Capítol 3: Xarxes Neurals

1

Soft-Computing

- Sistemes fuzzy
- XN
- Algorismes genètics

Noms alternatius que s'empreen per les XN:
- Connexionisme
- Processament paral.lel distribuït
- Computació neural
- Xarxes adaptatives
- Computació col.lectiva

Contingut

1. Introducció.
2. Fonaments
3. Perceptró
4. XN multicapa feedforward
5. Aplicacions de XN
6. Temes avançats

Inteligencia Artificial. Russell & Norvig, Prentice-Hall, 1994.
Capítol 19.

2

1. Introducció

- El camp de les xarxes neurals constitueix una tècnica de tractament de la informació basada en estudis sobre el funcionament del cervell i del sistema nerviós humà.
- Com funciona el cervell humà ?
  - Incògnita
  - Pensament, memòria consciència
  - Humà

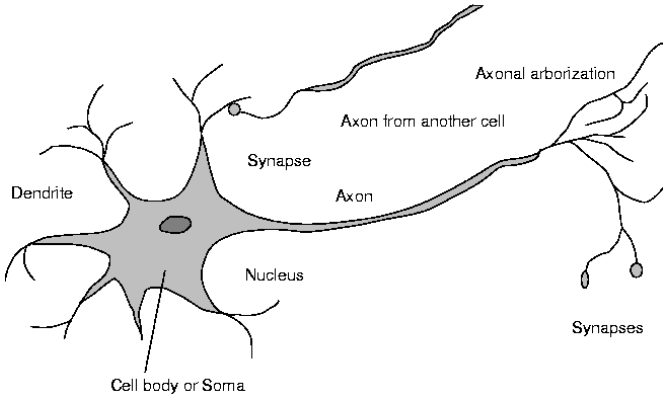
3

* En el cervell resideix el pensament, la memòria, la consicència.

* Humà: Cervell molt gran, cosa que fa pensar que d'això depèn la seva intel.ligència

(excepció: balenes, dofins)

Unitat fonamental: neurona



L'axon pot fer des d'1 cm fins a 1 m !!

Propagació del senyal

- Reacció electromagnètica complexa
  - Transmissió d'una substància química des dela sinapsis que entra a la dendrita
  - Augment o disminució del potencial elèctric del cos de la neurona
  - Quan aquesta variació supera un llindar --> pols electrònic o acció potencial que s'envia a l'axon
  - El pols s'estén en tot l'arbre axonal alliberant substàncies transmissores cap a altres sinapsis

5

Augment del potencial --> sinapsis excitadores

Disminució del potencial --> sinapsis inhibidores

- Plasticitat: canvis a llarg termini en les connexions en resposta a un estímul (--> aprenentatge)
- Còrtex: part externa del cervell on es processa més informació
- Diferents àrees del cervell tenen funcionalitats diferents.
  - Desconeixement de la correspondència
- Desconeixement de l'emmagatzematge de la informació de forma individual
- Però si es coneix que:

Una col.lecció de cel.lules simples proporciona pensament, acció i consciència

6

Cervell humà vs. Computadores

	Computadores	Cervells
Unitats còmput	10^5 portes	10^{11} neurones
Unitats emmagat.	10^8 bits RAM 10^{10} bits disc	10^{11} neurones 10^{14} sinapsis
Temps de cicle	10^{-8} segons	10^{-3} segons
Amplada de banda	10^9 bits/seg	10^{14}/bits/seg
Actualització neural per segon	10^5	10^{14}

7

Pero:

- El cervell huma evoluciona molt le ntament
- Les computadores evolucionen molt ràpidament

Dades de Rusell&Norvig, 1995.

Actualement estem parlant d'ordinadors de capacitat de disc molt superior (2 Gigas, on 1Giga=10**12 bits).

Principals diferències:

- Grau de paral.lelisme en el procés
- Tolerància a fallides
- Degradació gràcil
- Mecanisme d'aprenentatge inductiu

8

Paral.lelisme:

Ordinadors --> 1 sola CPU

1 model neural s'executa seqüencialment en un ordinador i requereix de molts cicles de CPU per saber si s'activarà, mentre que en el cervell totes les neurones s'activen simultàneament.

--> La rapidesa de processament del cervell és molt superior

Ex.: Reconeixement d'una cara.

