



UNIVERSITATEA BABEȘ-BOLYAI
Facultatea de Matematică și Informatică



INTELIGENȚĂ ARTIFICIALĂ

Sisteme inteligente

Sisteme care învață singure

– rețele neuronale artificiale –

Laura Dioșan

Sumar

A. Scurtă introducere în Inteligența Artificială (IA)

B. Rezolvarea problemelor prin căutare

- Definirea problemelor de căutare
- Strategii de căutare
 - Strategii de căutare neinformate
 - Strategii de căutare informate
 - Strategii de căutare locale (Hill Climbing, Simulated Annealing, Tabu Search, Algoritmi evolutivi, PSO, ACO)
 - Strategii de căutare adversială

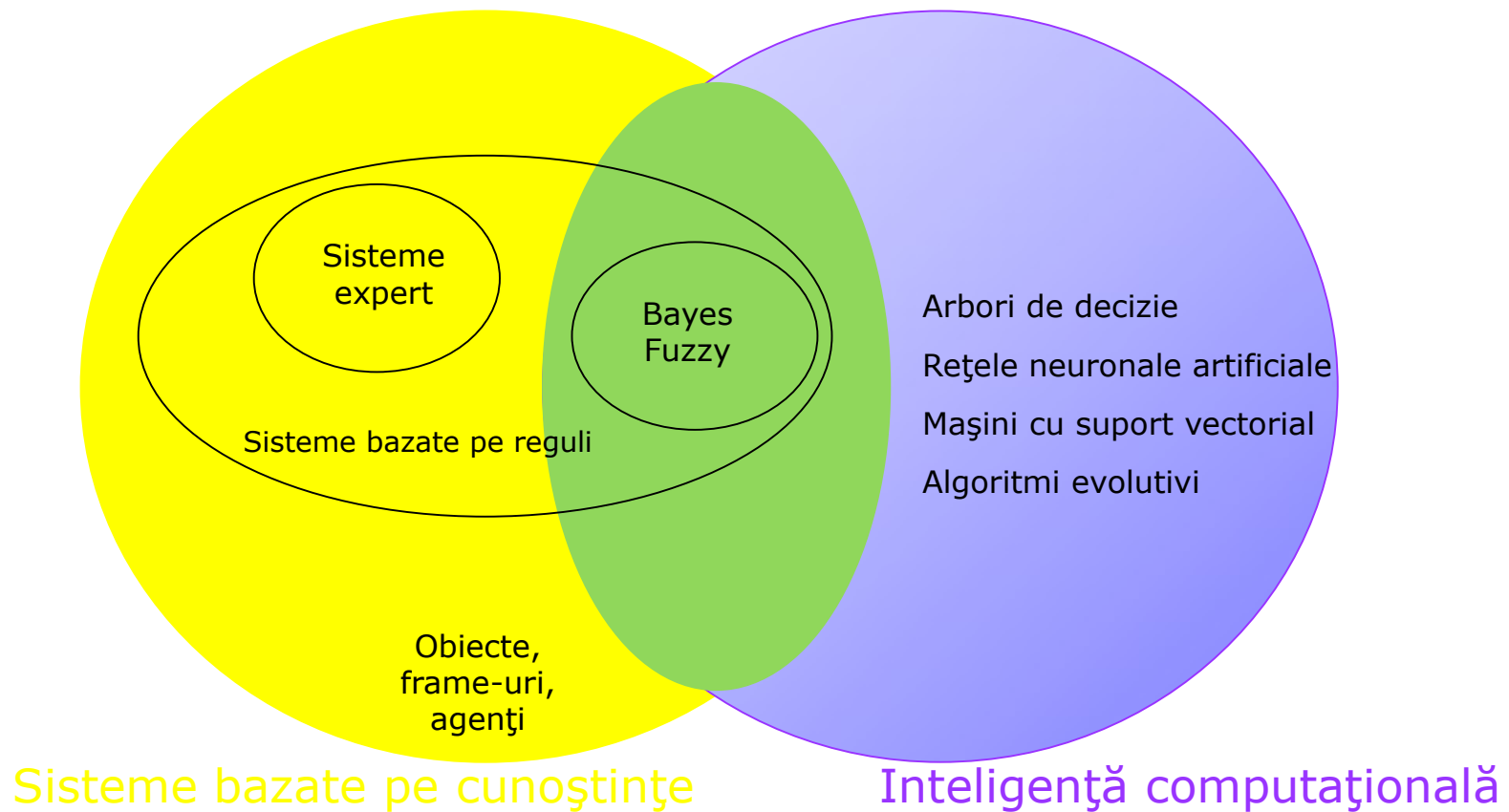
C. Sisteme inteligente

- Sisteme care învață singure
 - Arbori de decizie
 - Rețele neuronale artificiale
 - Mașini cu suport vectorial
 - Algoritmi evolutivi
- Sisteme bazate pe reguli
- Sisteme hibride

Materiale de citit și legături utile

- ❑ Capitolul VI (19) din *S. Russell, P. Norvig, Artificial Intelligence: A Modern Approach, Prentice Hall, 1995*
- ❑ capitolul 8 din *Adrian A. Hopgood, Intelligent Systems for Engineers and Scientists, CRC Press, 2001*
- ❑ capitolul 12 și 13 din *C. Groșan, A. Abraham, Intelligent Systems: A Modern Approach, Springer, 2011*
- ❑ Capitolul V din *D. J. C. MacKey, Information Theory, Inference and Learning Algorithms, Cambridge University Press, 2003*
- ❑ Capitolul 4 din *T. M. Mitchell, Machine Learning, McGraw-Hill Science, 1997*

Sisteme inteligente



Sisteme inteligente – SIS – Învățare automată

□ Tipologie

- În funcție de experiența acumulată în timpul învățării:
 - SI cu învățare supervizată
 - SI cu învățare nesupervizată
 - SI cu învățare activă
 - SI cu învățare cu întărire

- În funcție de modelul învățat (algoritmul de învățare):
 - Arbori de decizie
 - **Rețele neuronale artificiale**
 - Algoritmi evolutivi
 - Mașini cu suport vectorial
 - Modele Markov ascunse

Sisteme inteligente – SIS – RNA

□ Rețele neuronale artificiale (RNA)

- Scop
- Definire
- Tipuri de probleme rezolvabile
- Caracteristici
- Exemplu
- Proiectare
- Evaluare
- Tipologie

Sisteme inteligente – SIS – RNA

□ Scop

- Clasificare binară pentru orice fel de date de intrare (discrete sau continue)

- Datele pot fi separate de:

- o dreaptă $\rightarrow ax + by + c = 0$ (dacă $m = 2$)
- un plan $\rightarrow ax + by + cz + d = 0$ (dacă $m = 3$)
- un hiperplan $\sum a_i x_i + b = 0$ (dacă $m > 3$)

- Cum găsim valorile optime pt. a, b, c, d, a_i ?

