Neural Network driven Fuzzy Reasoning

Andrea Splendiani mat. 615434

27th June 2004

Abstract

La logica fuzzy consente di trattare informazioni incerte o vaghe e permette un ragionamento approssimato di alto livello attraverso regole di inferenza. Le reti neurali presentano capacita' di astrazione e di apprendimento da dati di esempio. Una integrazione di tali tecnologie porta ad una loro sinergia, particolarmentre efficace nel realizzare un ragionamento approssimato di tipo umano.

Verranno qui presentate le basi della logica fuzzy, orientandosi sui suoi sviluppi piu' utilizzati in sistemi esperti e sistemi di controllo, quindi verrano presentate le reti neurali e le loro caratteristiche essenziali. Infine si mostrera' come le caratteristiche di questi due sistemi possano essere combinate, in particolare mostrando come possano essere realizzati sistemi di inferenza fuzzy attraverso reti neurali.

Contents

1 Introduzione

Nella realizzazione di sistemi esperti e di sistemi di controllo si riscontra spesso la necessita' di riprodurre o formalizzare ragionamenti di tipo vago o approssimato, operanti su dati non sempre certi o precisi, tipici di situazoni reali e dell'esperienza umana.

Considerato un ragionamento di tipo umano, esso operera' su concetti in parte ben definiti, come "piu' alto di", in parte difficilmente quantizzabili o soggettivi come "molti", "pesante". Inoltre tipici ragionamenti saranno del tipo "Se la pressione supera il valore x allora esegui l'azione y" oppure "Se il paziente e' spesso stanco allora forse sta male".

Mentre la logica tradizionale, ad esempio una teoria formale del primo ordine, puo' essere efficacemente utilizzata per modellare la prima delle due proposizioni, e quindi permettere una realizzazione algoritmica di un sistema di controllo, sono evidenti le limitazioni che si incontrano nel tentativo di formalizzare la seconda delle due frasi. In primo luogo il concetto "stanco" non e' precisamente definito, e' soggettivo, inoltre "spesso" e "forse" sono quantificatori imprecisi, non direttamente rappresentabili in una logica "tradizonale".

Naturalmente il poter formalizzare una affermazione con una teoria formale consente di ottenere un ragionamento preciso e rigoroso, soggetto ad una verifica formale, tuttavia e' spesso accettabile sacrificare tale rigore, pur di poter trattare piu' propriamente informazioni imprecise o incerte.

Un'area di ricerca denominata SOFT COMPUTING accomuna dievrsi metodi di ragionamento imprecisi, tra cui LOGICA FUZZY, RETI NEURALI, ALGORITMI GENETICI, PROGRAMMAZIONE EVOLUZIONISTICA.

La logica Fuzzy si basa sull'estensione del concetto di insieme: mentre comunemente l'appartenenza ad un insieme e' o vera o falsa, nella logica fuzzy si considera un'apparteneza parziale o sfmuata. Da qui partono numerosi sviluppi che portano alla rappresentazione di un ragionamento approssimato, in grado di rappresentare propriamente proposizioni come "se la pressione x e' molto alta, allora occore eseguiremolto l'operazione y", oppure dedurre qunato sia vera la proposizione "un uomo e' giovane" data la "verita'" della proposizione "l'uomo ha tra i 30 e i 40 anni".

Le reti neoronali sono strutture computazionali modellate ispirandosi alla struttura del sistema cerebrale umano. Esse sono caratterizzate da numerose unita' di elaborazione elementari fortemente interconnesse fra loro. Dalla loro interconnessione si sviluppa un comportamento in grado di effettuare elaborazioni complesse. Le reti neurali apprendono la loro funzione attraverso una fase di apprendimento, esse non vengono cioe' programmate per eseguire una detrminata funzione algoritmicamente, ma vengono addestrate attraverso es-

empi. Esse si mostrano quindi particolarmente indicate per apprendere concetti o relazioni, ad esempio il concetto di "pressione alta", data una certa serie di esempi, o la correlazione tra stanchezza e malattia.

Sebbene Logica Fuzzy e Reti Neurali abbiano molte caratteristiche in comune, i loro punti di forza sono diversi. In particolare la logica fuzzy consente un ragionamento di alto livello, attraverso regole di inferenza, mentre le reti neurali rappresentano un sistema di basso livello (o sub-simbolico), efficace nel'astrarre relazioni o concetti da dati sensoriali. I due sistemi sono quindi suscettibili di una integrazione che ne sfrutti le sinergie, e di fatto numerose applicazioni, specie nel campo del controllo, hanno utilizzato un loro accoppiamento.

Gli algoritmi genetici, di cui non si parlera' oltre, sono algoritmi ispirati all'evoluzione naturale. Una popolazione (una collezione) di soluzioni ad un problema (o una popolazione di algoritmi nel caso di Programmazione Genetica), viene confrontata con il problema da risolvere, e ad ogni elemento viene assegnato un valore di fitness, una misura del grado di buona capicita' di risolvere il problema. Quinidi vengono selezionate le soluzioni migliori, eliminate le peggiori, e ne vengono reintrodotte di nuove, originate combinando le soluzioni con i valori di fitness piu' elevati. La combinazione delle soluzioni avviene attraverso una combinazione delle rispettive codifiche, nonche' attraverso alcune mutazioni casuali di queste ultime. Anche gli algoritmi genetici mostrano capacita' di apprendimento, ma anche di ottimizzazione.

Di seguito si presentera' una breve introduzione alla Logica Fuzzy e alle Reti Neurali, intesi come sistemi atti a rappresentare un ragionamnto approssimato. Ma e' bene ricordare che tali sistemi di ragionamento, pur permettendo dui trattare in modo piu' coerente situazoni "reali", rinunciano al rigore e alla precisione delle logiche formali. Queste ultime a fronte di forti approssimazioni nella definizione dei concetti e delle relazioni tra essi, garantiscono un ragionamento rigoroso. I sistemi di ragionamento approssimato (che sono essenzialmente numerici e non simbolici) evitano tali approssimazioni semplicemente accettandole e rendendo "approssimato" l'intero sistema.

2 Logica Fuzzy

La logica Fuzzy si basa sull'estensione del concetto di insieme. Mentre nella logica tradizonale un elemento appartiene o non aoppartiene ad un insieme, nella logica fuzzy l'appartenenza puo' essere parziale. In realta' insieme a questa estensione vi e' anche un impoverimento, dovuto all'abbandono di una approssimazione molto forte (appartenenza o non appartenenza) in gradi di garantire forti risultati.

Nell'ottica dellla costruzione ragionamento approssimato conviene tuttavia presentare tale logica come una graduale estensione di concetti tradizionali.

2.1 Estendendo il concetto di insieme

Un insieme tradizionale o *nitido* puo' essere completamente specificato dalla sua funzione caratteristica. Per un insieme **A**, sottoinsieme dell'insieme universo del discorso **U**, questa e':

$\mu_A = \begin{cases} 1 \Leftrightarrow x \in A \\ 0 \Leftrightarrow x \notin A \end{cases}$

Nella logica fuzzy analogamente un insieme puo' essere rappresentato da una FUNZIONE DI APPARTENENZA, simile alla funzione descrittiva, ma avente valori in [0,1] anziche' solo in {0,1}. Quindi un insieme fuzzy puo' essere rappresentato come un insieme (finito o infinito) di coppie ordinate:

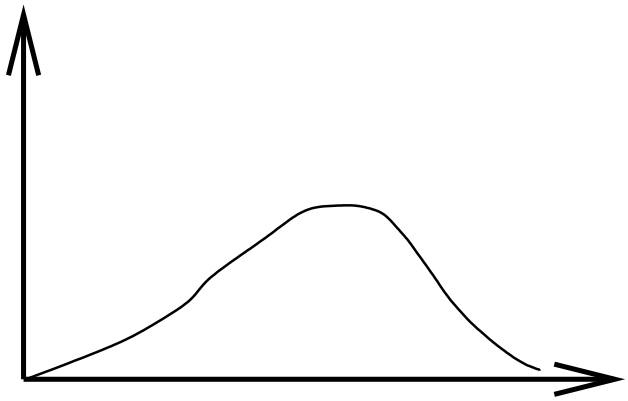
$A = \{(x, \mu_A(x)) \mid x \in U\}$

Un insieme fuzzy e' spesso espresso con una notazione piu' compatta, ad esempio:

$A = 0.5/1 + 0.7/2 + 0.5/3$ se U e' discreto.

$A = \int_U \mu_A(x)/x$ se U e' continuo.

Un altro modo di rappresentare un insieme fuzzy e' tramite la sua funzione di appartenenza:



Per un insieme fuzzy si possono poi introdurre le seguenti definizioni:

SUPPORTO: $Supp(A) = \{x \in U \mid \mu_A(x) \geq 0\}$

ALTEZZA: $Height(A) = \sup_{x \in U} \mu_A(x)$

Un insieme A si dice NORMALIZZATO se $Height(A) = 1$

ALFA TAGLIO: $A_\alpha = \{x \in U \mid \mu_A(x) \geq \alpha\}$

ALFA TAGLIO STRETTO: $A_\alpha = \{x \in U \mid \mu_A(x) > \alpha\}$

Da notare che gli alfa tagli sono insiemi "nitidi". Grazie agli Alfa-tagli e' poi possibile esprimire un insieme fuzzy in funzione

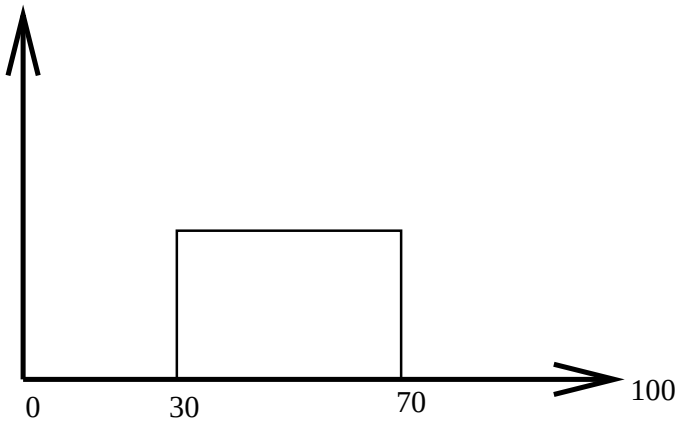
di piu' insiemi nitidi, si ha infatti:

$\mu_A(x) = \sup_{\alpha \in (0,1]} \min(\alpha, \mu_{A_\alpha}(x)\ \forall x \in U$

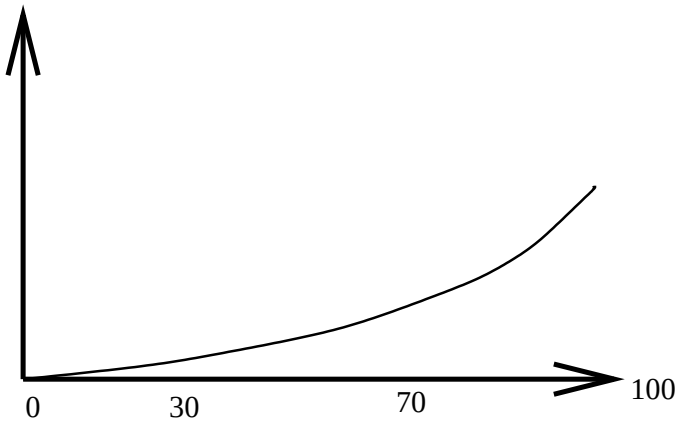
dove si e' considerato $\mu_{A_\alpha}(x) = \begin{cases} 1 \Leftrightarrow x \in A_\alpha \\ 0 \Leftrightarrow x \notin A_\alpha \end{cases}$

Si potrebbero poi introdurre cardinalita' (fuzzy e non), convessita' e altre proprieta' utili per ulteriori sviluppi della teoria. Questi aspetti non verranno qui presentati. Invece, poiche' il concetto di insieme fuzzy e' il fulcro della logica fuzzy, e' bene soffermarsi ancora su cio' che esso significa.

Consideriamo U={anni 1...100} e consideriamo l'insieme {tra i 30 e i 70 anni}, esso puo' essere cosi' rappresentato:



notiamo che in questo caso l'insieme fuzzy coincide con l'insieme nitido. Consideriamo ora l'insieme: {anni in cui un uomo e' considerato vecchio}, esso puo' essere cosi' rappresentato:



e' evidente che tale insieme e' intrinsecamente fuzzy, e qualunque tentativo di darne una rappresentazione come insieme nitido porterebbe ad una eccessiva approssimazione.

2.1.1 Ancora sull'estensione del concetto di insieme

Si possono definire anche insiemi fuzzy di ordine superiore, ad esempio gli insiemi fuzzy di tipo 2, ovvero un insieme fuzzy per cui il grado di appartenenza di un elemento ad un insieme e' a sua volta espresso da un insieme fuzzy.

Ad esempio si potrebbe avere l'insieme *Funzionale*, e un elemento potrebbe appartenervi con grado *poco*, *molto*, *etc.* Formalmente per questi insiemi si ha:

$\mu_A : U \rightarrow [0,1]^{[0,1]}$

cosi' definita:

$\mu_A(x) = \int f(u)/u$ con $u \in [0,1]$ e $f : [0,1] \rightarrow [0,1]$

Un altro tipo di estensione consiste nel definire insiemi fuzzy i cui elementi siano a loro volta insiemi fuzzy, ad esempio:

BuoneCaratteristicheDiUnaAuto ={*TenutaDiStrada*,*BbassiConsumi*, *etc.*}

2.1.2 Su come gli insiemi fuzzy rappresentino l'incertezza

Gli insiemi fuzzy costituiscono una cornice ideale per rappresentare informazioni incerte, ma e' bene stabilire cosa si intende per incertezza. Consideriamo le frasi:

A :<*e' probabile che* venga estratto il numero 15>

B: <*e' stata una buona gara*>

C: <*il corridore e' buono ?*>

consideriamo A: non e' noto se ilnumero 15 verra' estratto o meno, ma chiaramente sara' o non sara' estratto. Questo tipo di incertezza e' legato al verificarsi o meno di un evento ben definito, e porta a ragionamenti di tipo probabilistico.

C: anche se buono e' un termine fuzzy, e' da notare che qui l'informazione non e' nota: si e' di fronte a "non conoscenza", non incertezza.

B: una buona gara e' un termine fuzzy, ovvero icerto, non ben definito: questa e' la vaghezza che consente di trattare la logica fuzzy, quella associata alla *impossibilita' di definire esattamente le caratteristiche di un qualcosa.*

Tuttavia al di la' della definizione di insieme fuzzy si associa spesso alla Logica Fuzzy la TEORIA DELLA POSSIBILTA', che consente di trattare anche non solo informazioni vaghe, ma imprecise in un senso piu' lato.

2.1.3 Sul concetto di Fuzzyness e le sue implicazioni

C'e' chi e' arrivato acorrelare la relazione tra Logica Tradizionale e Logica Fuzzy alla relazione tra Cultura Occidentale, permeata dalla bivalnte Logica Aristotelica (basata sul "E' o non e"') e cultura orientale, simboleggiata dal Tao, che ammette che in ogni cosa vi sia sempre una parte del suo opposto. In particolare KOSKO, nel suo noto testo IL FUZZY-PENSIERO si muove su queste linee, arrivando a presentare la Logica Fuzzy come una rivoluzione nella cultura occidentale.

Vi e' una correlzione tra la cultura occidentale e logica tradizionale e tra la logica fuzzy e la cultura occidentale, tuttavia e' anche da notare che la logica fuzzy non solo estende la logica tradizionale, ma anche ne toglie il rigore, e come si vedra' la maggior parte degli sviluppi che si muovono da questa abbandonano le caratteristiche delle logiche formali, per diventare in definitiva sistemi numerici.

La logica fuzzy si propone quindi come un formalismo piu' idoneo per rappresentare ragionamenti di tipo approssimato, in molte applicazioni e' preferibile alla logica tradizionle, ma e' su un piano da essa distinta, e non direttamente paragonabile.

2.2 Estendendo le operazioni sugli insiemi

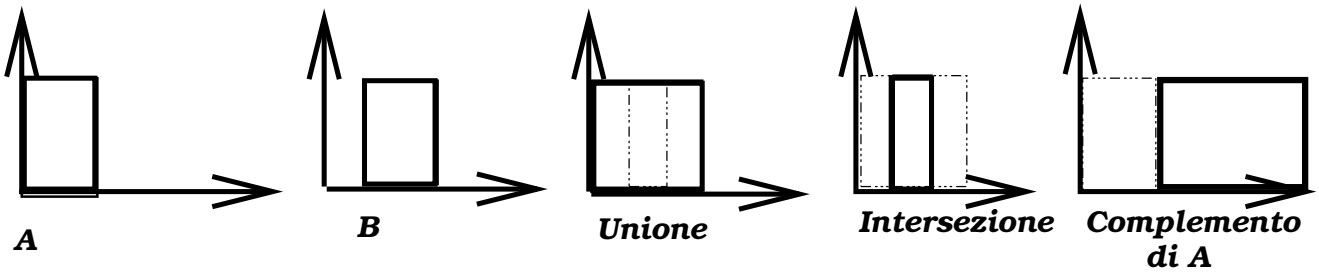
Sugli insiemi nitidi sono notoriamente definite alcune operazioni, tra cui unione, intersezione e complemento. Dati due insiemi **A** e **B**, sottoinsiemi di un insieme universo **U**, esse sono cosi' definitie:

Unione: $A \cup B = \{x \mid x \in A \vee x \in B\}$, $x \in U$

Intersezione: $A \cap B = \{x \mid x \in A \wedge x \in B\}$, $x \in U$

Complemento: $\sim A = \{x \mid x \notin A\}$, $x \in U$

E' possibile poi "visualizzare" tali operazioni tramite le funzioni caratteristiche dei vari insiemi coinvolti.