Tolerància a fallides

--> Un error de hardware pot deixar de funcionar l'ordinador

--> Les cel.lules del cervell es van morint a poc a poc sense que el cervell deixer de funcionar de cop

Degradació gràcil

--> Davant de nous inputs, els ordinadors no saben què fer si no se'ls programa

--> El cervell és capaç de donar respostes a inputs no coneguts. Quan més desconeguts la resposta serà pitjor.

Aprenentatge:

Important des del punt de vista psicològic.

2. Fonaments de les X.N.
Components

- Una X.N. està composada d'unitats de processament (equivalents a les neurones) organitzades de diferents formes tot formant l'estructura de la xarxa.
- Components
  - Nodes o unitats, algunes connectades a l'entorn (entrada/sortida)
  - Enllaços: d'entrada i sortida a cada unitat
  - Pesos associats a cada enllaç
    - Primer nivell de memòria a llarg termini en la XN
    - Aprenentatge: actualització dels pesos
      - Objectius: ajustar el comportament d'entrada/sortida al que s'espera de l'entorn.
- Nivell d'activació:
  - cada unitat en té un
  - mètode de calcular el nivell d'activació d'acord amb les entrades i sortides
  - Computació local: cada neurona calcula el grau d'activació en funció de les entrades de les seves veïnes, sense control global

9

Entrades:

- Cada entrada correspon a un atribut individual. El valor numèric de l'atribut és l'entrada a la xarxa.

- Les xarxes neurals només poden processar números. Si per resoldre un problema la xarxa ha de treballar amb valors qualitatius o imatges, aquests s'han de preprocessar fins a convertir-los en un format numèric equivalent.

- Exemples d'entrades a xarxes neurals poden ésser: valor dels pixels de caràcters o d'altre tipus de gràfics, imatges o veu digitalitzades, senyals digitalitzats provenint de sensors d'un sistema de monitorització, etc.

Sortides:

- La sortida de la xarxa és la solució del problema. Per exemple, en el cas d'utilitzar una XN per a reconeixement de caràcters la sortida seria el valor del caràcter reconegut.

Construcció d'una XN per a desenvolupar una tasca

- # de neurones que calen
- Com s'han de col.locar les neurones per formar la xarxa
- Procés:
  - Inicialitzar els pesos
  - Obtenir els pesos definitius mitjançant un algorisme d'aprenentatge
    - Aprenentatge supervisat
    - Aprenentage no supervisat

10

Aprenetatge Supervistat

- Utilitza un conjunt d'entrades per les quals les sortides (desitjades) són conegudes.
- La diferència entre la sortida actual i la desitjada s'utilitza per a calcular les correccions dels pesos de la XN.
- Exemples de d'aquest tipus d'aprenentatge són la Xarxa de Hopfield i el Backprogragation

