Università degli Studi di Perugia

Reti neurali: applicazioni in Fisica ed oltre

M. Alvioli - M. Biasini

XVI ciclo di Dottorato

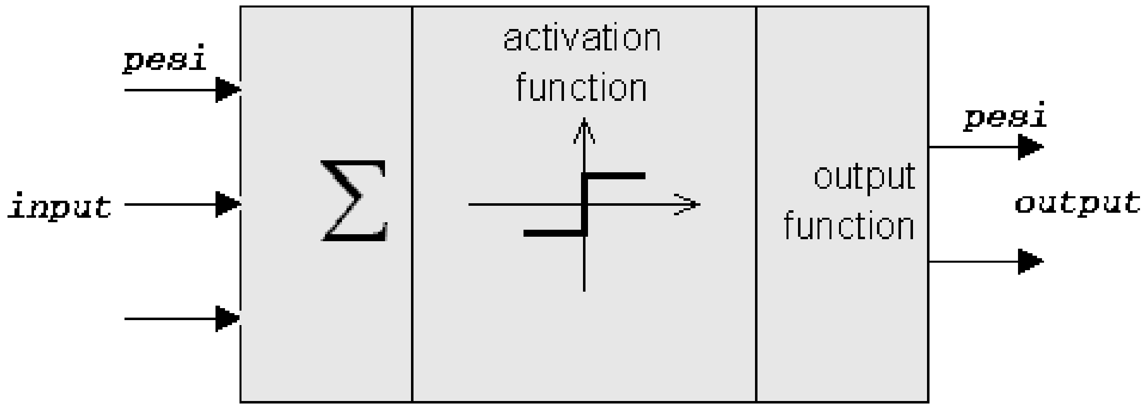
1 Il modello biologico

Struttura

- Le unità costituenti del cervello sono le cellule neurali
- Esse sono in numero enorme, tutte collegate fra loro

Funzionamento

- Le connessioni fra neuroni trasportano impulsi elettrici ai nuclei delle cellule
- Ogni neurone può *attivarsi* in base alla quantità di impulsi ricevuti
- Un neurone attivato è in grado di trasmettere a sua volta un impulso verso tutti gli altri neuroni ai quali è connesso



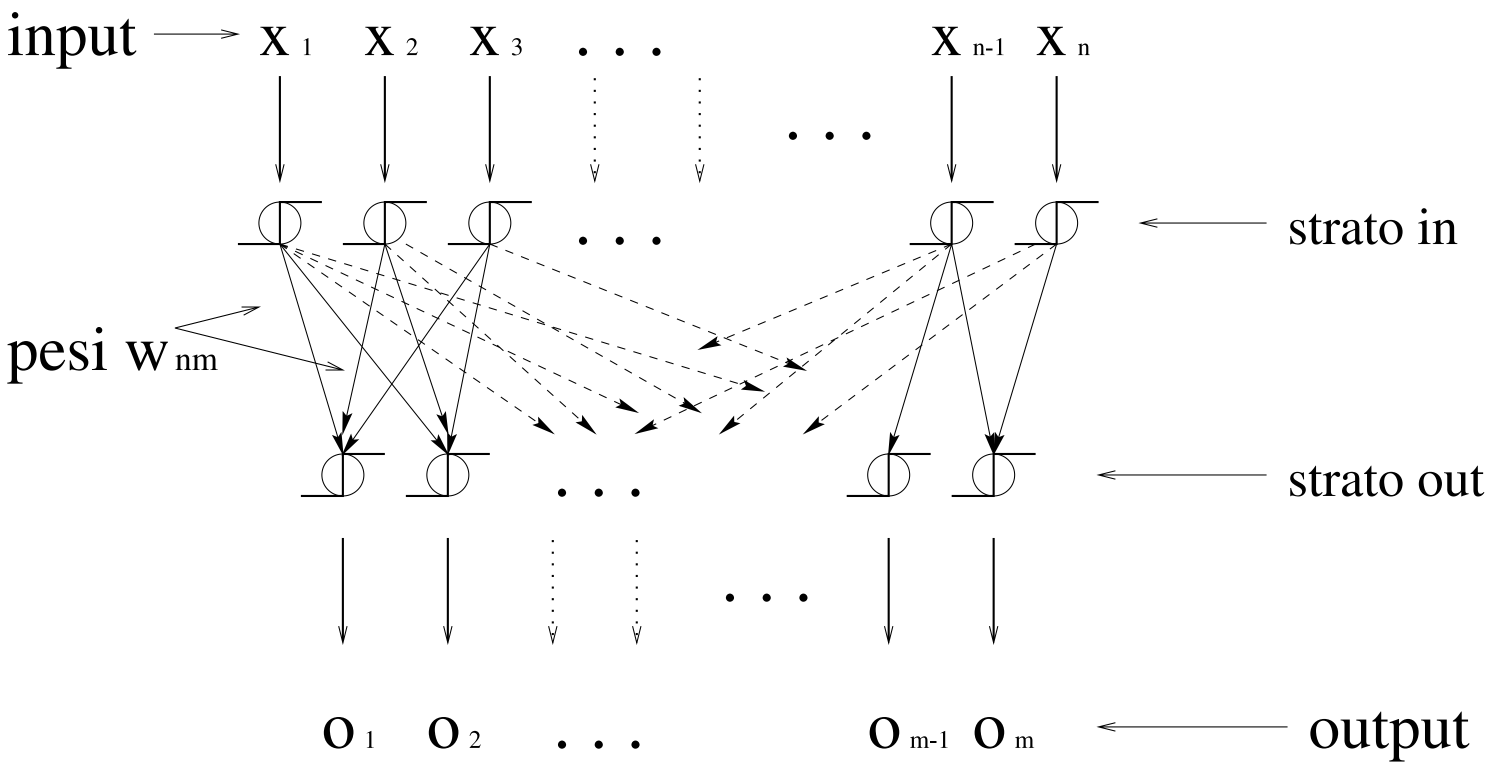
Tale struttura è in grado di:

- Memoria associativa: apprendimento
- Capacità di auto-organizzazione

L'emulazione di una tale struttura è oggetto di studio e l'uso di reti artificiali è risultato utile in molte applicazioni scientifiche.

2 Reti neurali artificiali (ANN)

Struttura di base:



Definiamo per le uscite delle *feature functions*, funzioni degli ingressi $\boldsymbol{x}$:

$$\mathbf{O_i = F_i(x_1, ..., x_n), \;\; i = 1, ..., m}$$

per ogni neurone del primo strato:

$$\mathbf{v_i = g\left(\sum_{j=1}^{n} \omega_{ij} x_j + \theta_j\right) = g\left(\sum_{j=0}^{n} \omega_{ij} x_j\right), \;\; \omega_{i0} = \theta_i}$$

Questa rete ha un comportamento **lineare** ed uscite binarie. Le variabili sono i **pesi**; il numero di ingressi ed uscite; la *funzione di attivazione* g, qui a gradino.

In cosa consiste il processo di apprendimento?

1 Inizializzare **a caso** il valore dei pesi

2 Calcolare la funzione uscita su un set di dati di **allenamento**

3 Calcolare un **errore** in base al risultato che si voleva ottenere

4 Cambiare il valore dei pesi utilizzando opportunamente l'errore ottenuto, facendo entrare un parametro che misuri la **velocità di apprendimento** della rete

5 Ripetere il procedimento per quante **epoche** si vuole, con i set di dati a disposizione, fermandosi in base ad un criterio da stabilire (dati di *train* e di *test*)

6 fase di **generalizzazione**: si congelano i valori dei pesi e la rete è pronta per essere utilizzata.

Strutture più complicate

Reti più flessibili in grado di assolvere a compiti molto generici:

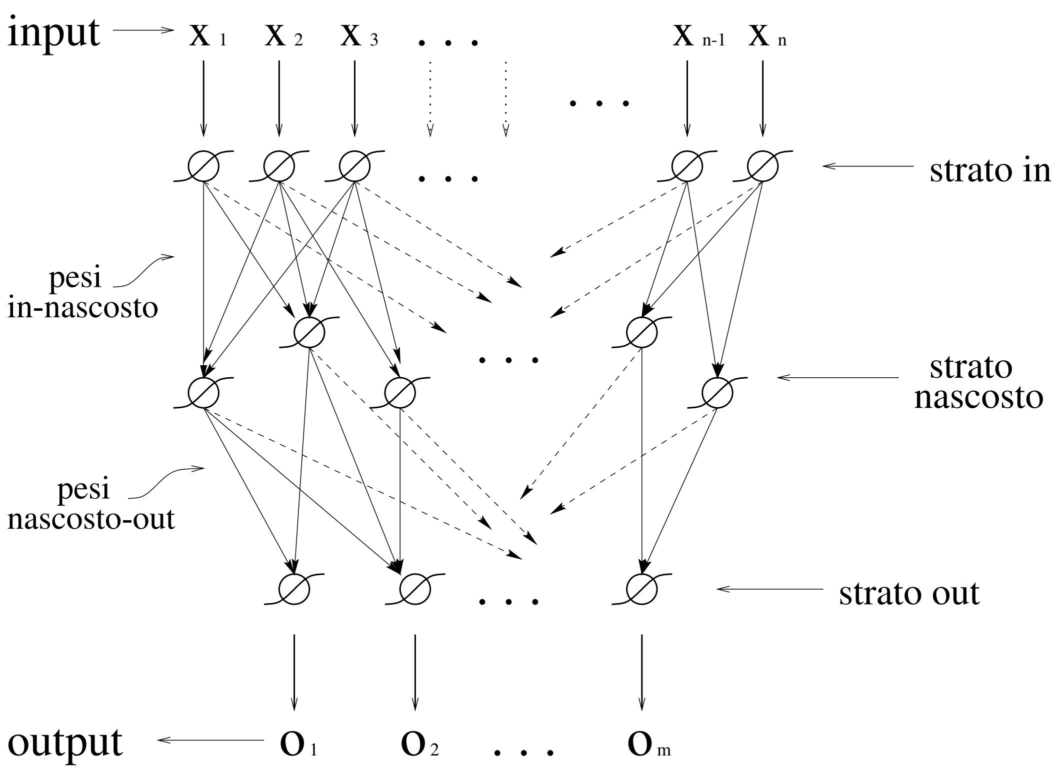
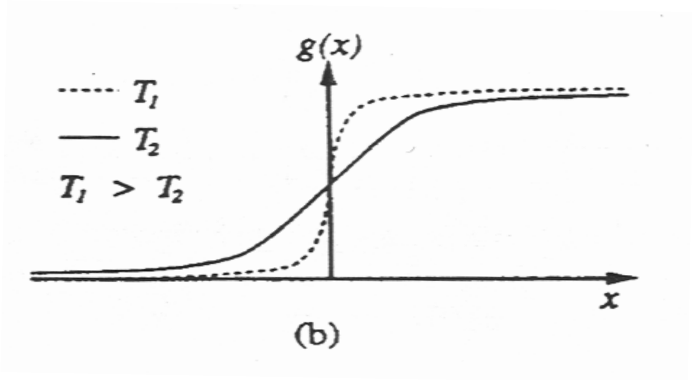


