Tecniche di riconoscimento statistico

Teoria e applicazioni industriali

Parte 4 – Reti neurali per la classificazione

Ennio Ottaviani
On AIR srl
ennio.ottaviani@onairweb.com
http://www.onairweb.com/corsoPR

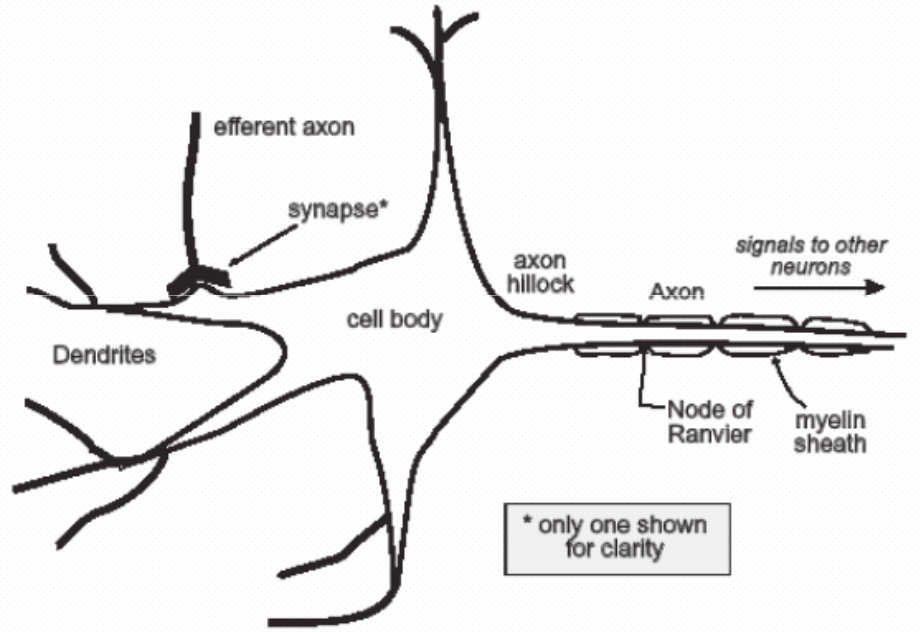
A.A. 2018-2019

Origine delle reti neurali

- Le reti neurali (artificiali) nascono come uno dei campi della intelligenza artificiale (IA)
- Obiettivo IA: replicare per mezzo di macchine (hw+sw) le attività mentali umane
- Tesi IA forte: l'attività mentale consiste nella esecuzione di qualche sequenza ben definita di operazioni di manipolazione di simboli (algoritmo)
- La complessità del cervello umano come computer appare ancora aggi irraggiungibile !

Il neurone biologico

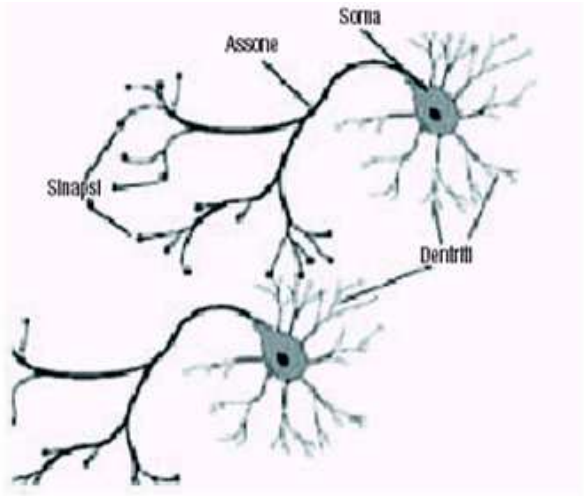
- Un neurone è composto da tre parti: coro (soma), dendriti ed assone



- La comunicazione avviene tramite segnali emessi dall'assone e ricevuti dalle dendriti di altri neuroni tramite interfacce (sinapsi)

Attivazione del neurone

- Il corpo del neurone integra le migliaia di segnali elettrici microscopici provenienti dalle dendriti
- Se l'integrazione produce una effetto globale superiore ad una soglia di attivazione, un impulso elettrico viene propagato lungo l'assone
- L'impulso viene ricevuto da un altro neurone, che può quindi modificare il suo stato e così via a cascata …



Apprendimento hebbiano

- Gli studi di Hebb (anni ’40) combinano dati fisiologici e studi comportamentali sui primati, formando una prima teoria biologica dell’apprendimento

- *«se un neurone A è abbastanza vicino ad un neurone B da contribuire ripetutamente e in maniera duratura alla sua eccitazione, allora ha luogo in entrambi i neuroni un processo di crescita o di cambiamento metabolico tale per cui l'efficacia di A nell'eccitare B viene accresciuta»*

