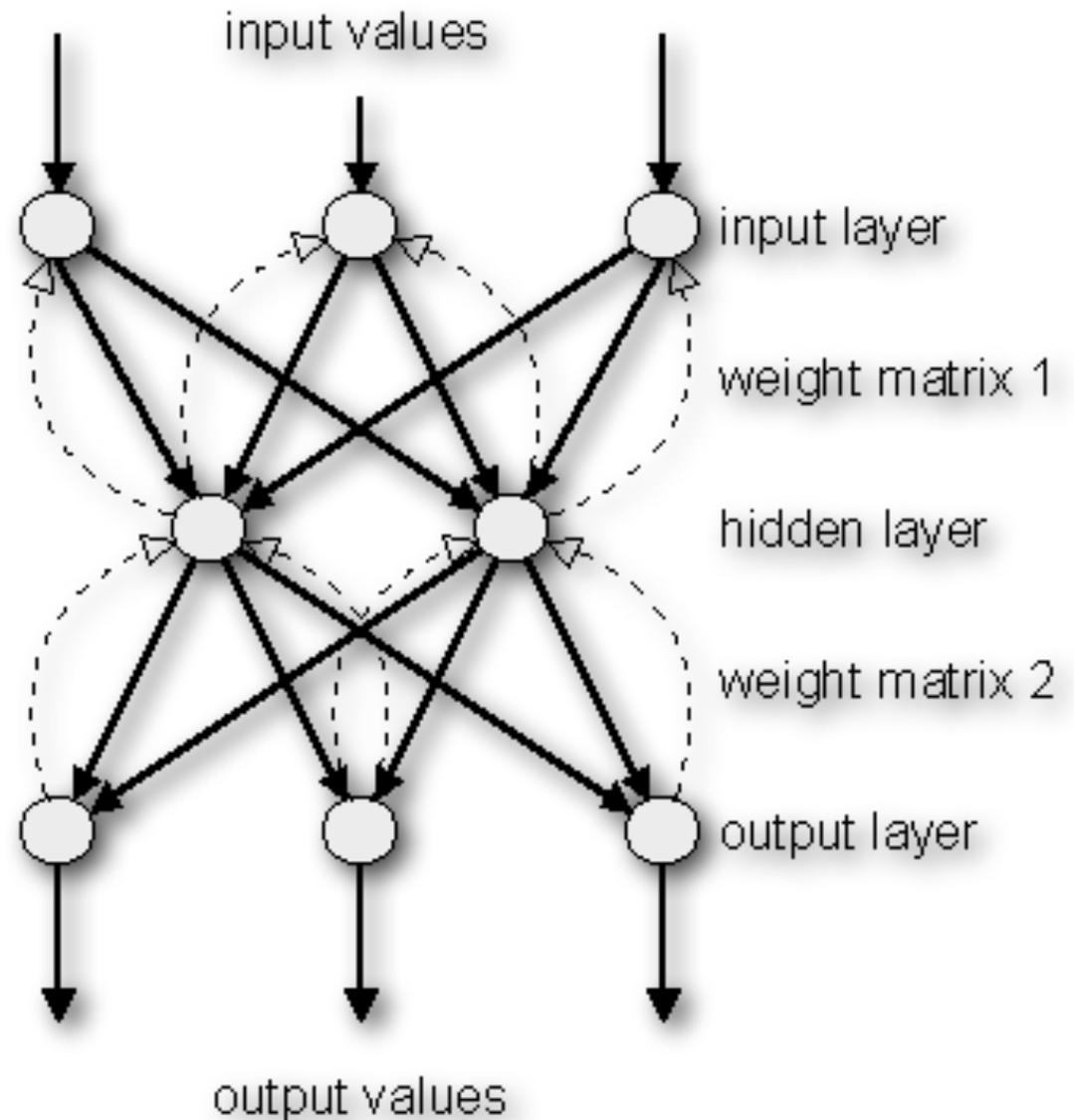


# Tipologie di Reti Neurali

## Backpropagation

- La rete Backpropagation è stata introdotta da Hinton, Rumelhart e Williams nel 1986 ed è uno dei tipi più potenti di rete neurale.
- Ha la stessa struttura del Multi-Layer Perceptron e utilizza l'algoritmo di apprendimento "backpropagation".

tipo	Feedforward
strati	1 strato di input, 1 o più strati nascosti, 1 strato di output
valori di input/output	binari
funzione di attivazione	sigmoidea 
metodo di apprendimento	supervisionato
algoritmo di apprendimento	backpropagation
utilizzo	operazioni logiche complesse, pattern classification, analisi del parlato

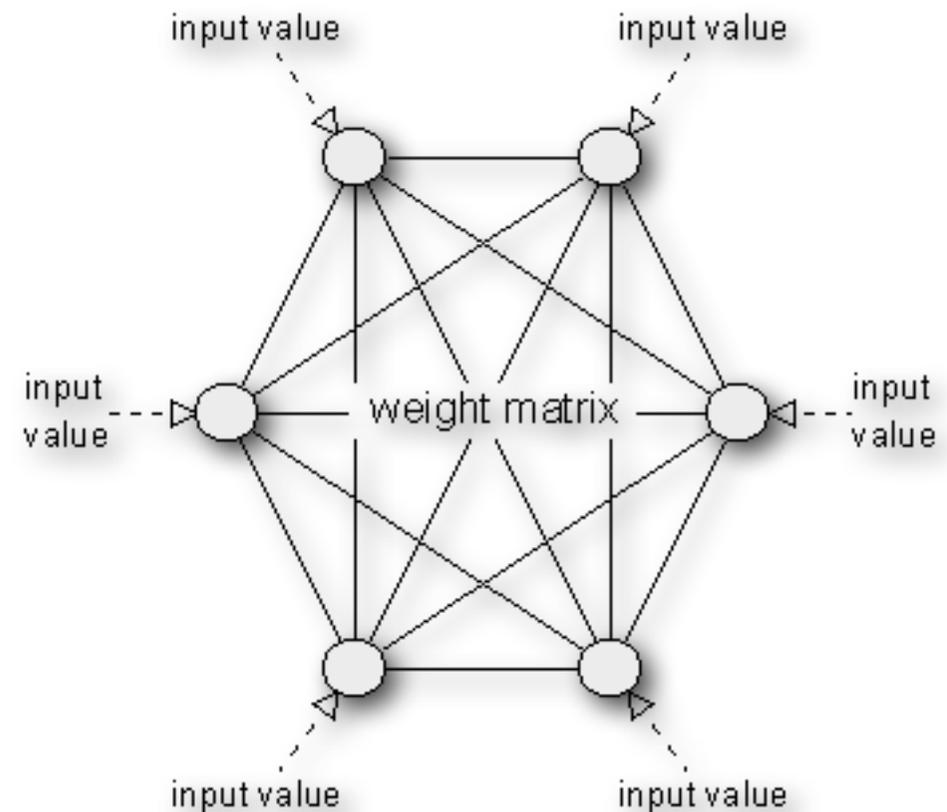


# Tipologie di Reti Neurali

## Hopfield

- Questa rete è stata introdotta dal fisico Hopfield nel 1982 e appartiene ai tipi di reti neurali denominate “modelli termodinamici”.
- Si compone di un insieme di neuroni, in cui ogni neurone è collegato ad ogni altro neurone.
- Non c'è differenziazione tra neuroni di input e di output.
- L'applicazione principale di una rete di Hopfield è la conservazione e il riconoscimento di modelli, per esempio immagini.

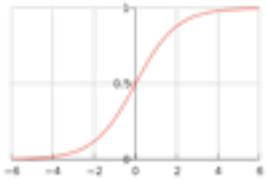
tipo	Feedback
strati	1 matrice di connessioni
valori di input/output	binari
funzione di attivazione	segno, hard limiter 
metodo di apprendimento	non supervisionato
algoritmo di apprendimento	delta rule simulated annealing (più usata)
utilizzo	pattern association, problemi di ottimizzazione