11

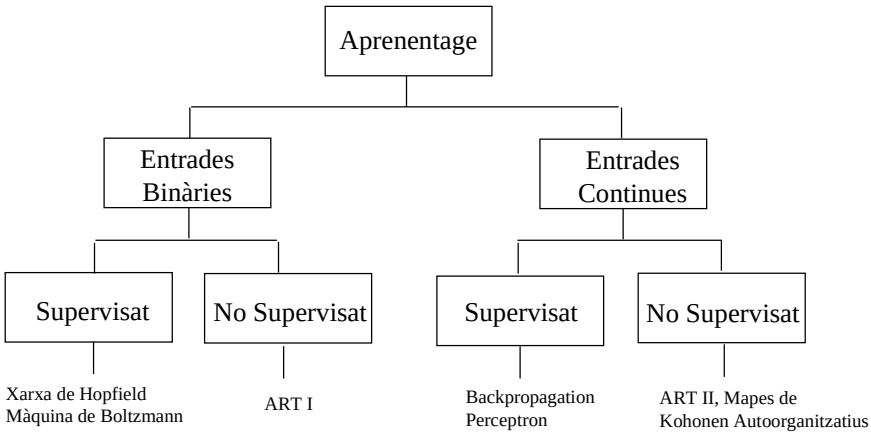
Aprenetatge No Supervistat

- Només es mostra una entrada a la XN. La xarxa s'autoorganitza, és a dir, s'organitza internament de manera que els elements de processament interns responen estratègicament a diferents entrades.
- No es proporciona cap coneixement quines classificacions són correctes, així com les classificacions que obté la XN poden o no tenir significat per la persona que l'està entrenant.
- Encara que el nombre de categories en que la xarxa classifica les sortides es pot controlar variant determinats paràmetres del model. De tota manera, la persona que ha entrenat la XN ha d'examinar les categories finals el significat a cadascuna d'elles i determinant la utilitat dels resultats obtinguts.
- S'usen per identificar grups de dades
- Exemples d'aquest aprenentatge són ART (Adaptive Resonance Theory) i els mapes autoorganitzatius de Kohonen

12

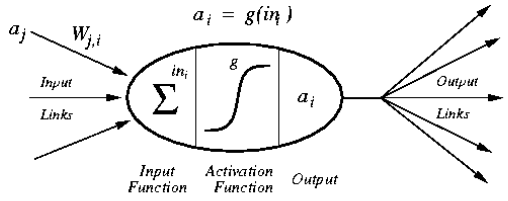
En aquest curs ens centrarem en aprenentatge SUPERVISTAT.

Classificació dels Algorismes d'Aprenentatge



13

Unitat simple



$$in_i = \sum_j W_{j,i} a_j \qquad a_i = g(in_i)$$

4

Càlcul a dos nivells:

- Lineal (ini): funció d'entrada --> suma ponderada dels inputs

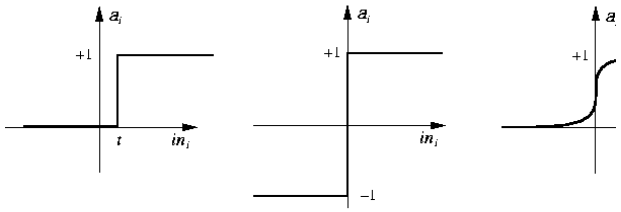
- No lineal: funció d'activació
  --> Transforma la suma ponderada en el valor final d'activació de la unitat
  --> Acostuma a ser la mateixa per a tots els nodes de la xarxa

Notació

Notation	Meaning
a_i $\mathbf{a}_i$	Activation value of unit i (also the output of the unit) Vector of activation values for the inputs to unit i
g g'	Activation function Derivative of the activation function
Err_i Err^e	Error (difference between output and target) for unit i Error for example e
I_i $\mathbf{I}$ $\mathbf{I}^e$	Activation of a unit i in the input layer Vector of activations of all input units Vector of inputs for example e
in_i	Weighted sum of inputs to unit i
N	Total number of units in the network
O O_i $\mathbf{O}$	Activation of the single output unit of a perceptron Activation of a unit i in the output layer Vector of activations of all units in the output layer
t	Threshold for a step function
T $\mathbf{T}$ $\mathbf{T}^e$	Target (desired) output for a perceptron Target vector when there are several output units Target vector for example e
$W_{j,i}$ W_i $\mathbf{W}_i$ $\mathbf{W}$	Weight on the link from unit j to unit i Weight from unit i to the output in a perceptron Vector of weights leading into unit i Vector of all weights in the network

5

Funcions matemàtiques per g



$$step_t(x) = \begin{cases} 1 & x \geq t \\ 0 & x < t \end{cases} \quad sign(x) = \begin{cases} +1 & x \geq 0 \\ -1 & x < 0 \end{cases} \quad sigmoid(x) = \frac{1}{1+e^{-x}}$$

16

Cada funció --> un model diferent

STEP

El resultat de la funció és 1 quan la suma ponderada de l'entrada supera t
Altrament retorna 0.

Motivació biològica:

1--> pols en l'axon

0 --> no pols

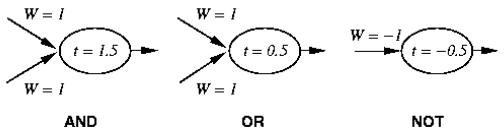
Llindar: també anomenat BIAS.