Queste operazioni possono essere estese in piu' modi nel caso in cui si operi su insiemi fuzzy. Forse l'approccio piu' generale e' stabilire quali proprieta' debbano essere soddisfatte da funzioni di *"intersezione"*, *"unione"*, *"complemento"*. Introduciamo quindi T-NORME e T-CONOMRME, come "generalizzazione" di intersezione ed unione.

una T-NORMA e' una funzione $t : [0,1]X[0,1] \rightarrow [0,1]$ che soddisfa le seguenti proprieta':

- Condizioni al confine: $t(0,0) = 0,\ t(\mu_A(x),1) = t(1,\mu_A(x)) = 1$
- Commutativita': $t(\mu_A(x),\mu_{B(x)}) = t(\mu_B(x),\mu_A(x))$
- Monotonia: se $\mu_A(x) \leq \mu_C(x)$ e $\mu_B(x) \leq \mu_D(x)$ allora $t(\mu_A(x),\mu_B(x)) \leq t(\mu_C(x),\mu_D(x))$
- Associativita': $t(\mu_A(x),t(\mu_B(x),\mu_C(x)) = t(t(\mu_A(x),\mu_B(x)),\mu_C(x))$

Analogamente una T-CONORMA, generalizzazione del "concetto" di unione, puo' essere caratterizzata come una funzione $s : [0,1]X[0,1] \rightarrow [0,1]]$ che soddisfa le seguenti proprieta':

- Condizioni al confine : $s(1,1) = 1,\ s(\mu_A(x),0) = s(0,\mu_A(x)) = \mu_A(x)$
- Commutativita' : $s(\mu_A(x),\mu_{B(x)}) = s(\mu_B(x),\mu_A(x))$
- Monotonia : se $\mu_A(x) \leq \mu_C(x)$ e $\mu_B(x) \leq \mu_D(x)$ allora $s(\mu_A(x),\mu_B(x)) \leq s(\mu_C(x),\mu_D(x))$
- Associativita' : $s(\mu_A(x),s(\mu_B(x),\mu_C(x)) = s(s(\mu_A(x),\mu_B(x)),\mu_C(x))$

alcuni esempi di t-norme e t-conorme utilizzate come estensione dei concetti di intersezione e unione sono i seguenti (si considerano due insiemi **A** e **B**, sottoinsiemi di un umiverso del discorso **U**):

Unione e intersezione

INTERSEZIONE:

t-norma cosi' definita $\mu_{A\cap B}(x) = \min(\mu_A(x),\mu_B(x)),\ \forall x \in U$

UNIONE: t-conorma cosi' definita

$\mu_{A\cup B}(x) = \max(\mu_A(x),\mu_B(x)),\ \forall x \in U$

Somma e prodotto

PRODOTTO : t-norma cosi' definita

$\mu_{A\bullet B}(x) = \mu_A(x) \bullet \mu_B(x),\ \forall x \in U$

SOMMA : t-conorma cosi' definita

$\mu_{A+B}(x) = \mu_A(x) + \mu_B(x) - \mu_A(x) \bullet \mu_B(x),\ \forall x \in U$

Unione ed intersezione vincolate

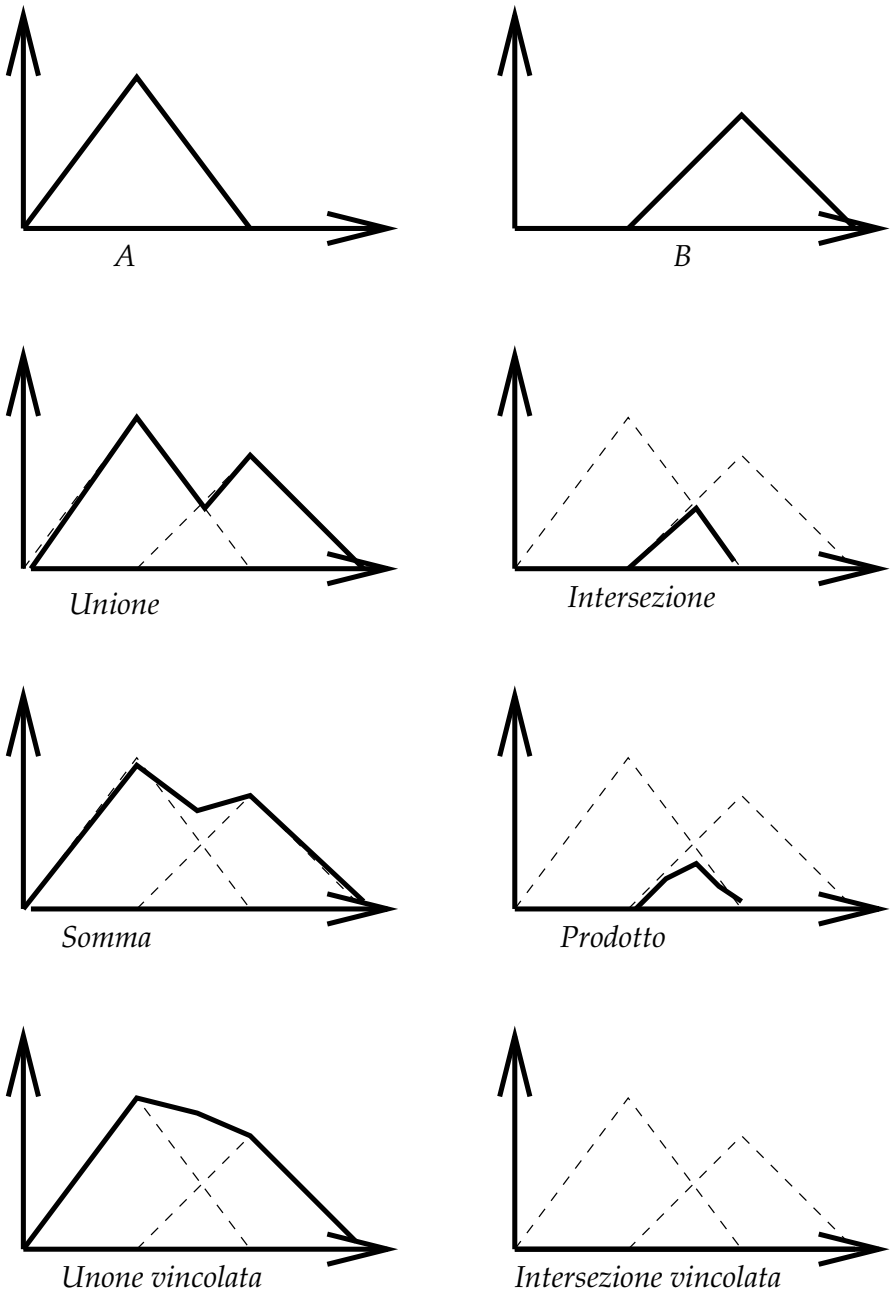
INTERSEZIONE: t-norma cosi' definita

$\mu_{A\wedge B}(x) = \max(0, \mu_A(x) + \mu_B(x) - 1)$, $\forall x \in U$

UNIONE: t-conorma cosi' definita

$\mu_{A\vee B}(x) = \min(1, \mu_A(x) + \mu_B(x))$, $\forall x \in U$

Si noti che le tre classi di operazioni sopra introdotte coincidono con le usuali operazioni insiemistiche nel caso gli insiemi coinvolti siano "nitidi". (Ma nel caso di insiemi fuzzy esse hanno, in generale, diverse proprieta'). Ecco come tali operazioni possono essere visualizzate tramite le funzioni di appartenenza dei vari insiemi coinvolti:



Queste operazioni possono poi essere viste come *"operazioni di aggregazione"*. Per mostrare esaurientemente il quadro di tali "operazioni di aggregazione" introduciamo due nuove operazioni:

PRODOTTO DRASTICO : e' una t-norma cosi' definita:

$$a \odot b = \begin{cases} a \Leftrightarrow b = 1 \\ b \Leftrightarrow a = 1 \\ 0 \Leftrightarrow a, b < 1 \end{cases}$$ [1]

SOMMA DRASTICA : e' una t-conorma cosi' definita:

$$a \uplus b = \begin{cases} a \Leftrightarrow b = 0 \\ b \Leftrightarrow a = 0 \\ 1 \Leftrightarrow a, b > 0 \end{cases}$$ [2]

Si puo' allora dimostrare che tutte le t-norme sono "comprese" tra il prodotto drastico e l'intersezione prima definita, nel seguente senso:

$t_{dp}(a,b) = t_{minima(a,b)} \leq t_{generica}(a,b) \leq t_{massima}(a,b) = \min(a,b)$

Analogamente si pou' dimostare che tutte le t-conorme sono "comprese" tra l'unione prima definita, e la somma drastica, ovvero:

$\max(a,b) = s_{minima}(a,b) \leq s_{generica}(a,b) \leq s_{massima}(a,b) = s_{ds}(a,b)$[3]

Si possono anche introdurre t-norme e t-conorme "parametriche", che in base al valore del parametro si "muovano" tra gli estremi prima definiti. Introduciamo come esempio l'INTERSEZIONE e l'UNIONE di YAGER.

L'INTERSEZIONE DI YAGER e' una t-norma cosi' definita:

$t_w(a,b) = 1 - \left[1, \left((1-a)^w, (1-b)^w\right)^{\frac{1}{w}}\right]$

P er $w \to 0$ essa tende al prodotto drastico, per $w \to \infty$al'intersezione standard. w puo' essere interpretato come grado della forza dell'intersezione prodotta.

L'UNIONE DI YAGER e' una t-conorma cosi' definita:

$s_w(a,b) = \min\left[1, (a^w + b^w)^{\frac{1}{w}}\right]$

Per $w \to \infty$ essa tende all'unione prima definita, per $\to 0$ alla somma drastica.[4]

L'insieme delle operazioni di aggregazione puo' poi essere completato da una classe di operatori di media, tali da essere compresi tra l'intersezione e l'unione standard. Un tipico esempio e' la MEDIA GENERALIZZATAche, generalizzata ad n insiemi, assume la seguente forma:

$h_\alpha(a_1, a_2, \ldots, a_n) = \left(\frac{(a_1)^\alpha + (a_2)^\alpha + \ldots (a_n)^\alpha}{n}\right)^{\frac{1}{\alpha}}$

Il quadro delle operazioni di aggregazione puo' in definitiva essere cosi' sintetizzato:

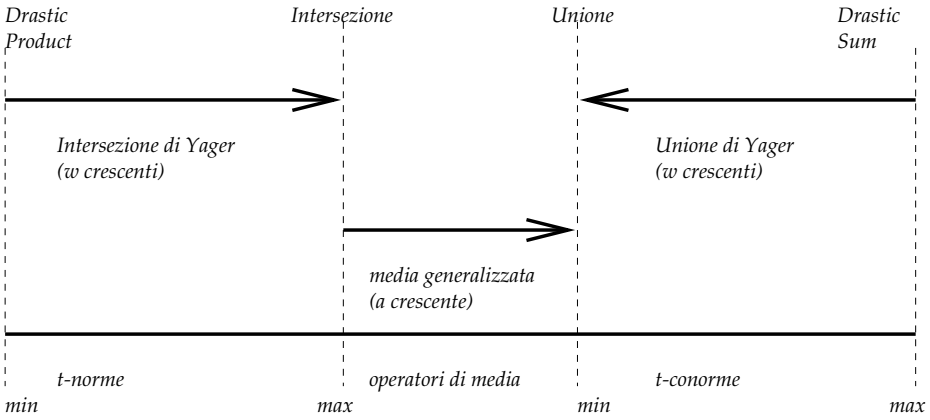
---

[1] con a e b si intende una notazione piu' compatta per $\mu_A(x)$ e $\mu_B(x)$, rispettivamente

[2] Per esigenze tipografiche non sono stati utilizzati i simboli che vengono utilizzati in letteraturz per tali operazioni

[3] con $t_{dp}(a,b)$ e $s_{ds}(a,b)$ si intendono rispettivamente lil prodotto drastico e la somma drastica. $\min(a,b)$ e $\max(a,b)$ corrispondono all'intersezione e all'unione prima definite.

[4] Da notare che per w = 1 le due operazioni coincidono con uninoe e intersezione vincolate.



Si puo' ora definire il COMPLEMENTO, ovvero una funzione $c : [0,1] \rightarrow [0,1]$ che soddisfa le seguenti proprieta':

- Condizioni di confine: $c(0) = 1$,$c(1) = 0$
- Proprieta' di monotonia: se $\mu_A(x_1) \leq \mu_A(x_2)\ \forall x_1, x_2 \in U$, allora $c(\mu_A(x_1)) \geq c(\mu_A(x_2))$
- Continuita' : $c(\bullet)$ e' continua
- Involuzione : n $c(c(\mu_A(x))) = \mu_A(x)$[5]

Il complemento standard, per un generico insieme A in U e' cosi' definito:
$\mu_{\sim A}(x) = 1 - \mu_A(x),\ \forall x \in U$[6]

Si e' detto in precedenza che le varie estensioni mostrate delle operazioni sugli insiemi sono compatibili con le operazioni nitide in caso di insiemi nitidi, tuttavia le possibili generalizzazioni differiscono per le proprieta' presentate. Consideriamo quindi unione, intersezione e complemento standard. Per queste operazioni valgono le proprieta':

- Commutativa : $A \cup B = B \cup A,\ A \cap B = B \cap A$
- Associativa : $(A \cup B) \cup C = A \cup (B \cup C),\ (A \cap B) \cap C = A \cap (B \cap C)$
- Di assorbimento : $A \cup (A \cap B) = A,\ A \cap (A \cup B) = A$[7]

[5] a volte si definisce come negazione forte quella presentata, e si richiede alla negazione la meno restrittiva proprieta' di involuzione: $c(c(\mu_A(x))) \leq \mu_A(x)$

[6] $\sim A$ e' piu' comunemente scritto come A segnato

[7] si noti che il soddisfare queste tre proprieta' consente di strutturare i sottoinsiemi di U come un reticolo, rispetto alle operazioni standard

- Idempotenza : $A \cup A = A$, $A \cap A = A$
- Distribuitivita' : $A \cap (B \cup C) = (A \cap B) \cup (B \cap C)$, $A \cup (B \cap C) = (A \cup B) \cap (A \cup C)$
- Leggi di de Morgan : $\sim (A \cup B) = \sim A \cap \sim B$, $\sim (A \cap B) = \sim A \cup \sim B$
- Legge dello zero : $A \cup U = U$, $A \cap U = A$
- Legge di identita' : $A \cup \emptyset = A$, $A \cap U = A$
- Legge di doppia negazione : $\sim (\sim A)) = A$

Si noti che non valgono le leggi de terzo escluso e di non contraddizione, fondamentali nella teoria degli insiemi nitidi. Tuttavia una diversa scelta delle operazioni puo' far mantenere tali leggi anche nella teoria fuzzy, a discapito di altre. Ad esempio scegliendo il complemento standard, unione e intersezione vincolate, si puo' facilmente dimostrare che le leggi del terzo escluso e di non contraddizione sono valide, tuttavia si sacrificano idempotenza e disribuitivita'.

Altre operazioni spesso utilizzate nella teoria degli insiemi fuzzy sono le seguenti:

PRODOTTO CARTESIANO:

$\mu_{A_1 \times A_2 \times \ldots \times A_n}(x_1, x_2, \ldots, x_n) = \min [\mu_{A_1}(x), \mu_{A_2}(x), \ldots, \mu_{A_n}(x)]$, $x_1 \in U_1, x_2 \in U_2, \ldots, x_n \in U_n$

PROIEZIONE

dato $A = A_1 \times A_2 \times \ldots \times A_n$e $P = A_i$, si definisce $\mu_{[A \downarrow P]} = \sup_{y \in A_i} \mu_A(x_1, \ldots, x_i, \ldots x_n)$

Infine si accenna a come anche le relazioni possano essere estese a RELAZIONI FUZZY. Una relazione nitida tra piu' insiemi $X_1, X_2, \ldots, X_n$ puo' essere vista come un sottoinsieme dell'insieme prodotto cartesiano di detti insiemi. Analogamente una RELAZIONE FUZZY tra piu' insiemi nitidi $X_1, X_2, \ldots, X_n$ puo' essere vista come un sottoinsieme fuzzy di $U = X_1 \times X_2 \times \ldots \times X_n$.

2.3 Estendendo la logica: K-SEQ

Un primo approccio ad una logica del ragionamento approssimato parte dal considerare il calcolo proposizionale classico, valutando le proposizioni su [0,1] anziche' su {0,1}. Si ottengono varie logiche, a seconda degli operatori fuzzy utilizzati al posto dei corrispettivi operatori logici (che varieranno, in genere, nele loro proprieta'). Come esmpio si propone la logica K-SEQ o LOGICA SEQUENZIALE STANDARD, ottenuta utilizzando intersezione, unione e complemento standard.

Per questa, dette P,Q,R lettere enunciative, i valori di verita' sono dati da:[8]

- $v(\sim P) = 1 - v(P)$
- $v(P \Rightarrow Q) = v(\sim P \cup Q) = \max(1 - v(P), v(Q))$
- $v(P \cup Q) = \max(v(P), v(Q))$
- $v(P \cap Q) = \min(v(P), v(Q))$

Per la logica K-SEQ valogono i seguenti risultati:

- La logica K-SEQ e' essenzialmente trivalente, ovvero se due f.b.f. coincidono su $\left\{0, \frac{1}{2}, 1\right\}$ allora coincidono su [0,1]
- Tutti e soli i teoremi di L^9 hanno in K-SEQ valore maggiore o uguake a $1/2^{10}$

La logica K-SEQ presentata permette di valutare la verita' su [0,1] e quindi si tratta in parte di una logica fuzzy[11]. Tuttavia essa non tratta direttamente su insiemi fuzzy e non permette un ragionamento approssimato da premesse incerte.

2.4 Introduzione alla teoria della possibilta'

La teoria della possibilta', in congiunzione con la teoria degli insiemi fuzzy, consente di sviluppare una logica piu' propriamente fuzzy

2.4.1 Misure fuzzy

Si consideri una misura del tipo "l'oggetto x pesa piu' di 10Kg". Gli insiemi [0,10] e (10,∞) necessari a rappresenare tale misura sono insiemi nitidi, non si habisogno di ricorrere ad insiemi fuzzy per rappresentare la vaghezza (come

[8] $v(\bullet)$ rapprsenta la funzione di verita', AND, OR e NOT sono rappresentati rispettivamente da intersezione, unione e complemento standard

[9] con L ci si riferisce solitamente alla teoria assiomatica del calcolo proposizionale classico

[10] si considera, in una deduzione $\dfrac{\begin{array}{c}P\\P \Rightarrow Q\end{array}}{Q}$ che il valore di verita' della deduzione sia pari al minimo dei valori di verita' delle premesse.

[11] Piu' spesso ci si riferisce a logiche come quella proposta come a logiche multivalenti

ad esempio per "pesante", "leggero"). Tuttavia si ha avra' sempre una incetezza nell'attribuire la misua a [0,10] o (10,∞). Per rappresentare tale tipo di incertezza si ricorre allora ad una MISURA FUZZY,ovvero una funzione che associ ad ogni possibile sottoinsieme[12] dell'universo del discorso un grado di appartenenza dell"'elemneto misurato".

In altri termini una misura fuzzy e' definita come una funzione $g : P(X) \rightarrow [0, 1]$ tale che:

- $g(0) = 0$
- $g(U) = 1$
- se $A \subseteq B$, allora $g(A) \leq g(B)$
- se $A_1 \subseteq A_2 \subseteq \ldots \subseteq A_n$ o se $A_1 \supseteq A_2 \supseteq \ldots \supseteq A_n$, allora $\lim_{i\rightarrow\infty} g(A_i) = g\left(\lim_{i\rightarrow\infty} A_i\right)$

2.4.2 Misure di Plausibilta' e Credenza

Si consideri una funzione m, detta ASSEGNAMENTO DI BASEcosi' definita:

$m : P(U) \rightarrow [0, 1]$[13] tale che $m(0) = 0$ e $\sum_{A\in P(X)} m(A) = 1$

ad essa si associa la seguente semantica: $m(A)$ misura l'evidenza che un elemento appartenga all'insieme A, ma non ad un suo sottoinsieme. Un einsieme A tale che $m(A) > 0$prende il nome di ELEMENTO FOCALE.

Si possono a questo punto introdurre due misure, sia sulla base dell'assegnamento di base, o equivalentement caratterizzandole con due ulteriori proprieta', esse sono la MISURA DI CREDENZA, e la MISURA DI PLAUSIBILTA'.

La Misura di Credenza e' caratterizzata dall'ulteriore proprieta':

$Bel(A_1 \cup A_2 \cup \ldots \cup A_n) \geq \sum_i Bel(A_i) - \sum_{i<j} Bel(A_i \cap A_j) + \ldots + (-1)^n Bel(A_1 \cap A_2 \cap \ldots \cap A_n)$

questa misura rappresenta la credenza che un elemento x appartenga all'insieme A e ai vari sottoinisiemi di A. Tale misura puo' anche essere espressa in funzione dell'assegnamento di base: $Bel(A) = \sum_{B\subseteq A} m(B)$

La misura di Plausibilta' e' caratterizzata dall'ulteriore proprieta':

[12]le misure fuzzy sono efinite su insiemi nitidi, ma possono essere estese ad insiemi fuzzy

[13]In realta' si richiede meno restrittivamente che $m(\bullet)$ sia definita su un'algebra di Borel, o anche una $\sigma - algebra$. In sostanza si richiede che il dominio di $m(\bullet)$ contenga insieme vuoto, insieme universo, complemento di ogni elemento, e sia chiuso rispetto all'unione. Non e' un caso che la definizione su un'algebra di Borel sia richiesta anche dall Teoria della Probabilita', di fatto questa puo' essere interpretata come una misura fuzzy, in cui l'assegnamento possibilistico di base si basa sull'evidenza derivata da assegnamenti ripetuti, e tale da rispettare piu' strettamente i due assiomi di subbaditivita' introdotti in seguito (di fatto richiedendo l'additivita').

$Pl(A_1 \cap A_2 \cap \ldots \cap A_n) \leq \sum_i Pl(A_i) - \sum_{i<j} Pl(A_i \cup A_j) + \ldots + (-1)^{n-1} Pl(A_1 \cup A_2 \cup \ldots A_n)$

questa misura rappresenta la plausibilita' che un elemento x appartenga all'insieme A, ad un sottoinsieme di A, o ad un insieme intersecantesi con A. Tale misura puo' essere espressa in funzione dell'assegnamento di base come: $Pl(A) = \sum_{B \cap A \neq \emptyset} m(B)$.