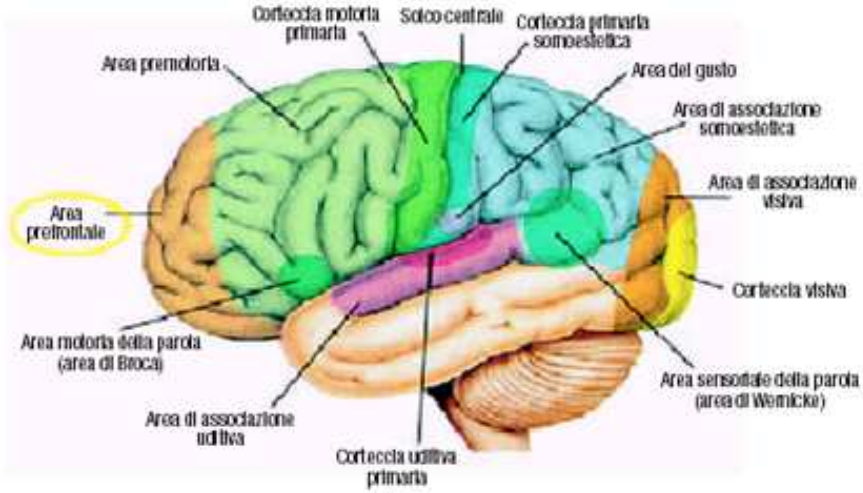
- Più direttamente … due neuroni che scaricano assieme formano una connessione potenziata

La metafora neurobiologica

- È opinione condivisa che i segnali elettrici presenti nei neuroni siano alla base dell'elaborazione dell'informazione a livello cerebrale

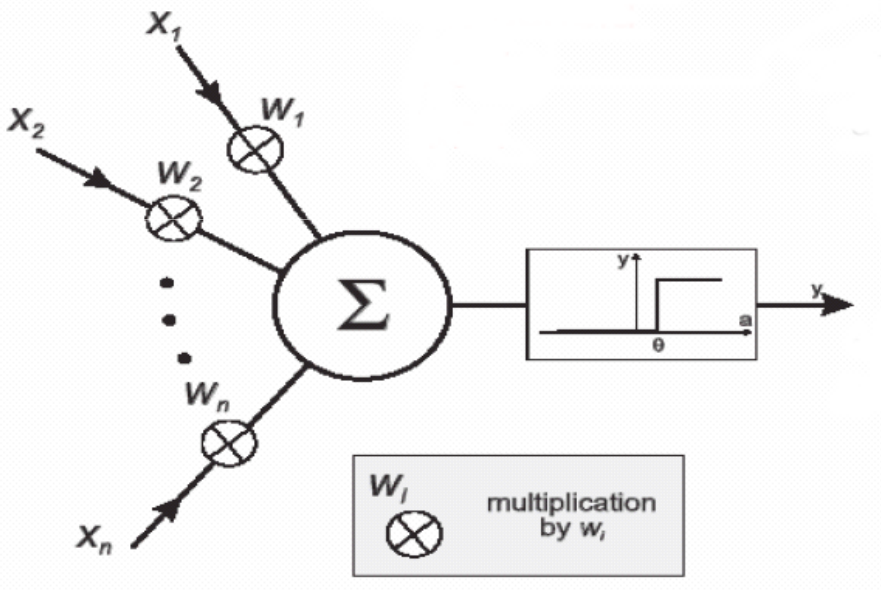
- C'è evidenza sperimentale che le sinapsi siano influenzate dalla dalle esperienze, dall'apprendimento di compiti specifici, etc…

- È il particolare pattern di interconnessioni e di forza delle sinapsi, che definisce le proprietà funzionali di ogni porzione del cervello



Il neurone artificiale

- Un semplice modello di neurone artificiale emula questa funzionalità di base in termini formali (percettrone di McCulloch-Pitts, 1943)



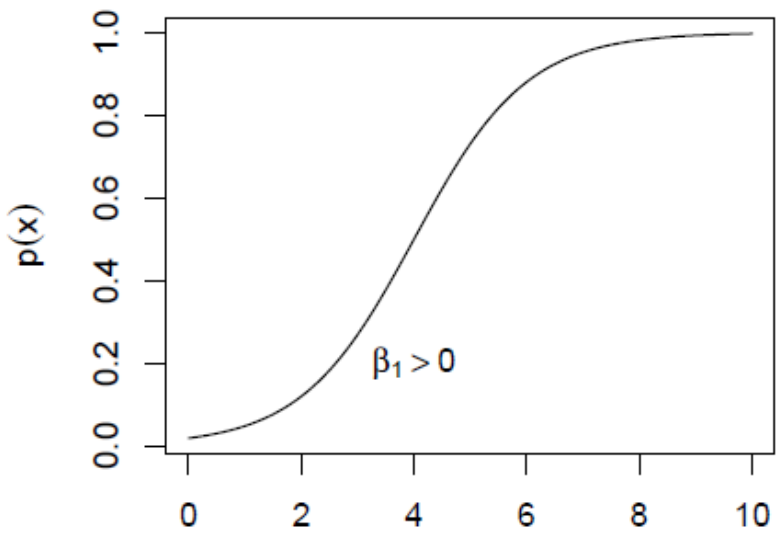
- Ingressi ed uscita sono variabili binarie. La funzione di attivazione a è una combinazione lineare degli ingressi. L'uscita è determinata da una soglia θ

Legame con funzione logistica

- Versioni successive utilizzano funzioni non-lineari di tipo *soft* anziché *hard* (dette sigmoidi)
- Ingressi ed uscita sono ora variabili numeriche
- Analogia diretta con la regressione logistica (caso particolare di variabili che esprimono la probabilità del verificarsi di un evento)

$$E(Y|x) = p(x) = \frac{e^{\beta \cdot \mathrm{x}}}{1 + e^{\beta \cdot \mathrm{x}}}$$