Figura 1: Funzione di attivazione a soglia morbida.



Ora, dato che abbiamo tre strati:

$$\mathbf{O_l} = \mathbf{F}_\omega(\mathbf{x_1}, ..., \mathbf{x_n}) = \mathbf{g}\left(\sum_{\mathbf{k=0}}^{\mathbf{n}} \omega_{\mathbf{lk}}^{(\mathbf{2})} \mathbf{g}\left(\sum_{\mathbf{j=0}}^{\mathbf{n}} \omega_{\mathbf{ij}}^{(\mathbf{1})} \mathbf{x_i}\right)\right)$$

- comportamento **non lineare** della rete in funzione degli ingressi;
- la procedura che si utilizza ora è simile a quella di un *fit* per le x_i con delle funzioni dipendenti da parametri, i pesi della rete; si massimizza cioè la corrispondente distribuzione di probabilità congiunta, o *likelihood*:

$$\mathbf{p}\left(\mathbf{out}[\boldsymbol{y}] \,|\, \mathbf{in}[\boldsymbol{x}]; \boldsymbol{\omega}\right) \propto \prod_{\mathbf{j=1}}^{\mathbf{n}} \mathbf{e}^{(\mathbf{F}_\omega(\mathbf{x_j}) - \mathbf{y_j})^2/2\sigma^2},$$

che porta alla minimizzazione della ***funzione costo*** rispetto ai pesi $\boldsymbol{\omega}$:

$$E(\boldsymbol{\omega}) = \sum_{j=1}^{n} (F_\omega(x_j) - y_j)^2,$$

cioè l'**errore** prima menzionato.

2.1 Tipi di ANN: a cosa servono

Ad distinguere i vari tipi di reti neurali sono l'**algoritmo** tramite il quale avviene l'aggiornamento dei pesi ad ogni epoca di apprendimento ed il tipo di architettura.