Due notevoli proprieta' delle misure ora introdotte sono:

- $Pl(A) \geq Bel(A) \geq m(A)$
- $Pl(A) = 1 - Bel(\sim A)$
- $Bel(A) = 1 - Pl(\sim A)$

Queste misure sono state introdotte su insiemi nitidi. definendo pero' una misura di sottoinsiemita' $S(A, B)$ ed una di similarita' $E(A, B)$tra insiemi fuzzy, quanto visto finora puo' essere esteso da $P(X)$a $Fuzzy(X)$, insieme degli insiemi fuzzy su X, universo del discorso. Ad esempio si avranno:

- $Bel(A) = \sum_{B \subseteq X} S(A, B) m(B)$
- $Pl(A) = \sum_{B \subseteq X} E(A, B) m(B)$

2.4.3 Misure di Possibilta' e Necessita'

Quando l'insieme degli elementi focali per una misura di plausibilita' o di credenza e' tale da potersi ordinare secondo una relazione di inclusione, ovvero quando $A_1 \subset A_2 \subset \ldots \subset A_n$, dove $A_i \in P(X)$rappresenta un generico elemento focale, allora valgono le seguenti proprieta':

$Bel(A \cap B) = \min[Bel(A), Bel(B)]$

$Pl(A \cup B) = \max[Pl(A), Pl(B)]$, per ogni $A, B \in \sigma - algebra$ su $P(X)$[14][15]

In queste condizioni le misure di plausibilita' e di credenza vengono rispettivamente denominate MISURA DI POSSIBILTA' e MISURA DI NECESSITA' e vengono indicate con i simboli $\prod, N$[16].

Le misure di possibilita' e necessita possono anche essere introdotte come misure fuzzy caratterizzate dalle ulteriori proprieta':

Per la necessita' : $N(A \cap B) = \min[N(A), N(B)]$

[14] Piu' in generale non e' necessario che gli elementi focali siano annidati, ma basta che essi siano tutti o alcuni dei sottoinsiemi in una sequenza di insiemi annidati.

[15] Si dice corpo di evidenza l'insieme degli elementi focali e del corrispettivo valore.

[16] vengono anche dette misure di plausibilta' e di credenza consonanti

Per la possibilita' : $\prod(A \cup B) = \max[\prod(A), \prod(B)]$, per ogni $A, B \in \sigma - algebra$ su $P(X)$[17][18]

Tali misure ovviamente ereditano tutte le proprieta' delle misure fuzzy e delle misure di plausibilta' e credenza. In particolare:

- $\prod(A) = 1 - N(\sim A)$
- $N(A) = 1 - \prod(\sim A)$

Tali proprieta' corrispondono alla semantica secondo cui A e' possibile se e solo se non A non e' necessario, e viceversa A e' necessario se e solo se non A non e' possibile. Altre proprieta' interessanti sono :

- $\prod(A \cup \sim A) = 1$
- $N(A \cap \sim A) = 0$
- se $N(A) > 0$allora $\prod(A) = 1$
- se $\prod(A) < 1$ allora $N(A) = 0$

Come per la misura di probabilita' e la funzione di distribuzione di probabilita', cosi' anche ogni misura di possibilta' puo' essere caratterizzata da una FUNZIONE DI DISTRIBUZIONE DI POSSIBILTA' $\pi : X \to [0, 1]$[19] , in particolare $\prod(A) = \max_{x \in A} \pi(x)$, $\forall A \in X$[20].

2.4.4 Possibilita' e insiemi fuzzy

A questo punto e' possibile mettere in relazione insiemi fuzzy e teoria della posiibilta' . Data la frase "X e' A" in cui X e' una VARIABILE FUZZY e A e' un insieme fuzzy (entrambi definiti su un universo del discorso U), tale assegnamento produce una distribuzione di possibilita' su X, pari alla fuznione di appartenenza di A, in altri termini $\pi_X = \mu_A$.

[17] Piu' in generale non e' necessario che gli elementi focali siano annidati, ma basta che essi siano tutti o alcuni dei sottoinsiemi in una sequenza di insiemi annidati.

[18] Si dice corpo di evidenza l'insieme degli elementi focali e del corrispettivo valore.

[19] X ovviamente e' una $\sigma - algebra$

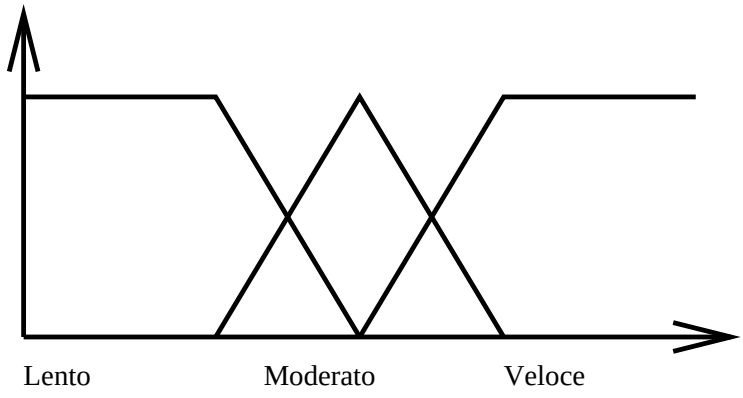
[20] Si puo' dimostrare che $m(A_i) = \pi(x_i) - \pi(x_{i+1})$

2.5 Rappresentazione di proposizioni fuzzy

Proposizioni vaghe e imprecise possono quindi essere rapresenate nella logica fuzzy attraverso distribuzioni di possibilta'. Ad esempio dato un universo del discorso $U = \{10, 20, 30, 40, 50, 60, 70, 80\}$ ed il seguente insieme fuzzy $Giovane = \{1/10 + 0.8/20 + 0.5/30 + 0.2/40 + 0.1/50 + 0.1/60 + 0/70\}$ *"Giovanni e' giovane"* e' una proposizione che associa alla variabile eta' di giovanni la distribuzione $1/10 + 0.8/20 + 0.5/30 + 0.2/40 + 0.1/50 + 0.1/60 + 0.1/70$. Quindi *"l'eta' di Giovanni e' quaranta anni"* ha un valore di possibilta' pari a $\pi_{EtaDiGiovanni}(40)$ pari a 0.2. Analogamente la possibilita' di *"Giovann abbia tra i trenta e i cinquantra anni"* sara' pari a $\sup_{x \in \{30,40,50\}} \pi_{EtaDiGiovani}(x)$, che risulta 0.5.

2.5.1 Variabili linguistiche

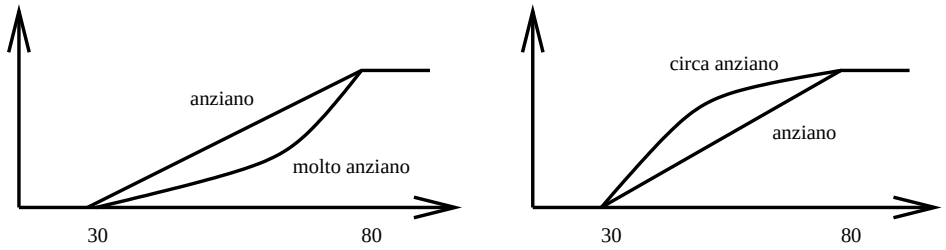
Una variabile linguistica e' una variabile i cui possibili valori sono variabili fuzzy. Formalmente puo' essere definita come una quintupla $(x, T(x), U, G, M)$, con x nome della variabile, U universo del discorso, T(x) insieme dei termini di x (insieme dei nomi dei valori linguistici di x, ognuno dei quali e' una variabile fuzzy definita su U), G e' una regola sintattica per generare i nomi dei valori di x, M e' una regola semantica per associare ad ogni valore il suo significato. Come esempio si consideri la variabile *Velocita'*, x="Velocita"', ={0,...,100},T(Velocita')={piano,moderato,veloce},G ed M sono intuitivi (vedi figura).



Un MODIFICATORE LINGUISTICO $h(\bullet)$e' un operatore che modifica un insieme fuzzy A (ed il significato ad esso associato). Alcuni modificatori comunemente usati sono ad esempio:

- $molto(A)$ e' cosi' definito: $\mu_{molto(A)}(u) = (\mu_A(u))^2$

- $circa(A)$ e' cosi' definito: $\mu_{circa(A)}(u) = \sqrt{\mu_A(u)}$

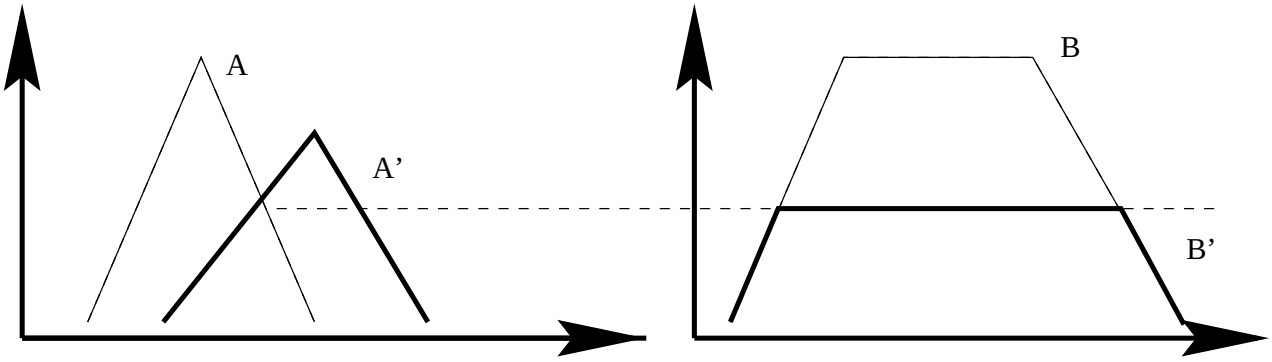


2.5.2 La variabile Verita'

Una particolare variabile linguistica e' *Verita'*={assolutamente falso, molto falso, circa falso,falso,circa vero,vero,molto vero, assolutamente vero}[21].. I valori che essa assume sono cosi' definiti (si ricordino i modificatori prima introdotti):

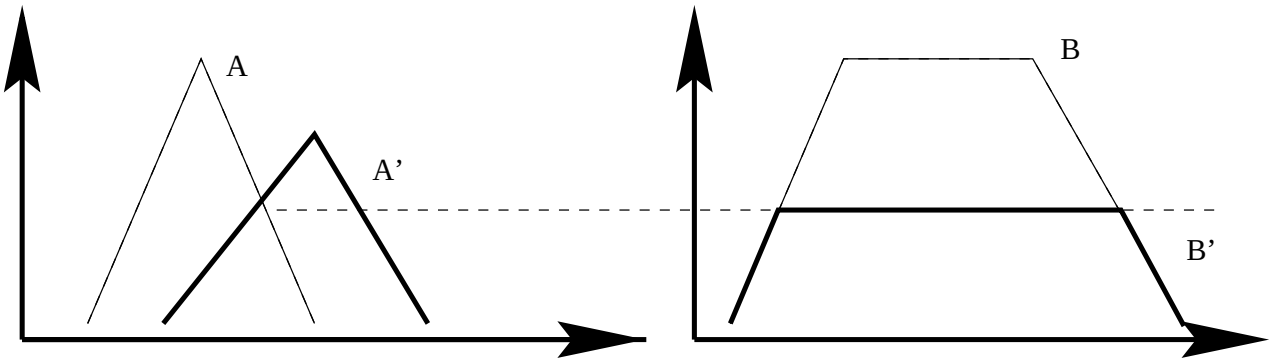
- $Vero(u) = u,\ u \in [0,1]$
- $Falso(u) = u,\ u \in [0,1]$
- $AssolutamenteFalso(u) = \begin{cases} 1 \Leftrightarrow u = 0 \\ 0 \Leftrightarrow u \neq 0 \end{cases},\ u \in [0,1]$
- $AssolutamenteVero(u) = \begin{cases} 1 \Leftrightarrow u = 1 \\ 0 \Leftrightarrow u \neq 1 \end{cases},\ u \in [0,1]$

[21]Il valore di verita' fuzzy di una proposizione fuzzy puo' anche essere introdotto in una forma piu' estesa. Introdotto come variabile linguistica esso e' utile per mostrare come rappresentare proposizioni del tipo (A e' vero).



Distrobuzione della possibilta' di B' dedotta considerando la premessa A' e la relzione tra A e B data dalla Implicazione di Mandani

Detto τ un termine della variabile Verita', si puo' allora interpretare la proposizione "x e' A e' τ" come "x e' $\tau(A(u))$". Ad esempio dato un generico A, "x e' A e' vero" equivale a "x e' vero(A)" quindi ad x viene associata la distribuzione di posibilita' $vero(A(u)) = A(u)$, corrispondente ad "x e' A". Quindi di default ad una proposizione e' associato il valore di verita' vero. Un esempio mostrera' come A venga trasformato in "A e' assolutamente vero" e "A e' assolutamente falso".



Distrobuzione della possibilta' di B' dedotta considerando la premessa A' e la relzione tra A e B data dalla Implicazione di Mandani

2.5.3 Ancora sulle proposizioni fuzzy

Piu' proposizioni fuzzy possono essere combinate fra loro, ad esempio "X e' A e X e' B", in questo caso si puo' rocorrere ad una t-norma per rappresentarne la distribuzione di possibilita', analogamente una disgiunzione puo' essere ricondotta ad una t-conorma. Anche la negazione ed altre operazioni possono essere introdotte, in ogni caso si puo' determinare una distribuzione di possibilta' per ogni proposizone prodotta (eventualmente multidimensinale). Non vengono qui trattate verie forme di quantificazione o qualificazione, che portano a vari tipi di ragionamento, tra cui uno di tipo sillogistico.

2.6 Inferenza fuzzy

Si pone ora il problema di determinare la distribuzione di possibilita' di una conseguenza, a partire da piu' premesse. Alcune regole sono le seguenti:

- REGOLA DI INCLUSIONE : x e' A, A$\subset$B, allora x e' B
- REGOLA DI CONGIUNZIONE : x e' A, x e' B, allora x e' A$\cap$B
- REGOLA DI DISGIUNZIONE : x e' A, x e' B, allora x e' A$\cup$B
- REGOLA DI NEGAZIONE : non "x e' A", allora x e' non A
- REGOLA DI PROIEZIONE : (x,y) sono in relazione R, x e' $\prod_X(R)$(ed amche y e' $\prod_Y(R)$

2.6.1 Regola di inferenza composizionale

La REGOLA DI INFERENZA COMPOSIZIONALE e' cosi' definita:

$$\frac{\begin{array}{ccc} X & e' & A \\ (X,Y) & e' & R \end{array}}{\begin{array}{ccc} Y & e' & A \circ R \end{array}}$$

dove $\mu_{A\circ R}(v) = \max_u \min(\mu_A, \mu_R(u,v))$[22]

Tale regola puo' essere vista come una composione della regola di congiunzione e della regola di proiezione, e puo' essere generalizzata utilizzando una norma arbitaria al posto dell'intersezione standard. Tuttavia e' utile darne un'altra interpretazione, che ne evidenzi la semantica.

Considerata la frase *"se X e' A, allora Y e' B"*[23] e' possibile darne una interpretazione di tipo possibilistico. Piu' esattamente si puo' introdurre

[22]Tale operazione e' anche datta max-min composition tra un insieme A e una relazione R

[23]In generale le due variabili fuzzy varieranno su due insiemi distinti, qui U e V rispettivamente per x e y

LA DISTRIBUZIONE DI POSSIBILITA' CONDIZINALE $\prod_{Y|X}(x,y)$ definta come possibilta' che Y assuma il valore y, dato che X assuma il valore x. Si puo' anche introdurra la DISTRIBUZIONE DI POSSIBILTA' CONGIUNTA $\prod_{X,Y}(x,y)$ per cui vale $\prod_{X,Y}(x,y) = t(\prod_X(x), \prod_{Y|X}(x,y))$.

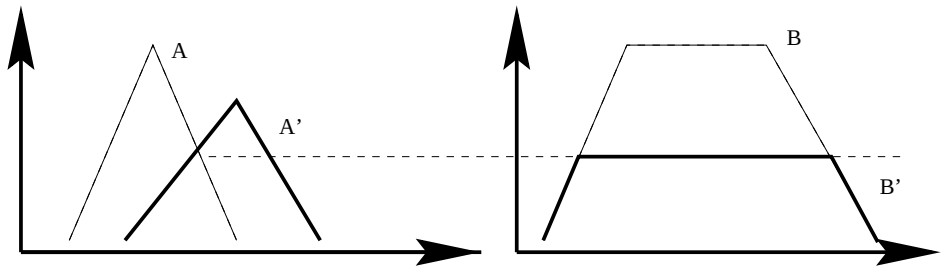
Allora dato un assegnamento di possibilta' alla frase *"se X e' A, allora Y e' B"*[24] ed un assegnamento di possibilta' ad X da *"X e' A"* si puo' misurare la possibilta' di *"Y e' B"* come misura di possibilta' della proiezione della distribuzione di possibilta' congiunta $\prod_{X,Y}(x,y)$ su Y, ovvero $\prod(Y) = \sup_{x \in U} t(\prod_x(x), \prod_{Y|X}(x,y))$.[25]

Le proprieta' di tale ragionamento dipenderanno in generale dalla t-norma adottata e dalla $\prod_{Y|X}(x,y)$, che puo' essere denotata anche come una FUNZIONE DI IMPLICZIONE ed indicata con $X \rightarrow Y(x,y)$.

Si presentano due esempi di ragionamento che sfruttano l'intersezione standard come t-norma e due diverse fuznioni di implicazione, l'implicazione di Mandani e l'implicazione di Larsen[26].

- Implicazione di Mandani $\mu_{A \rightarrow B}(u,v) = \mu_A(u) \cap \mu_B(v)$
- Implicazione di Larsen $\mu_{A \rightarrow B}(u,v) = \mu_A(u) \bullet \mu_B(v)$

premessa e conclusione sono rappresentate da A' e B'.

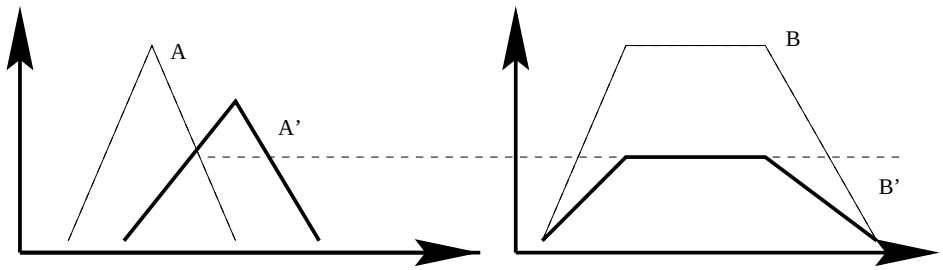


Distrobuzione della possibilta' di B' dedotta considerando la premessa A' e la relzione tra A e B data dalla Implicazione di Mandani

[24] X e Y possono variare in due insiemi in genere distinti.

[25] Si puo' piu' in generale richiedere che la distribuzione di possibilta' di Y sia maggiore od uguale a quella cosi' misurata, corrispondentemente al fatto che non e' necessario che la regola porti alla distribuzione di possibilta' piu' restrittiva per Y

[26] Tali funzioni di implicazione sono spesso usate nei sistemi di controllo



Distrobuzione della possibilta' di B' dedotta considerando la premessa A' e la relzione tra A e B data dalla Implicazione di Larsen

2.6.2 Modus Ponens Generalizzato

Il *Modus Ponens* e' la regola di inferenza per eccellenza, esse permette, date *"se X e' A allora Y e' B"* e *" X e' A"*, di dedurre *"Y e' B"*. La rappresentazione della proposizone *"se X e' A allora Y e' B"* nella logica tradizionale e' un proposizone che e' vera sempre, fuorche' se lal'antecedente e' vero e la conseguenza falsa, ovvero, generalizzando, il valore della verita' dell'antecedente e' sempre superiore di quello della conseguenza.

Generalizzandolo al caso della logica fuzzy, si ha una regola di deduzione cosi' posta, detta MODUS PONENS GENERALIZZATO (GMP):

da "se X e' A allora Y e' B" e da "X e' A" si deduce "Y e' B*"*[27]

e si vuole che siano soddisfatte le seguenti proprieta':

* PROPRIETA' DI BASE: *se* $A* \subseteq A$ *e A* e' normalizzato allora* $B^*=B$. Tale prorieta' stabilisce che il Modus Ponens Generalizzato contenga il modus ponens classico come caso particolare.

- PROPRIETA' DI SOPRAINSIEMITA': *se* $A^* \neq A$ *e A* non e' inclusoi in A, allora* $B^* \supset B$. Ovvero se la premessa differisce dall'antecedente essendo in parte esterna ad essa, la distribuzione di possibilta' della conseguenza dedotta e' maggiore. Di fatto piu' la premessa e' distante dall'antecedente piu' aumenta l'indeterminazione della conseguenza dedotta, ovvero aumenta la sua distribuzione di possibilta', che al limite sara' pari solo a 1 in tutto l'universo del discorso, situazionepari all'indeterminazione totale.

[27] vi e' qui una generalizzazione rispetto al modus ponens, in cui si richiede che la premessa sia uguale all'antecedente. Nel modus ponens generalizzato premessa e antecedente saranno piu' in generale raffrontati.