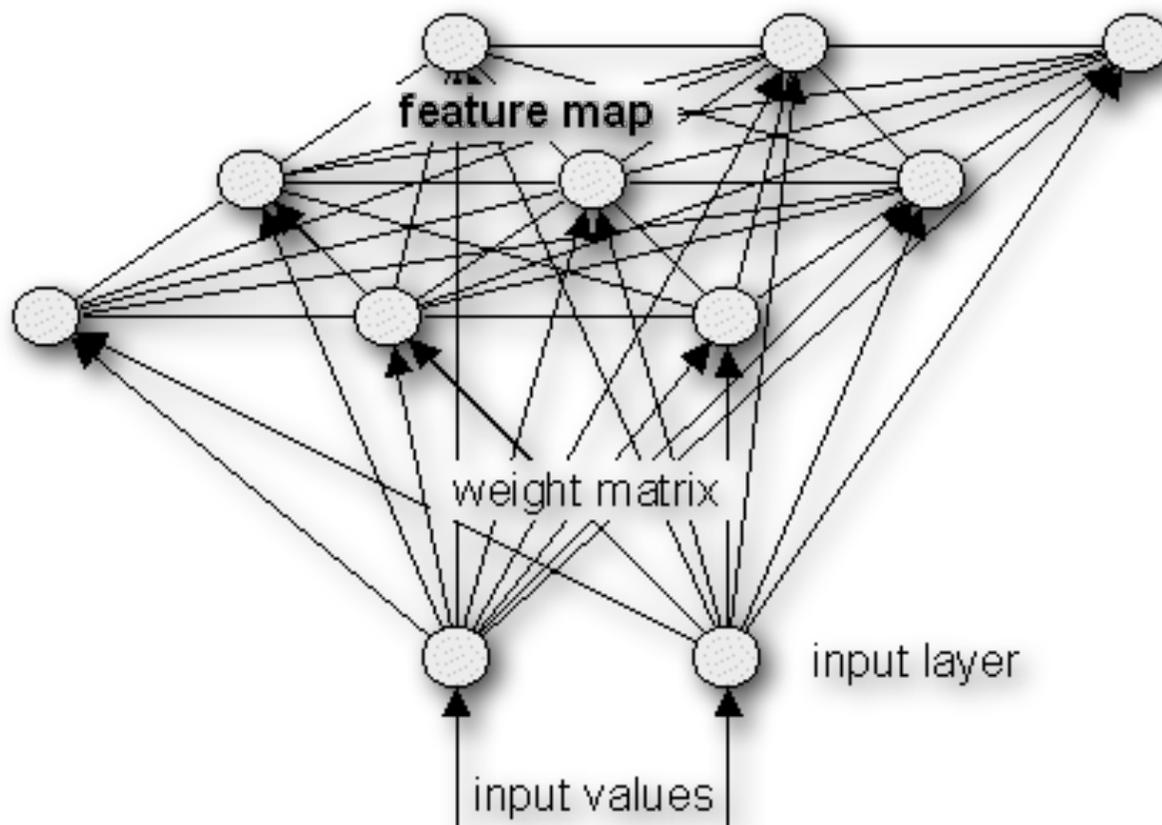


# Tipologie di Reti Neurali

## Mappa di Kohonen

- Questa rete è stata introdotta dal professore finlandese Teuvo Kohonen (Università di Helsinki) nel 1982.
- È probabilmente il tipo di rete neurale più utile per la simulazione del processo di apprendimento del cervello umano.
- Il suo “cuore” è la mappa caratteristica, uno strato di neuroni in cui i neuroni si organizzano secondo certi valori di input.
- Questo tipo di rete neurale è sia feedforward (neuroni di input → mappa) e feedback (neuroni della mappa interconnessi).

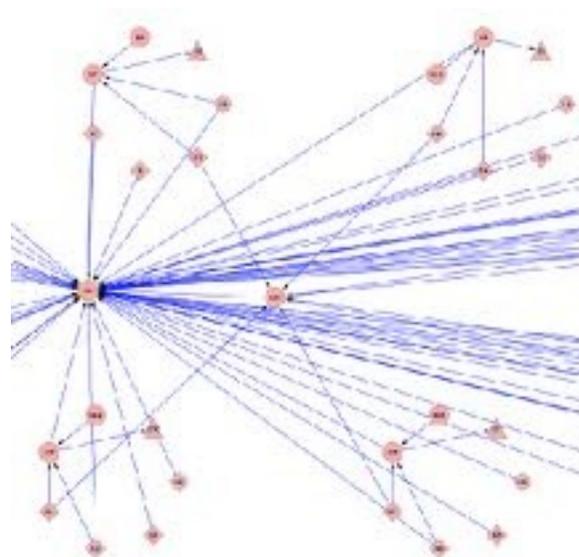
tipo	Feedforward/Feedback
strati	1 strato di input, 1 strato “mappa”
valori di input/output	binari, reali
funzione di attivazione	sigmoidea 
metodo di apprendimento	non supervisionato
algoritmo di apprendimento	auto-organizzazione (esempio)
utilizzo	pattern classification, problemi di ottimizzazione simulazione



*Durante il processo di apprendimento, i neuroni sulla mappa si organizzano in funzione dei valori di ingresso. Questo si traduce in una struttura neuronale clusterizzata, in cui i neuroni con proprietà (valori) simili si dispongono in settori connessi sulla mappa.*

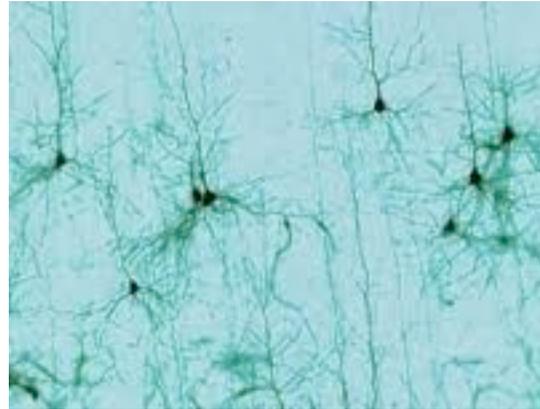
# Il processo di apprendimento

- Questa sezione descrive le capacità di apprendimento delle reti neurali.
- In primo luogo, viene illustrato il termine ***apprendimento***.
- Successivamente, viene presentata una panoramica di specifici algoritmi di apprendimento per reti neurali.



# Il processo di apprendimento

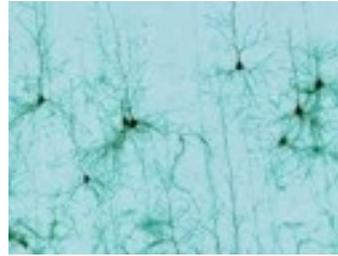
Cosa significa "apprendimento" per una rete neurale?



- ▶ Nel cervello umano, l'informazione viene passata tra i neuroni in forma di **stimolazione elettrica** lungo i dendriti.
- ▶ Se una certa quantità di stimolazione viene ricevuta da un neurone, esso **genera un output** verso tutti gli altri neuroni connessi e così le informazioni prendono la loro strada verso la destinazione, dove si verificherà qualche reazione.
- ▶ Se lo **stimolo** in entrata è **troppo basso**, nessuna uscita è generata dal neurone e il successivo trasporto delle informazioni sarà **bloccato**.

# Il processo di apprendimento

## Cosa significa "apprendimento" per una rete neurale?



Spiegare come il cervello umano impari certe cose è molto difficile e nessuno lo sa esattamente.

Si suppone che durante il processo di apprendimento la struttura di collegamento tra i neuroni venga modificata, in modo che certe stimolazioni sono accettate solo da certi neuroni.

Ciò significa che esistono solidi legami tra le cellule neurali una volta che hanno imparato un fatto specifico, consentendo il richiamo rapido delle informazioni.

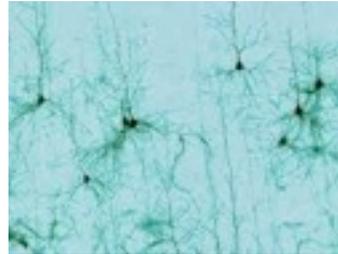
Se informazioni analoghe arrivano successivamente vengono stimulate le stesse cellule nervose, che adatteranno la loro struttura di connessione in funzione di queste nuove informazioni.

D'altra parte, se una informazione specifica non viene richiamata per molto tempo, la struttura di connessione stabilita tra le cellule neurali responsabili diverrà più "debole".

Questo accade ad es. quando si "dimentica" un fatto in precedenza acquisito o lo si ricorda solo vagamente.

# Il processo di apprendimento

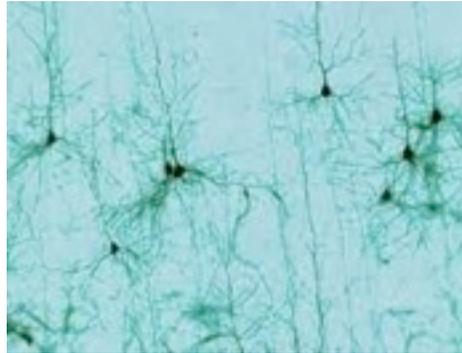
Cosa significa "apprendimento" per una rete neurale?



- Come accennato prima, **le reti neurali cercano di simulare** la capacità di apprendimento del cervello umano.
- Infatti, la rete neurale artificiale è fatta anch'essa di neuroni e dendriti. A differenza del modello biologico, **una rete neurale ha una struttura immutabile**, costruita su un determinato numero di neuroni e un determinato numero di connessioni tra di loro (chiamate "pesi"), che hanno certi valori.
- Ciò che cambia nel **processo di apprendimento** sono i valori di tali pesi.

# Il processo di apprendimento

Cosa significa "apprendimento" per una rete neurale?