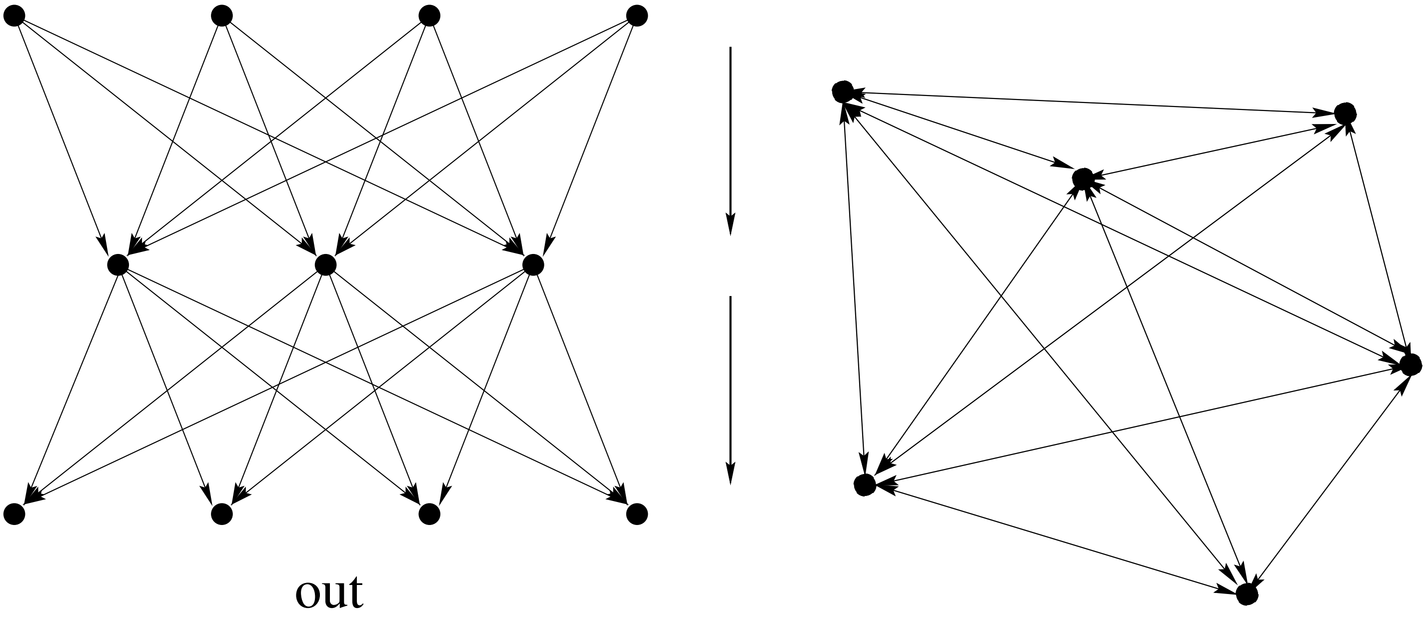


Figura 2: Schema *feed-forward* (sinistra) e *feed-back* (destra).

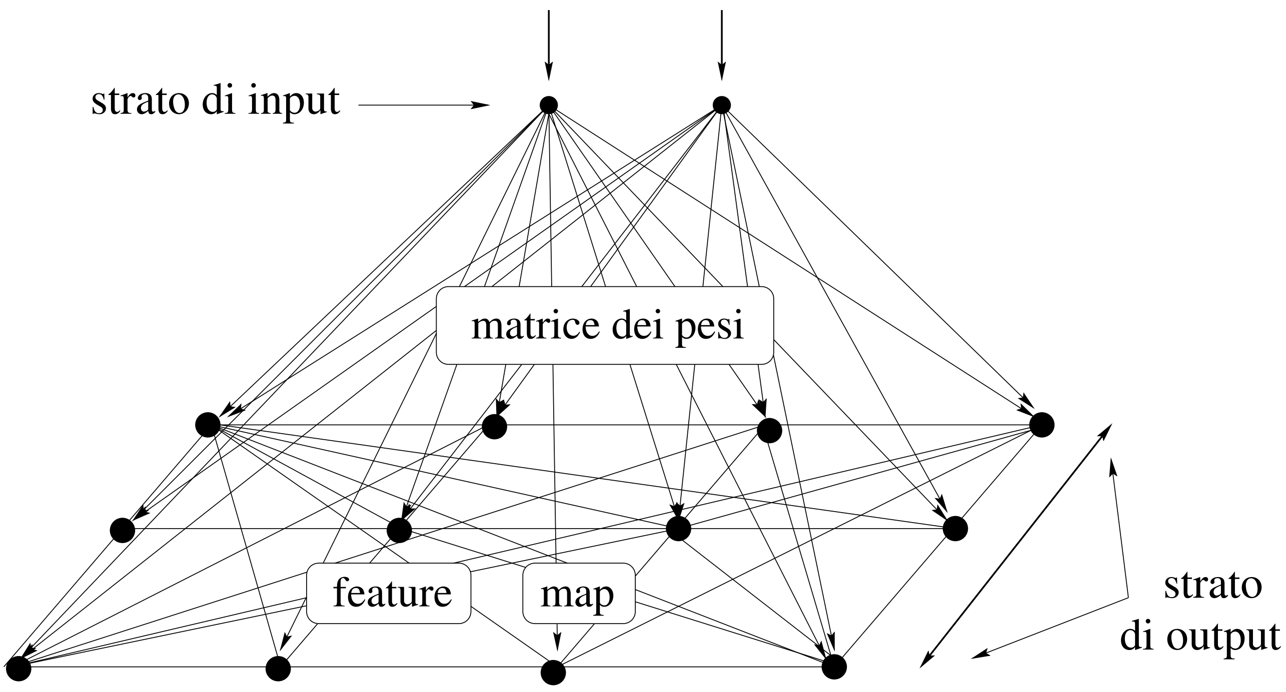


Figura 3: ***Feature map***: i nodi dello strato di input sono collegati a tutti i nodi dello strato della mappa, anch'essi tutti collegati fra loro. In questo modo si possono utlizzare algoritmi *auto-organizzanti*.