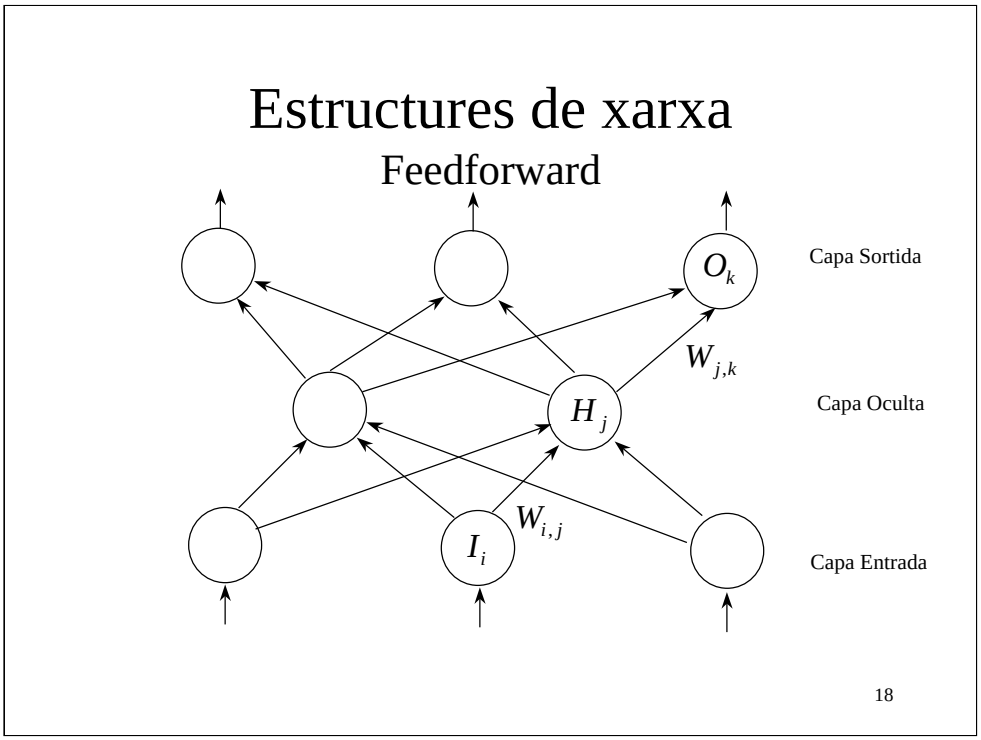
Convé tractar el llindar com un pes d'input addicional, de forma que l'aprenentatge s'aplica a pesos sense distingir si es tracta d'entrades o llindars.

El valor de l'input pel "bias" és sempre 1 (sempre està activat).

SIGMOIDALl: la més usada

Equivalènica de comportament amb portes lògiques





Cada estructura proporciona propietats computacionals diferents.

Feedforward:

- Enllaços unidireccionals (no hi ha cicles)

- Normalment estructurades per nivells: cada unitat està enllaçada exclusivament a unitats del nivell següent

No hi ha enllaços entre unitats d'un mateix nivell

- Els càlculs flueixen de forma uniforme des de l'entrada cap a la sortida (carència de cicles)

--> No hi ha feedback en els càlculs

--> La XN calcula una funció simple dels inputs depenent dels valors dels pesos

- Aplicacions:

Reflexius/adaptatius simples

Comportaments simples dins d'un agent més complex

Funció no lineal:

Aprende els pesos = Ajustar els paràmetres a les dades del conjunt d'aprenentatge --> punt de vista de l'estadística: regressió no lineal

Estructures de xarxa

Recurrent

- Els enllaços s'estructuren en tipologies arbitràries
- És realment com funciona el cervell, ja que permet pensar en memòria a curt termini: hi ha informació emmagatzemada en estats interns de la xarxa
- Permeten el disseny de sistemes més complexos
- Són inestables, oscil.len i mostren un caràcter caòtic.
- L'aprenentatge és difícil.

X. De Hopfield

Màquines de Boltzmann

19

X. de Hopfield

- Usen connexions bidireccinals amb valors simètrics

$$W_{i,j} = W_{j,i}$$

- Totes les unitats són al mateix temps unitats d'entrada i de sortida
- La funció d'activació és sign(x)
- Té una memòria associativa.
- Nombre d'exemples que pot emmagatzemar: 0.138N

20

Memòria associativa:

--> Després de realitzar l'aprenentatge amb un conjunt d'exemples, un nou estímul produirà una sortida equivalent a l'exemple més semblant.