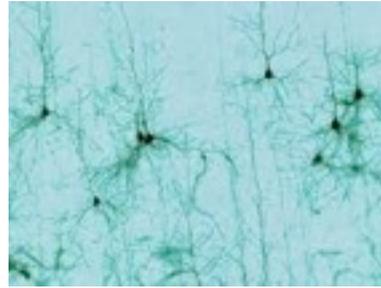


Rispetto all'originale biologico questo significa che:

1. Le **informazioni in arrivo** "stimolano" (superano un determinato valore di soglia di) certi neuroni
2. Questi **passano le informazioni** ai neuroni collegati **o prevengono** l'ulteriore trasporto lungo le connessioni "ponderate" in uscita.
3. Il **valore di un peso** sarà aumentato se le informazioni devono essere trasportate e ridotto in caso contrario.

# Il processo di apprendimento

## Cosa significa "apprendimento" per una rete neurale?



Durante l'apprendimento di diversi input, **i valori dei pesi vengono cambiati dinamicamente** fino a quando non sono in equilibrio, in modo tale che ogni ingresso porterà all'output desiderato.

L'addestramento di una rete neurale risulta in una **matrice** che contiene i valori **dei pesi tra i neuroni**.

Una volta che una rete neurale è stata correttamente addestrata, sarà probabilmente in grado di trovare il risultato desiderato per un ingresso precedentemente appreso, utilizzando i valori della matrice dei pesi.

Perché probabilmente? Purtroppo non si può garantire che una rete neurale produca i risultati corretti in ogni caso. Molto spesso vi è un certo errore residuo dopo il processo di apprendimento, per cui l'output generato è solo una **buona approssimazione** dell'uscita perfetta nella maggior parte dei casi.

# Il processo di apprendimento

## Apprendimento supervisionato

L' algoritmo di apprendimento di una rete neurale può essere con o senza **supervisione**.

Una rete neurale è supervisionata, se l'output desiderato è già noto.

**Esempio:** associazione di modelli (*pattern association*)

Supponiamo che, una rete neurale debba imparare ad associare le seguenti coppie di modelli. I modelli d'ingresso (*input pattern*) sono numeri decimali, ognuno rappresentato da una sequenza di bit. I modelli di uscita (*output pattern*) sono forniti in forma di valori binari dei numeri decimali:

input pattern	output pattern
0001	001
0010	010
0100	011
1000	100

1. Durante l'apprendimento, uno dei pattern di input viene “presentato” al livello di input della rete.
2. Questo pattern si propaga attraverso la rete (indipendentemente dalla sua struttura) sino al livello di uscita.

3. Lo strato di uscita genera un pattern di output che viene poi confrontato con il pattern reale. A seconda della differenza tra il risultato prodotto dalla rete e l'obiettivo, viene calcolato un valore di errore.
4. Questo **errore** indica lo sforzo di apprendimento della rete, che può essere controllato dal “supervisore immaginario”.
5. Quanto maggiore è il valore di errore calcolato, tanto più i valori dei pesi verranno modificati.

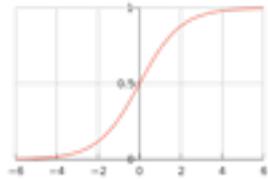
# Il processo di apprendimento

## Apprendimento non supervisionato

- ▶ Le **reti neurali non supervisionate** (cioè che “imparano” senza supervisione) non hanno uscite predefinite.
- ▶ Il risultato del processo di apprendimento non può essere determinato a priori.
- ▶ Durante il processo di apprendimento, le unità (i pesi) della rete neurale vengono “disposte” all'interno di un determinato intervallo di valori, a seconda dei valori in ingresso.
- ▶ L'obiettivo è quello di raggruppare insieme unità simili, “vicine” in certe aree del range di valori.
- ▶ Questo effetto può essere utilizzato in modo efficace ai fini della classificazione di modelli (*clustering*).
- ▶ Il seguente esempio di rete di Kohonen “*self organizing*” chiarisce il funzionamento in dettaglio.

# Tipologie di Reti Neurali

## Mappa di Kohonen



### A 3D Kohonen Feature Map



input values: 10	learning cycle: 0
input layer: 3 neurons	learning rate: 0.6
map size: 9x9 neurons	activation area: 3.0
weights: 243	elapsed time: 0.0010 sec

# Il processo di apprendimento

## Forward propagation

Si tratta di un algoritmo di apprendimento supervisionato che descrive il “flusso di informazioni” attraverso una rete neurale dal suo strato di ingresso al suo livello di output.

L'algoritmo funziona nel modo seguente:

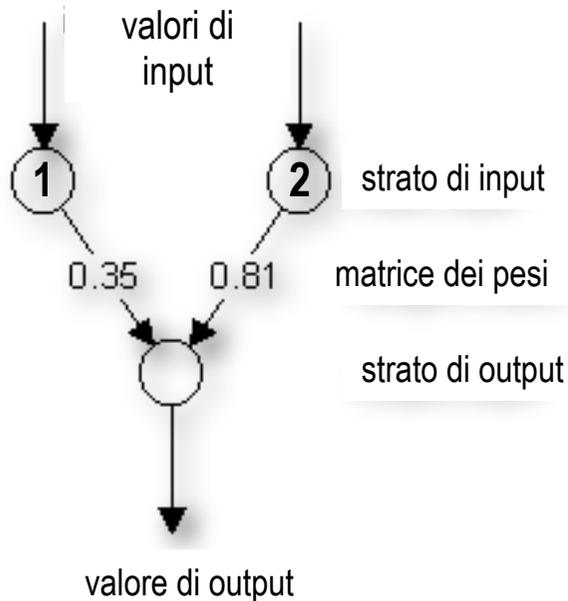
1. Impostare tutti i pesi a valori casuali compresi tra -1.0 e +1.0
2. Fornire un pattern di ingresso (valori binari) ai neuroni del livello di input
3. Attivare ogni neurone dello strato successivo nel modo seguente:
  - *moltiplicare i valori di peso delle connessioni che portano al neurone con i valori di output dei neuroni precedenti;*
  - *sommare questi valori;*
  - *passare il risultato ad una funzione di attivazione, che calcola il valore di uscita del neurone.*
4. Ripetere questa operazione fino a quando lo strato di uscita non viene raggiunto
5. Confrontare il pattern di uscita calcolato con il pattern previsto e calcolare l'errore
6. Modificare tutti i pesi aggiungendo il valore di errore ai (vecchi) valori di peso
7. Tornare al punto 2
8. L'algoritmo termina se tutti i pattern di output corrispondono ai loro pattern previsti

# Il processo di apprendimento

## Forward propagation

### Esempio.

Supponiamo di avere il seguente Perceptron a due strati:



**(1)** Inizialmente i pesi vengono impostati a valori casuali (0.35 e 0.81) e il “tasso di apprendimento” (learning rate)  $\mu$  della rete è impostato a 0.25; quindi vengono forniti in ingresso i valori del primo pattern di input (0 1). Il neurone dello strato di output viene così attivato:

Input 1 del neurone di output	$0 \times 0.35 = 0$
Input 2 del neurone di output	$1 \times 0.81 = 0.81$
Somma degli input	$0 + 0.81 = 0.81$ (= output)
Calcolo dell'errore	$0 - 0.81 = -0.81$
Valore di modifica del peso 1	$0.25(\mu) \times 0(\text{input1}) \times (-0.81) = 0$
Valore di modifica del peso 2	$0.25(\mu) \times 1(\text{input2}) \times (-0.81) = -0.2025$
Modifica del peso 1	$0.35 + 0 = 0.35$ (immutato)
Modifica del peso 2	$0.81 + (-0.2025) = 0.6075$

Pattern da “apprendere”:

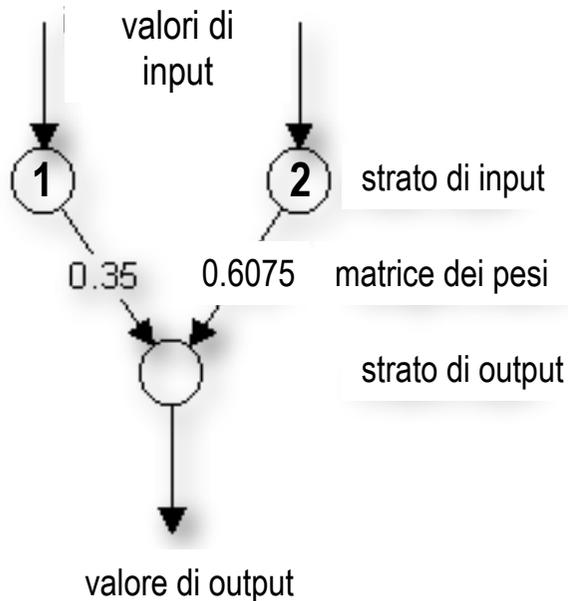
input	output previsto (target)
<b>0 1</b>	<b>0</b>
1 1	1

# Il processo di apprendimento

## Forward propagation

### Esempio.

Supponiamo di avere il seguente Perceptron a due strati:



Pattern da "apprendere":

input	output previsto (target)
0 1	0
<b>1 1</b>	<b>1</b>

**(2)** Dopo che i pesi sono stati modificati viene fornito in ingresso il secondo pattern di input (1 1) e viene nuovamente eseguita l'attivazione del neurone di output, con i nuovi valori dei pesi:

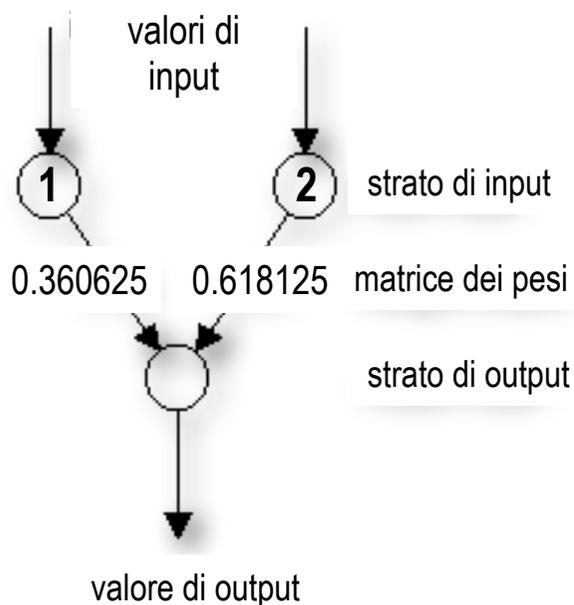
Input 1 del neurone di output	$1 \times 0.35 = 0.35$
Input 2 del neurone di output	$1 \times 0.6075 = 0.6075$
Somma degli input	$0.35 + 0.6075 = 0.9575$ (= output)
Calcolo dell'errore	$1 - 0.9575 = \mathbf{0.0425}$
Valore di modifica del peso 1	$0.25(\mu) \times 1(\text{input1}) \times 0.0425 = 0.010625$
Valore di modifica del peso 2	$0.25(\mu) \times 1(\text{input2}) \times 0.0425 = 0.010625$
Modifica del peso 1	$0.35 + 0.010625 = 0.360625$
Modifica del peso 2	$0.6075 + 0.010625 = 0.618125$

# Il processo di apprendimento

## Forward propagation

### Esempio.

Supponiamo di avere il seguente Perceptron a due strati:



Pattern da “apprendere”:

input	output previsto (target)
0 1	0
1 1	1

**(3)** Si conclude così il primo passo di apprendimento (learning step, o “epoca”), nel quale ciascun pattern di input è stato propagato attraverso la rete ed i pesi sono stati aggiornati.

L'errore complessivo della rete può essere ora calcolato mediando i quadrati degli errori di output (MSE=Mean Squared Error):

Errore della rete (MSE)	$[(-0.81)^2 + (0.0425)^2] / 2 = \mathbf{0.32895313}$
-------------------------	--

Oppure può essere utilizzato l'RMSE (Root Mean Squared Error):

Errore della rete (RMSE)	$\text{sqrt}[0.32895313] = \mathbf{0.57354436}$
--------------------------	---

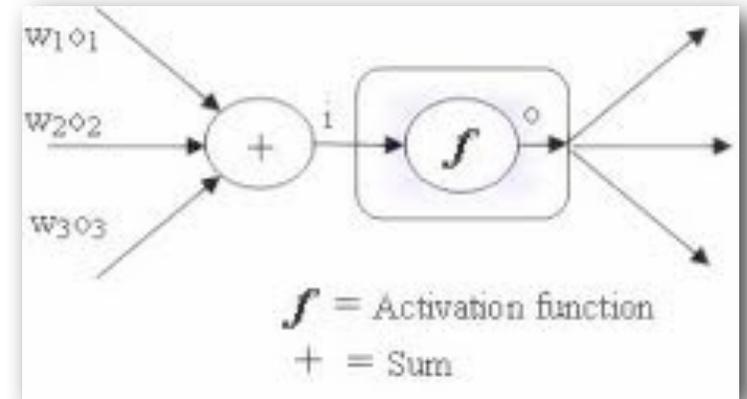
Ripetendo tale procedura, l'errore della rete si riduce progressivamente.

L'algoritmo termina correttamente (converge) quando l'errore della rete è nullo (situazione ideale) o quasi...

# Il processo di apprendimento

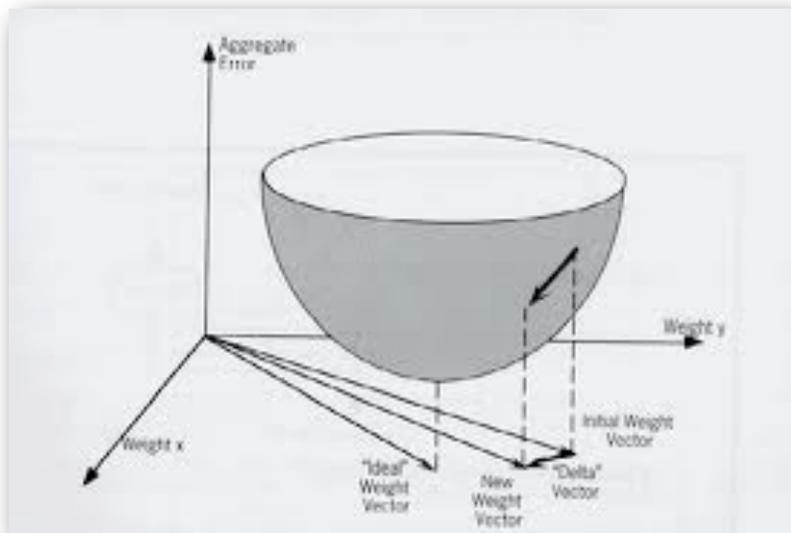
## Backward propagation

Il **Backpropagation** è un algoritmo di apprendimento supervisionato ed è principalmente usato dal Multi-Layer Perceptron per cambiare i pesi collegati agli strati di neuroni nascosti.



L'algoritmo di backpropagation utilizza l'errore di output per cambiare i valori dei pesi a ritroso.

Per ottenere l'errore deve essere stata precedentemente eseguita una fase "forward propagation", durante la quale i neuroni vengono attivati utilizzando la funzione di attivazione sigmoide.



# Il processo di apprendimento

## Backward propagation

La formula dell'**attivazione sigmoidea** è:  $f(x) = \frac{1}{1 + e^{-x}}$

L'algoritmo funziona nel modo seguente.

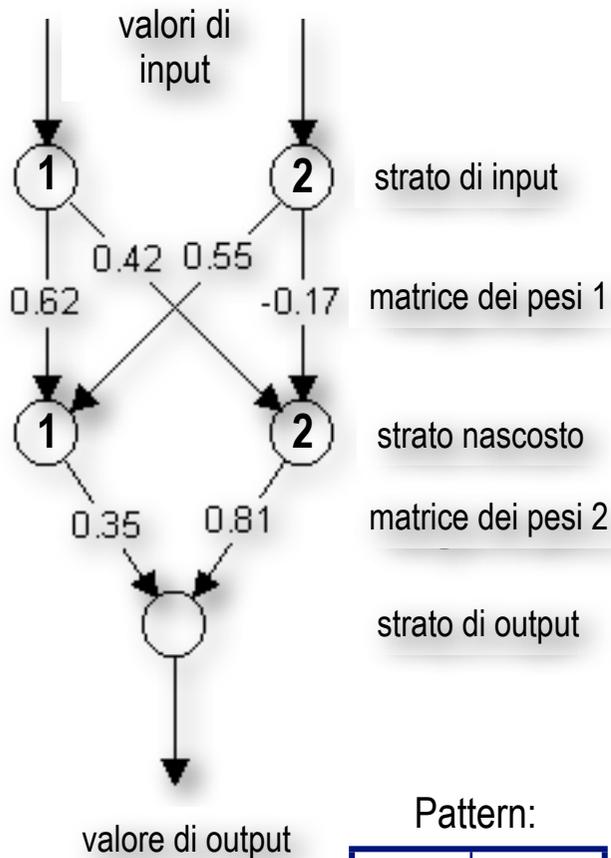
1. Eseguire la fase di forward propagation per un pattern di input e calcolare l'errore di uscita
2. Cambiare tutti i valori di ogni matrice dei pesi utilizzando la formula:  
 $(\text{vecchio})\text{peso} + \text{tasso di apprendimento} \times \text{errore di output} \times \text{neurone di output } (i) \times \text{neurone di output } (i+1) \times [1 - \text{neurone di output } (i+1)]$
3. Tornare al punto 1
4. L'algoritmo termina se tutti i pattern di output corrispondono ai target

# Il processo di apprendimento

## Backward propagation

### Esempio.

Supponiamo di avere il seguente Multi-Layer Perceptron a tre strati:



Pattern:

input	target
0 1	0
1 1	1

**(1)** Inizialmente i pesi vengono impostati ai valori casuali (0.62, 0.42, 0.55, -0.17) per la matrice dei pesi 1 e (0.35, 0.81) per la matrice 2, con  $\mu = 0.25$ .