- Rețele neuronale artificiale (RNA)
- Mașini cu suport vectorial (MSV)

- De ce RNA?

- Cum învață creierul?

Sisteme inteligente – SIS – RNA

□ Scop → De ce RNA?

- Unele sarcini pot fi efectuate foarte ușor de către oameni, însă sunt greu de codificat sub forma unor algoritmi
 - Recunoașterea formelor
 - vechi prieteni
 - caractere scrise de mână
 - vocea
 - Diferite raționamente
 - conducerea autovehiculelor
 - cântatul la pian
 - jucarea baschetului
 - înotul
- Astfel de sarcini sunt dificil de definit formal și este dificilă aplicarea unui proces de raționare pentru efectuarea lor

Sisteme inteligente – SIS – RNA

□ Scop → Cum învață creierul?

■ Creierul uman - componentă

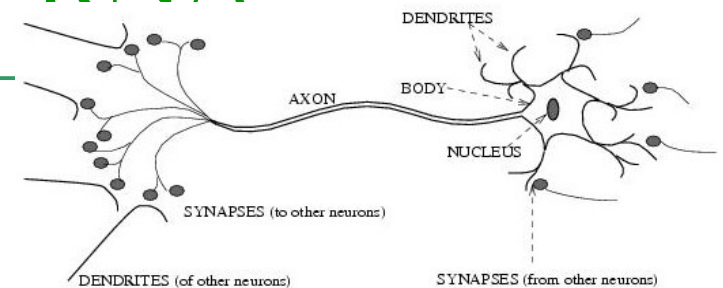
- Aproximativ 10.000.000.000 de neuroni conectați prin sinapse

□ Fiecare neuron

- are un corp (soma), un axon și multe dendrite
- poate fi într-una din 2 stări:
 - activ – dacă informația care intră în neuron depășește un anumit prag de stimulare –
 - pasiv – altfel

□ Sinapsă

- Legătura între axon-ul unui neuron și dendritele altui neuron
- Are rol în schimbul de informație dintre neuroni
 - 5.000 de conexiuni / neuron (în medie)
- În timpul vieții pot să apară noi conexiuni între neuroni



Sisteme inteligente – SIS – RNA

□ Scop → Cum învață creierul?

■ Cum "învață" (procesează informații)?

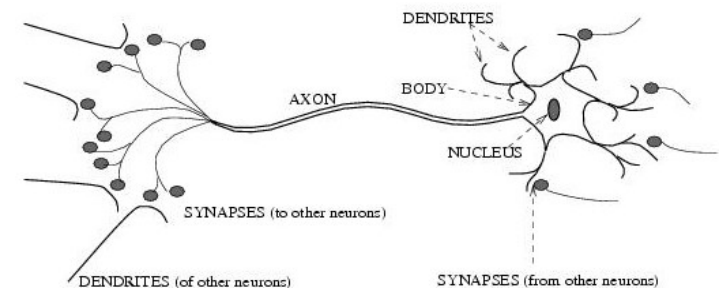
- Conexiunile care de-a lungul trăirii unor experiențe s-au dovedit utile devin permanente (restul sunt eliminate)
- Creierul este interesat de noutăți
- Modelul de procesare a informației
 - Învățare
 - Depozitare
 - Amintire

□ Memoria

- Tipologie
 - De scurtă durată
 - Imediată → 30 sec.
 - De lucru
 - De lungă durată
- Capacitate
 - Crește odată cu vârsta
 - Limitată → învățarea unei poezii pe strofe
- Influențată și de stările emoționale

□ Creierul

- rețea de neuroni
- sistem foarte complex, ne-liniar și paralel de procesare a informației
- Informația este depozitată și procesată de întreaga rețea, nu doar de o anumită parte a rețelei → informații și procesare globală
- Caracteristica fundamentală a unei rețele de neuroni → învățarea → rețele neuronale artificiale (RNA)



Sisteme inteligente – SIS – RNA

□ Definire

- Ce este o RNA?
- RN biologice vs. RN artificiale
- Cum învață rețeaua?

Sisteme inteligente – SIS – RNA

□ Definire → Ce este o RNA?

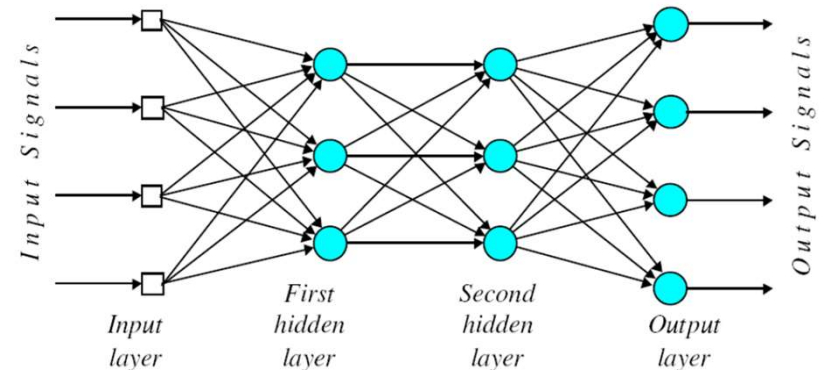
- O structură similară unei rețele neuronale biologice
- O mulțime de noduri (unități, neuroni, elemente de procesare) dispuse ca într-un graf pe mai multe straturi (*layere*)

□ Nodurile

- au intrări și ieșiri
- efectuează un calcul simplu prin intermediul unei funcții asociate → funcție de activare
- sunt conectate prin legături ponderate
 - Conexiunile între noduri conturează structura (arhitectura) rețelei
 - Conexiunile influențează calculele care se pot efectua

□ Straturile

- Strat de intrare
 - Conține m (nr de atribute al unei date) noduri
- Strat de ieșire
 - Conține r (nr de ieșiri) noduri
- Straturi intermediare (ascunse) – rol în “complicarea” rețelei
 - Diferite structuri
 - Diferite mărimi



Sisteme inteligente – SIS – RNA

- Definire → RN biologice vs. RN artificiale

RNB	RNA
Soma	Nod
Dendrite	Intrare
Axon	Ieșire
Activare	Procesare
Synapsă	Conexiune ponderată

- Definire → Cum învață rețeaua?