Exemple:

Si el conjunt són fotografies

L'estímul és una part d'una fotografia

--> La sortida és la fotografia a la qual pertany la part

Observar que la fotografia no està emmagatzemada "tal qual" en memòria sinó que cada pes és una codificació parcial de cada fotografia

N: # d'unitats

Màquines de Boltzmann

- Pesos simètrics

$$W_{i,j} = W_{j,i}$$

- Tenen unitats ocultes (*hidden*)
- Utilitzen una funció d'activació estocàstica

21

Funció d'activació estocàstica:

--> Simulen estats de transició = cerca "simulated annealing" en el cas que aproxima millor el conjunt d'aprenentatge

--> Simulen un cas particular de xarxes de creences (belief networks)

Estructures de xarxa
Òptima

- El punt dèbil de les XN és que tenen una estructura fixa, determinada per una autoritat externa.
- El nombre d'unitats en cada nivell pot créixer de forma exponencial amb el nombre d'inputs
- El problema de trobar l'estructura adequada es pot plantejar com un problema de cerca
  - Algorismes genètics
  - Mètodes "hill-climbing"
- Es fan servir tècniques de validació creuada per determinar quan s'ha trobat la XN de mida adequada

22

Topologial: per què?

--> Una XN petita pot ser incapaç de representar la funció que desitgem.

--> Una XN massa gran pot emmagatzemar tots els exemples però no generalitzarà prou bé els inputs per tractar casos no vistos amb anterioritat.

Overfitting: massa paràmetres (pesos) en el model.

Nombre d'unitats:

XN feedforward amb 1 capa oculta --> funcions continues dels inputs

XN feedforward amb 2 capes ocultes --> qualsevol funció

Falta teoria per decidir sobre el nombre d'unitats

Problema de cerca:

- Al plantejar-lo com a problema de cerca, cada cop que avaluem un estat significa executar el mecanisme d'aprenentatge de la XN --> Procés molt costós en temps de CPU

- Alternatives de refinament dels mètodes de HC:

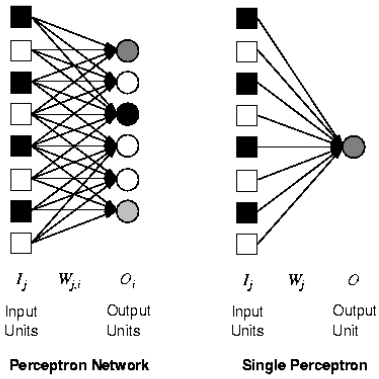
  --> Començar per una XN gran i fer-la cada cop més petita

  --> Començar per una XN petita i fer-la cada cop més gran

Tècniques de validació creuada: es separa una part de les dades d'aprenentage per usar-les de test i verificar el grau de predicció de la hipòtesis induïda per la resta de dades.

3. Perceptró

- XN d'un nivell, feed forward



$$O = Step_0(\sum_j W_i I_j)$$

23

1950's

Dibuix de la dreta: perceptró simple.

Dibuix de l'esquerra: xarxa de peceptrons.

Cada unitat d'input és independent de les altres --> Estudiarem perceptrons simples de forma aïllada

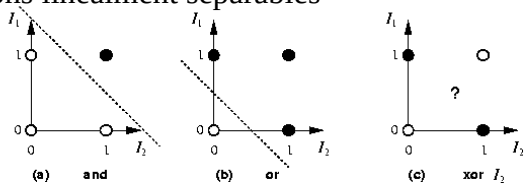
Perceptró: Què poden representar?

- Funcions booleanes simples
- Funció major-que

$$\left\{\begin{matrix} 1 & \text{si la majoria dels inputs són 1} \\ 0 & \textit{altrament} \end{matrix}\right\}$$

$$W_j = 1 \qquad t = \frac{n}{2}$$

- Funcions linealment separables



Funcions booleanes simples: AND, OR,NOT.

I la seva compinació si es crea una xarxa de diferents capes.

Funció major-que:

Notar que per n entrades requeriríem un arbre de decisió de O(2**n) nodes!

Funcions linealment separables: existeix una línia que permet separar les solucions (1) e les no solucions (0).

[Les solucions estan posades en boles negres i les no solucions en boles blanques]

En l'exemple es pot veure que no es pot representar XOR.