Quindi viene fornito il primo ingresso (0 1) allo strato di input e vengono attivati i neuroni dello strato nascosto:

Input del neurone nascosto 1	$0 \times 0.62 + 1 \times 0.55 = 0.55$
Input del neurone nascosto 2	$0 \times 0.42 + 1 \times (-0.17) = -0.17$
Output del neurone nascosto 1	$1 / (1 + \exp(-0.55)) = 0.634$
Output del neurone nascosto 2	$1 / (1 + \exp(+0.17)) = 0.458$

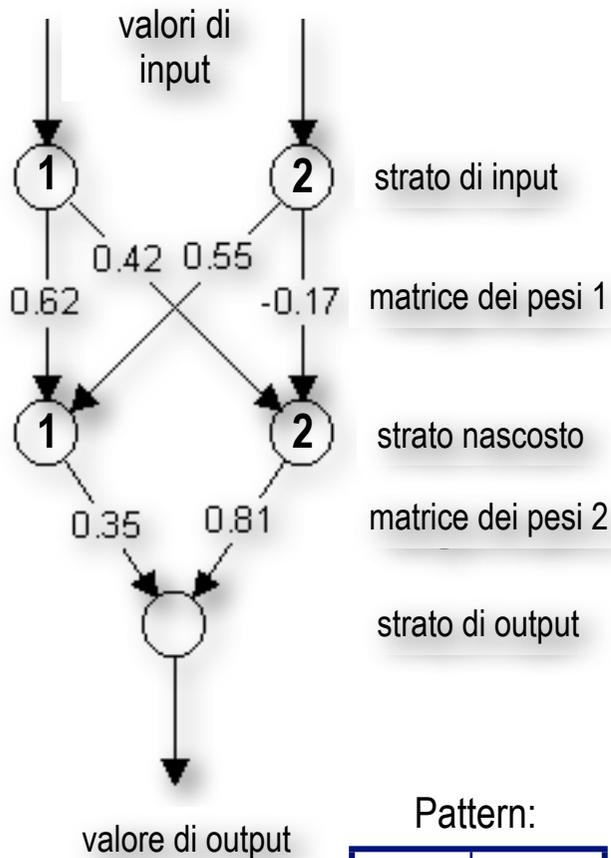
*I calcoli sono arrotondati alla terza cifra decimale*

# Il processo di apprendimento

## Backward propagation

### Esempio.

Supponiamo di avere il seguente Multi-Layer Perceptron a tre strati:



**(2)** Vengono ora attivati i neuroni dello strato di output:

Input del neurone di output	$0.634 \times 0.35 + 0.458 \times 0.81 = 0.593$
Uscita del neurone di output	$1 / (1 + \exp(-0.593)) = 0.644$
Calcolo errore di output	$0 - 0.644 = -0.644$

*I calcoli sono arrotondati alla terza cifra decimale*

Pattern:

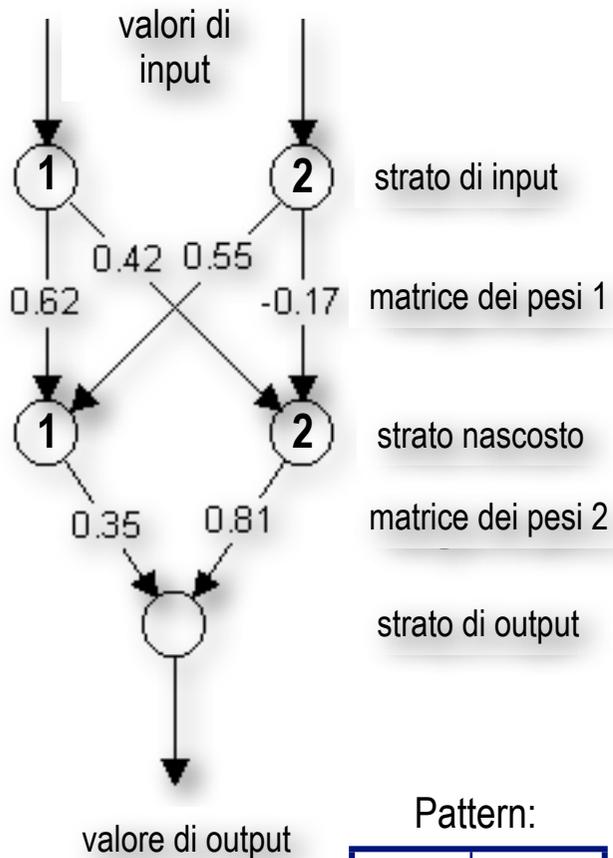
input	target
0 1	0
1 1	1

# Il processo di apprendimento

## Backward propagation

### Esempio.

Supponiamo di avere il seguente Multi-Layer Perceptron a tre strati:



Pattern:

input	target
0 1	0
1 1	1

**(3)** Dopo aver ottenuto l'errore di output, possiamo eseguire la backpropagation. Iniziamo modificando i pesi della matrice 2:

Valore di modifica del peso 1	$0.25 \times (-0.644) \times 0.634 \times 0.644$ $\times (1-0.644) = -0.023$
Valore di modifica del peso 2	$0.25 \times (-0.644) \times 0.458 \times 0.644$ $\times (1-0.644) = -0.017$
Modifica del peso 1	$0.35 + (-0.023) = 0.327$
Modifica del peso 2	$0.81 + (-0.017) = 0.793$

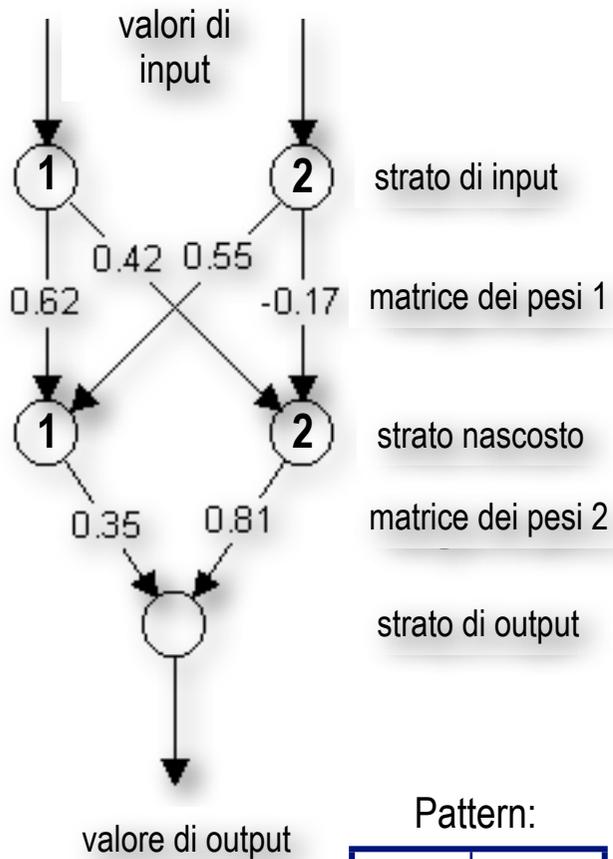
*I calcoli sono arrotondati alla terza cifra decimale*

# Il processo di apprendimento

## Backward propagation

### Esempio.

Supponiamo di avere il seguente Multi-Layer Perceptron a tre strati:



Pattern:

input	target
0 1	0
1 1	1

**(4)** Ora possiamo aggiornare i pesi della matrice 1:

Valore di modifica del peso 1	$0.25 \times (-0.644) \times 0 \times 0.634 \times (1-0.634) = 0$
Valore di modifica del peso 2	$0.25 \times (-0.644) \times 0 \times 0.458 \times (1-0.458) = 0$
Valore di modifica del peso 3	$0.25 \times (-0.644) \times 1 \times 0.634 \times (1-0.634) = -0.037$
Valore di modifica del peso 4	$0.25 \times (-0.644) \times 1 \times 0.458 \times (1-0.458) = -0.040$
Modifica del peso 1	$0.62 + 0 = 0.62$ (immutato)
Modifica del peso 2	$0.42 + 0 = 0.42$ (immutato)
Modifica del peso 3	$0.55 + (-0.037) = 0.513$
Modifica del peso 4	$-0.17 + (-0.040) = -0.210$