- PROPRIETA' DI INDETERMINAZIONE TOTALE: *Se A* e' non A, allora Y e' sconosciuto*, ovvero pari a una distribuzione sempre uguale all'unita' su tutto l'universo del discorso.

Si puo' dimostrare che la regola di deduzione composizionale, con le funzioni di implicazione di Mandani o Larsen, non soddisfa le condizioni del GPM. Tuttavia il modus ponens generalizzato e' un caso particolare della regola di deduzione composizionale. Ad esmpio utilizzando come t-norma l'intersezione standard, e come funzione di implicazione $\mu_{A\to B}(u,v) = \sup_{u\in U}\{z \in [0,1] \mid (\mu_A(u), z$ la funzione di deduzione composizionale soddisfa le proprieta' del modus ponens generalizzato e si ha: $\mu_{B^*}(v) \geq \sup_{u\in U}(\mu_{A^*}(u), \mu_{A\to B}(u,v))$, con la funzione di implicazione prima definita.

Piu' in genarale il modus ponens generalizzato puo' essere visto come una funzione m, che soddisfa le seguenti proprieta', relative ad una funzione di implicazione $\mu_{A\to B}(u,v)$, che ora indicheremo come $I[\mu_A(u), \mu_B(v)]$:

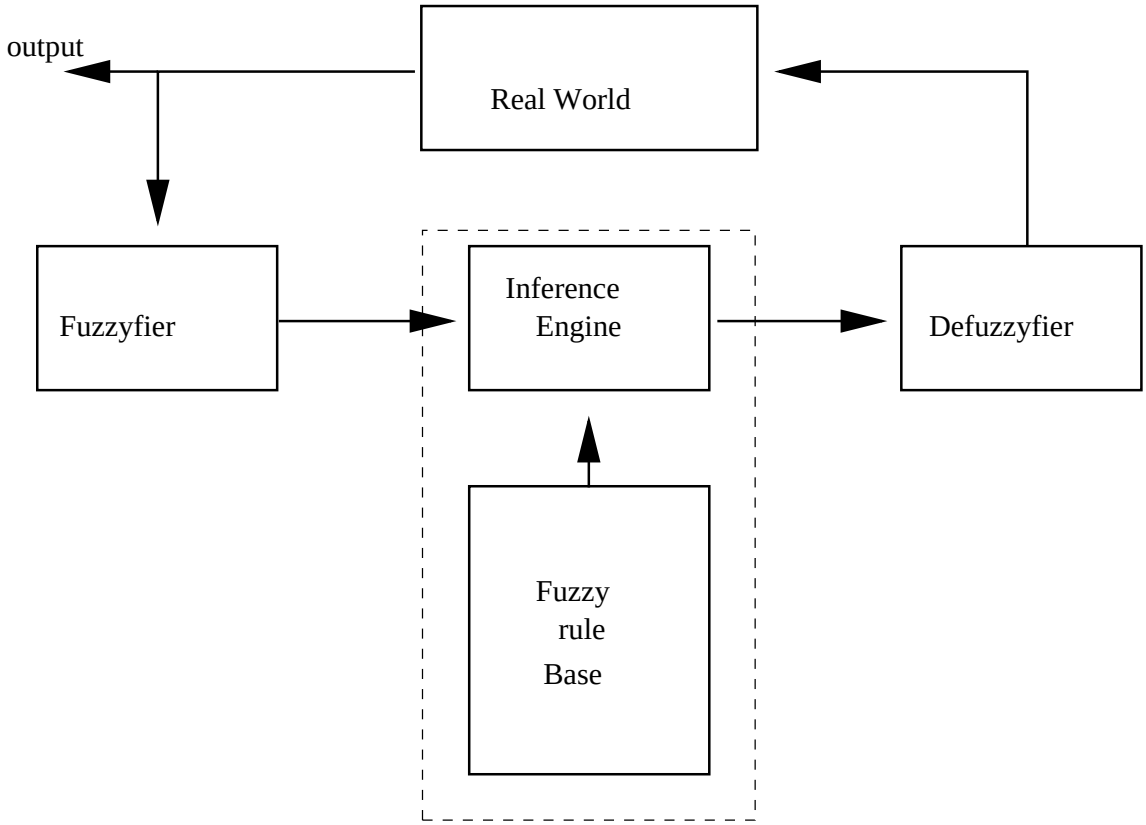
- $\mu_B(v) \geq m(\mu_A(u), I[\mu_A(u), \mu_B(v)])$
- $m(1,1) = 1$
- $m(0, I[\mu_A(u), \mu_B(v)]) = 0$
- se $\mu_{A'}(u) \leq \mu_A(u)$, allora $m(\mu_{A'}(u), I[\mu_A(u), \mu_B(v)]) \leq m(\mu_A(u), I[\mu_A(u), \mu_B(v)])$

Una funzione m che sodddisfa tali proprieta' viene detta FUNZIONE CHE GENERA IL MODUS PONENS. Alcuni risultati consentono di garantirne l'esistenza

2.7 Cenno ai sistemi di controllo Fuzzy

Il sucesso piu' grande dei sistemi fuzzy e' nell'ambito del CONTROLLO e dei SISTEMI ESPERTI. Qui si utilizzano spesso sistemi di ragionamento qualitativo, basati su un insieme di regole, che non sempre si basano sul Modus Ponens Generalizzato, ma ad esempio usano spesso la regola di inferenza composizionale con la norma di Mandani o di Larsen.

Si presenta lo schema di un tipoco sistema di controllo, e alcune delle istanze in esso coinvolte.



2.7.1 Fuzzyfier

Spesso le variabili osservate potrebbero essere non fuzzy, quindi si pone il problema di una loro conversione in termini fuzzy. Il piu' semplice FUZZYFICATORE converte un detreminato valore $x_0$in un FUZZY SINGLETON, overo un insieme fuzzy la cui funzione di appartenenza vale 1 in x_0, 0 altrimenti. Nei sistemi esperti spesso l'input e' di tipo linguistico e quindi gia' fuzzy.

2.7.2 Defuzzyfier

Analogamente il risultato di un controllore fuzzy (o di un sistema esperto fuzzy) e' a sua volta un'azione fuzzy, che, specie in un impianto di controllo, deve essere tradotta in un'azione nitida. Il DEFUZZIFICATORE traduce l'output del controllore, rappresentato da un insieme fuzzy, in un valore "nitido". Un esempio tipico di metodo di defuzzificazione e' il COA (Center of Gravity) in cui l'azione di controllo e' calcolata come il baricentro della distribuzione di possibilta' rappresentante l'azione di controllo. in altri termini,

dato un universo del discorso U discreto , un'azione di controllo A, il valore defuzicato $u*$ e' dato da:
$U^* = \frac{\int_u \mu_A(u) u \partial u}{\int_u \mu_A(u) \partial u}$

2.7.3 Rule base

La BASE DELLE CONOSCENZE o BASE DELLE REGOLE e' rappresentata da un insieme di regole della forma:

- R^i: SeX e' A_i e ... eY e' B_i, allora z = C_i

in questo caso si e' mostrata la i-esima regola di un sistema MISO (Multiple Input, Single Output). Una generica base di conoscenze sara' rappresentata da un insieme di regole connesse dal connettivo OR.

2.7.4 Inference engine

Il MOTORE INFERENZIALE genera un output dagli ingressi e consultando la base delle regole. Presentiamo tre tipici metodi di ragionamento:

- RAGIONAMENTO FUZZY DI PRIMO TIPO. Considerando la funzione di implicazione di Mandani o di Larsen[28] la conclusione C' puo' essere espressa come: $C' = (A' \cap \ldots \cap B') \circ [(A_1 \cap \ldots \cap B_1 \rightarrow C_1) \cup \ldots \cup (A_n \cap \ldots \cap B_n \rightarrow C_n)]$, dove $\circ$denota la composizione max-min, precedentemente introdotta.

- RAGIONAMENTO FUZZY DEL SECONDO TIPO.Tale metodo si basa sull'assunzione che le funzioni di apparteneza degli insiemi $C_i$siano monotone. Detta allora α_i la FORZA DI ATTIVAZIONE della i-esima regola, calcolabile come $\alpha_i = \min(\mu_{A_i}(x_0), \ldots, \mu_{B_i}(x_0))$ (dove $\mu_{X_i}(x_0)$ indica il grado di matching tra i dati nella base di regole e i dati forniti), si ha un'azione di controllo calcolata come $u_{control} = \frac{\alpha_1 u_1 + \ldots + \alpha_n u_n}{\alpha_1 + \ldots \alpha_n}$, dove $u_i = \mu_{C_i}^{-1}(\alpha_i)$. Si noti che l'azione di controllo e' nitida e non fuzzy.

- RAGIONAMENTO FUZZY DEL TERZO TIPO. Definita la forza di attivazioe della i-esima regola come al punto precedente, e condiderate regole della forma: R^i: Se x e' A_i e ... e y e' B_i, allora z = $f_i(x, y)$, allora l'azione di controllo e' data da: $u_{control} = \frac{\alpha_1 f_1(x_0, \ldots, y_0) + \ldots + \alpha_n f_n(x_0, \ldots, y_0)}{\alpha_1 + \ldots + \alpha_n}$. Si noti che in questo caso ne' l'input ne' l'output sono fuzzy.

[28] quanto segue non e' vero per tutti i tipi di implicazione, in particolare per alcuni di essi il ruolo di congiunzione ed disgiunzione e' invertito

3 Reti Neurali

Le RETI NEURALI sono un sistema di trattamento dell'informazione costruito facendo uso di alcuni principi organizzativi tipici del cervello umano. Esse sono composte da un alto numero di semplici unita' di elaborazione collegati insieme ed operanti in parallelo, ed il loro comportamento collettivo mostra la capcita' di apprendere, ricordare e generalizzare da un insieme di DATI DI ADDESTRAMENTO.

Da notare che le reti neurali sono molti utili in campi in cui e' molto abile l'uomo (ad esempio nel pattern matching), mentre non sono indicate per computazioni rigorose ed algoritmiche (dove anche la mente umana ha difficolta').

Una rete neurale e' costituita da un insieme di elementi, detti NEURONI, ognuno dei quali riceve piu' inputs (dagli ingressi o da altri neuroni) e, pesandoli, li combina in un valore che , passando attraverso un elemento non lineare, determina l'output dell'elemento, che a sua volta puo' essere l'input di altri neuroni o essere l'uscita[29].

I pesi con cui vengono misurati gli ingressi (uno per ogni connessione da un neurone ad un altro) vengono variati, sulla base dei dati forniti, fintanto che la rete non "calcola" la funzione desiderata. L'algoritmo con cui viene effettuato l'adeguamento dei pesi prende il nome di REGOLA DI APPRENDIMENTO.

L'informazione appresa dalla rete risiede quindi nei pesi delle connessioni, tale INFORMAZIONE e' DISTRIBUITA, ovvero la sua rappresentazione e' sparsa nei pesi della rete, in modo tale che la sua organizzazione dipenda dalla rete stessa, e non sia fornita dall'esterno.

Per queste caratteristiche ci si riferisce alle reti neurali come ad un SISTEMA SUBSIMBOLICO di trattamento della conoscenza. Essa infatti elabora valori numerici anziche' simboli, e immagazzina le informazioni in modo destrutturato e non percepibile dall'esterno della rete.[30]

3.1 Elemenenti caratteristici delle reti neurali

Una rete neurale e' caratterizzata da tre elementi: il modello di neurone utilizzato[31], la topologia della rete, la regola di apprendimento. Di seguito verranno presentate queste tre entita'.

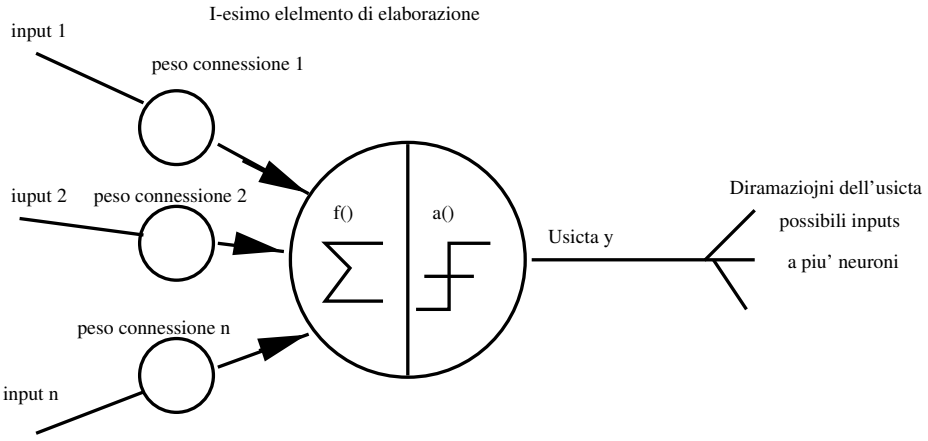
[29] Tale operazione puo' comportare o meno un ritardo temporale

[30] Per inciso una applicazione molto frequente delle reti neurali e' quella di modelli black box per sistemi non lineari.

[31] Piu' di un modello di neurone puo' essere utilizzato all'interno di una stessa rete

3.1.1 Il neurone

Un modello base di neurone e' il NEURONE MP. La sua struttura e' mostrata nel seguente grafo:



considerando un ritardo temporale tale elemento i-esimo , dati gli inputs $x_1, x_2, \ldots, x_n$e detti w_{in}il generico peso associato alla n-esima connessione dell-iesimo elemento, calcola la funzione:

$y_i(t+1) = a\left(\sum_{j=1}^{n} w_{ij}x_j(t) - \theta_i\right)$, dove la funzione a() e' definita come:

$a(f) = \begin{cases} 1 \Leftrightarrow f \geq 0 \\ 0 \Leftrightarrow f < 0 \end{cases}$

In tale neurone si puo' distinguere una prima parte, lineare, che combina gli inputs a seconda dei pesi delle corrispondenti connessioni (il prodotto di tale prima operazione prende il nome di NET-INPUT, o POTENZIALE DI ATTIVAZIONE), e una seconda parte, non lineare (in questo caso uno scalino),che attiva il segnale di uscita solo se il net-input supera una determinata soglia di attivazione (tale parte prende il nome di FUNZIONE DI ATTIVAZIONE o FUNZIONE DI TRASFERIMENTO).

La struttura del neurone MP e' mantenuta da tutti i tipi di neurone, che possono essere ottenuti variando le funzioni caratteristiche della prima e della seconda parte.

Alcune diffuse funzioni per il calcolo del net-input sono:

- Combinazione lineare: $f_i = \sum_{j=1}^{n} w_{ij}x_j - \vartheta_i$[32]

[32] ϑ_i puo' essere considerato come un ulteriore peso associato ad ingresso costante pari a -1. Tale considerazione consente di trattare in modo uniforme tutti i parametri della rete. Per tale ragione d'ora in poi il valore di tale perametro non sara' piu' preso in considerazione, in particolare per gli algoritmi di apprendimento. Si noti che tale approssimazione e' anche in vario modo generalizzabile alle altre funzioni di net-input

- Funzione quadratica: $f_i = \sum_{j=1}^{n} w_{ij} x_j^2 - \vartheta_i$
- Funzione polinomale: $f_i = \sum_{j=1}^{n} \sum_{k=1}^{n} w_{ijk} x_j x_k + x_j^{\alpha_j} + x_k^{\alpha_k} - \vartheta_i$

Tipiche funzioni di trasferimento utilizzate sono le seguenti[33]:

- Funzione a scalino: $a(f) = \begin{cases} 1 \Leftrightarrow f \geq 0 \\ 0 \Leftrightarrow f < 0 \end{cases}$
- Funzione di soglia : $a(f) = \begin{cases} 1 \Leftrightarrow f \geq 0 \\ -1 \Leftrightarrow f < 0 \end{cases}$
- Funzione di rampa : $a(f) = \begin{cases} 1 \Leftrightarrow f > 1 \\ f \Leftrightarrow 0 \leq\leq 1 \\ 0 \Leftrightarrow f < 0 \end{cases}$
- Sigmoide unipolare : $a(f) = \frac{1}{1+e^{-\lambda f}}$
- Sigmoide bipolare : $a(f) = \frac{1}{1+e^{-\lambda f}} - 1$, $\lambda > 0$ determenina quanto sia ripida la sigmoida, all'infinito essa coincide con lo scalino (o con la soglia, a seconda che sia unipolare o bipolare).

I neuroni piu' frequentemente utilizzati sono LTU (LINEAR THRESHOLD UNIT) che combina lineramente gli inputs e sfrutta una funzione di trasferimento a soglia o a scalino, e il LGU (LINEAR GRADED UNIT) che ha la stessa prima parte e una funzione di trasferimento a sigmoide.

3.1.2 Topologia

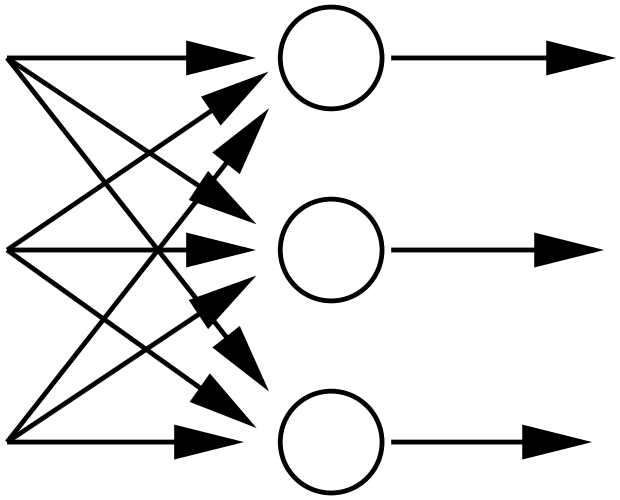
Le reti neurali sono costituite da neuroni fortemente interconnessi, in cui gli output di alcuni fungono da inputs ad altri (con funzione eccitante o inibitoria[34]). E' fondamentale per una rete specificare come tali connessioni sono disposte, nonche' la funzione dei vari neuroni[35]. Le principali topologie utilizzate sono le seguenti:

- SINGLE LAYER FEEDFORWARD NETWORK. Piu' neuroni vengono affiancati in parallelo, operando sugli stessi input.

[33] non vengono qui considerate le reti a simmetria radiale

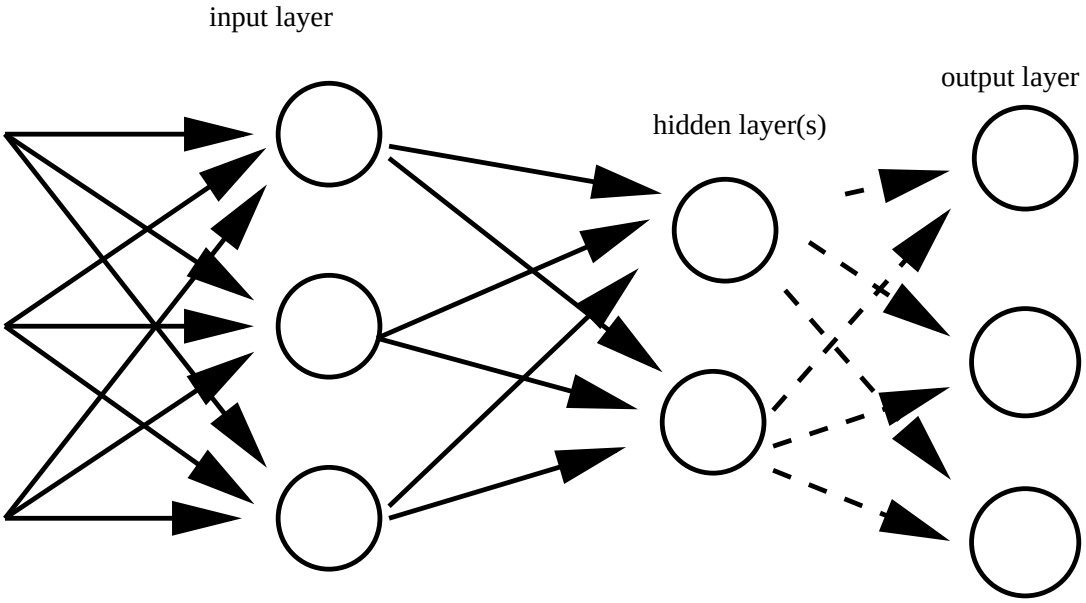
[34] di fatto l'uscita di un neurone ha funzione inibitoria su una latro neurone quando la connessione fra il primo e il secondo ha un peso negativo

[35] Qui si trascende dalla dimensione del tempo, eventualmente sono da specificarsi anche eventuali ritardi.