--> MOLT LIMITAT: hi han poques funcions linealment separables

Aprenentatge de funcions linealment separables

- XN inicials: pesos amb valors aleatoris entres -0.5 i 0.5
- Actualització dels pesos diverses vegades
- Època: actualització de tots els pesos per a tots els exemples.
- Algorisme:

  Per a cada exemple e fer

  Calcular els outputs (O)

  Valors observats (T)

  Actualitzar els pesos en la XN segons e,O,T

  fins que es prediguin correctament tots els exemples o s'arribi a un criteri d'aturada

25

Regla d'actualització del perceptró

Err = T-O

Si Err > 0, llavors augmentar W_j

Si Err < 0, llavors disminuir W_j

Actualització:

$$W_j = W_j + \alpha I_j Err$$

on αés la velocitat d'aprenentatge

26

REcordar que O=WI

1960: Frank Rosenblatt

Teorema de convergència: el sistema convergirà a un conjunt de pesos que representaran correctament els exemples

1967: Minsky & Papert

Limits dels perceptrons

Perceptró
Convergència

- El mètode del perceptrò està realitzant una cerca del gradient descendent en l'espai de pesos.
- Els mètodes de cerca de gradient tenen problemes de mínims locals
- Però l'espai de pesos no té mínims locals, per la qual cosa sempre trobarà la solució.

$\Rightarrow$Convergència

27

Mètode del gradient

= Hill climbing quan la funció d'avaluació es mesura en funció del cost enlloc de la qualitat o altres propietats

Es tracta de moure sempre en la direcció del valor creixent

No cal mantenir un arbre de cerca: només l'estat i la seva avaluació

Problemes:

-> Màxims locals

-> Pla: qual la funció d'avaluació es plana dins de l'espai, l'algorisme tindrà un comportament aleatori

->Crestes pronunciades --> oscil.lacions

Perceptró vs. Arbres de decisió

Funció major-que

- És una funció linealment separable --> El perceptró es comporta millor que l'arbre de decisió generat per un mecanisme inductiu

28

Perceptró vs. Arbres de decisió

Esperem una taula en el restaurant?

Alternativa	Si hi ha l'alternativa d'algun restaurant a prop
Bar	Si hi ha algun bar o àrea d'espera
Fri/Sat	Cert en Divendres i Dissabtes
Fam	Estem afamats
Patrons	Quanta gent hi ha al restaurant (ningu, algu, ple)
Preu	$,$$,$$$
Pluja	Si plou fora
Reserva	Si hem fet una reserva
Tipus	Francès, Italià, Tailandès, Burguer
Est	Temps estimat d'espera (minuts): 0-10,10-30,30-60,>60

29

Inputs del problema

Perceptró vs. Arbres de decisió

Esperem una taula en el restaurant?

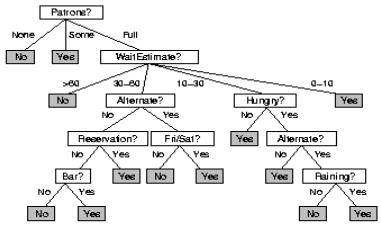
Example	Attributes										Goal
	Alt	*Bar*	*Fri*	*Hun*	*Pat*	*Price*	*Rain*	*Res*	*Type*	*Est*	*WillWait*
X_1	*Yes*	*No*	*No*	*Yes*	*Some*	$$$	*No*	*Yes*	*French*	*0–10*	*Yes*
X_2	*Yes*	*No*	*No*	*Yes*	*Full*	$	*No*	*No*	*Thai*	*30–60*	*No*
X_3	*No*	*Yes*	*No*	*No*	*Some*	$	*No*	*No*	*Burger*	*0–10*	*Yes*
X_4	*Yes*	*No*	*Yes*	*Yes*	*Full*	$	*No*	*No*	*Thai*	*10–30*	*Yes*
X_5	*Yes*	*No*	*Yes*	*No*	*Full*	$$$	*No*	*Yes*	*French*	*>60*	*No*
X_6	*No*	*Yes*	*No*	*Yes*	*Some*	$$	*Yes*	*Yes*	*Italian*	*0–10*	*Yes*
X_7	*No*	*Yes*	*No*	*No*	*None*	$	*Yes*	*No*	*Burger*	*0–10*	*No*
X_8	*No*	*No*	*No*	*Yes*	*Some*	$$	*Yes*	*Yes*	*Thai*	*0–10*	*Yes*
X_9	*No*	*Yes*	*Yes*	*No*	*Full*	$	*Yes*	*No*	*Burger*	*>60*	*No*
X_{10}	*Yes*	*Yes*	*Yes*	*Yes*	*Full*	$$$	*No*	*Yes*	*Italian*	*10–30*	*No*
X_{11}	*No*	*No*	*No*	*No*	*None*	$	*No*	*No*	*Thai*	*0–10*	*No*
X_{12}	*Yes*	*Yes*	*Yes*	*Yes*	*Full*	$	*No*	*No*	*Burger*	*30–60*	*Yes*