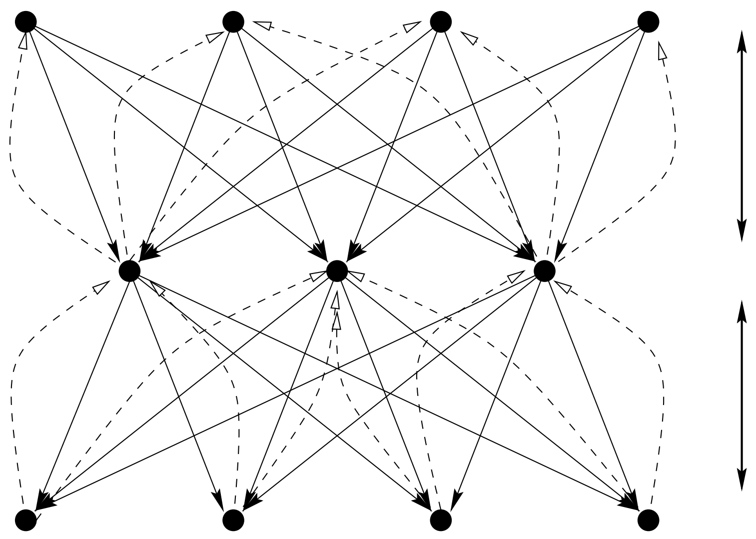


Figura 4: Schema *feed-forward* con ***back-propagation***

Con i vari schemi descritti si possono costruire reti estremamente flessibili che si prestano agli utilizzi i più disparati:

- **Classificazione**: suddivisione in classi con determinate caratteristiche; riconoscimento di immagini;
- **Approssimazione di funzioni**: si possono usare le $F(\boldsymbol{x})$ come funzioni da interpolare. Con ingressi ed uscite binari, si possono emulare funzioni booleane tipo OR, NOT, ecc. e loro combinazioni;
- Problemi di **ottimizzazione**, da ricondurre alla minimizzazione di una ***funzione energia***.
  Ad esempio, in un modello di Ising unidimensionale:

$$\mathbf{E = -\frac{J}{2}\sum_i s_i s_{i+1}}$$

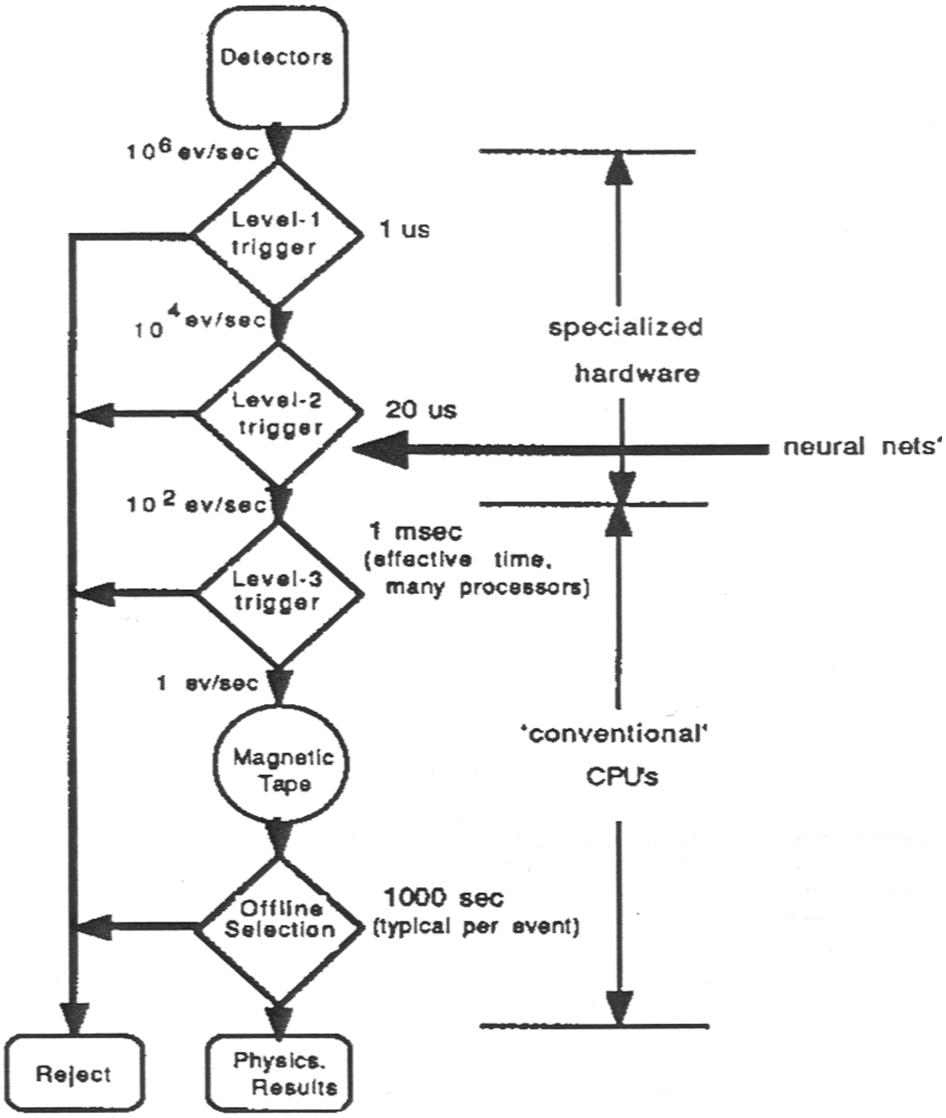
  oppure, più in generale, per uno *spin glass* con interazioni non-locali:

$$\mathbf{E = -\frac{1}{2}\sum_{i\neq j}\sum_j \omega_{ij} s_i s_j}$$

  caratterizzato da stati fondamentali difficilmente identificabili.

2.2 Reti neurali in HEP

- Come ***trigger*** in tempo reale in fase di acquisizione dati:



- come strumento di analisi *off-line* dei dati raccolti.
- Problema dell'apprendimento: l'allenamento si fa su simulazioni tipo *monte-carlo*, che introducono una *dipendenza dal modello fisico* utilizzato.

La natura intrinsecamente parallela di una ANN permette l'utilizzo di architetture parallele di CPU convenzionali.

2.3 JetNet

E' il programma utilizzato per il riconoscimento di jet originatisi da quark b contro il fondo di c, u, d, s.

La rete è di tipo feed-forward, con algoritmo back-propagation e apprendimento di tipo *supervised* con 30 ingressi.

I parametri sono: il numero di **epoche** di allenamento/test, la **rapidità** di apprendimento, il numero di **strati** e di **nodi nascosti**.

Caratteristiche del caso sono l'***efficienza*** con la quale vengono riconosciuti i jet:

$$\varepsilon_{\mathrm{b}} = \frac{\#\ \mathrm{b\ riconosciuti}}{\mathrm{b\ presenti\ nei\ dati}}\,,$$

calcolata in base ad un '*taglio*' sulla ***purezza*** del segnale ottenuto per il b:

$$\mathrm{p} = \frac{\#\ \mathrm{b\ riconosciuti}}{\#\ \mathrm{b\ riconosciuti}\ +\ \mathrm{fondo}}\,;$$

tale purezza è un altro parametro in ingresso: si specifica all'inizio qual è la purezza che si vuole per calcolare l'efficienza.

3 Riconoscimento di sequenze di DNA

Le sequenze genetiche *splice-junction* sono quelle parti di DNA rilevanti nel processo di creazione delle proteine.
La rete deve capire a quale fra le seguenti possibilità corrisponde la sequenza che gli si presenta:

EI - *Exon-Intron*: è l'*inizio* di una sequenza non rilevante nel processo di clonazione delle proteine;

IE - *Intron-Exon*: è la *fine* di una sequenza non rilevante nel processo di clonazione delle proteine;

Nel lavoro originale veniva sviluppato un algoritmo dedicato, (*Knowledge Based ANN*, KBANN) prendendo in considerazione un set di 3200 sequenze di 60 nucleotidi (A, T, G, C) ognuna.
La bontà del metodo veniva valutata confrontando un ***error rate*** con quello ottenuto con una rete inizializzata a caso.
Cerchiamo di riprodurre i risultati visti usando la rete utilizzata per i jet del quark b.
Abbiamo costruito l'*error rate* come la quantità:

$$\mathbf{E_{rate}} = \frac{\#\,\textbf{EI identificati erroneamente}}{\#\,\textbf{EI id. correttamente} + \#\,\textbf{EI id. erron.}}\,;$$

utilizzando come purezza del segnale

$$\mathbf{p = 1 - E_{rate}}\,.$$

facendo attenzione al fatto che in $KBANN$ non si considera l'efficienza della classificazione.