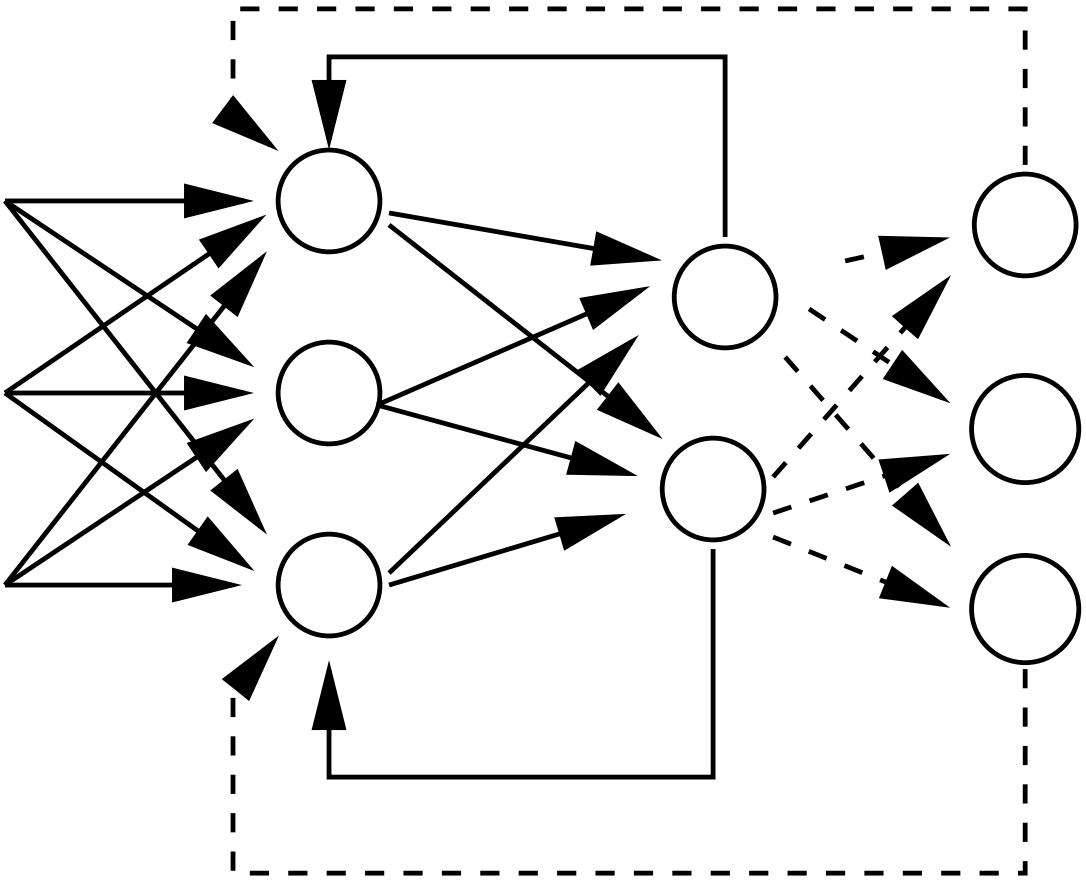
Single layer feedforward network

- MULTI LAYER FEEDFORWARD NETWORK. Piu' reti single layer sono connesse in cascata. Il primo strato prende il nome di INPUT LAYER, l'ultimo di OUTPUT LAYER, gli strati intermedi di HIDDEN LAYER. Una rete si dice TOTALMENTE CONNESSA se ogni output di tale strato e' connesso ad ogni input dello strato successivo.



Multi layer feedforward network

- FEEDBACK NETWORK. Le topologie precedenti sono dette feedforward perche' ogni livello funge da input a livelli siccessivi. Qualora le uscite di un livello possano essere ingressi per neuroni di livelli precedenti si parla di feedback networks.
- LATERAL FEEDBACK NETWORK. Si tratta di feedback networks in cui gli output di un livello fungono da input a neuroni dello stesso livello.
- RECURRENT NETWORKS. Si tratta di feedback networks che presentano anelli chiusi.

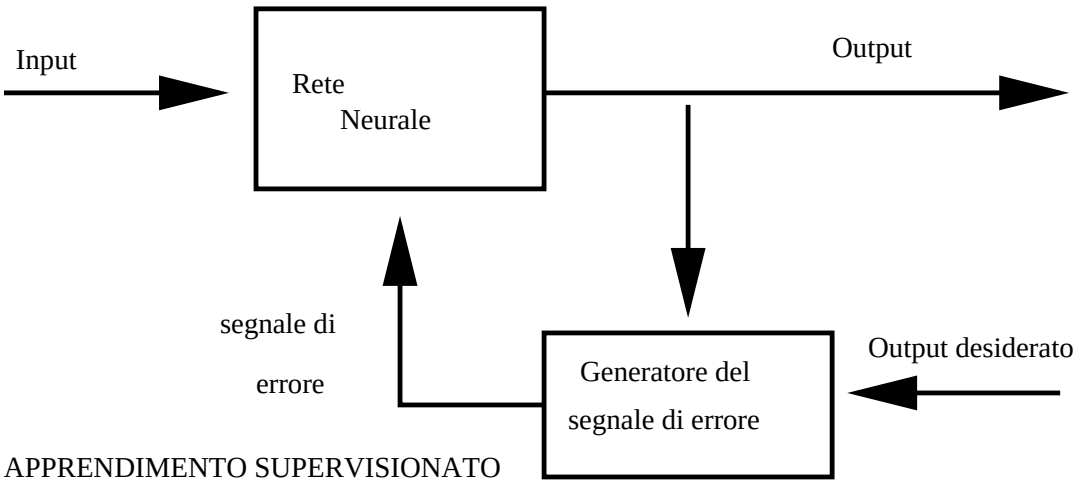


Multi layer recurrent network

3.1.3 Regola di apprendimento

Per una rete neurale sono possibili due tipi di apprendimento: APPRENDIMENTO DEI PARAMETRI e APPRENDIMENTO DELLA STRUTTURA[36] (l'ultimo dei quali non verra' qui trattato). L'apprendimento dei parametri consiste essenzialmente nell'apprendimento dei pesi delle connessioni, che, partendo da una condizine iniziale, vengono aggiornati in base agli esempi presentati. Si possono distinguere tre tipi di apprendimento:

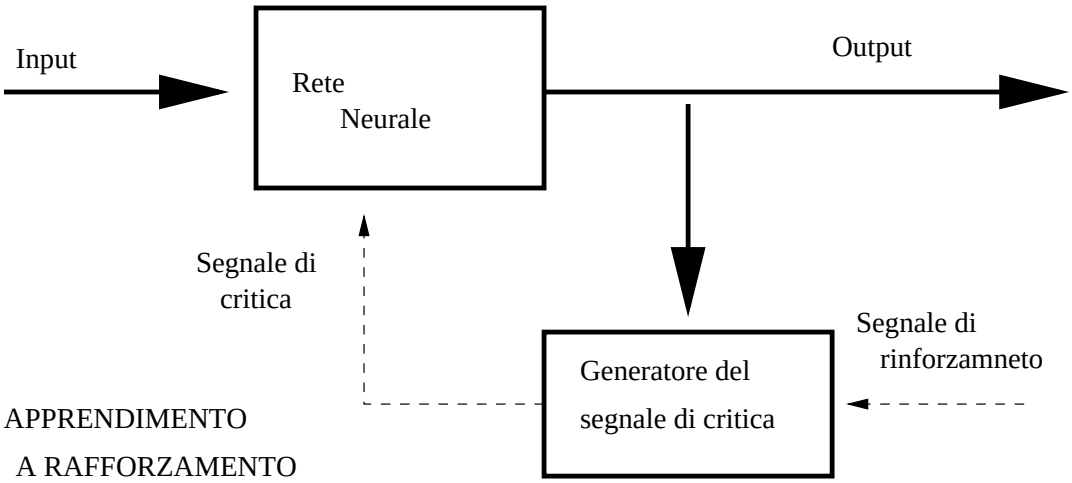
- APPRENDIMENTO SUPERVISIONATO. Sono presenti come esempi dati di input e dati di output, quindi e' possibile stabilire ad ogni esempio o iterazione (spesso infatti la presentazione degli esempi e l'applicazione della regola di apprendimento avvengono iterativamente) un segnale di errore. Tale tipo di apprendimento viene anche detto apprendimento con insegnante.



APPRENDIMENTO SUPERVISIONATO

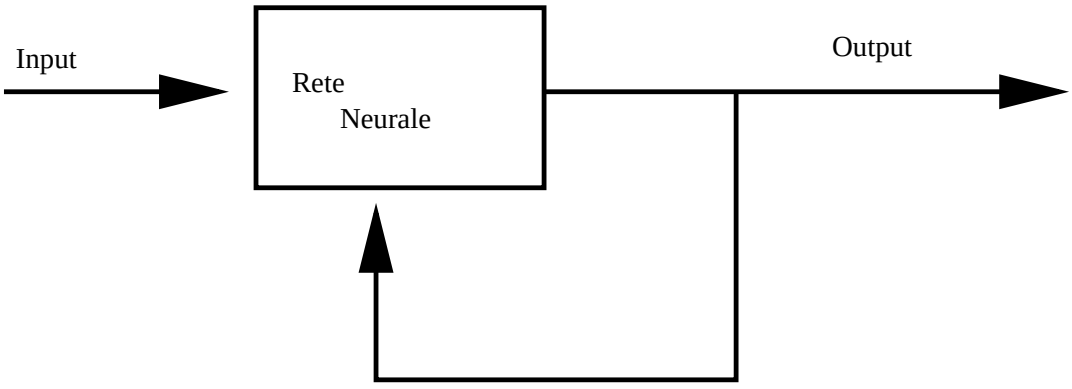
- APPRENDIMENTO A RAFFORZAMENTO. Non e' presente l'output desiderato, ma solo una critica dell'output prodotto (ad esempio del tipo Positivo/Negativo), quindi il segnale di errore prodotto ha un contenuto piu' debole rispetto al caso dell'apprendimento supervisionato (tuttavia in ambedue i casi e' presente un feedback). Viene anche detto apprendimento con critica.

[36]Per l'apprendimento strutturale vengono spesso utilizzate tecniche evolutive quali gli algoritmi genetici o la programmazione evolutiva



APPRENDIMENTO
A RAFFORZAMENTO

- APPRENDIMENTO NON SUPERVISIONATO. Sono presenti solo dati in input, ed e' la rete stessa a dover trovare relazioni tra questi (schemi, relazioni, regloraita', etc.). Un tipico esempio e' la classificazione di oggetti, senza stabilire a priori le classi. Durante questo apprendimento la rete muta i propri pesi in un processo detto AUTO ORGANIZZAZIONE.

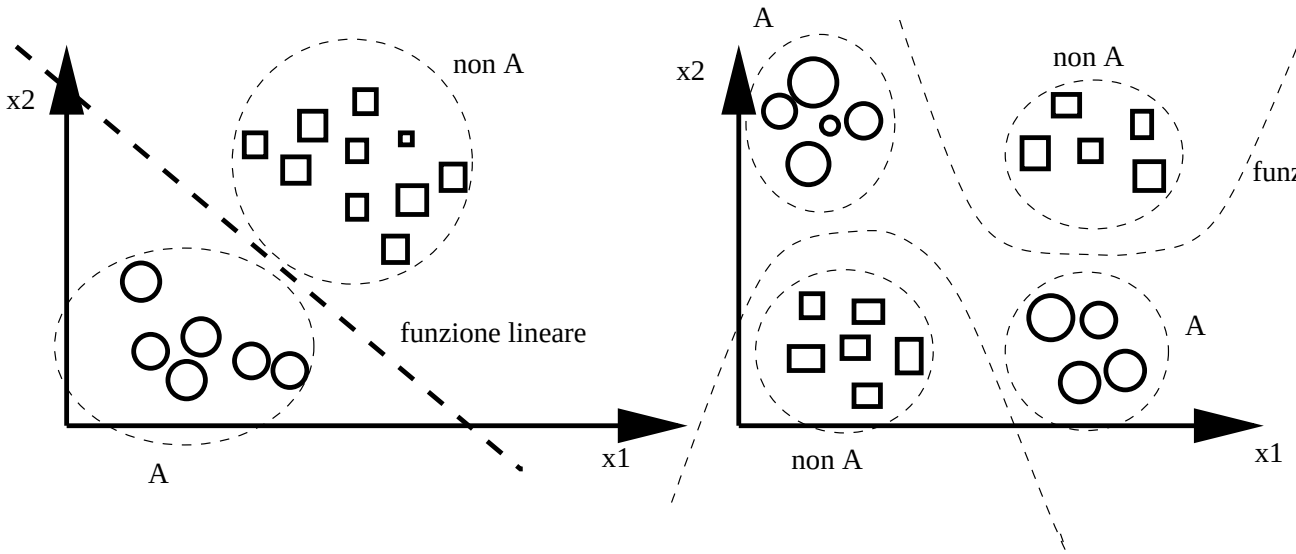


APPRENDIMENTO
NON SUPERVISIONATO

3.2 Una semplice rete.

La piu' semplice rete che si puo' pensare e' costituita da un solo neuronle LTU, avente come funzione di trasferimento lo scalino, e come net-input una combinazione linerare degli ingressi. Tale rete vene detta PERCEPTRONE ed il suo output e':

$o = \begin{cases} 1 \Leftrightarrow \sum_{i=1}^{n} w_i x_i - \vartheta \geq 0 \\ 0 \Leftrightarrow \sum_{i=1}^{n} w_i x_i - \vartheta < 0 \end{cases}$ dove $x_1, \ldots, x_n$ sono gli n-input, $w_1, \ldots, w_n$i corrispondenti pesi e ϑ la soglia di attivazione. In sostanza tale rete separa lineramente lo spazio in due sottospazi. Ad esempio per un ingresso bidimensionale si puo' vedere come l'output della rete sia 1 se l'input si trova in uno dei due sottospazi, 0 nell'altro caso. I due spazi risultano suddivisi da : $x_1 w_1 + x_2 w_2 - \theta = 0$, variando i pesi delle connessioni si puo' allora variare l'inclinazione della retta, mentre variando ϑ si effettua una sua traslazione. Durante la fase di apprendimento un perceptrone puo' quindi apprendere come partizionare linearmente lo spazio a seconda degli input presentati (si parla anche di RETE A SEPARAZIONE LINEARE). Ovviamente non tuttti gli input sono linaremente separabili, come mostra il seguente grafo in cui i dati presentati sono divisi in due gruppi, A e $\overline{A}$, la cui distinzione e' quanto puo' apprendere la rete.



Per inciso si noti che una tale rete puo' apprendre gli operatori AND e OR, ma non l'operatote XOR. Naturalmente una funzione non lineare puo' ancora partizoonare adeguatamente lo spazio (come anche una rete multilayer).

3.2.1 Apprendimento per un perceptrone

La regola di apprendimento base per un perceptrone e' la seguente[37]:

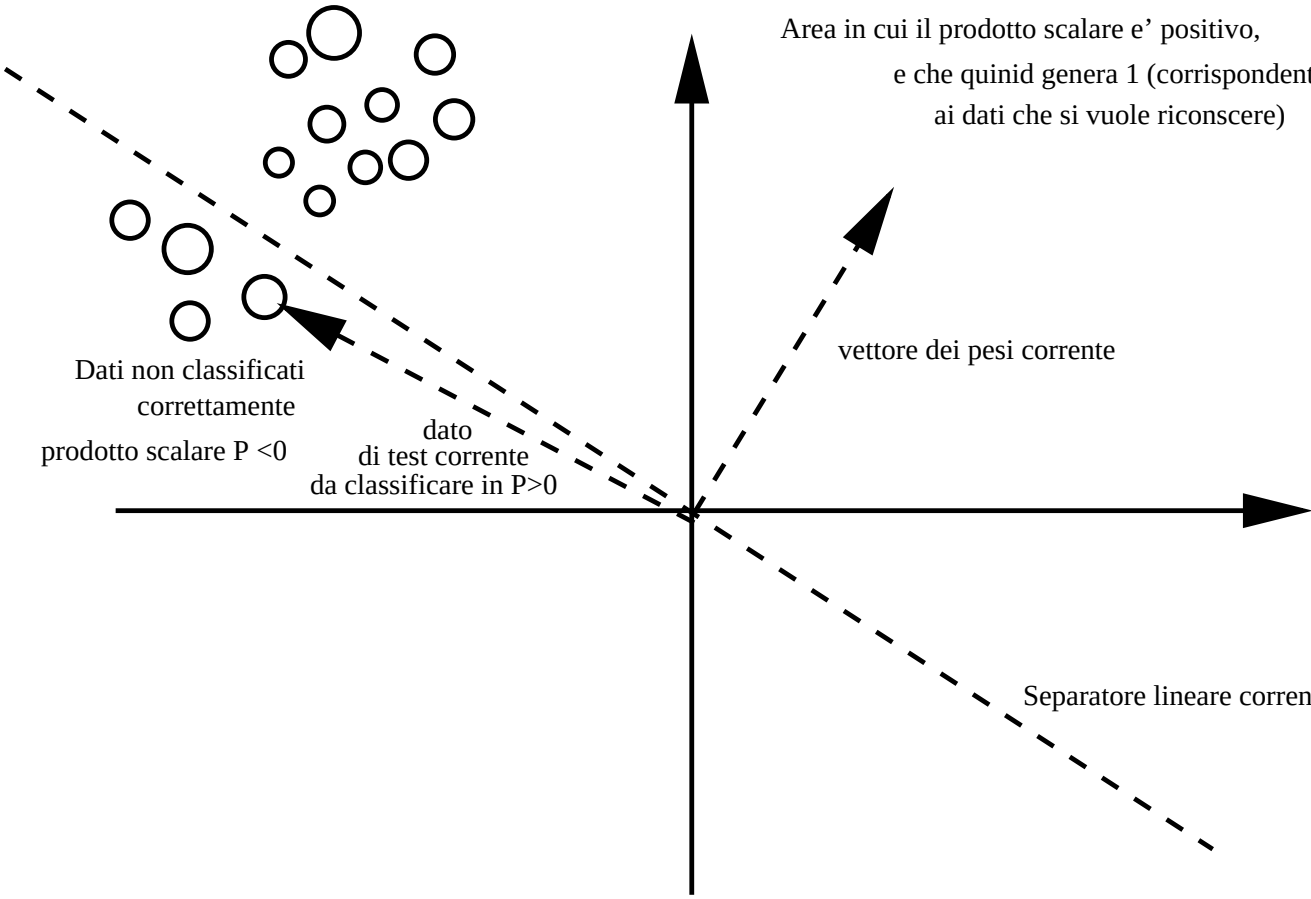
1. Presentazioni di un insieme un input dall'insieme dei dati di addestramneto: $\overline{x_1}, \ldots, \overline{x_n}$, $\overline{o}$
2. Calcolo del potenziiale di attivazione $P = \sum_{i=1}^{n} w_i \overline{x_i}$
3. Calcolo dell'ouput: $o = \begin{cases} 1 \Leftrightarrow P \geq 0 \\ 0 \Leftrightarrow P < 0 \end{cases}$
4. Se $o = \overline{o}$ tornare al punto 1, altrimenti aggiornare la stima dei pesi. ed esettamente se $\overline{o} < o$ (i.e.: $\overline{o} = 0$ e $o = 1$) diminuire i pesi di $\Delta w_i = -\eta x_i$ (in senso algebrico, ovvero sommare), se $\overline{o} > o$ (i.e.: $\overline{o} = 1$ e $o = 0$)aumentare i pesi di $\Delta w_i = \eta x_i$. η prende il nome di LEARNING RATE o TASSO DI APPRENDIMENTO.