- Plecând de la un set de n date de antrenament de forma

$$((x_{p1}, x_{p2}, \dots, x_{pm}, y_{p1}, y_{p2}, \dots, y_{pr}))$$

cu $p = 1, 2, \dots, n$, m – nr atributelor, r – nr ieșirilor

- se formează o RNA cu m noduri de intrare, r noduri de ieșire și o anumită structură internă
 - un anumit nr de nivele ascunse, fiecare nivel cu un anumit nr de neuroni
 - cu legături ponderate între oricare 2 noduri
- se caută valorile optime ale ponderilor între oricare 2 noduri ale rețelei prin minimizarea erorii
 - diferența între rezultatul real y și cel calculat de către rețea

Sisteme inteligente – SIS – RNA

- Tipuri de probleme rezolvabile cu RNA
 - Datele problemei se pot reprezenta prin numeroase perechi atribut-valoare
 - Funcția obiectiv poate fi:
 - Unicriterială sau multicriterială
 - Discretă sau cu valori reale
 - Datele de antrenament pot conține erori (zgomot)
 - Timp de rezolvare (antrenare) prelungit

Sisteme inteligente – SIS – RNA

□ Proiectare

- Construirea RNA pentru rezolvarea problemei P
- Inițializarea parametrilor RNA
- Antrenarea RNA
- Testarea RNA

Sisteme inteligente – SIS – RNA

□ Proiectare

■ Construirea RNA pentru rezolvarea unei probleme P

- pp. o problemă de clasificare în care avem un set de date de forma:

- (x^d, t^d) , cu:
- $x^d \in \mathbf{R}^m \rightarrow x^d = (x^d_1, x^d_2, \dots, x^d_m)$
- $t^d \in \mathbf{R}^R \rightarrow t^d = (t^d_1, t^d_2, \dots, t^d_R)$,
- cu $d = 1, 2, \dots, n, n+1, n+2, \dots, N$

- primele n date vor fi folosite drept bază de antrenament a RNA
- ultimele $N-n$ date vor fi folosite drept bază de testare a RNA

- se construiește o RNA astfel:

- stratul de intrare conține exact m noduri (fiecare nod va citi una dintre proprietățile de intrare ale unei instanțe a problemei – $x^d_1, x^d_2, \dots, x^d_m$)
- stratul de ieșire poate conține R noduri (fiecare nod va furniza una dintre proprietățile de ieșire ale unei instanțe a problemei $t^d_1, t^d_2, \dots, t^d_R$)
- unul sau mai multe straturi ascunse cu unul sau mai mulți neuroni pe fiecare strat

Sisteme inteligente – SIS – RNA

□ Proiectare

- Construirea RNA pentru rezolvarea problemei P
- **Inițializarea parametrilor RNA**
- **Antrenarea RNA**
- Testarea RNA

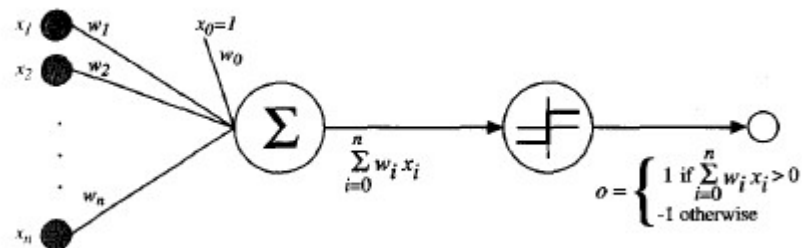
Sisteme inteligente – SIS – RNA

□ Proiectare

- Inițializarea parametrilor RNA
 - Inițializarea ponderile între oricare 2 noduri de pe straturi diferite
 - Stabilirea funcției de activare corespunzătoare fiecărui neuron (de pe straturile ascunse)
- Antrenarea (învățarea) RNA
 - Scop:
 - stabilirea valorii optime a ponderilor dintre 2 noduri
 - Algoritm
 - Se caută valorile optime ale ponderilor între oricare 2 noduri ale rețelei prin minimizarea erorii (diferența între rezultatul real y și cel calculat de către rețea)
 - Cum învață rețeaua?
 - Rețeaua = mulțime de unități primitive de calcul interconectate între ele →
 - Învățarea rețelei = \cup învățarea unităților primitive
 - Unități primitive de calcul
 - Perceptron
 - Unitate liniară
 - Unitate sigmoidală

Sisteme inteligente – SIS – RNA

- Proiectare → Antrenarea RNA → Cum învață rețeaua?
 - Neuronul ca element simplu de calcul
 - Structura neuronului
 - Fiecare nod are intrări și ieșiri
 - Fiecare nod efectuează un calcul simplu
 - Procesarea neuronului
 - Se transmite informația neuronului
 - Neuronul procesează informația
 - Se citește răspunsul neuronului
 - Învățarea neuronului – algoritmul de învățare a ponderilor care procesează corect informațiile
 - Se pornește cu un set inițial de ponderi oarecare
 - Cât timp nu este îndeplinită o condiție de oprire
 - Se procesează informația și se stabilește calitatea ponderilor curente
 - Se modifică ponderile astfel încât să se obțină rezultate mai bune



Sisteme inteligente – SIS – RNA

□ Proiectare → Antrenarea RNA → Cum învață rețeaua?