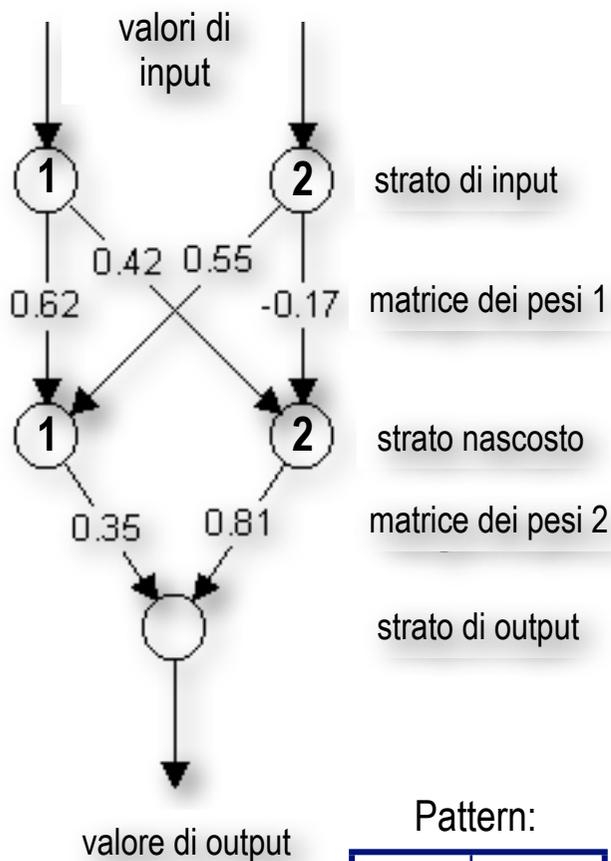
*I calcoli sono arrotondati alla terza cifra decimale*

# Il processo di apprendimento

## Backward propagation

### Esempio.

Supponiamo di avere il seguente Multi-Layer Perceptron a tre strati:



Pattern:

input	target
0 1	0
1 1	1

**(5)** Il primo pattern di input è stato propagato attraverso la rete.

- La stessa procedura verrà utilizzata per il successivo pattern di input, questa volta con i valori dei pesi modificati.
- Dopo la propagazione forward e backward del secondo input viene completato un passo di apprendimento (learning step, o “epoca”) e quindi si può calcolare l'errore di rete mediando gli errori quadratici di output.
- Eseguendo questa procedura più volte, l'errore diventa sempre più piccolo.
- L'algoritmo termina correttamente se l'errore della rete è pari a zero (situazione perfetta) o quasi.
- Si noti che questo algoritmo è applicabile anche per Multi-Layer Perceptron con più di un livello nascosto.

# Il processo di apprendimento

## Backward propagation

**Cosa succede se tutti i valori di un pattern di input sono zero?**



Se tutti i valori di un pattern di input sono zero, i pesi della matrice 1 non saranno mai modificati per esso e la rete non apprenderà mai quel pattern.

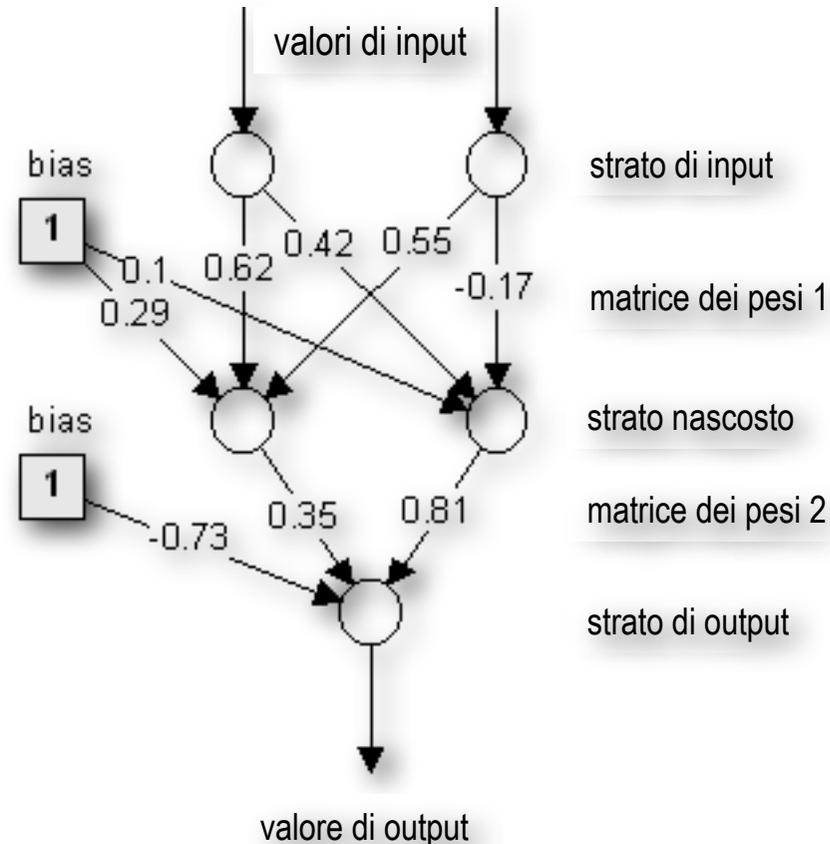
Per evitare ciò, viene creato uno “pseudo input” chiamato **Bias** con un valore costante pari a 1.

Questo cambia la struttura della rete nel modo seguente:

# Il processo di apprendimento

## Backward propagation

### MLP a tre strati con Bias



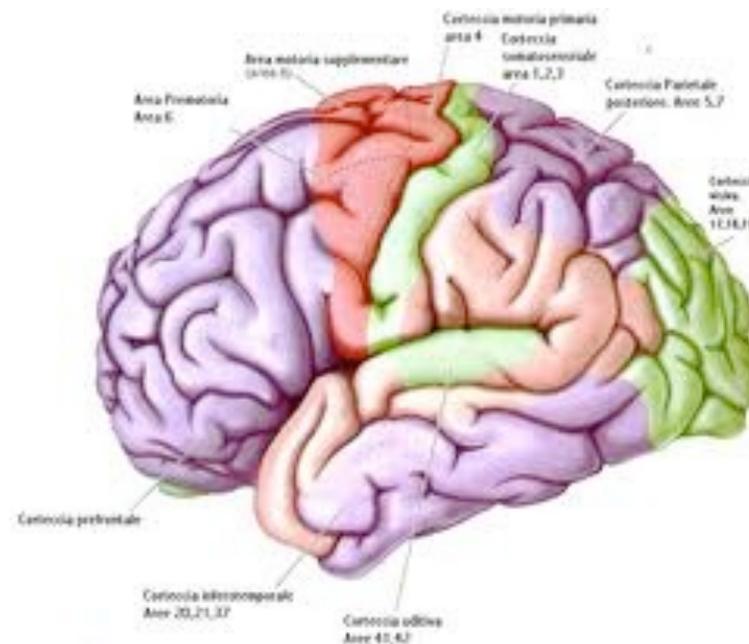
- ▶ I **pesi aggiuntivi**, associati ai Bias in ingresso ai neuroni degli strati nascosti e di output, hanno valori iniziali casuali e vengono aggiornati analogamente agli altri.
- ▶ Inviando un valore costante 1 ai neuroni, viene garantito che i loro valori di input siano sempre diversi da zero.

# Il processo di apprendimento

## Self organization

“**Self organization**” (auto-organizzazione) è un algoritmo di apprendimento non supervisionato utilizzato dalle reti neurali con mappa caratteristica di Kohonen (Kohonen Feature Map).

Come accennato nei paragrafi precedenti, una rete neurale cerca di simulare il cervello biologico umano, e l'auto-organizzazione è probabilmente il modo migliore per rendersene conto.



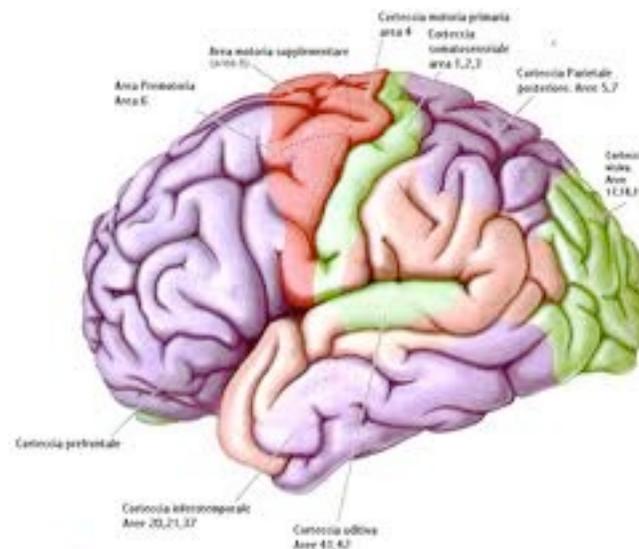
# Il processo di apprendimento

## Self organization

È comunemente noto che la corteccia del cervello umano è suddivisa in **diverse regioni**, ognuna responsabile di determinate funzioni. Le cellule neurali si organizzano in **gruppi**, secondo le informazioni in arrivo.

Le informazioni in input non solo sono ricevute dalle singole cellule neurali, ma influenzano anche le altre cellule vicine.

Questa organizzazione si traduce in una specie di **mappa**, nella quale le cellule neurali con funzioni simili vengono disposte in raggruppamenti omogenei (cluster).