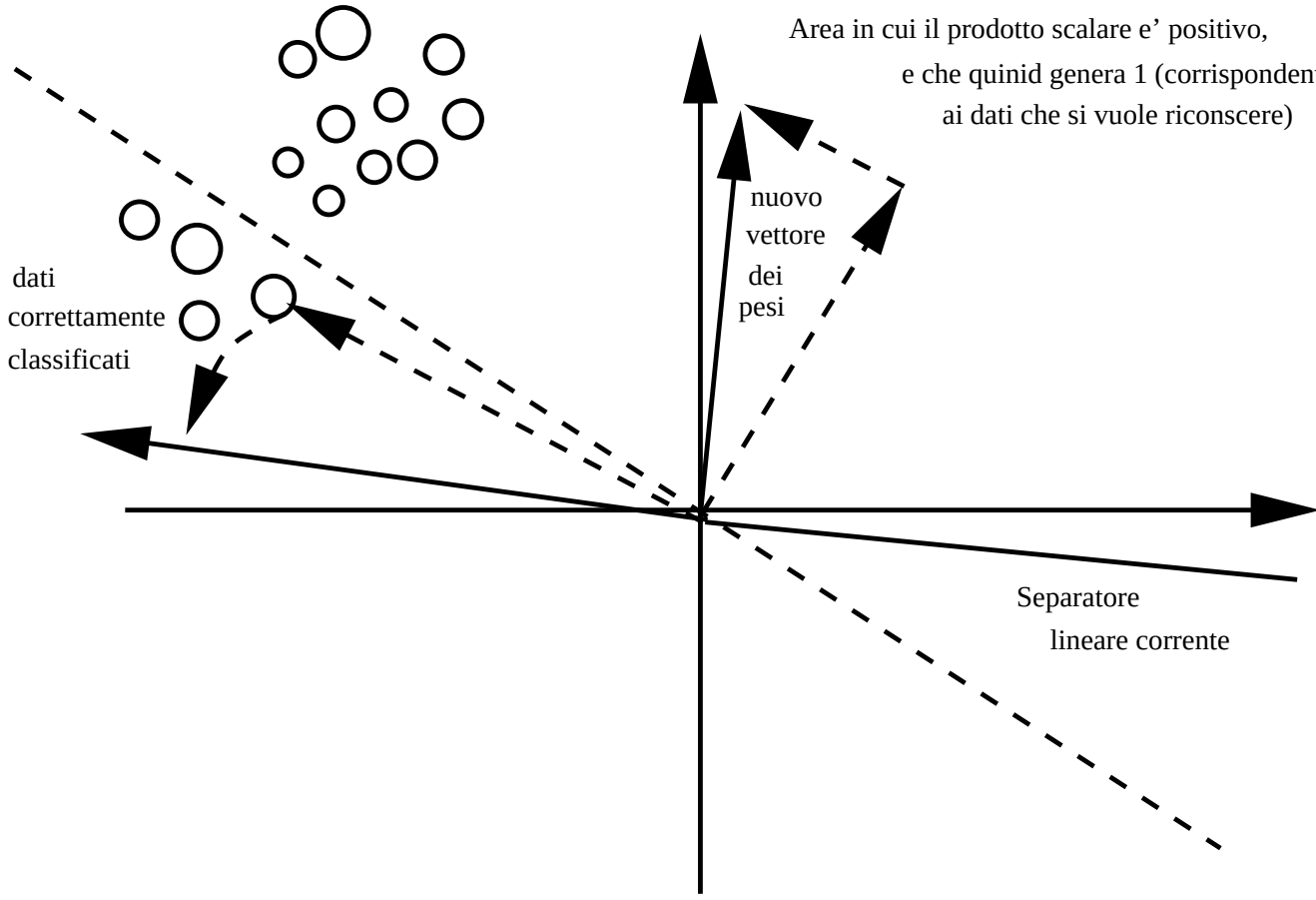
L'algoritmo, se la soluzione esiste, vi converge dopo un numero finito di esempi.

Tale regola di apprendimento dei pesi puo' essere cosi' spiegata. Si consideri il caso in cui l'output desiderato e' 1 e l'output prodotto e' 0. Allora si puo' pensare che il potenziale di attivazione generato dall'input in esame sia troppo basso, e quindi occorre incrementare il valore, di conseguenza i pesi.

Una visione geometrica chiarisce meglio perche' venga scelto un simile incremento dei pesi. Il potenziale di attivazione, per come e' definito, e' dato dal prodotto scalare tra il vettore dei pesi e il vettore dei dati di esempio (il vettore dei pesi e' a sua volta perpendicolare all'iperpiano di separazione). Inoltre, non considerando θ, tutti i vettori in gioco hanno origine nell'origine. Incrementare i pesi di $\Delta w_i = \pm \eta x_i$ equvale a ruotare il vettore dei pesi (e l'iperpiano di separazione) nella direzione dei dati in esame.

[37]Non si fa rifrimento al parametro θ in quanto questo e' assimilibile ad un peso, il cui corrispettivo input sia forazato a -1, come precedentemente accennato


Area in cui il prodotto scalare e' positivo,
e che quinid genera 1 (corrispondent
ai dati che si vuole riconscere)
vettore dei pesi corrente
Dati non classificati
correttamente
prodotto scalare P <0
dato
di test corrente
da classificare in P>0
Separatore lineare corren



3.3 Regola di apprendimento di Widrow-Hoff[38]

Piu' perceptroni possono essere parallelizzati in una configurazione single-layer. In questo modo ognuno di essi puo' essere istruito a rinoscere una particolare forma lineare, e l'intera rete puo' rinoscere piu' forme lineari, pari al numero di neuroni utilizzati.

Sia data una rete single layer feedforward, formata da m LGU (o altri neuroni con funzione di net-input lineare e funzione di attivazine differenziabile), ognuno avente n inputs. Si indichi con w_{ij}il peso della connessione tra

[38]La presentazione di questa regola e' utile introduzione all'algoritmo di Back-Progation , che e' la fuznione di addestramento per le reti multi layer feedforward, che godono della proprieta' di aprossimazione universale. In se quste reti: syngle layer feedforward sono intermedie, basterebbe una combinazione lineare delle loro uscite (e quindi un ulrerior "strato" per garantire anche a queste tale proprieta'.

l'input i e il neurone j, inoltre sia ogni singolo dato di esempio nella forma $\overline{x_1}, \ldots, \overline{x_n}, \overline{o_1}, \ldots, \overline{o_m}$[39].

L'algoritmo di Widrow-Hoff e' un METODO A MINIMAZZIONE DELL'ERRORE DI PREDIZIONE. Ad ogni presentazione dei dati di input, viene calcolato l'output $o_1, \ldots, o_n$ e questo viene confrontato con l'output desiderato per generare una CIFRA DI MERITO. Una tipica scelta e' l'ERRORE QUADRATICO MEDIO $E = \frac{1}{2} \sum_{j=1}^{m} (o_j - \overline{o_j})^2$.

L'algortimo procede iterativamente spostandosi nella direzione di maggior decrescita di E, ovvero in dirazione opposta al suo gradiente. Ad ogni iterazone quindi si calcola il gradiente dell'errore corrente in funzione dei pesi, quindi ci si muove nella direzione opposta al gradiente (di una quantita' pari al learning rate).

Tale algoritmo puo' essere cosi' esposto:

1. Presentazione di un input : $\overline{x_1}, \ldots, \overline{x_n}, \overline{o_1}, \ldots, \overline{o_m}$
2. calcolo degli output corrsipondenti $o_1, \ldots, o_n$, dove $o_j = a(P_j(\overline{x_1}, \ldots, \overline{x_n})$, essendo $a()$ la funzione di attivazione e $P = \sum_{i=1}^{n} w_{ij} x_i$ la fuznione di potenziale di attivazione.
3. calcolo dell'errore quadratico medio $E = \frac{1}{2} \sum_{j=1}^{m} (o_j - \overline{o_j})^2$.
4. calcolo della variazione $\Delta w_{ij} = w_{ij}(t+1) - w_{ij}(t) = -\eta \frac{\partial E}{\partial w_{ij}}$, con η tasso di apprendimento, solitamente comnpreso tra 0 e 1[40].
5. Ritorno al punto 1, finche' l'errore non risulta inferiore ad un minimo stabilito[41].

Alcune considerazioni sull'algoritmo presentato:

- L'algoritmo cosi' presentato e' detto ON-LINE, perche' viene applicato mano a mano che gli esempi vengono presentati. A rigore tuttavia l'errore di predizione va calcolato sull'intero lotto, quindi al termine della presentazione degli esempi occorre eseguire l'operazione di correzione dei pesi, e di seguito si puo' ripetere piu' volte l'intero procedimento. Questa seconda modalita' di applicare l'algoritmo e' detta BATCH. Tuttavia la versione on-line e' quella ampiamente piu' usata.

[39] Si tratta di un tipo di apprendimento supervisionato, pertanto dovranno essere forniti sia gli inputs, che gli outputs desiderati.

[40] Si noti che qui il tempo avanza con l'iterare dell'algoritmo

[41] L'intero procedimento puo' essere ripetuto piu' volte, e ogni ripresentazione del lotto di dati, insieme al relativo apprendimento, viene denominato EPOCA

- Il valore dell'aggiornamento $\Delta w_{ij} = -\eta \frac{\partial E}{\partial w_{ij}}$ e' pari a $\Delta w_{ij} = -\eta D_j x_i$ con $D_j = (o_j - \overline{o_j})a'(\overline{P_j})$, Infatti $\frac{\partial E}{\partial w_{ij}} = (o_j - \overline{o_j})\frac{\partial o_j}{\partial w_{ij}}$, $\frac{\partial o_j}{\partial w_{ij}} = \frac{\partial o_j}{\partial P_j}\frac{\partial P_j}{\partial w_{ij}} = a'(\overline{P})x_i$.[42]

- Il valore dell'aggiornamento, nel caso la funzione di attivazione sia lineare, diventa: $\Delta w_{ij} = -\eta(o_j - \overline{o_j})x_i$[43]

- Nel caso la funzione di attivazione sia una sigmoide, si ha:[44] $\Delta w_{ij} = -\eta(o_j - \overline{o_j})o_j(1 - o_j)x_i$.

- In questa sede si accenna solo al fatto che, dato un modello lineare del sistema fisco, l'errore quadratico medio si comporta come un iperparabolide, garantendo cosi' l'esistenza della soluzione (nel caso il sistema non sia lineare e' opportuno turbare i pesi per evitare di individuare eventuali minimi globali, e non il minimo locale,).

3.4 Teorema di approssimazione universale

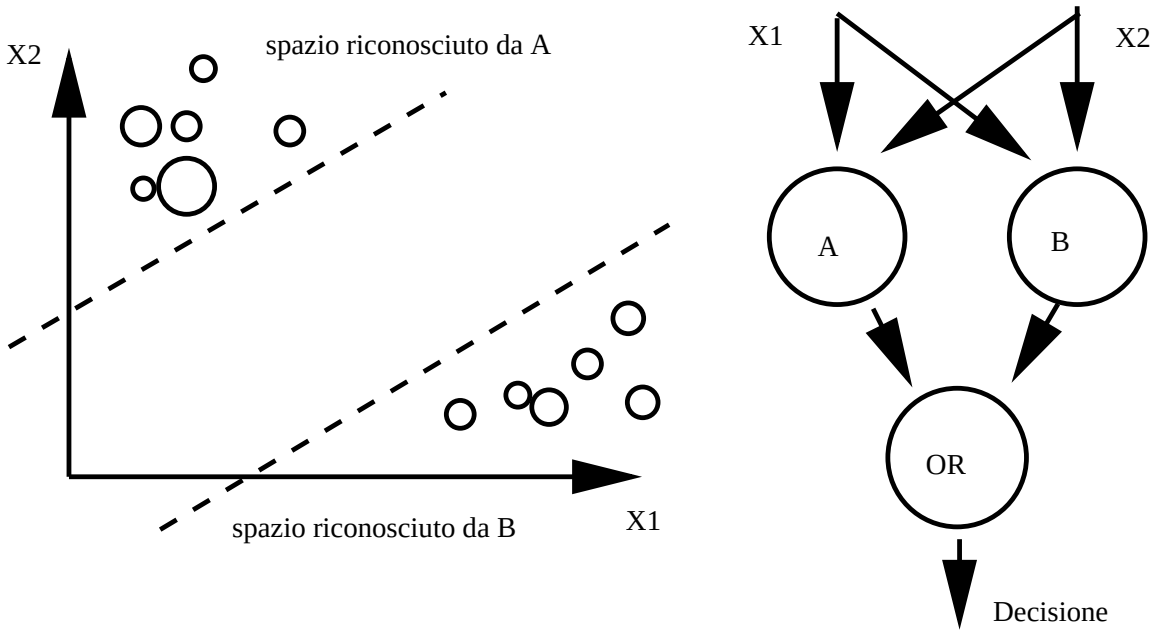
Si e' vesto che un singolo perceptrone non puo' essere addestrato per eseguire l'operazione XOR[45], tuttavia reti multilayer feedforward possono apprendere facilmente tale operazione. Ad esempio una rete con due neuroni A e B, e un terzo neurone OR (effettuante l'OR dei sui ingressi) riconoscela partizione delllo spazio associato all'operzione XOR, come si vede nella seguente fgura.

[42] $a'(\overline{P})$ rappresenta la derivata della funzione di attivazione calcolata per un net-input dato da $\overline{x_1}, \ldots, \overline{x_n}$.

[43] In questo caso si parla anche di regola di apprendimento del perceptrone continuo

[44] Si noti che si hanno dei problemi numerici se l'output e' vicino a 0 o a 1

[45] In realta' sarebbe possibile, fornendo in input oltre ai due ingressi anche il loro prodotto, in pratica cio' pero' e' equivalente a introdurre un'ulteriore semplice neurone che applica una t-norma agli ingressi, quindi si ricadrebbe in un caso particolare di multilyer feedforward network



Per una rete di perceptroni separazioni dello spazio piu' complesse possono richiedre piu' livelli intermedi (fino ad un massimo di due, effettuanti due operzioni consecuitive AND OR o viceversa).Piu' in generale vale il seguente

Teorema di approssimazione universale

Una rete multilayer feedforward, con almeno uno strato intermedio, avente una arbitraria FUNZIONE DI ATTIVAZIONE DI SQUASHING[46] e e una funzine di net-input linear o polinomiale puo' approssimare ogni funzione fino ad un arbitrario livello di accuratezza, ammesso che vi siano sufficenti elementi nello srato intermedio.[47]

[46]Una funzione di attivazione di squashing $a()$ e' cosi' definta:

- $a : [\Re \rightarrow [0, 1]$ (o anche $[-1, 1]$
- e' non crescente
- $\lim_{\lambda \rightarrow \infty} a(\lambda) = 1$
- $\lim_{\lambda \rightarrow -\infty} a(\lambda) = 0$ (o anche -1)

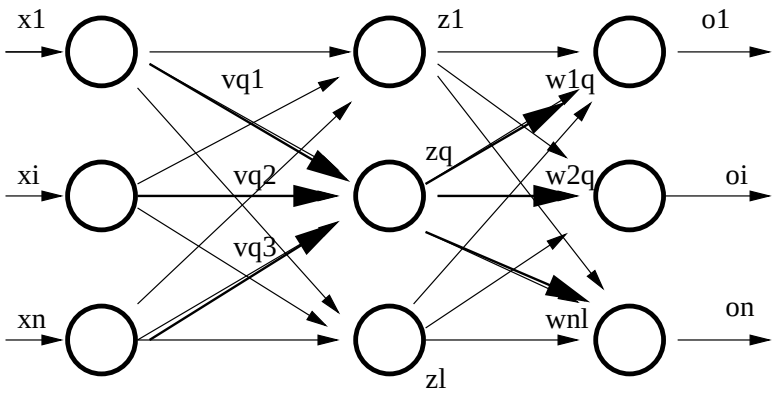
Ad esempio le sigmoidi unipolare o bipolare sono funzione di squashing

[47]Piu' formulazioni di questa proprieta' sono presenti, ma sostanzialmente equivalenti.

3.5 Regola di apprendimento di back-propagation

Si e' visto che le reti universali feedforward possono approssimare qualsiasi funzione[48]. E' chiara quindi l'importanza di questo tipo di rete. Tuttavia l'algoritmo di Widrow-Hoff non puo' essere direttamente applicato, in quanto e' qui necessario anche aggiornare i pesi dei link intermedi, che non dipendono dirattamente dall'errore. Tuttavia tale algoritmo puo' essere esteso in questa direzione, cosa che essenzialmente viene fatta nell'ALGORITMO DI BACK-PROPAGATION. Tale algoritmo e' l'algoritmo fondamentale di apprendimento supervisionato per reti multilayer feedforward, pertanto il suo ruolo e' centrale nell'ambito delle reti neurali[49]

Per una sua introduzione si consideri la seguente rete (le connessioni marcate sono quelle cui e' associato il nome):



Tale rete ha un primo strato di input, che riceve gli ingressi $x_1, \ldots, x_n$. Tali ingressi vengono trasmessi dal primo strato di neuroni al secondo, attraverso connessioni che hanno il generico peso v_{qj} indicante il peso che va dal j-esimo input al q-esimo neirone del secondo strato. Un neurone del secondo strato applica la funzione non lineare $a()$ ai sui ingressi[50], e genera un input, z_q per il q-esimo neurone, che e' in input all'ultimo strato. w_{iq} indica il peso della connessione tra il q-esimo neurone del secondo strato, equello i-esimo del primo. A sua volta ul neurone dell'ultimo strato calcolera' l'output $o()$, dopo aver combinato linearmente i suoi ingressi. Per la rete presentata i tre strati hanno rispettivamente m,l,n neuroni.

[48]Per lo meno qualsiasi funzione caraterizzabile da una superficie di controllo su un dominio

[49]Di fatto la sua introduzione si e' rivitalizzato l'interesse per le reti neurali, in primo tempo bloccate dalla difficolta' di trattare partizioni dello spazio non lineari

[50]Si suppone che la funzione di attivazione sia la stessa per tutti i neuroni della rete

L'algoritmo di back-propagation opera essenzialmente in due fasi, iin una prima fase si calcola l'output a partire dai dati di esempio, tale operazione procede in direzione forward, cioe' il flusso del segnale va dal primo strato all'ultimo. Segue una seconda fase in cui l'azione di correzione dei pesi viene propagata all'indietro, in direzione back.

La cifra di merito utilizzata per calcolare l'errore e' l'errore qudratico medio, e il metodo di ottiizzazione essenzialmente quello a discesa di gradiente, utilizxzato nell'algoritmo di Widrow-Hoff.

Il calcolo dell'errore quadratico medio prosegue nel seguente modo:

1. il generico neurone q del secondo strato calcola un potenziale di attivazione pari a $P_q = \sum_{j=1}^{m} v_{qj} xj$, e prodce un output pari a : $z_q = a(P_q) = \left(\sum_{j=1}^{m} w_{qj} x_j\right)$.

2. il generico neurone i del terzo ed ultimo strato calcola allora un potenziale di attivazione pari a $P_i = \sum_{q=1}^{l} w_{iq} z_q = \sum_{q=1}^{l} w_{iq} a\left(\sum_{j=1}^{m} v_{qj} x_j\right)$, l'output generato risulta ora $y_i = a(P_i) = a\left(\sum_{q=1}^{l} w_{iq} z_q\right) = a\left(\sum_{q=1}^{l} w_{iq} a\left(\sum_{j=1}^{m} v_{qj} x_j\right)\right)$

A questo punto il segnale in ingresso e' stato *propagato* all'uscita della rete. L'errore qudaratico medio (funzioe dei pesi) e' ora: $E = \frac{1}{2}\sum_{i=1}^{n}\left[\overrightarrow{o_i} - a(P_i)\right]^2$, ovvero $E = \frac{1}{2}\sum_{i=1}^{n}\left[\overrightarrow{o_i} - a\left(\sum_{q=1}^{l} w_{iq} z_q\right)\right]^2$.

Nella seconda fase l'errore viene ustao per distribuire l'azione di correzione all'indietro lungo la rete.

1. Per il metodo di minimizzazione utilizato si ha: $\Delta w_{iq} = -\eta \frac{\partial E}{\partial w_{iq}}$per i pesi relativi agli input dello strato di uscita. Dunque $\Delta w_{iq} = -\eta \frac{\partial E}{\partial y_i}\frac{\partial y_i}{\partial P_i}\frac{\partial P_i}{\partial w_{iq}} = \eta(\overline{o_i} - o_i)(a'(P_i))\, z_q$, quindi $\Delta w_{iq} = -\eta D_{output_i} z_q$, con ovvia definizionedi di D_{output_i}[51]. D viene anche detto SEGNALE DI ERRORE. Si noti che fin qui l'algoritmo e' identico a quello di Widrow Hoff.