■ Neuronul ca element simplu de calcul

□ Structura neuronului

- Fiecare nod are intrări și ieșiri
- Fiecare nod efectuează un calcul simplu prin intermediul unei funcții asociate

□ Procesarea neuronului

- Se transmite informația neuronului → se calculează suma ponderată a intrărilor

$$net = \sum_{i=1}^n x_i w_i$$

- Neuronul procesează informația → se folosește o funcție de activare:

- Funcția constantă
- Funcția prag
- Funcția rampă
- Funcția liniară
- Funcția sigmoidală
- Funcția Gaussiană
- Funcția Relu

Sisteme inteligente – SIS – RNA

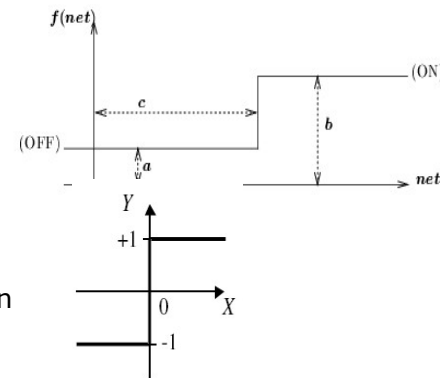
□ Proiectare → Antrenarea RNA → Cum învață rețeaua?

■ Funcția de activare a unui neuron

- Funcția constantă $f(net) = \text{const}$
- Funcția prag (c - pragul)

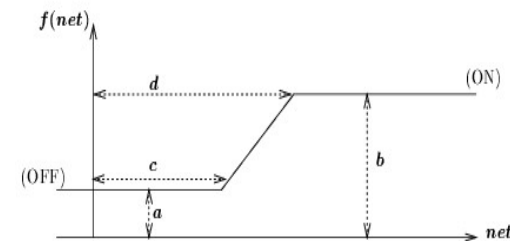
$$f(net) = \begin{cases} a, & \text{dacă } net < c \\ b, & \text{dacă } net > c \end{cases}$$

- Pentru $a=+1$, $b=-1$ și $c=0$ → funcția semn
- Funcție discontinuă



□ Funcția rampă

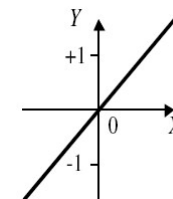
$$f(net) = \begin{cases} a, & \text{dacă } net \leq c \\ b, & \text{dacă } net \geq d \\ a + \frac{(net-c)(b-a)}{d-c}, & \text{altfel} \end{cases}$$



□ Funcția liniară

$$f(net) = a * net + b$$

- Pentru $a = 1$ și $b = 0$ → funcția identitate $f(net) = net$
- Funcție continuă



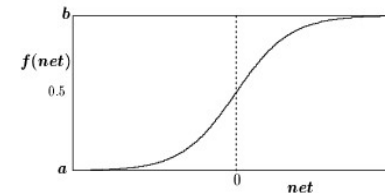
Sisteme inteligente – SIS – RNA

□ Proiectare → Antrenarea RNA → Cum învață rețeaua?

■ Funcția de activare a unui neuron

□ Funcția sigmoidală

- În formă de S
- Continuă și diferențiabilă în orice punct
- Simetrică rotațional față de un anumit punct ($net = c$)
- Atinge asimptotic puncte de saturație



$$\lim_{net \rightarrow -\infty} f(net) = a \quad \lim_{net \rightarrow \infty} f(net) = b$$

- Exemple de funcții sigmoidale:

$$f(net) = z + \frac{1}{1 + \exp(-x \cdot net + y)}$$

$$f(net) = \tanh(x \cdot net - y) + z$$

$$\text{unde } \tanh(u) = \frac{e^u - e^{-u}}{e^u + e^{-u}}$$

- Pentru $y=0$ și $z = 0 \rightarrow a=0, b = 1, c=0$
- Pentru $y=0$ și $z = -0.5 \rightarrow a=-0.5, b = 0.5, c=0$
- Cu cât x este mai mare, cu atât curba este mai abruptă

Sisteme inteligente – SIS – RNA

□ Proiectare → Antrenarea RNA → Cum învață rețeaua?

■ Funcția de activare a unui neuron

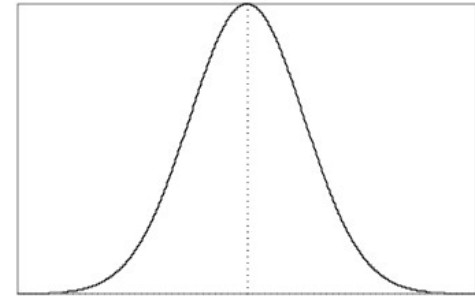
□ Funcția Gaussiană

- În formă de clopot
- Continuă
- Atinge asimptotic un punct de saturație

$$\lim_{net \rightarrow \infty} f(net) = a$$

- Are un singur punct de optim (maxim) – atins când $net = \mu$
- Exemplu

$$f(net) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left[-\frac{1}{2}\left(\frac{net - \mu}{\sigma}\right)^2\right]$$



Sisteme inteligente – SIS – RNA

□ Proiectare → Antrenarea RNA → Cum învață rețeaua?

■ Funcția de activare a unui neuron

□ Funcția ReLU

- În formă de rampă
- Continuă, monotonă
- Derivata ei este monotonă
- Codomeniu pozitiv $[0, \infty)$

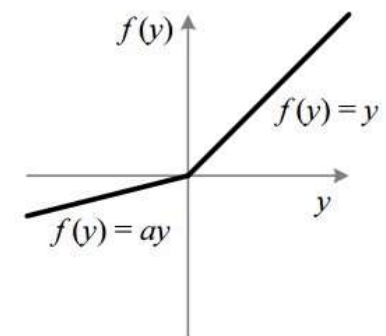
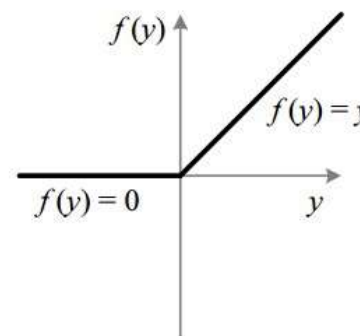
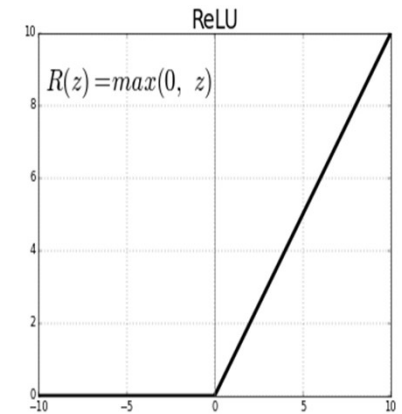
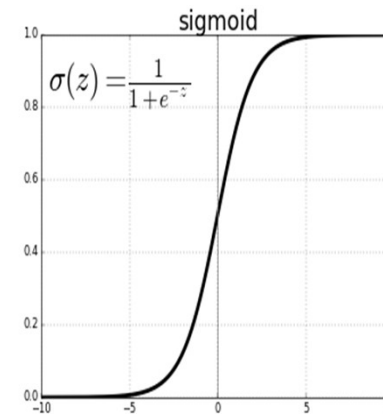
$$f(net) = \max(0, net)$$

$$f(net) = \begin{cases} 0, & \text{dacă } net < 0 \\ net, & \text{dacă } net \geq 0 \end{cases}$$