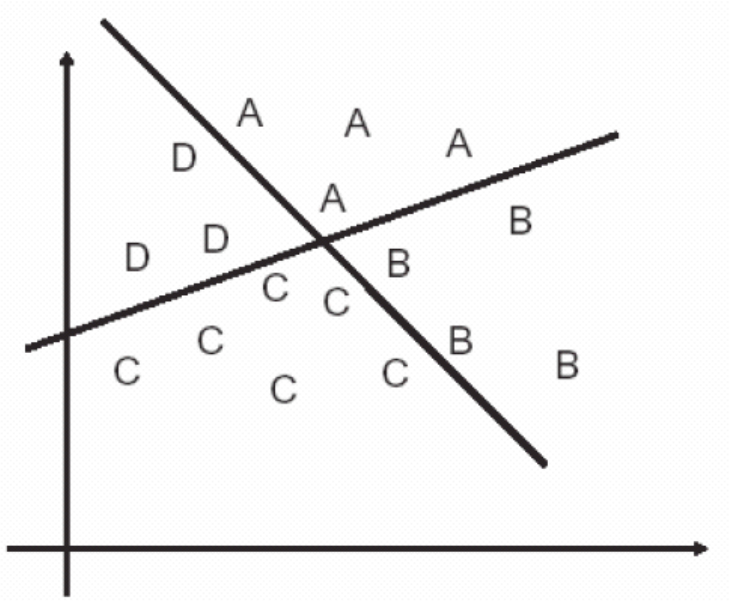
$$\mathrm{logit} p(x) = \ln \frac{p(x)}{1 - p(x)} = \beta \cdot \mathrm{x}$$



Neuroni per la classificazione

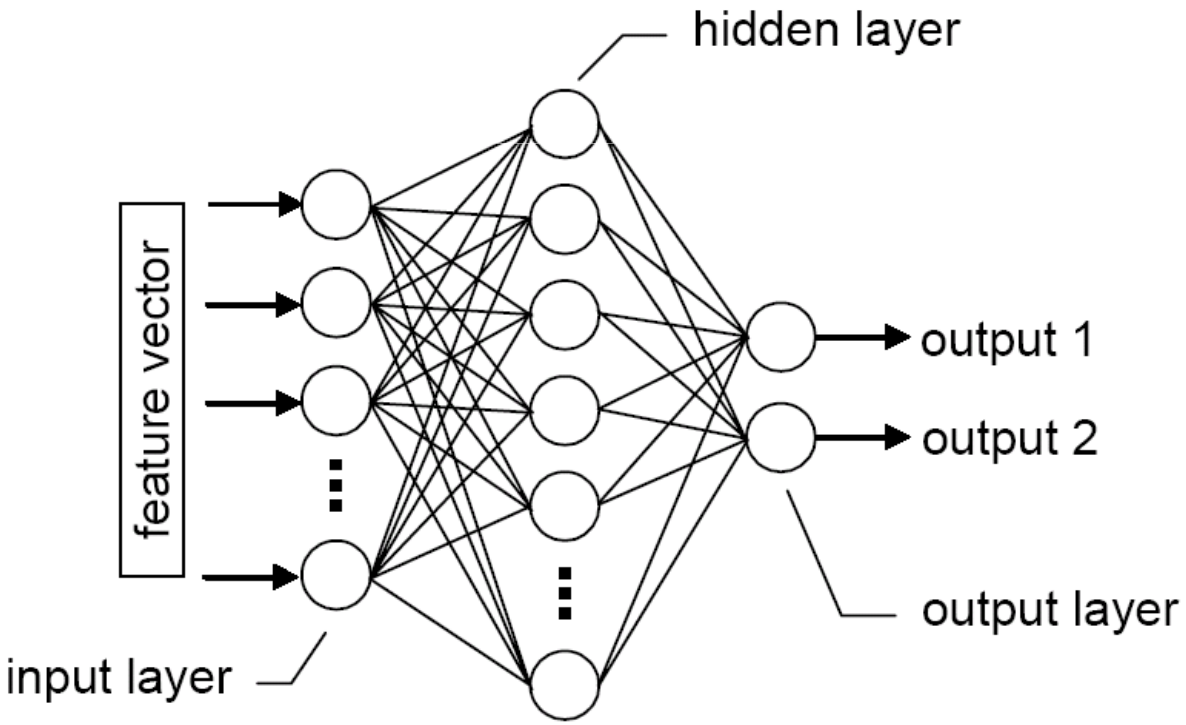
- Un singolo neurone può risolvere problemi di classificazione a 2 classi (se linearmente separabili)

- Per problemi con più classi si possono combinare più neuroni, ciascuno dei quali risolve un problema binario, con logiche booleane tipo AND/OR