2. A questo punto si procede a calcolare la correzione da apportare ai pesi in ingresso ai nodi intermedi. $\Delta v_{qj} = -\eta \frac{\partial E}{\partial v_{qj}} = -\eta \frac{\partial E}{\partial P_q}\frac{\partial P_q}{\partial v_{qj}} = -\eta \frac{\partial E}{\partial z_q}\frac{\partial z_q}{\partial P_q}\frac{\partial P_q}{\partial v_{qj}}$, quindi $\Delta v_{qj} = \eta \sum_{i=1}^{n}[(\overline{o_i} - o_i)a'(P_i)w_{iq}]a'(P_q)x_j$[52]. Anche qui si puo' ricavare $\Delta v_{qj} = \eta \sum_{i=1}^{n}[D_{output_i} w_{iq}]a'(P_q)x_j = \eta D_{hidden_q} x_j$, avendo posto $D_{hidden_i} = -\frac{\partial E}{\partial P_q} = -\frac{\partial E}{\partial z_q}\frac{\partial z_q}{\partial P_q} = a'(P_q)\sum_{i=1}^{n} D_{output_i} w_{iq}$.

[51]il pedice output sta ad indicare che si tratta del segnale di correzione corrispondente allo starto di output

[52]Si veda come E puo' essere espresso in funzione di w nel scondo punto della fase predente

Si noti che la correzione da apportare ai pesi di uno strato nascosto puo' essere determinata sulla base delle correzioni da apportare allo strato che essa alimenta, attraverso coefficenti pare al peso delle connessioni stesse coinvolte, ma questa volta percorse all'incontrario propagando segnali di errore.

L'algoritmo di back propagation puo' essere cosi' esposto, iterando sui dati di addestramento :

1. Propagazione in avanti: si considera il dato di addestramento corrente e si propaga lungo la rete, ottenedo un output

2. Si genera la cifra di merito E, confrontando l'output prodotto con quello fornito.

3. Propagazione all'indietro dell'errore: si aggiornano i pesi della rete. Considerati due strati successivi con usite z_qe y_i , e preso w_{iq} come peso generico tra idue neuroni, si aggiorna $w_{iq}^{nuovo} = w_{iq}^{vecchio} + \eta D_i z_q$. con $D_i = (\overline{o_i} - o_i)a'(P_i)$ se y_i e' l'uscita di un neurone di output, con $D_i = a'(P_i)\sum_k D_k w_{ki}$ se y_i e' l'uscita di un neurone intermedio.

L'algoritmo termina, anche dopo diverse epoche, quando l'errore diventa accettabile. Esistono diverse varianti utili a migliorare le caratteristiche dell'algoritmo, che qui non verranno esposte.

3.5.1 Algoritmo Winner takes all

Infine si presenta un esmpio di algoritmo di apprendimento non supervisonato, l'algoritmo WINNER TAKES ALL, detto anche REGOLA DI APPRENDIMENTO DI KOHOEN. L'apprendimento consiste nell'individuare, dato un insieme di oggetti, una loro classificazione. In questo algoritmo di apprendimento si assume che il numero di classi sia un dato, n.

Una rete di Kohen, qui e' applicato l'algoritmo, e' essenzialmente una rete single layer feedforward, avente m inputs $x_1, \ldots, x_m$ ed n neuroni (ognuno dei quali riceve tutti gli inputs). w_{ij} indica il peso della connessione tra il j-esimo input e lo i-esimo neurone. Gli output sono rispettivamente denominati con $o_1, \ldots, o_n$. La funzione dove la funzione di attivazione non e' rilevante per l'apprendimento.

L'algoritmo , dopo una inizializzazione, procede in due fasi:

- Inizializzazione. Gli n vettore dei pesi vengono perturbati casualmente, indi normalizzati.

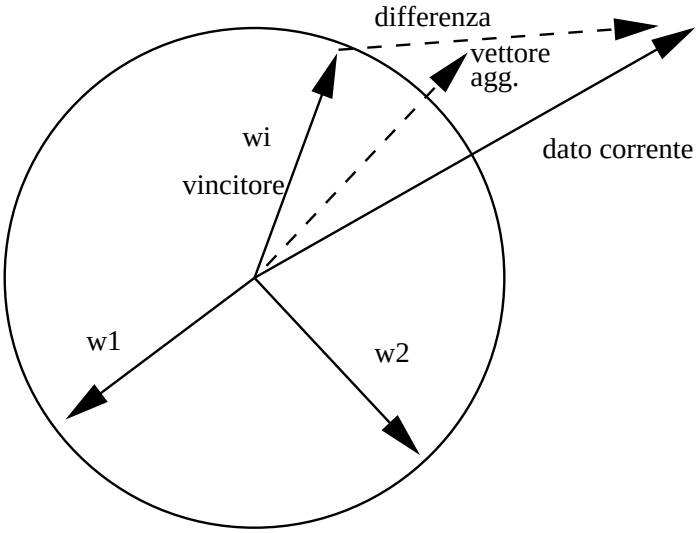
Per ogni dato presentato:

- Scelta del vincitore. Viene scelto il neurone i tale da presentare la minore distanza euclidea dal dato in quastuione, ovvero i tale che $\| x - \widehat{w_i}^{(k)} \| \leq \min_{1 \leq j \leq n} \left\{ \| x - \widehat{w_j}^{(k)} \| \right\}$[53]

- Aggiornameto dei pesi. Il wincitore viene premiato, e si ha $\widehat{w_i}^{(k+1)} = \widehat{w_i}^{(k)} + \eta^{(k)}(x - \widehat{w_i}^{(k)})$, mentre gli altri pesi rimangono invariati. Questo corrisponde ad un avvicinamento al dato in questione. η e' una opportuna costante di tempo che puo' variare nel tempo.[54]

L'apprendiento termina quando la variazione indotta nei pesi comincia ad essere inferior ad una prefissata soglia.

Un esempio grafico chiarifica la funzione di premiazione del vincitore.



La normalizzazione del vettore dei pesi serve ad evitare che nella scelta del piu' vicino ad un dato non influiscano l'ordine dei pesi correnti, ma solo la direzione del vettore dei pesi.

4 Integrazione tra sistemi fuzzy e reti neurali

Le reti neurali e i sistemi fuzzy presenatno molti elementi in comune. Ambedue sono sistemi numerici che si possono comportare come stimatori o come sis-

[53] con x e w_j si intendono i rispettivi vettori, $\widehat{w}$ sta ad indicare che il vettore e' normalizzato

[54] Sono possibili piu' varianti, che ad esmpio premino in parte anche i neuroni vicini. Di fatto cosi' come e' esposto l'algoritmo presenta vari problemi, far cui non ultimo la dipendenza dalle caratteristiche statistiche della distribuzione di dati di test

temi dinamici, ambedue possono stimare funzioni "campionate" e fungere da memorie associative, ambideue sono sistemi di trattamento dell'informazione non simbolici.

In aggiunta, si puo' dimostrare che un SISTEMA FUZZY[55], con caratteristiche del tutto generali, puo' approssimare una qualsiasi funzione, risultato analogo al teorema di approssimazione universale.

Oltre a queste analogie i due sistemi presentano pero' significative differenze:

- I sistemi fuzzy consentono un ragionamento di alto livello, attraverso regolo IF-THEN operanti su variabili linguistiche, essi sono sistemi strutturati, anche se operano su una struttura numerica sfumata.
- Un problema che si riscontra con i sistemi fuzzy e' la necessita' di definire le funzioni di memebrship o le regole logiche.
- Le reti neurali sono sistemi a struttura libera, presentano capacita' di apprendimento e di astrazione.
- Le reti neurali forniscono una rappresentazione distribuita della conoscenza ridondante e hanno ntrinsecamente doti di fault tollerance.
- Le reti neurali non consentono di dedurre una struttura dalle informazioni apprese. Queste sono immagazzinate in modo distribuito, assai difficile da "estrarre" dalla rete
- Un problema connesso al punto precedente e' la difficolta' di disegnare reti neurali per risolvere un determinato problema.

Si osserva che le differenze tra i due sistemi tendono ad essere complementari piu' che contrastanti. Cio' suggerisce una loro possibile combinazione che generi un sistema piu' ricco.

Dal lato della logica fuzzy, l'integrazione delle reti neurali fornisce l'abilita' di apprendere e la proprieta' di fault tollerance. Di fatto si arricchisce il framework fuzzy dell'abilita' di trattare informazioni di basso livello (quali quelle sensoriali)[56]

[55]Un sistema fuzzy e' fondamentalmenteil nucleo base di regole / motore inferenziale di un controllore fuzzy. Fuzzificatore e defuzzificatore possono essere presente o meno a seconda dell'esatta definizione o del tipo di ragionamento.

[56]Le reti neurali possono comunque trattare informazioni di alto livello, come distinguere, ad esempio, clienti affidabili da clienti inaffidabili. Naturalmente piu' l'informazione manipolata e' di alto livello, piu' la sua incapacita' di "spiegare" cioe' che e' stato appreso costituisce un limite.

Dal lato delle reti neurali, queste acquistano trasparenza, grazie ad una loro prestrutturazione o ad una post-interpretazione delle loro caratteristiche, resa possibile da o nel framework fuzzy.

Si possono individuare tre "vie" di integrare questi due sistemi:

- NEURAL FUZZY SYSTEMS: Le reti neurali vengono utilizzate come strumenti in modelli fuzzy. Qui i sistemi fuzzy acquistano le capacita' di apprendimento e ottimizzazione delle reti neurali. Una tipica area applicativa di tali sistemi e' il controllo.

- FUZZY NEURAL NETWORKS: I modelli di reti neurali convenzionali vengono fuzzificati, ovvero gli elementi tipici delle reti (funzioni di attivazione, funzione di potenziale,...) possono arricchirsi di elementi fuzzy. Una tipica applicazione di tali sistemi e' nel pattern recognition. Tali sistemi non verranno qui trattati.

- FUZZY-NEURAL HYBRID SYSTEMS: Si tratta di architetture orientate all'applicazione, in cui la commistione delle due tecnologie e' ad hoc per risolvere detreminate problemi. Le aree di applicazione comprendono quelle del controllo e del pattern recognition.

Di seguito verranno presentate alcuni concetti generali inerenti i neural fuzzy systems ed alcune integrazioni complete.

4.1 Realizzazione neurale di funzioni di memerships

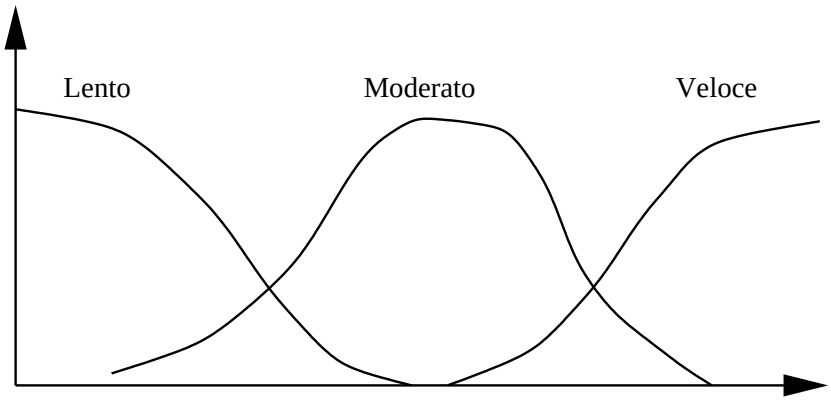
Si e' visto che una rete neurale piu' approssimare una arbitraria funzione continua , dato un numero adeguato di esempi di tale fuznione. Quindi una rete neurale , con almeno uno strato intermedio, e puo' essere utilizzate per costruire la funzione di appartenenza di un concetto, dato un adeguato numero di esempi (ed utilizzando la funzione di back-propagation)[57].

Spesso le funzioni di memberships utilizzate sono funzioni come sigmoidi o campane, allora un singolo neurone puo' essere utilizato per rappresentare una funzione di appartenenza. Un semplice esempio mostra i vantaggi di un'implementazione neurale delle funzioni di appartenenza

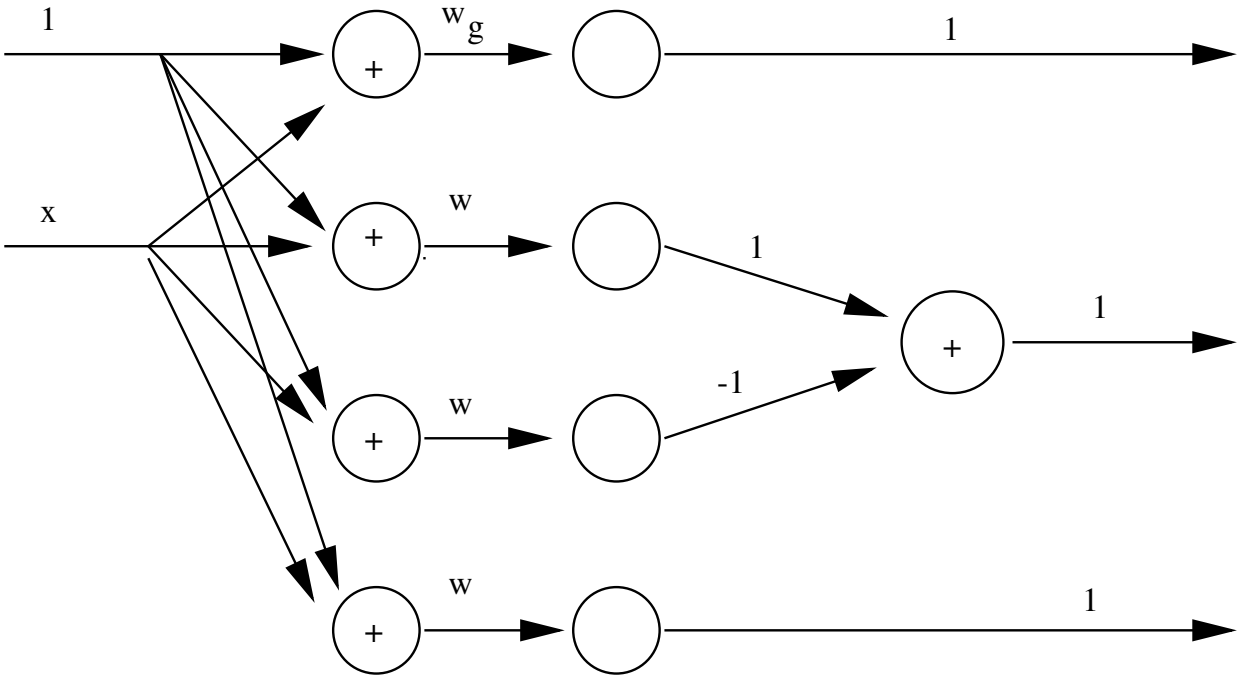
4.1.1 Un semplice esempio

Si consideri la variabile linguistica *Velocita'*, avente come insieme di termini: *{Lento, Medio, Veloce}*. Gli insiemi di appartenenza dei tre termini possono essere definiti su $\Re$ nel seguente modo:

[57] Per inciso anche una arbitraria relazione fuzzy, o una distribuzione di possibilta' congiunta, puo' essere appresa da una singola rete.



Una tale suddivisione fuzzy dello spazio puo' essere eseguita dalla rete mostrata:



Il primo strato di neuroni somma gli input, il secondo applica una funzione sigmoide unipolare al suo net-input. Come risultato ogni neurone calcola una delle tre funzioni di appartenenza. Ad esempio per l'insieme Lento, si avra' $\mu_L(x) = \frac{1}{1+e^{-w_g(x+w_c)}}$. Si noti che w_c e w_g sono parametri che determinano rispettivamente la posizione e il gradiente del sigmoide, e possono essere appresi in quanto pesi della rte neurale in questione. Dato un opportuno

numero di esempi tale rete puo' dunque imparare a suddividere lo spazio negli opportuni insiemi fuzzy.

4.2 Realizzazione basilare di operazioni fuzzy

La realizzazione neurale piu' immediata per una operazione AND, OR, NOT, e' considerare la funzione di attivazione tale da effettuare l'operazione in questione. In generale si puo' considerare un NEURONE FUZZY, che puo' effettuare operazioni tipiche degli insiemi fuzzy come t-norme e t-conorme. Allora ad esempio un neurone che computi l'operazione fuzzy AND potra' presentare una t-norma come funzione di attivazione (si noti che per esigenze legate all'apprendiemento si richiede che la funzione di attivazione sia differenziabile).

Un'altro modo di realizzare operazioni fuzzy e' utilizzare NEURONI OWA (ORDERED WEIGHTED AVERAGING), tali neuroni sono caratterizzati da un vettore di pesi di ingresso fisso, ma gli input vengono ordinati (per valore crescente o decrescente) prima di essere moltiplicati scalarmente per il vettore dei pesi (ovvero il primo peso computa sempre il valore piu' grande in input, l'ultimo il piu' basso e analogamente per i pesi intermedi). Un vettore OWA puo' effettuare una operazione compresa tra il standard fuzzy AND (min) e il standard fuzzy OR (max).

4.3 Realizzazione neurale dell'inferenza fuzzy

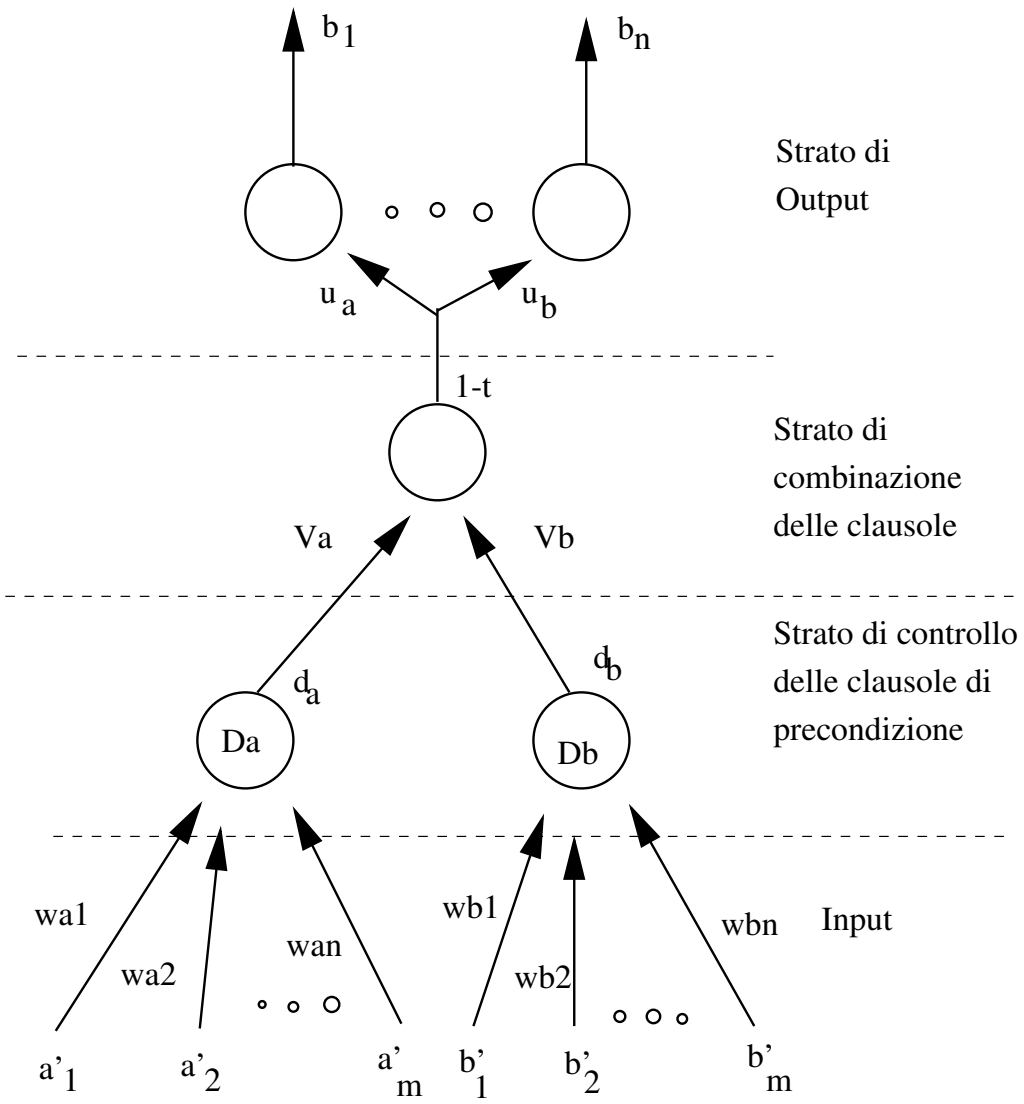
Sistemi di inferenza fuzzy realizati neuralmente possono essere utilizzati per estrapolare relazioni complesse tra le distribuzioni di possibilta' dell'antecedente e del conseguente di regole fuzzy. Una simile combinazione consente di sviluppare sistemi che mostrino un comportamento adattivo, mantenendo una rappresentazione forte della conoscenza. Le regole fuzzy noti vengono utilizzate per strutturare opportune reti neurali che implementino dette regole, e tali reti neurali posseggono quindi una struttura che direttamente rispecchia le regole fuzzy coinvolte, rendendo cosi' possibile capire "cosa" la rete apprenda.