■ Variantă: Leaky ReLU

- Compensează problemele cu argumentele negative dint ReLU

$$f(net) = \begin{cases} a \cdot net, & \text{dacă } net < 0 \\ net, & \text{dacă } net \geq 0 \end{cases}$$



Sisteme inteligente – SIS – RNA

□ Proiectare → Antrenarea RNA → Cum învață rețeaua?

■ Neuronul ca element simplu de calcul

- Structura neuronului
- Procesarea neuronului

- Se transmite informația neuronului → se calculează suma ponderată a intrărilor

$$net = \sum_{i=1}^n x_i w_i$$

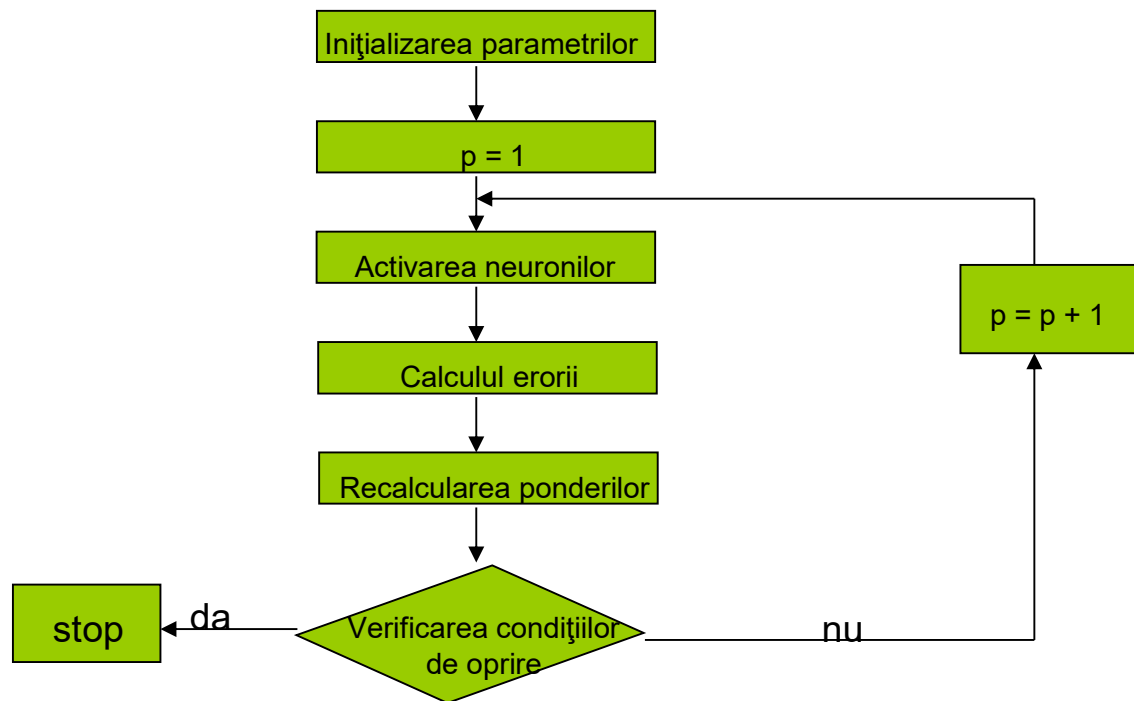
- Neuronul procesează informația → se folosește o funcție de activare:
 - Funcția constantă
 - Funcția prag
 - Funcția rampă
 - Funcția liniară
 - Funcția sigmoidală
 - Funcția Gaussiană
- Se citește răspunsul neuronului → se stabilește dacă rezultatul furnizat de neuron coincide sau nu cu cel dorit (real)

Sisteme inteligente – SIS – RNA

□ Proiectare → Antrenarea RNA → Cum învață rețeaua?

■ Neuronul ca element simplu de calcul

- Structura neuronului
- Procesarea neuronului
- Învățarea neuronului
 - Algoritm



Sisteme inteligente – SIS – RNA

- Proiectare → Antrenarea RNA → Cum învață RNA?
 - Învățarea neuronului
 - 2 reguli de bază
 - Regula perceptronului → algoritmul perceptronului
 1. Se porneste cu un set de ponderi oarecare
 2. Se stabilește calitatea modelului creat pe baza acestor ponderi pentru **UNA** dintre datele de intrare
 3. Se ajustează ponderile în funcție de calitatea modelului
 4. Se reia algoritmul de la pasul 2 până când se ajunge la calitate maximă
 - Regula Delta → algoritmul scăderii după gradient
 1. Se porneste cu un set de ponderi oarecare
 2. Se stabilește calitatea modelului creat pe baza acestor ponderi pentru **TOATE** dintre datele de intrare
 3. Se ajustează ponderile în funcție de calitatea modelului
 4. Se reia algoritmul de la pasul 2 până când se ajunge la calitate maximă
 - Similar regulii perceptronului, dar calitatea unui model se stabilește în funcție de toate datele de intrare (tot setul de antrenament)

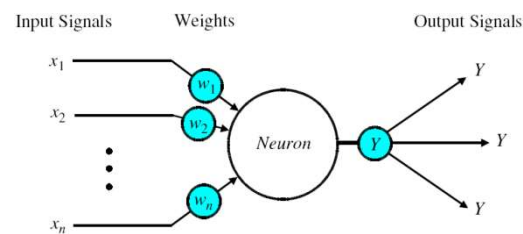
Sisteme inteligente – SIS – RNA

□ Proiectare → Antrenarea RNA → Cum învață RNA?