Taula amb els exemples.

Les dades són a la columna "Attributes" i la resposta esperada és a "Goal WillWait".

Perceptró vs. Arbres de decisió

- Solució arbres de decisió



31

Perceptró vs. Arbres de decisió

- Solució1 XN: codificació local

  Usar una única unitat d'input i associar valors diferents a cada valor discret

  ningú = 0.0 algú=0.5 ple=1.0

- Solució 2 XN: codificació distribuïda

  Usar una unitat d'input per a cada valor, posant en ON l'input corresponent al valor actual

32

Cal adequar els inputs ja que aquests es presenten en forma d'atributs multivaluats (que són fàcilment processables pels algorismes que indueixen l'arbre de decisió) mentre que les xarxes neurals només poden tractar números

--> Millors resultats en arbres de decisió.

4. Xarxes multicapa feedforward
Backpropagation

- És el mètode d'aprenentatge més popular per una XN multicapa
- Mètodes
  - Es passen els exemples per la xarxa
  - Si l'output és correcte, no es fa res
  - Altrament s'ajusten els pesos
    - Calcular la diferencia entre les unitas de sortida i la esperada
    - Començant pel nivell més extern, repartir l'error per a cada nivell de la xarxa i fins que s'arribi a la darrera capa oculta
      - Propagar l'error al nivell anterior
      - Actualitzar els pesos entre els dos nivells

33

Cap de les propietats dels perceptron es pot estendre a les XN multicapa.

--> Són xarxes poc eficients

--> No hi ha garantia de convergència cap a un òptim global

... però funcionen.

Backpropagation:

1969 Bryson & Ho

Ignorat fins als 80's per la decadència/decepció i pels límits tecnològics

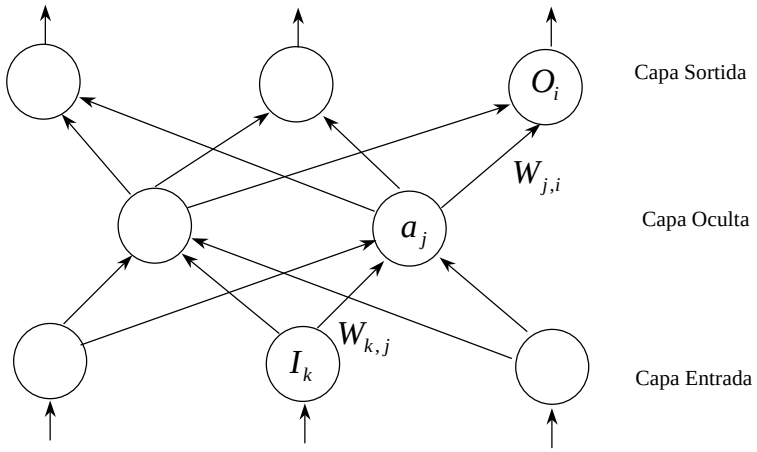
Pot resoldre el problema de decisió del restaurant amb 10 unitats d'entrada (hi ha 10 inputs). El problema és determinar el nombre d'unitats ocultes

Ajust de pesos: es tracta de determinar el grau de culpabilitat d'un error i dividir-lo entre els pesos que hi contribueixen.

En el perceptró és fàcil perquè només hi ha 1 pas entre la sortida i l'entrada.

En una XN multicapa hi ha molts pesos connectant cada entrada amb la sortida.