4.3.1 Fuzzy inference networks

Le FUZZY INFERENCE NETWORKS sono reti multlayer feedforward che riproducono l'azione di una regola di inferenza fuzzy. Esse sono composte da piu' sottoreti, ognuna delle quali e' una struttura di base rappresentate una regola fuzzy nella forma: *Se* X_1 *e'* A_1 *AND* X_2 *e'* A_2*... AND* X_n *e'* A_n *allora y e'* B.

La distribuzione di possibilta' delle variabili e degli insiemi nelle precondizioni viene campionata, ovvero rappresentanta da un numero finito di elementi dalla sua distribuzione di possibilta' [58]. In altri termini detto X_i un insieme "quantizzato", esso viene espresso come $X_i = \{x_{i1}, \ldots, x_{in}\}$. Sia allora data una regola con due soli antecedenti : *Se* X_1 *e' A*$_1$ *AND* X_2 *e' B allora y e' C.* e siano forniti i fatti "X_1 *e' A'*" e "X_2 *e' B'* ". Questi indurranno due distribuzioni di possibilita' quantizzate come input, rispettavimante pari a $\prod_{X_1} = \{a_1, \ldots, a_n\}$ per la variabile X_1, e $\prod_{X_2} = \{b_1, \ldots, b_n\}$ per la variabile X_2. Di seguito viene fornita una rete che calcola una regola con due soli antecedenti. Regole con piu' antecedenti non verranno qui trattate, ma e' facilmente intuibile come tali reti (e relative proprieta') possano essere estese.

[58] Di fatto cio' gia' aviene se l'universo del discorso e' discreto



Il primo strato nascosto, detto STRATO DI CONTROLLO DELLE CLASUSOLE DI PRECONDIZIONE, genera una misura del disaccordo tra gli antecedenti A e B e i fatti "rappresentati" da X_1 , X_2 (nel senso che tali variabili contengonono la distribuzione di possibilta' successiva agli assegnamenti "X_1 *e' A'*"e "X_2 *e' B'* " , rappresentanti i fatti). Sono possibili due implementazioni per questo strato.

In una prima implementazione i pesi relativi agli input sono il complemento dei valori di possibilta' desiderati nelle precondizioni. Ovvero, per la prima clausola "X_1 e' A" si ha: $w_{ai} = \overline{a_i} = 1 - a_i$[59] . Cosi' si nota che se gli

[59]Senza apice sono indicati i valori di A (e corrispettivi quantizzati a_i) dell' insieme

insiemi si allontano dalla distribuzione desiderata, il net-input del neurone relativo alla data precondizione tendera' a salire (si avvicina a 1).[60].

Quindi il neurone D_i , attraverso il suo net-input e la sua funzione di attivazione calcola una misura di disaccordo tra la distribuzione fattuale e la distribuzione desiderata. in particolare un neurone OWA puo' calocalre la seguente fuzione:

- $d_k^1 = \max_j \left\{(1 - a_j) * a'_j\right\}$

Estendendo le caratteristiche dei neuroni, considerando funzioni di net-input piu' complesse, o aggiungendo un ulteriore semplice strato di input[61] si potrebbe anche avere

- $d_k^2 = \max_j \left\{\min\left\{(1 - a_j), a'_j\right\}\right\}$

Una seconda implementazione dello strato di controllo delle clausole di precondizione utilizza come pesi una campionatura della distribuzione di possibilta' considerata, ovvero, per la prima clausola "X_1 e' A" si ha: $w_{ai} = a_i$. Considerando anche qui un neurone piu' elaborato, o un ulteriore strato di input, si consideracome funzione calcolata dal neurone D_i (e da un eventuale strato sottostante) :

- $d_k^3 = \max_j \left\{\left|a_j - a'_j\right|\right\}$

Nello strato successivo le misure di disaccordo di ogni nodo vengono combinate per dare una misura del disaccordo complessivo tra le clausole di input e i dati di input. I pesi v_i sui link coinvolti rappresentano l'importanza delle varie clausole, possono essere forniti esplicitamente o "appresi" dalla rete.Tale strato computa la funzione:

- $1 - t = max_i \{v_i * d_i\}$

presente nella precondizone, con l'apice sono indicati i valori assunti dalle variabili X_i sulla base degli assegnamenti di probabilita' indotti dai fatti.

[60]Si noti che in input viene effettuato il prodotto tra l'input e il peso della connessione, ovvero viene applicata una particolare t-norma. Piu' in generale si potrebbe introdurre un ulteriore strato di input, in cui ogni neurone riceve un input, e applica una t-norma tra tale input ed il peso ad esso associato.

[61]Si noti che in input viene effettuato il prodotto tra l'input e il peso della connessione, ovvero viene applicata una particolare t-norma. Piu' in generale si potrebbe introdurre un ulteriore strato di input, in cui ogni neurone riceve un input, e applica una t-norma tra tale input ed il peso ad esso associato.

che per il caso in esempio diventa: $1 - t = max\{v_a * d_a, v_b * d_b\}$.

Lo strato di output e' caratterizzato dai pesi u_i definiti come il complemento dei singoli valori di possibilta' della distribuzione di possibilta' desiderata nel conseguente. Ovvero se la conseguenza e' "Y e' C", allora si campiona C e si ha: $u_i = \overline{c_i} = 1 - c_i$. Allora ogni nodo di output da' un valore campinato della distribuzone di C' (conseguenza dedotta) , cosi' calcolato : $c'_i = 1 - u_1(1-t) = 1 - (1-c_i)(1-t) = c_i + t - c_i t$. Nell'insieme lo strato di output fornisce la distribuzione di possibilita' per la conseguenza C' .[62].

Per una rete di inferenza fuzzy possono essere dimostrate lle seguenti proprieta' (si considera una rete con un solo antecedente "X e' A" e un solo conseguente "Ye' C"):

- Se A e' un sottoinsieme nitido dell'universo del dicorso, e se $A' = A$, allora sia ha $Y = C' = C$, in analogia con il modus ponens classico.

- Se A e A' sono sottoinsiemi nitdii dell'universo del discorso, se $A' \neq A$si ha che, se la funzione calcolata dal primo strato e' d^3, il risultato e' sempre SCONOSCIUTO[63], altrimenti il risultato e' SCONOSIUTO se $\overline{A} \cap A' \neq \emptyset$, e' $Y = C' = C$ se $A' \subset A$. Ovvero se si applica d^3 si ha come risulato SCONOSCIUTO ogni volta che gli input non corrispondono alle precondizioni, come nel modus-ponens classico. Applicando invece d^1o d^2 si gode di una proprieta' che permette di dedurre, da *"se X e' rosso Y e' un pomodoro"*, e *"X e' molto rosso"*, che *"Y e' un pomodoro"*.

- Se A e A' , A" sono sottoinsiemi fuzzy dell'universo del discorso, e la rete di inferenza fuzzy usa neuroni d^1o d^2, si puo' dimostrare che, se $A'' \subset A' \subset A$ allora, consiederati gli output B', B" vale: $B \subset B'' \subset B'$. Ovvero, man mano che la premessa si fa piu' specifica, la deduzione si fa meno generale Di contro applicando d^3 si ha che, man mano che la premessa diverge dal la precondizione , la conclusione si fa piu' generale.

4.3.2 Fuzzy aggregation networks

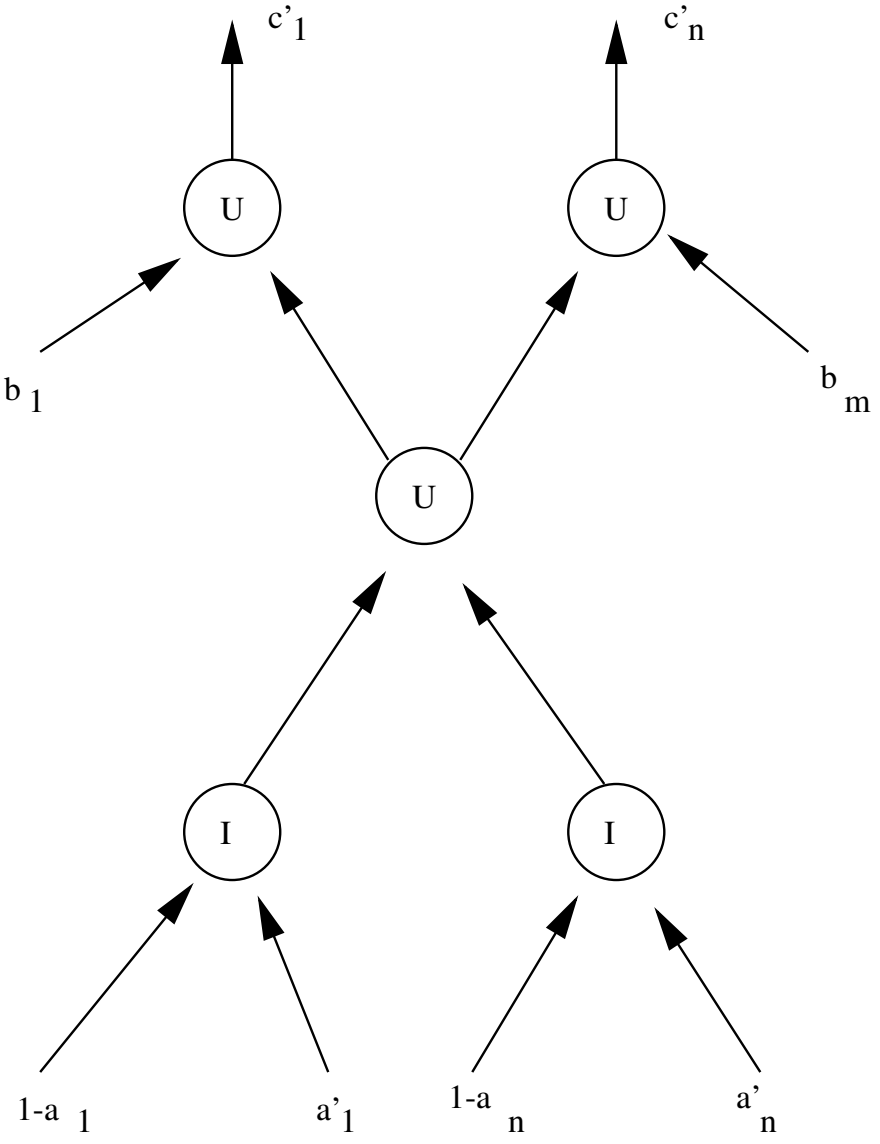
Le FUZZY AGGREGATION NETWORKS consistono in una generalizzazione delle reti di inferenza fuzzy, in cui le fuznioni computate dai nodi possono essere ogni possibile operazione fuzzy di intersezione, unione, aggregazione,

[62]si noti che se t=1, pari alla massima incertezza, la distribuzione di possibilta' di C' e' pari a 1 nell'intero universo del discorso, corrispondente al termine SCONOSCIUTO. Se invece t=0, si ha che $C = C'$.

[63]SCONOSCIUTO, come anche precedentemente presentato, corrisponde ad una distribuzione di possibilita' pari a 1 sull'intero universo del discorso.

eventaulmente parametrizzate. Si hanno cosi' molti piu' parametri apprendibili dalla rete e una maggiore potenzialita' di adattamento della rete stessa.

La struttura base di una rete di aggregzione fuzzy, che ricalca la struttura di una fuzzy inference network, e' mostrata nel seguente schema, per una regola del tipo *"Se X e' A allora Y e' C"* . Con I si indica un neurone che computa una generica fuzione di intersezione, con U di unione. Nella rappresentazione grafica scelta i pesi sono forniti come input ai nodi.



Una rete come quella mostrata, che ricalca la struttura di una fuzzy inference networks, conserva le stesse proprieta' di quest'ultima (neuroni d^1 e

d^2), indipendentemente dalle operazioni di unione o intersezione utilizzate, purche' nella relazione indicata.

4.4 Sistemi neuro fuzzy

Finora si e' visto come singole funzioni fuzzy possano essere implementate mediante reti neurali. Si presenta ora un sistema neuro fuzzy completo. Come prcedentemente visto, nei sistemi fuzzy non si utiliza un ragioanamento fuzzy categorico (cosa possibile cone reti di inferenza fuzzy), ma piuttosto un motore inferenziale, legato ad una base di regole, in cui i meccanismi di inferenza intressati non sempre si basano sul modes ponens generalizzato, e non sempre necessitano di inputs e outputs fuzzy.

Di fatto si preferiscono spesso strutture che, seppur sacrificando un ragionamento piu' fondato, garantiscano doti quali semplicita' di implementazione e adeguatezza allo scopo specifico dell'applicazione in questione (es. controllo).

A maggior ragione una forte componente empirica e una scarsa sistematizzazione si ritrovano nei sistemi neuro fuzzy.

4.4.1 Neural network driven fuzzy reasoning

Si considera qui una realizzazione neurale di un sistema fuzzy utilizzante un motore inferenziale con un ragionamento del terzo tipo o TSK. Mediante l'implementazione neurale si spera di risolvere i problemi legati alla mancanza di apprendimento per le funzioni di appartenenza e le regole di inferenza nei sistemi fuzzy. Lo schema qui proposto prende il nome di Neural Network driven Fuzzy Reasoning ed e' il seguente[64]:

- La s-esima regola della base sara' nella forma: R^s: Se $x = (x_1, \ldots, x_n)$ e' A_s , allora $y_s = f_s(x_1, \ldots, x_n)$, con $s \in 1, 2, \ldots, r$. (ovvero la base contiene r regole, utili ad un ragionamento del terzo tipo).

- I dati di addestramento, nella forma $(x_1, \ldots, x_n, y)_i$ (i identica l'i-esimo gruppo di dati) vengono classificati in r classi (corrispondenti alle r regole) da un algoritmo di apprendimento non supervisionato.

[64]Nella presente esposizone si fornisce un modello semplificato. Il modello completo introduce metodi di ottimizzazione della rete. In particolare vengono scartati, per ogni rete in gioco, gli input non ritenuti significativi. In sostanza si elimina un input a caso, e se questo non incide su una fuznione di costo prefissata, lo si elimina. Per valutare la fuznione di costo (di solito i minimi quadrati) si dividono i dati di addestramneto in dati di addestramento e dati di controllo.

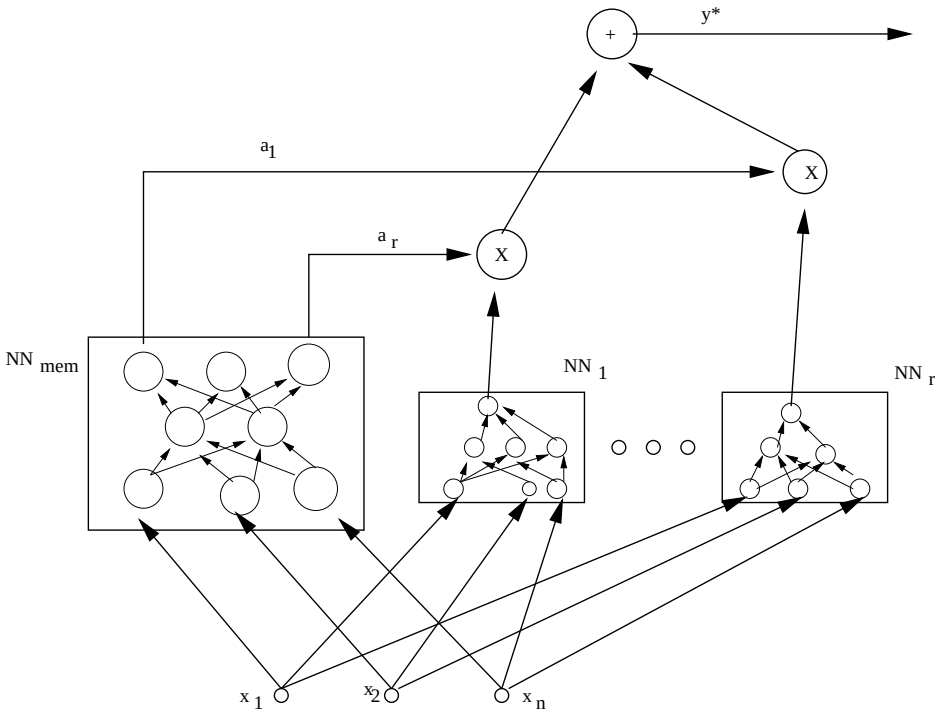
- Una rete, detta NN_{mem} viene addestrata sulla base di tale suddivisone. Dato ad esempio l'iesimo dato $(x_1, \ldots, x_n)_i$, classificato precedentemente nella classe r_j , si applica un apprendimento con supervisone (es. back propagation) in cui vengono forniticome input $(x_1, \ldots, x_n)_i$ e come output un vettore a r dimensioni con valori $r_k = 1$ se $k = j$, $r_k = 0$ altrimenti. In questo modo si addestra la rete a riconoscere il grado di matching tra input e precondizione, ovvero a calcolare la FORZA DI ATTIVAZIONE per ogni regola coinvolta, a seguito di un determinato input. Denominiamo tale segnale a_s (e sara' funzione del dat corrente)[65].

- Vengono addestrate r reti NN_s in modo tale da computare la fuznione $y_s = NN_s(x_1, \ldots, x_n)$. Anche tali reti sono multi layer feedforward e vengono adestrate con l'algoritmo di retropropagazione sui dati input/output relativi alla s-esima regola.

- La struttura complessiva della rtete e' la seguente, dove l'ultimo nodo calcola il valore di controllo: $y^* = \frac{\sum_{s=1}^{r} a_s y_s}{\sum_{s=1}^{r} a_s}$[66].

[65] Si ricorda che esso e' pari alla funzione di appartenenzadi A_i calcolata nell'input, per un sistema fuzzy con ragionamento del terzo tipo.

[66] Ovviamnet tutto questo andreabbe indicizzato nell'istante di tempo, ma non si e' voluta appesantire inutilmente la notazione.


+
y*
a1
X
a r
X
NN mem
NN 1
NN r
x 1
x2
x n

Index

5 Conclusioni

Si e' proposto un percorso attraverso le basi del Soft Computing, partendo da una introduzine alla logica fuzzy e alla teoria della possibilta', prestando attenzione ad evidenziare come ci si muove nel passare da un ragionamento rigoroso ed algoritmico ad uno approssimato. Quindi e' stato presentato un ragionamento fuzzy di tipo categorico, ma si sono anche messe in evidenza le approssimazioni utilizzate nei sistemi reali.

Sono state introdotte le reti neurali, nei loro aspetti generali e si sono visti alcuni tipici algoritmi di apprendimento.

Quindi si 'e mostrato come e perche' le reti neurali e logica fuzzy possano essere effettivamente combinati, e si sono mostrati in particolare i siemi neuro fuzzy. Per qusti sono state introdotte le realizzazioni di base delle funzioni fuzzy con reti neurali, e si e' visto un esmpio piu' completo.

Si e' voluto mostrare come questo connubio fra le due tecnologie possa portare a sistemi in grado di effettuare ragionamente approssimati di alto livello, simili a quelli della mente umana. Quella presentata non e' che una prima introduzione a tali sistemi, la cui progettazione e' peraltro fortemente empirica e una sua piu' dettagliata descrizione non avrebbe potuto prescindere dalla necessita' di riportare dati sperimentali. Inoltre nella progettazione di sistemi reali spesso si inseriscono altre istanze, dovute ai vincoli del problema che si cerca di affrontare, e influenzate dalla disponibilita' di realizzazioni circuitali di determinati sistemi o sottoparti di essi.

Tali sistemi hanno avuto recentemente un forte successo nell'ambito del controllo e dei sistemi esperti, e si e' voluto fornirne una prima introduzione, partendo da lla logica fuzzye quindi proponendo una impostazione ben fondata del modo di tratatare l'informzione incerta.

References

[1] Appunti del corso. Cherubini

[2] Neuro Fuzzy System. Chin-Teng Lin e C.S.George Lee, Prentice Hall

[3] Identificazione dei modelli e controllo adattivo. Bitanti, Pitagora

[4] Sistemi fuzzy. Camarrata, ETASLIBRI

[5] Intelligenza e vita. Mazzetti, Apogeo

[6] Il fuzzy Pensiero. Bart Kosko, Baldini & Castoldi

[7] Appunti On line del corso su sistemi Neuro Fuzzy di ??? address ???