■ Învățarea neuronului

□ Pp că avem un set de date de antrenament de forma:

- (x^d, t^d) , cu:
 - $x^d \in \mathbf{R}^m \rightarrow x^d = (x^d_1, x^d_2, \dots, x^d_m)$
 - $t^d \in \mathbf{R}^R \rightarrow t^d = (t^d_1, t^d_2, \dots, t^d_R)$, și $R = 1$ (adică $t^d = (t^d_1)$)
 - cu $d = 1, 2, \dots, n$
- RNA = unitate primitivă de calcul (un neuron) → o rețea cu:
 - m noduri de intrare
 - legate de neuronul de calcul prin ponderile w_i , $i = 1, 2, \dots, m$ și
 - cu un nod de ieșire



Sisteme inteligente – SIS – RNA

□ Proiectare → Antrenarea RNA → Cum învață RNA?

■ Învățarea neuronului

□ Algoritmul perceptronului

- Se bazează pe minimizarea erorii asociată unei instanțe din setul de date de antrenament
- Modificarea ponderilor pe baza erorii asociate unei instanțe din setul de antrenament

Inițializare ponderi din rețea

$w_i = \text{random}(a,b)$, unde $i=1,2,\dots,m$

$d = 1$

Cât timp mai există exemple de antrenament clasificate incorect

Se activează neuronul și se calculează ieșirea

Perceptron → funcția de activare este funcția semn (funcție prag de tip discret, nediferențiable)

$$o^d = \text{sign}(\mathbf{w}\mathbf{x}) = \text{sign}\left(\sum_{i=1}^m w_i x_i\right)$$

Se stabilește ajustarea ponderilor $\Delta w_i = \eta(t^d - o^d)x_i^d$, unde $i = 1,2,\dots,m$

unde η - rată de învățare

Se ajustează ponderile $w'_i = w_i + \Delta w_i$

Dacă $d < n$ atunci $d++$

Altfel $d = 1$

SfCâtTimp

Sisteme inteligente – SIS – RNA

□ Proiectare → Antrenarea RNA → Cum învață RNA?

■ Învățarea neuronului

□ Algoritmul scădere după gradient

- Se bazează pe eroarea asociată întregului set de date de antrenament
- Modificarea ponderilor în direcția dată de cea mai abruptă pantă a reducerii erorii $E(\mathbf{w})$ pentru tot setul de antrenament

$$E(\mathbf{w}) = \frac{1}{2} \sum_{d=1}^n (t^d - o^d)^2$$

- Cum se determină cea mai abruptă pantă? → se derivează E în funcție de \mathbf{w} (se stabilește gradientul erorii E)

$$\nabla E(\mathbf{w}) = \left(\frac{\partial E}{\partial w_1}, \frac{\partial E}{\partial w_2}, \dots, \frac{\partial E}{\partial w_m} \right)$$

- Gradientul erorii E se calculează în funcție de funcția de activare a neuronului (care trebuie să fie diferențiabilă, deci continuă)

- Funcția liniară $f(net) = \sum_{i=1}^m w_i x_i^d$

- Funcția sigmoidală $f(net) = \frac{1}{1 + e^{-\mathbf{w}\mathbf{x}}} = \frac{1}{1 + e^{-\sum_{i=1}^m w_i x_i^d}}$

- Cum se ajustează ponderile?

$$\Delta w_i = -\eta \frac{\partial E}{\partial w_i}, \text{ unde } i = 1, 2, \dots, m$$

Sisteme inteligente – SIS – RNA

□ Proiectare → Antrenarea RNA → Cum învață RNA?

■ Învățarea neuronului

□ Algoritmul scădere după gradient → calcularea gradientului erorii

■ Funcția liniară

$$f(net) = \sum_{i=1}^m w_i x_i^d$$

$$\frac{\partial E}{\partial w_i} = \frac{\partial \frac{1}{2} \sum_{d=1}^n (t^d - o^d)^2}{\partial w_i} = \frac{1}{2} \sum_{d=1}^n \frac{\partial (t^d - o^d)^2}{\partial w_i} = \frac{1}{2} \sum_{d=1}^n 2(t^d - o^d) \frac{\partial (t^d - \mathbf{w}\mathbf{x}^d)}{\partial w_i}$$

$$\frac{\partial E}{\partial w_i} = \sum_{d=1}^n (t^d - o^d) \frac{\partial (t^d - w_1 x_1^d - w_2 x_2^d - \dots - w_m x_m^d)}{\partial w_i} = \sum_{d=1}^n (t^d - o^d) (-x_i^d)$$

$$\Delta w_i = -\eta \frac{\partial E}{\partial w_i} = \eta \sum_{d=1}^n (t^d - o^d) x_i^d$$

■ Funcție sigmoidală

$$f(net) = \frac{1}{1 + e^{-\mathbf{w}\mathbf{x}}} = \frac{1}{1 + e^{-\sum_{i=1}^m w_i x_i^d}} \quad y = s(z) = \frac{1}{1 + e^{-z}} \Rightarrow \frac{\partial s(z)}{\partial z} = s(z)(1 - s(z))$$

$$\frac{\partial E}{\partial w_i} = \frac{\partial \frac{1}{2} \sum_{d=1}^n (t^d - o^d)^2}{\partial w_i} = \frac{1}{2} \sum_{d=1}^n \frac{\partial (t^d - o^d)^2}{\partial w_i} = \frac{1}{2} \sum_{d=1}^n 2(t^d - o^d) \frac{\partial (t^d - sig(\mathbf{w}\mathbf{x}^d))}{\partial w_i} = \sum_{d=1}^n (t^d - o^d) (1 - o^d) o^d (-x_i^d)$$

$$\Delta w_i = -\eta \frac{\partial E}{\partial w_i} = \eta \sum_{d=1}^n (t^d - o^d) (1 - o^d) o^d x_i^d$$

Sisteme inteligente – SIS – RNA

- Proiectare → Antrenarea RNA → Cum învață RNA?
 - Învățarea neuronului
 - Algoritmul scădere după gradient (ASG)