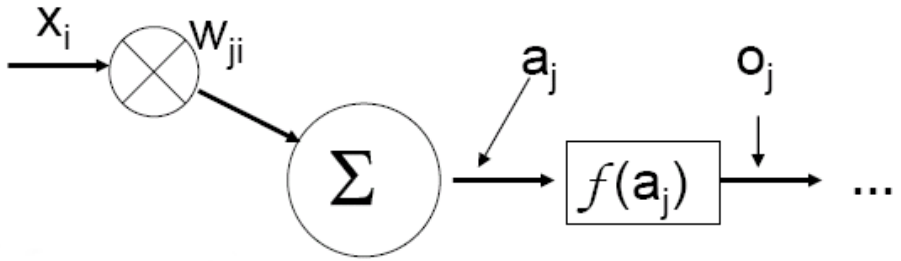
Schema di rete multistrato

- La generalizzazione dell'approccio neuronale alla classificazione prevede la realizzazione di strutture a strati, in cui i neuroni di ogni strato ricevono come dato di ingresso, l'uscita dello strato inferiore



Struttura locale

- Ogni nodo della rete multistrato svolge lo stesso tipo di elaborazione



- Esso dispone di un certo numero di gradi di libertà w_{ji} (pesi delle connessioni) tramite i quali la rete si può adattare ai vari problemi

- La relazione tra l'attivazione a_j e l'uscita o_j è espressa da una specifica funzione f (in generale non lineare)

- Il modello è noto come MLP (Multi Layer Perceptron)

Apprendimento

- L'apprendimento comporta un processo di stima dei pesi ottimale per ottenere le uscite desiderate sui campioni del *training set*

- L'obiettivo è in genere quello di minimizzare una funzione di errore differenziabile (*loss function*)

$$E = \frac{1}{2}\sum_{k}(d_k - o_k)^2$$

- Per problemi di classificazione su N classi si usano N neuroni di uscita, con funzione sigmoidale tale da generare valori in [0,1]. In questo modo E risulta proporzionale all'errore di classificazione

- Nota: le uscite della rete NON sono probabilità a posteriori !

Algoritmo di apprendimento

- Seguendo Rumelhart (1986), viene usata la tecnica della discesa del gradiente

$$w_{ji}^{n+1} = w_{ji}^{n} - \eta \left. \frac{\partial E}{\partial w_{ji}} \right|_{w^n}$$

- Applicando la regola di concatenazione delle derivate si ha

$$\frac{\partial E}{\partial o_k} = -(d_k - o_k) \qquad \frac{\partial o_k}{\partial a_k} = \frac{\partial f(a_k)}{\partial a_k} = f'(a_k)$$

$$\frac{\partial a_k}{\partial w_{kj}} = \frac{\partial}{\partial w_{kj}} \sum_i w_{ki} x_i = x_j$$

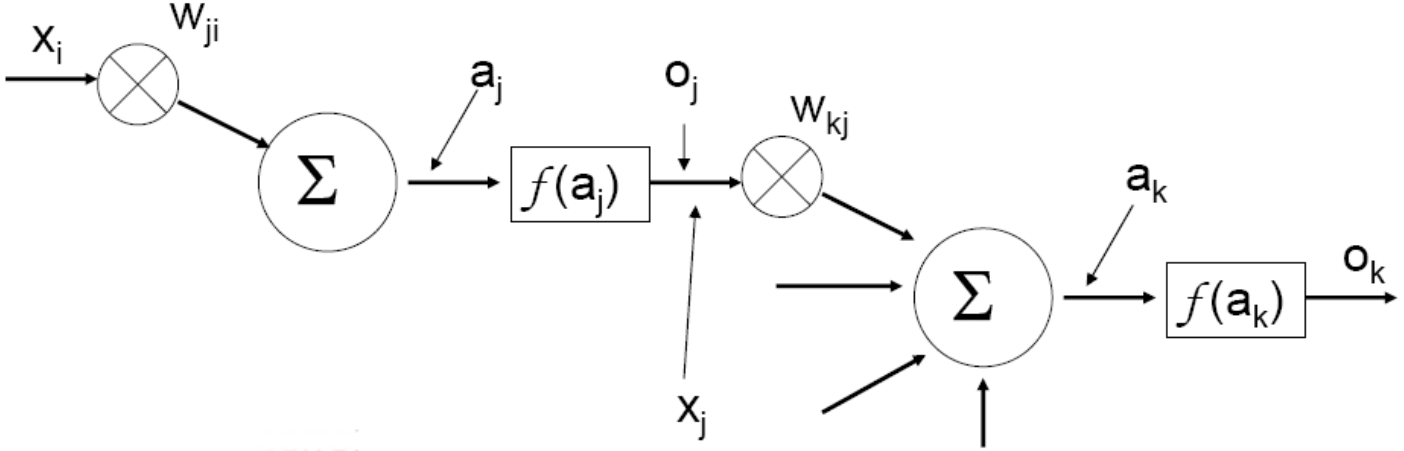
Algoritmo di apprendimento

- Combinando i risultati si ottiene la regola di apprendimento finale

$$w_{kj}^{n+1} = w_{kj}^{n} + \eta(d_k - o_k) f'(a_k) x_j \Big|_{w^n}$$

- Essa è valida per i neuroni dell'ultimo strato, per i quali è disponibile un valore atteso per la variabile di uscita