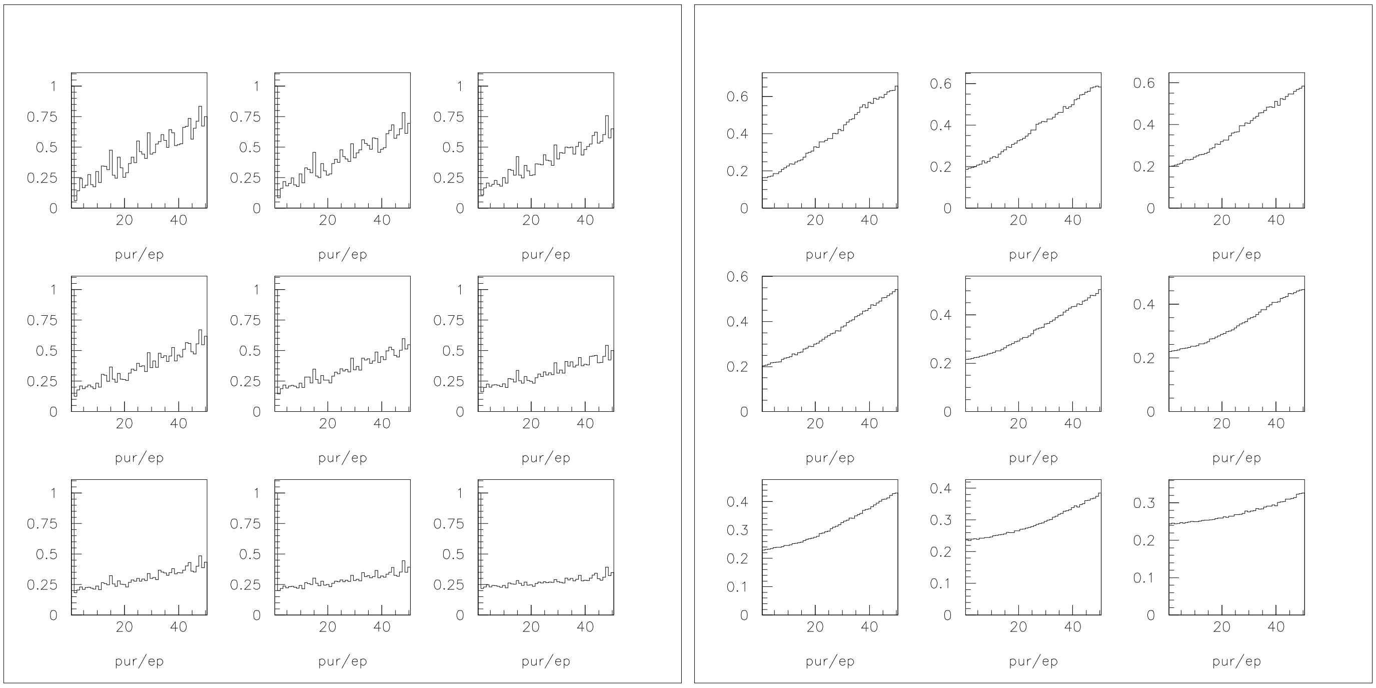


Figura 5: **purezza vs. epoca** ad efficienza crescente; set *train* (sinistra) e set di *test* (destra). Curve relative ad un ciclo di 50 epoche.

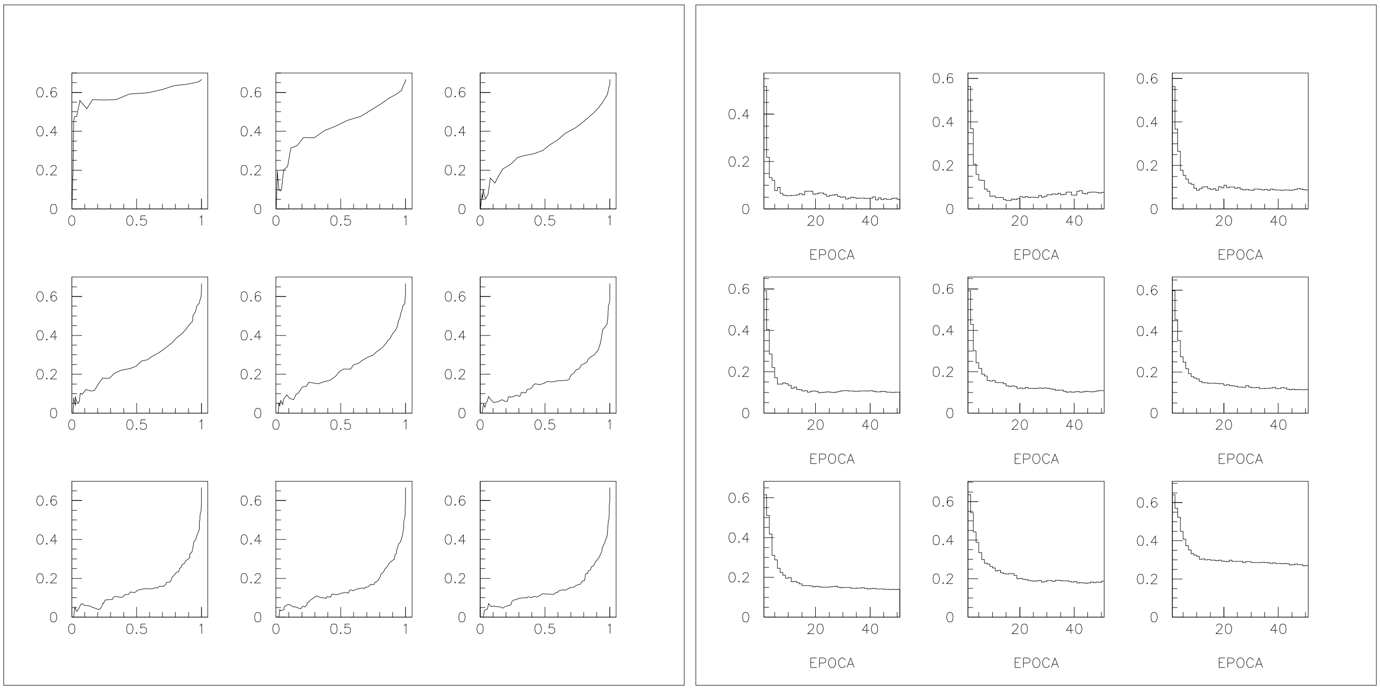


Figura 6: $\mathbf{E_{rate}}/\varepsilon_{\mathbf{EI}}$ (sinistra) per diverse epoche; $\mathbf{E_{rate}}$**/epoca** per ε_{EI} che varia dal 10% al 90% (destra). Curve relative ad un ciclo di 1000 epoche.