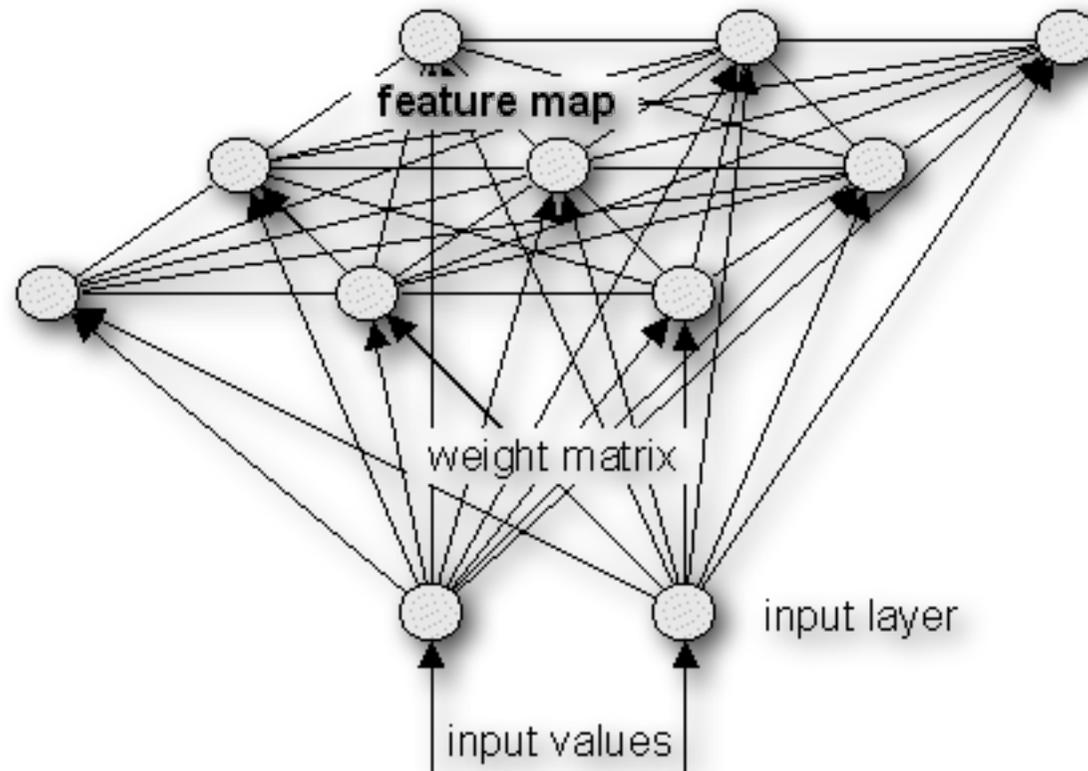
# Il processo di apprendimento

## Self organization

Questo processo di auto-organizzazione può essere eseguito anche da una rete neurale.

Questo tipo di rete neurale è per lo più utilizzato per la **classificazione**, perché i valori di input simili vengono agglomerati in zone della mappa.

Una **struttura tipica** di una mappa di Kohonen che utilizza l'algoritmo self organization è la seguente:

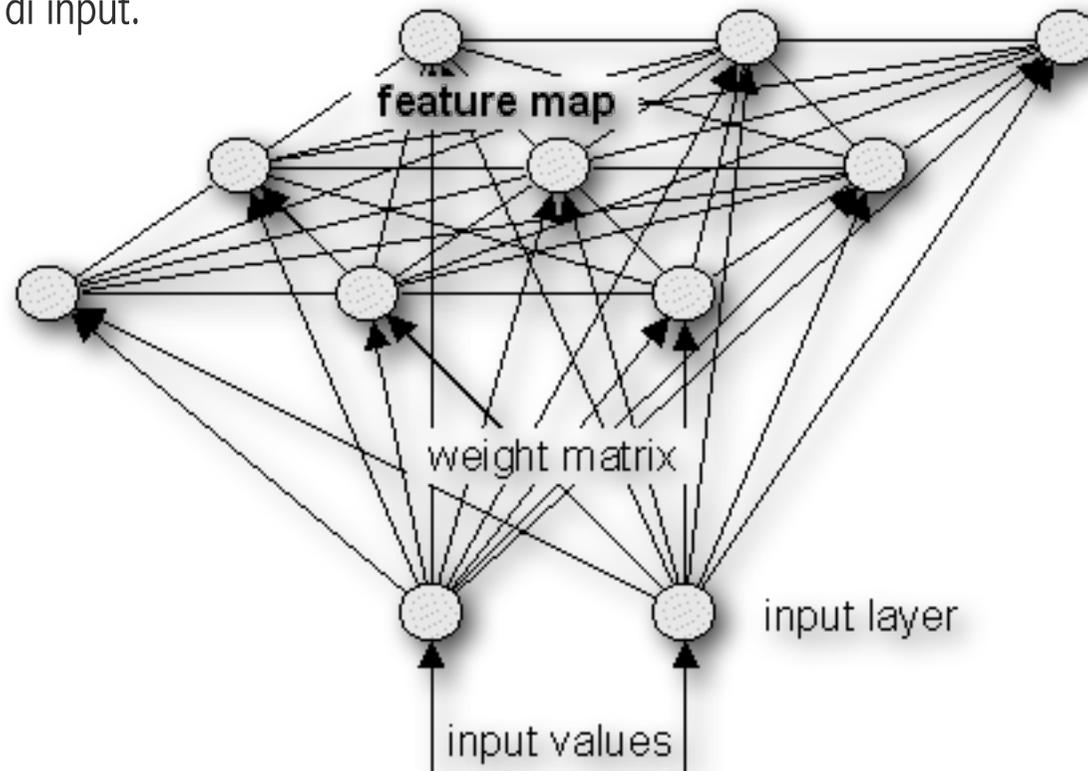


*Kohonen Feature Map con input a due dimensioni e mappa bidimensionale (3x3 neuroni)*

# Il processo di apprendimento

## Self organization

- Come si può vedere, **ogni neurone del livello di input è collegato ad ogni neurone sulla mappa.**
- La matrice dei pesi risultante è usata per propagare i valori di input della rete ai neuroni della mappa.
- Inoltre, **tutti i neuroni nella mappa sono collegati tra loro.** Questi collegamenti vengono utilizzati per influenzare i neuroni in una certa area di attivazione intorno al neurone con la massima attivazione, ricevuta dall'uscita del livello di input.



*Kohonen Feature Map con input a due dimensioni e mappa bidimensionale (3x3 neuroni)*

# Il processo di apprendimento

## Self organization

La quantità di feedback tra i neuroni della mappa viene di solito calcolata mediante la **funzione di Gauss**:

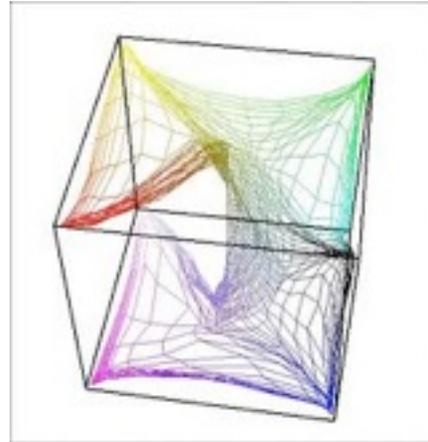
$$\text{feedback}_{ci} = e^{\frac{-|x_c - x_i|^2}{2r^2}}$$

dove

- $x_c$  è la posizione del neurone più attivato (centroide);
- $x_i$  sono le posizioni degli altri neuroni della mappa;
- $r$  è l'area di attivazione (raggio).

# Il processo di apprendimento

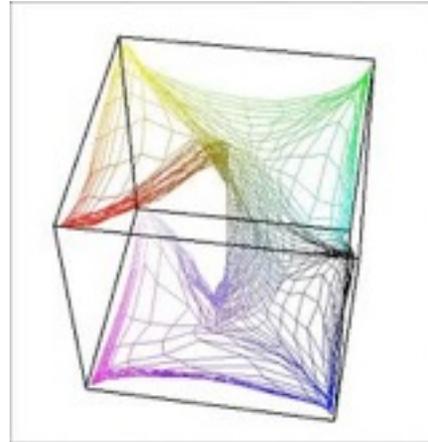
## Self organization



- Inizialmente, l'**area di attivazione** è di grandi dimensioni come anche il feedback tra i neuroni della mappa.
- Ciò si traduce in una attivazione dei neuroni in una vasta area intorno al **neurone più attivo**.
- Con il progredire dell'apprendimento, l'area di attivazione viene costantemente diminuita e solo i neuroni più vicini al centro di attivazione sono influenzati dal neurone più attivo.

# Il processo di apprendimento

## Self organization



- A differenza del modello biologico, **i neuroni della mappa non cambiano la loro posizione** sulla mappa.
- L' "organizzazione" è simulata modificando i valori nella matrice dei pesi (allo stesso modo delle altre reti neurali).

Poichè questo è un algoritmo di apprendimento non supervisionato, non vi sono pattern di ingresso/target preesistenti.

I valori in input alla rete sono presi da un range predefinito, e rappresentano i "dati" che devono essere organizzati.

# Il processo di apprendimento

## Self organization

L'algoritmo "self organization" funziona nel modo seguente.