ASG simplu	ASG stocastic
<p>Inițializare ponderi din rețea $w_i = \text{random}(a,b)$, unde $i=1,2,\dots,m$</p> <p>$d = 1$</p> <p>Cât timp nu este îndeplinită condiția de oprire</p> <p>$\Delta w_i = 0$, unde $i=1,2,\dots,m$</p> <p>Pentru fiecare exemplu de antrenament (x^d, t^d), unde $d=1,2,\dots,n$</p> <p>Se activează neuronul și se calculează ieșirea o^d</p> <p>funcția de activare = funcția liniară $\rightarrow o^d = \mathbf{w}\mathbf{x}^d$</p> <p>funcția de activare = funcția sigmoid $\rightarrow o^d = \text{sig}(\mathbf{w}\mathbf{x}^d)$</p> <p>Pentru fiecare pondere w_i, unde $i = 1,2,\dots,m$</p> <p>Se stabilește ajustarea ponderii $\Delta w_i = \Delta w_i - \eta \frac{\partial E}{\partial w_i}$</p> <p>Pentru fiecare pondere w_i, unde $i = 1,2,\dots,m$</p> <p>Se ajustează fiecare pondere w_i $w_i = w_i + \Delta w_i$</p>	<p>Inițializare ponderi din rețea $w_i = \text{random}(a,b)$, unde $i=1,2,\dots,m$</p> <p>$d = 1$</p> <p>Cât timp nu este îndeplinită condiția de oprire</p> <p>$\Delta w_i = 0$, unde $i=1,2,\dots,m$</p> <p>Pentru fiecare exemplu de antrenament (x^d, t^d), unde $d=1,2,\dots,n$</p> <p>Se activează neuronul și se calculează ieșirea o^d</p> <p>funcția de activare = funcția liniară $\rightarrow o^d = \mathbf{w}\mathbf{x}^d$</p> <p>funcția de activare = funcția sigmoid $\rightarrow o^d = \text{sig}(\mathbf{w}\mathbf{x}^d)$</p> <p>Pentru fiecare pondere w_i, unde $i = 1,2,\dots,m$</p> <p>Se stabilește ajustarea ponderilor $\Delta w_i = -\eta \frac{\partial E}{\partial w_i}$</p> <p>Se ajustează ponderea w_i $w_i = w_i + \Delta w_i$</p>

Sisteme inteligente – SIS – RNA

- Proiectare → Antrenarea RNA → Cum învață RNA?
 - Învățarea neuronului

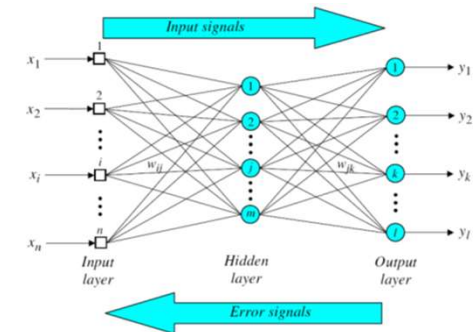
Diferențe	Algoritmul perceptronului	Algoritmul scădere după gradient (regula Delta)
Ce reprezintă o^d	$o^d = \text{sign}(\mathbf{w}\mathbf{x}^d)$	$o^d = \mathbf{w}\mathbf{x}^d$ sau $o^d = \text{sig}(\mathbf{w}\mathbf{x}^d)$
Cum converge	Într-un nr finit de pași (până la separarea perfectă)	Asimptotic (spre eroarea minimă)
Ce fel de probleme se pot rezolva	Cu date liniar separabile	Cu orice fel de date (separabile liniar sau ne-liniar)
Ce tip de ieșire are neuronul	Discretă și cu prag	Continuă și fără prag

Sisteme inteligente – SIS – RNA

□ Proiectare → Antrenarea RNA

■ Cum învață rețeaua?

- Rețeaua = mulțime de unități primitive de calcul interconectate între ele →
 - Învățarea rețelei = \cup învățarea unităților primitive
- Rețea cu mai mulți neuroni așezați pe unul sau mai multe straturi → RNA este capabilă să învețe un model mai complicat (nu doar liniar) de separare a datelor
- Algoritmul de învățare a ponderilor → backpropagation
 - Bazat pe algoritmul scădere după gradient
 - Îmbogățit cu:
 - Informația se propagă în RNA înainte (dinspre stratul de intrare spre cel de ieșire)
 - Eroarea se propagă în RNA înapoi (dinspre stratul de ieșire spre cel de intrare)



Se inițializează ponderile

Cât timp nu este îndeplinită condiția de oprire

Pentru fiecare exemplu (x^d, t^d)

Se activează fiecare neuron al rețelei

Se propagă informația înainte și se calculează ieșirea corespunzătoare fiecărui neuron al rețelei

Se ajustează ponderile

Se stabilește și se propagă eroarea înapoi

Se stabilesc erorile corespunzătoare neuronilor din stratul de ieșire

Se propagă aceste erori înapoi în toată rețeaua → se distribuie erorile pe toate conexiunile existente în rețea proporțional cu valorile ponderilor asociate acestor conexiuni

Se modifică ponderile

Sisteme inteligente – SIS – RNA

□ Proiectare → Antrenarea RNA

■ Cum învață o întreagă RNA?

□ Pp că avem un set de date de antrenament de forma:

■ (x^d, t^d) , cu:

- $x^d \in \mathbf{R}^m \rightarrow x^d = (x^d_1, x^d_2, \dots, x^d_m)$
- $t^d \in \mathbf{R}^R \rightarrow t^d = (t^d_1, t^d_2, \dots, t^d_R)$
- cu $d = 1, 2, \dots, n$

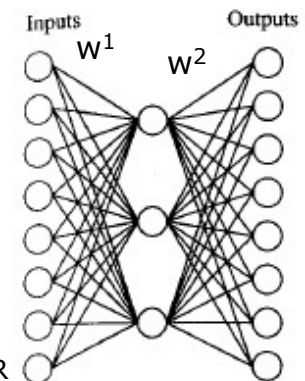
□ Presupunem 2 cazuri de RNA

■ O RNA cu un singur strat ascuns cu H neuroni → RNA₁

- m neuroni pe stratul de intrare,
- R neuroni pe stratul de ieșire,
- H neuroni pe stratul ascuns
- Ponderile între stratul de intrare și cel ascuns w_{ih}^1 cu $i=1,2,\dots,m, h=1,2,\dots,H$
- Ponderile între stratul ascuns și cel de ieșire w_{hr}^2 cu $h=1,2,\dots,H$ și $r=1,2,\dots,R$

■ O RNA cu p straturi ascunse, fiecare strat cu H_i (i = 1, 2, ..., p) neuroni → RNA_p