Backpropagation



34

Backpropagation: pas1

- Objectiu: minimitzar l'error entre el resultat desitjat i l'obtingut
- Pas 1: En la capa de sortida

$$Err_i = (T_i - O_i)$$

$$W_{i,j} = W_{j,i} + \alpha\, a_j Err_i g'(in_i)$$

- Si anomenem

$$\Delta_i = Err_i g'(in_i) \qquad \text{delta}$$

- Llavors

$$W_{j,i} = W_{j,i} + \alpha\, a_j \Delta_i$$

Regla delta

35

Es fa bàsicament el mateix que en el cas del perceptró, però començant per la unitat d'activació més externa.

Regla delta: També anomenada regla del mínim errror quadràtic.

Backpropagation: pas2

- En la capa oculta s'ha de definir una quantitat similar a l'error dels nodes de sortida
- Cada node ocult és responsable, en alguna fracció, de l'error Δ_i dels nodes de sortida als quals està connectat
- Cada Δ_i es divideix d'acord amb la relació de connexió entre el node ocult i l'output i el resultat és el valor d'error per el node ocult: Δ_j

$$\Delta_j = g'(in_i)\sum_i W_{j,i}\Delta_i$$

- La regla d'actualització dels pesos és:

$$W_{k,j} = W_{k,j} + \alpha\ I_k \Delta_j$$

36

Amb això queda acabat els detalls de l'algorisme.

Backpropagation

- Proporciona una forma de dividir el càlcul del gradient entre les unitats, de forma que un canvi en cada pes es pot calcular a través de la unitat al que pertany només usant informació local

  --> Permet la paral.lelització

  --> És plausible pensar-ho com a mecanisme d'aprenentatge biològic.
- Normalment s'usa la funció sigmoidal

37

La funció signe i step no es poden derivar.

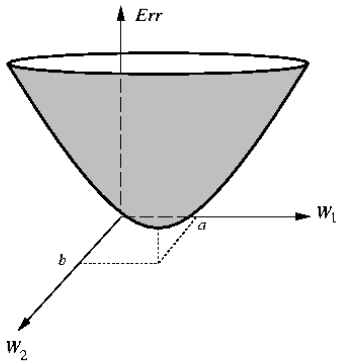
A més, la funció sigmoidal cumpleix que:

$g'=g(1-g)$

per la qual cosa es necessita poc esforç per calcular la derivada

Perceptró

Mètode del gradient descendent



Les equacions usades en el mètode de Backpropagation tenen una interpretacio com a metode de gradient descendent en l'espai de pesos, aplicat a la superficie de l'error.

Backpropagation.
Propietats.

- Expressivitat
- Eficiència computacional
- Generalització
- Sensibles al soroll
- Transparència
- Coneixement anterior

1) Expressivitat

XN careixen de l'expressivitat que té la representació lògica

Adequades per inputs/outputs continuus

Una XN multicapa pot representar qualsevol funció d'un conjunt d'atributs amb un nombre relativament petit d'unitats ocultes.

(2**n)/n per representar funcions booleanes de n inputs

O(2**n) pesos

Mentre que necessitariem 2**n botsen un entorn clàssic

El problema està en conèixer el nombre d'unitats; el disseny de XN és una incògnita.

2) Eficiencia computacional

Teoria --> Depèn del temps que es requereix per entrenar la XN de forma que s'ajusti al conjunt d'exemples.

n exemples, |W| pesos --> Temps d'una època: O(n|W|)

En general és un temps exponencial en funció del nombre d'inputs

Pràctica --> El temps de convergència és variable

--> Problemes de mínims locals (--> usar tècniques de "simulated annealing")

3) Generalització

Generalitzen bé per funcions per les quals "funcionen bé" (i.e. funcions en les quals les interaccions entre inputs no estan molt inbricades i l'output varia sensiblement respecte de l'input).

Han funcionat bastant bé en un ampli conjunt de problemes.

4) Sensibles al soroll

XN estan fent regressió no lienal --> són tolerants al soroll

El valor de certesa de la sortida és desconegut --> xarxes de creences

5) Transparència

Són caixes negres. No poden explicar el "perquè" d'un valor de sortida.

6) Coneixement anterior

Topologia de la xarxa (p.ex., en tractament d'imatges s'acostuma a connectar pixels veins entre les unitats de la primera capa)