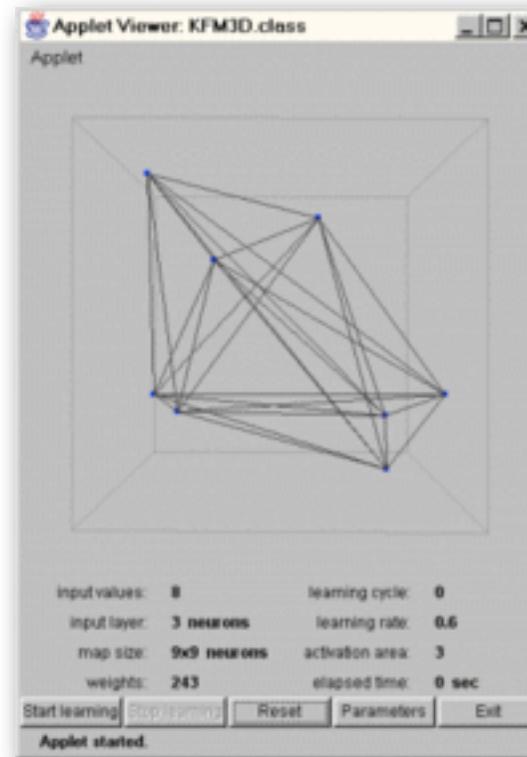
1. Definire l'intervallo dei valori di input
2. Impostare tutti i pesi a valori casuali presi nel campo dei valori di ingresso
3. Definire l'area iniziale (raggio) di attivazione
4. Prendere un valore a caso dal set di input e passarlo ai neuroni dello strato di input
5. Determinare il neurone più attivo sulla mappa:
  - *moltiplicare l'uscita del livello di input con i valori dei pesi;*
  - *il neurone mappa con il maggior valore risultante è detto essere "il più attivo";*
  - *calcolare il valore di feedback di ogni altro neurone mappa utilizzando la funzione di Gauss.*
6. Ottenere i nuovi valori dei pesi  $w_{i+1}$  con la formula:
$$w_{i+1} = w_i + \text{feedback} \times (\text{valore di ingresso} - w_i) \times \text{tasso di apprendimento}$$
7. Ridurre l'area di attivazione
8. Tornare al punto 4
9. L'algoritmo termina se l'area di attivazione è più piccola di un valore specificato

# Il processo di apprendimento

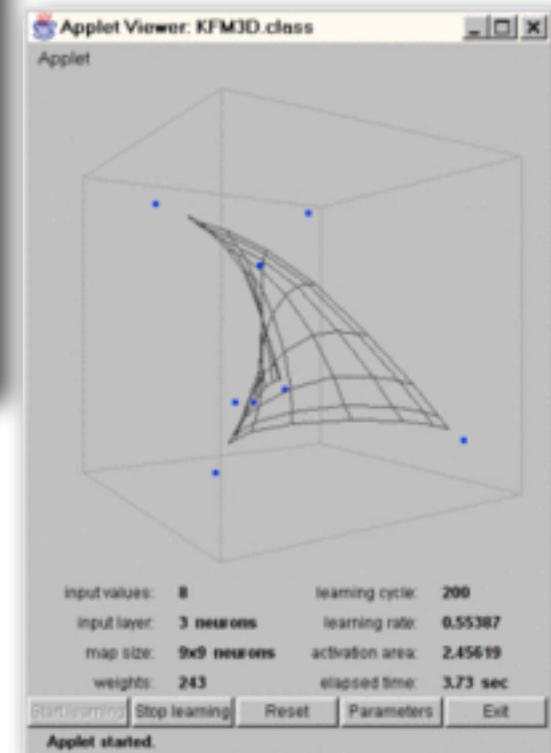
## Self organization

**Esempio:** vedi animazione.

- ▶ La Kohonen Feature Map dell'esempio ha tre neuroni nel suo strato di input che rappresentano i valori delle dimensioni spaziali x, y e z.
- ▶ La mappa caratteristica è inizialmente bidimensionale, e ha  $9 \times 9$  neuroni.
- ▶ La matrice risultante ha  $3 \times 9 \times 9 = 243$  pesi, perché ogni neurone di input è collegato ad ogni neurone della mappa.



*Schermate iniziali dell'animazione*

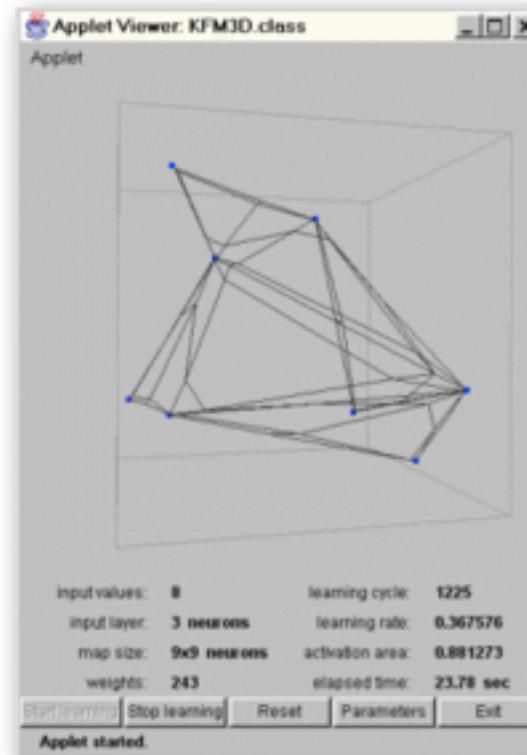


# Il processo di apprendimento

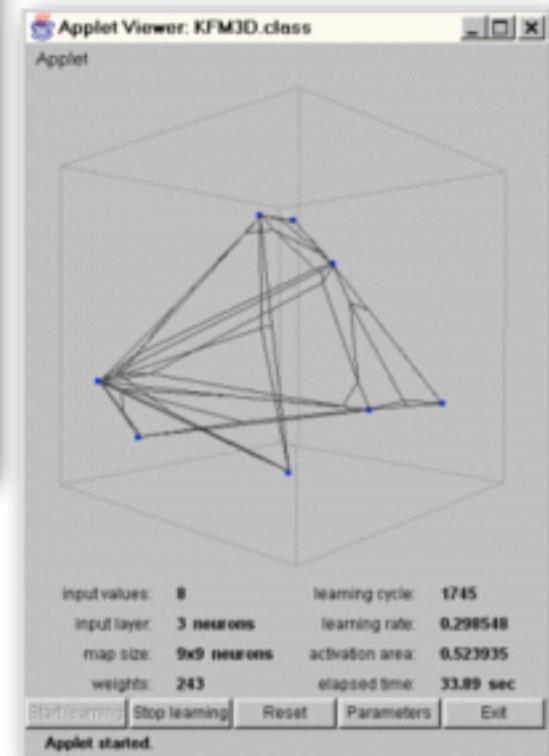
## Self organization

**Esempio:** vedi animazione.

- ▶ Man mano che l'apprendimento progredisce, la mappa diventa sempre più strutturata.
- ▶ Dall'animazione si può osservare che i neuroni della mappa "tentano" di avvicinarsi il più possibile al loro valore di input (punto bu) più vicino.



*Schermate successive  
dell'animazione*

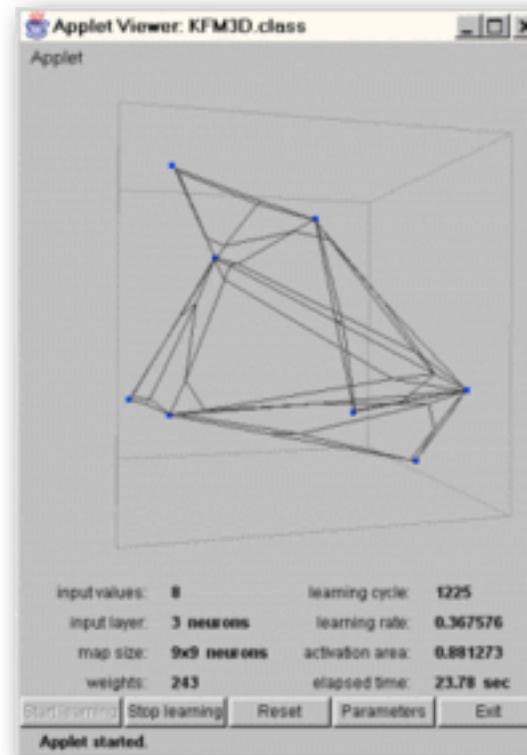


# Il processo di apprendimento

## Self organization

**Esempio:** vedi animazione.

- ▶ Al termine del processo di apprendimento, la mappa caratteristica “aggancia” tutti i valori di input (punti blu).
- ▶ La griglia non è geometricamente regolare poiché i neuroni interni alla mappa tentano anch’essi di avvicinarsi ai punti di input.
- ▶ Questo porta ad una visione distorta della griglia.



*Schermate successive dell'animazione*