- Per i neuroni interni la situazione è più complicata



Algoritmo di apprendimento

- La quantità da calcolare è

$$\frac{\partial E}{\partial o_j} = \frac{\partial}{\partial o_j}\frac{1}{2}\sum_k (d_k - o_k)^2 \qquad o_k = f\left(\sum_j w_{kj} o_j\right)$$

$$\frac{\partial E}{\partial o_j} = -\sum_k (d_k - o_k) f'(a_k) w_{kj}$$

- Combinando i termini otteniamo finalmente

$$\frac{\partial E}{\partial w_{ji}} = -\left[\sum_k (d_k - o_k) f'(a_k) w_{kj}\right] f'(a_j) x_i$$

$$w_{ji}^{n+1} = w_{ji}^{n} + \eta\left[\sum_k (d_k - o_k) f'(a_k) w_{kj}\right] f'(a_j) x_i \Big|_{w^n}$$

Back Propagation

- Vediamo in sintesi il processo di apprendimento noto come *back propagation*
  - Si inizializzano i pesi della rete a valori qualunque (es. random)
  - Si presenta in ingresso alla rete un campione del training set
  - Si calcolano le uscite dei neuroni nascosti, che forniscono l'ingresso per lo strato di uscita
  - Si calcolano i valori di uscita e si confrontano con i valori attesi
  - Si aggiornano i pesi dello strato di uscita
  - Si aggiornano i pesi dello strato nascosto
  - Si ripete per tutti gli elementi del training set
  - Si verifica una condizione di terminazione e, se non è raggiunta, si ripete ancora ripresentando ancora gli stessi campioni (epoca)

Curva di apprendimento

- All'inizio del training, l'errore è ovviamente alto, ma al progredire del processo esso cala fino ad un valore asintotico

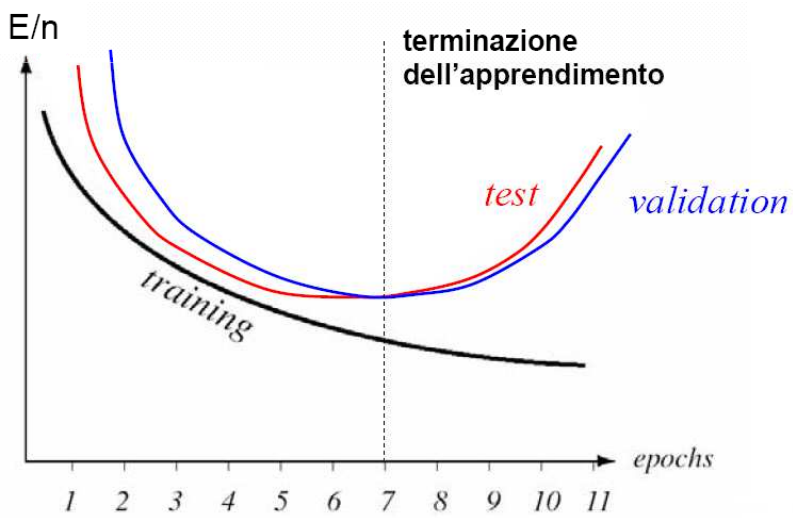
- Questo valore dipende da:
  - Caratteristiche intrinseche del training set
  - Numero di campioni
  - Numero di pesi della rete
  - Inizializzazione utilizzata
  - Parametri dell'algoritmo

- La condizione di terminazione non deve essere troppo stringente, per evitare l'*overtraining*
- Nota: la convergenza all'ottimo globale non è garantita

Criterio di terminazione

- Per garantire la capacità di generalizzazione, si utilizza una parte del *training set* come insieme di validazione

- L'insieme di validazione non viene usato per aggiornare i pesi ma il suo errore viene comparato con quello ottenuto sui campioni di training.



- La curva di apprendimento è in generale molto irregolare

Uso del “momentum”

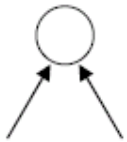
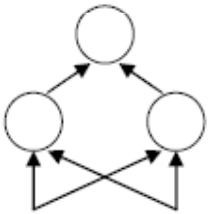
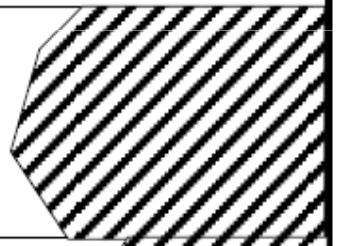
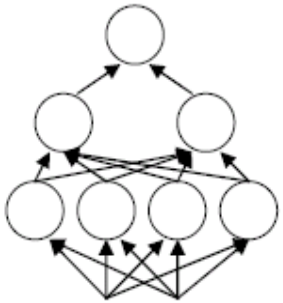
- Tecnica utilizzata nella backpropagation per far reagire la rete alla storia degli stimoli oltre che allo stimolo singolo

$$\Delta w^{n+1} = \mu\,(-\eta \nabla E^{n}) + (1-\mu)\Delta w^{n}$$

- Si forma una combinazione convessa ($0<\mu<1$) tra l’aggiornamento corrente e quello al passo precedente (filtro passa-basso)

- La scelta accurata del valore di momentum μ rende l’apprendimento più “regolare” e migliora la convergenza verso l’ottimo globale

Regioni di decisione

Struttura	Regioni di decisione	Forma generale
	Semispazi delimitati da iperpiani	
	Regioni convesse	
	Regioni di forma arbitraria	