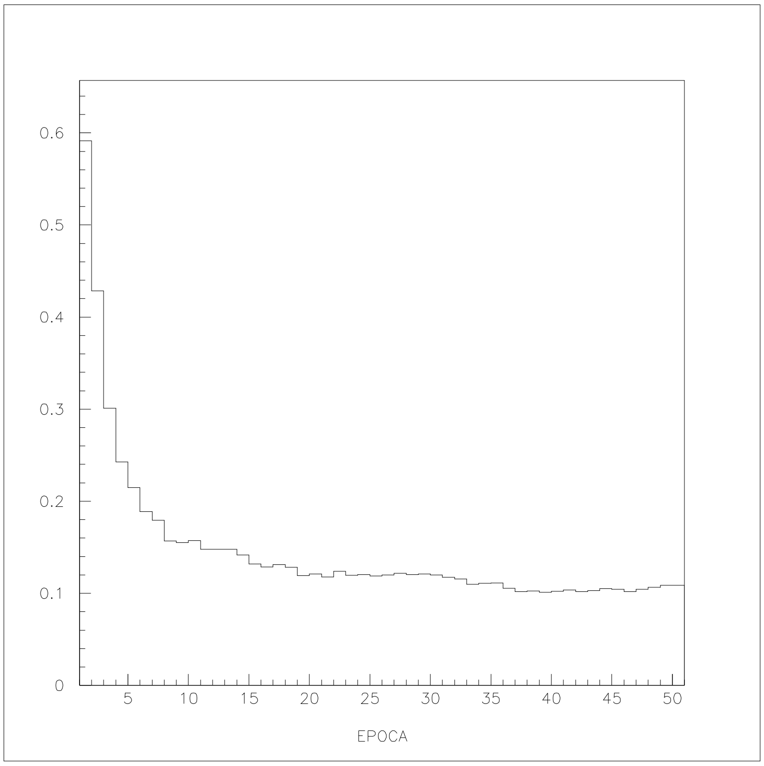


Figura 7: E_{rate}/epoca @ ε_{EI}=50% Curve relative ad un ciclo di 1000 epoche; le epoche sono da leggersi in unità di 20.

Riferimenti bibliografici

[1] M. O. Noordevier, G. G. Towell and J. W. Shavlik, *Training in Knowledge-Based Neural Networks to Recognize Genes in DNA Sequences*, Adv. Neural Information Processing Systems (1991)

[2] C. Peterson and T. Rögnvaldsson *An Introduction to Artificial Neural Networks*, Proc. 1991 CERN Summer School of Computing, CERN Yellow Report **92-02**, 113-170 (1992)

[3] L. Lnnblad, C. Peterson, and T. Rgnvaldsson *JETNET 3.0 - A Versatile Artificial Neural Network Package* Computer Physics Communications **81**, 185-220 (1994)

[4] M. Pohl, *Lezioni di statistica*, CERN, `http://pohl.home.cern.ch/pohl/statistique.ps.gz`

[5] G. Massinelli, *Tesi di Laurea*, Università degli Studi di Perugia, A. A. 1999/2000

[6] J. Frölich, *Fachhochschule Regensburg*, *Department of Computer Science*, Tesi di Diploma, `http://rfhs8012.fh-regensburg.de/~saj39122/jfroehl/diplom/e-index.html`

[7] DELVE - *Data for Evaluating Learning in Valid Experiments*, Progetto software, collezioni di dati, algoritmi di apprendimento, `http://www.cs.toronto.edu/~delve/`

[8] C. E. Rasmussen, *The linear model*, *http://www.cs.toronto.edu/~delve/methods/lin-1/home.html* (1996)

[9] C. E. Rasmussen, *Multilayer Perceptron Ensembles trained with Early Stopping*, `http://www.cs.toronto.edu/~ delve/methods/mlp-ese-1/home.html` (1996)

[10] UCI - *Machine Learning Rerpository*, University of California, Irvine, da cui il set di dati sulle sequenze di DNA *Splice*, `http://www.ics.uci.edu/~mlearn/MLRepository.html`

[11] B. Denby, *The use of Neural Networks in High Energy Physics*, Fermilab-pub-**92/215-E**, (1992)