Problemi realizzativi

- L'approccio risulta vincente nei problemi in cui esistano poche informazioni modellistiche sulle classi ma molti esempi già classificati
- I risultati dipendono criticamente dal preprocessing utilizzato (es. standardizzazione / normalizzazione)
- La costruzione del classificatore comporta comunque diverse scelte, che possono essere condotte solo per tentativi:
  - Funzioni di attivazione (es. sigmoidi)
  - Topologia della rete (quanti strati, quanti nodi)
  - Parametri di learning
  - Condizione di terminazione
  - Operazioni di preprocessing sui dati
- Si osserva a volte che: "*neural network researchers routinely reinvent methods that have been known in the statistical or mathematical literature for decades*"

Dimensionamento della rete

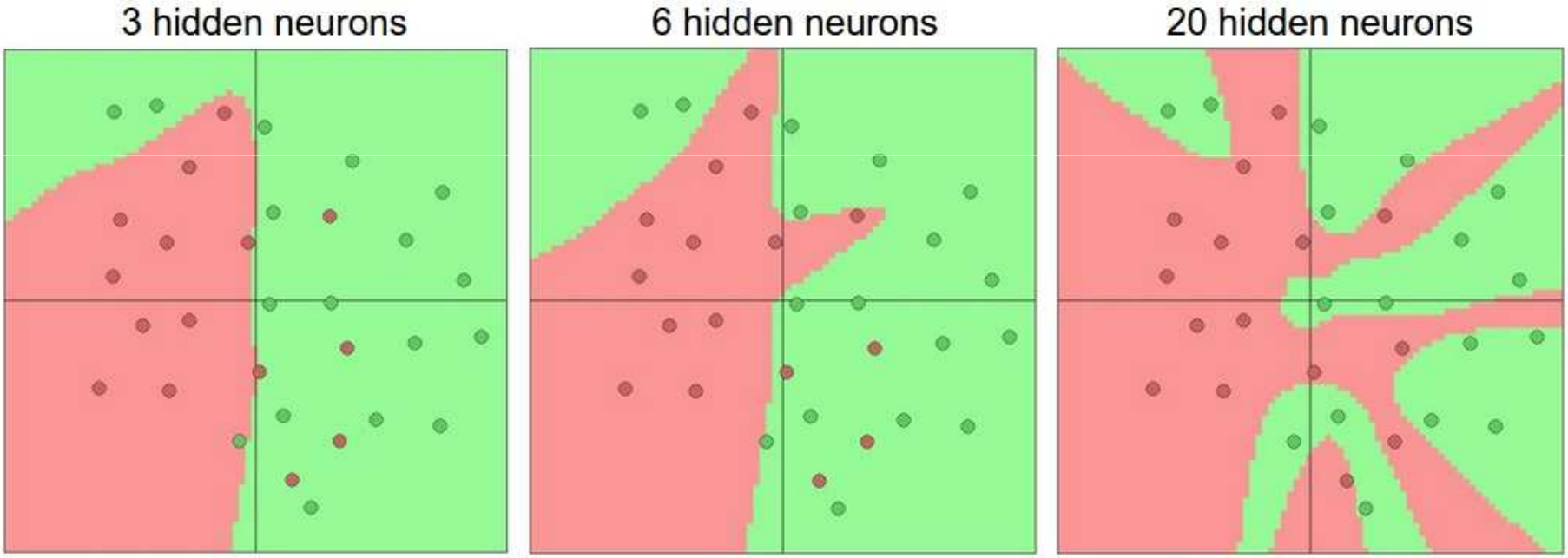
- Una rete MLP a singolo strato con numero arbitrario N di nodi hidden sigmoidali può approssimare qualunque funzione continua su un dominio finito (teorema di approssimazione universale, Cybenko 1989)

- In pratica il valore di N è difficile da stimare e si procede per tentativi
  - N troppo grande → la rete generalizza male
  - N troppo piccolo → la rete sbaglia troppo

- Una stima grezza di N si può avere in base al numero di autovalori "significativi" della matrice di covarianza delle features utilizzate

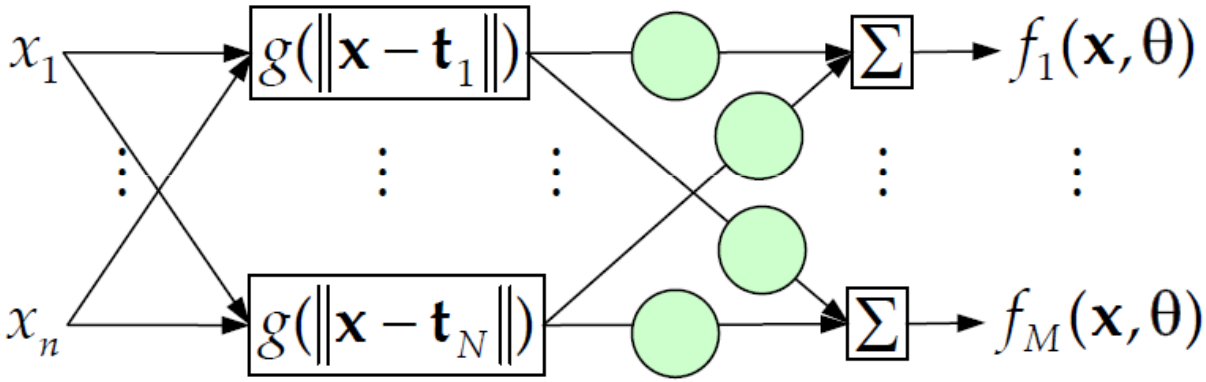
Dimensionamento della rete

- Nelle reti MLP, il numero di nodi dello strato nascosto è sempre il parametro più critico da stimare.
- La scelta migliore può essere individuata tramite cross-validation



Reti RBF

- Gli L nodi hidden rappresentano particolari vettori $t_1 \ldots t_L$ analoghi a quelli di input (centri) e la loro attivazione dipende dalla distanza secondo un funzione opportuna *g* (Radial Base Function)

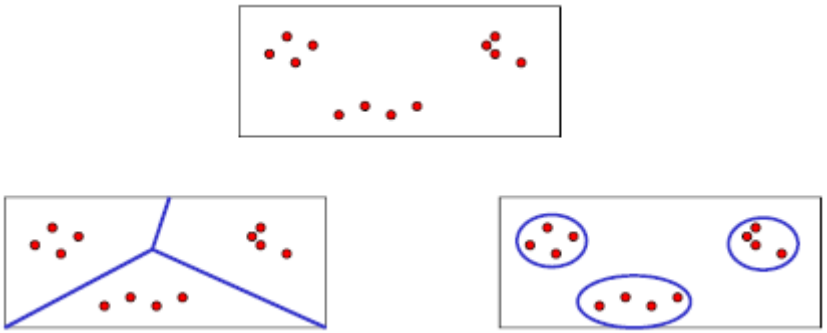


- La uscita j-esima vale

$$f_j(\mathbf{x},\theta) = \sum_{\ell=1}^{L} c_{j\ell}\, g(\|\mathbf{x}-\mathbf{t}_\ell\|)$$

Reti RBF

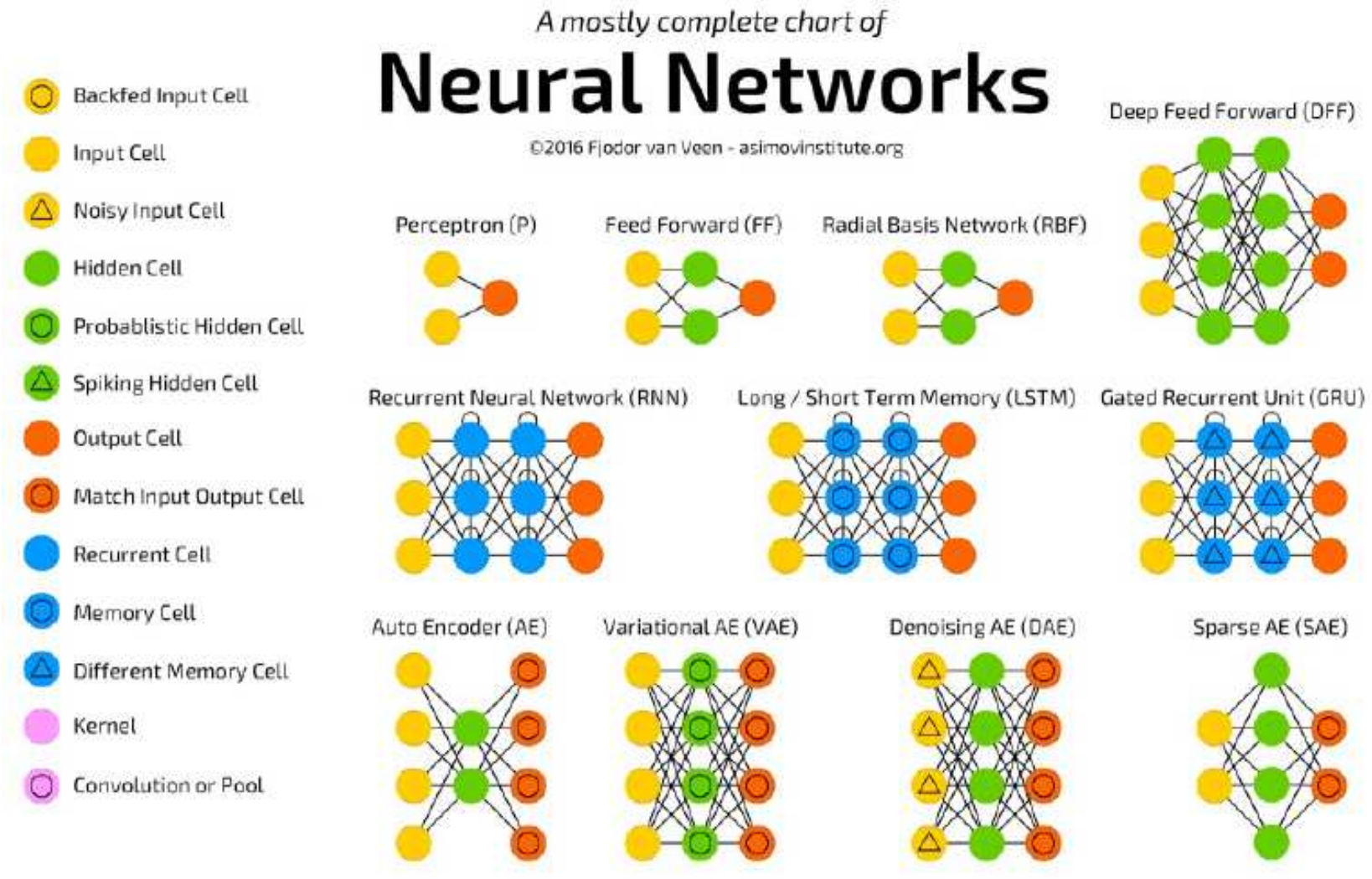
- Nei problemi di classificazione consentono di modellare bene classi complesse con pochi nodi hidden, risultando spesso più efficienti di MLP
- Dati i centri (determinati ad es. con un clustering), il calcolo dei pesi è molto semplice e senza problemi di convergenza (sistema lineare sovradeterminato risolto ai minimi quadrati)



Reti PNN

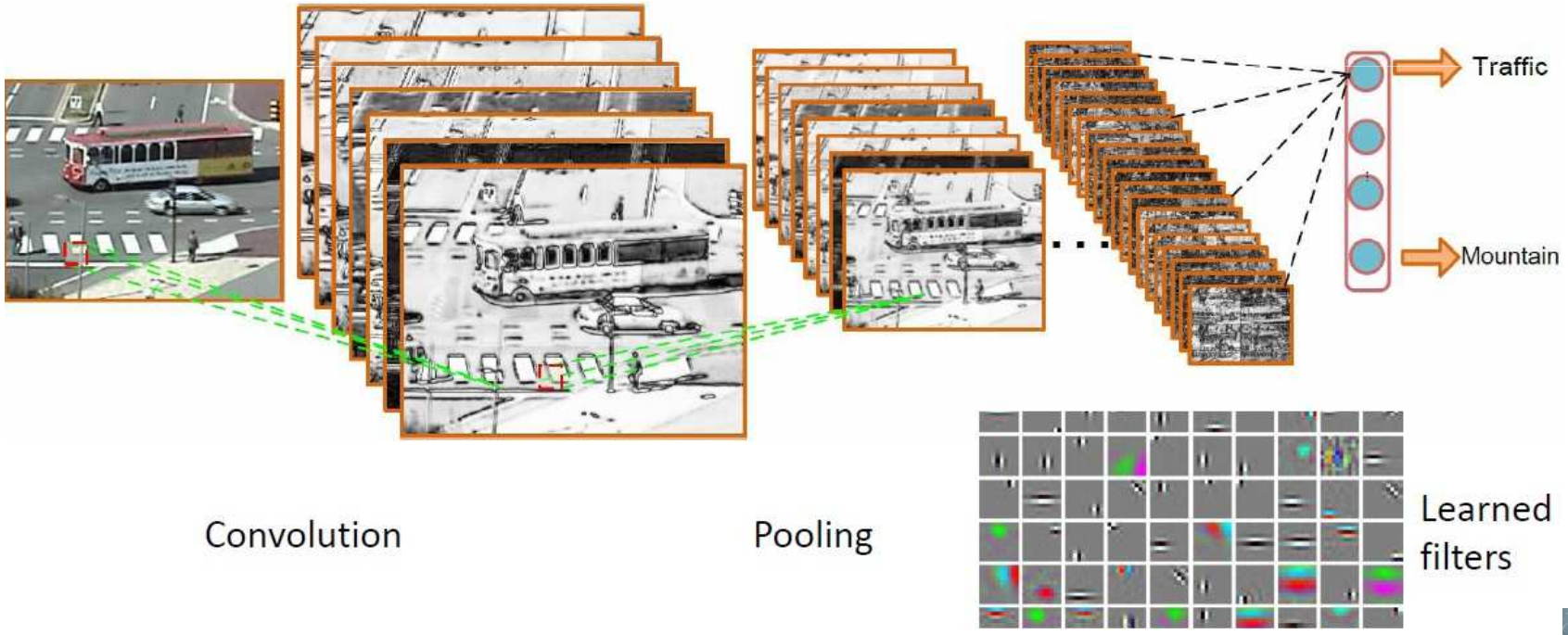
- Sono una evoluzione delle RBF orientate alla stima diretta della probabilità a posteriori di ogni classe (Probabilistic Neural Network)
- Utilizzano una funzione di attivazione di tipo RBF ma i centri sono tutti i vettori del training set (fattibile per dataset piccoli)
- Non dovendo scegliere i centri, gli unici parametri da scegliere sono quelli della RBF, ed i pesi sono calcolati direttamente risolvendo un sistema di equazioni lineari.
- Sono quindi un modo semplice per approssimare il classificatore bayesiano ideale

Lo zoo neurale



Evoluzione: deep learning

- Rappresenta una evoluzione delle macchine MLP con l'obiettivo di catturare in modo più profondo il problema dell'apprendimento, includendo anche l'estrazione delle features
- Lo schema più diffuso è la rete convolutiva (CNN)



Evoluzione: deep learning

- Tutto il PR è stato sviluppato separando l'estrazione delle features dalla classificazione
- Le features vengono selezionate spesso in modo del tutto a-priori, spesso per il loro significato "antropico"
- Apprendere le features insieme alla classificazione permette la loro integrazione ottimale
- Questo complica l'apprendimento ma, una volta completato, il sistema finale raggiunge prestazioni eccellenti anche in casi "difficili"