- m neuroni pe stratul de intrare,
- R neuroni pe stratul de ieșire,
- P straturi ascunse
- H_p neuroni pe stratul ascuns p, $p = 1, 2, \dots, P$
- Ponderile între stratul de intrare și primul strat ascuns $w_{ih_1}^1$ cu $i=1,2,\dots,m, h_1 = 1,2,\dots,H_1$
- Ponderile între primul strat ascuns și cel de-al doilea strat ascuns $w_{h_1h_2}^2$ cu $h_1 = 1,2,\dots,H_1$ și $h_2 = 1,2,\dots,H_2$
- Ponderile între cel de-al doilea strat ascuns și cel de-al treilea strat ascuns $w_{h_2h_3}^3$ cu $h_2 = 1,2,\dots,H_2$ și $h_3 = 1,2,\dots,H_3$
- ...
- Ponderile între cel de-al p-1 strat ascuns și ultimul strat ascuns $w_{h_{p-1}h_p}^p$ cu $h_{p-1} = 1,2,\dots,H_{p-1}$ și $h_p = 1,2,\dots,H_p$
- Ponderile între ultimul strat ascuns și cel de ieșire $w_{h_p r}^{p+1}$ cu $h_p = 1,2,\dots,H_p$ și $r = 1,2,\dots,R$



Sisteme inteligente – SIS – RNA

- ❑ Proiectare → Antrenarea RNA → Cum învață o întreagă RNA?
 - Algoritmul backpropagation pentru RNA₁

Se inițializează ponderile w_{ih}^1 și w_{hr}^2 cu $i=1,2,\dots,m$, $h=1,2,\dots,H$ și $r=1,2,\dots,R$

Cât timp nu este îndeplinită condiția de oprire

Pentru fiecare exemplu (x^d, t^d)

Se activează fiecare neuron al rețelei

Se propagă informația înainte și se calculează ieșirea corespunzătoare fiecărui neuron al rețelei

$$o_h^d = \sum_{i=1}^m w_{ih}^1 x_i^d \text{ sau } o_h^d = \text{sig}\left(\sum_{i=1}^m w_{ih}^1 x_i^d\right), \text{ cu } h=1,2,\dots,H$$

$$o_r^d = \sum_{h=1}^H w_{hr}^2 o_h^d \text{ sau } o_r^d = \text{sig}\left(\sum_{h=1}^H w_{hr}^2 o_h^d\right), \text{ cu } r=1,2,\dots,R$$

Se ajustează ponderile

Se stabilește și se propagă eroarea înapoi

Se stabilesc erorile corespunzătoare neuronilor din stratul de ieșire

$$\delta_r^d = t_r^d - o_r^d \text{ sau } \delta_r^d = o_r^d (1 - o_r^d)(t_r^d - o_r^d), \text{ cu } r=1,2,\dots,R$$

Se modifică ponderile între nodurile de pe stratul ascuns și stratul de ieșire

$$w_{hr}^2 = w_{hr}^2 + \eta \delta_r^d o_h^d, \text{ unde } h=1,2,\dots,H \text{ și } r=1,2,\dots,R$$

Se propagă erorile nodurilor de pe stratul de ieșire înapoi în toată rețeaua → se distribuie erorile pe toate conexiunile existente în rețea proporțional cu valorile ponderilor asociate acestor conexiuni

$$\delta_h^d = \sum_{r=1}^R w_{hr}^2 \delta_r^d \text{ sau } \delta_h^d = o_h^d (1 - o_h^d) \sum_{r=1}^R w_{hr}^2 \delta_r^d$$

Se modifică ponderile între nodurile de pe stratul de intrare și stratul ascuns

$$w_{ih}^1 = w_{ih}^1 + \eta \delta_h^d x_i^d, \text{ unde } i=1,2,\dots,m \text{ și } h=1,2,\dots,H$$

Sisteme inteligente – SIS – RNA

- Proiectare → Antrenarea RNA → Cum învață o întreagă RNA?
 - Algoritmul backpropagation pentru RNA_p

Se inițializează ponderile $w_{ih_1}^1, w_{h_1h_2}^2, \dots, w_{h_{p-1}h_p}^p, w_{h_p r}^{p+1}$

Cât timp nu este îndeplinită condiția de oprire

Pentru fiecare exemplu (x^d, t^d)

Se activează fiecare neuron al rețelei

Se propagă informația înainte și se calculează ieșirea corespunzătoare fiecărui neuron al rețelei

$$o_{h_1}^d = \sum_{i=1}^m w_{ih_1}^1 x_i^d \text{ sau } o_{h_1}^d = \text{sig} \left(\sum_{i=1}^m w_{ih_1}^1 x_i^d \right), \text{ cu } h_1 = 1, 2, \dots, H_1$$

$$o_{h_2}^d = \sum_{h_1=1}^{H_1} w_{h_1h_2}^2 o_{h_1}^d \text{ sau } o_{h_2}^d = \text{sig} \left(\sum_{h_1=1}^{H_1} w_{h_1h_2}^2 o_{h_1}^d \right), \text{ cu } h_2 = 1, 2, \dots, H_2$$

...

$$o_{h_p}^d = \sum_{h_{p-1}=1}^{H_{p-1}} w_{h_{p-1}h_p}^p o_{h_{p-1}}^d \text{ sau } o_{h_p}^d = \text{sig} \left(\sum_{h_{p-1}=1}^{H_{p-1}} w_{h_{p-1}h_p}^p o_{h_{p-1}}^d \right), \text{ cu } h_p = 1, 2, \dots, H_p$$

$$o_r^d = \sum_{h_p=1}^{H_p} w_{h_p r}^{p+1} o_{h_p}^d \text{ sau } o_r^d = \text{sig} \left(\sum_{h_p=1}^{H_p} w_{h_p r}^{p+1} o_{h_p}^d \right), \text{ cu } r = 1, 2, \dots, R$$

Sisteme inteligente – SIS – RNA

- Proiectare → Antrenarea RNA → Cum învață o întreagă RNA?
 - Algoritmul backpropagation pentru RNA_p

Se inițializează ponderile $w_{ih_1}^1, w_{h_1h_2}^2, \dots, w_{h_{p-1}h_p}^p, w_{h_p r}^{p+1}$

Cât timp nu este îndeplinită condiția de oprire

Pentru fiecare exemplu (x^d, t^d)

Se activează fiecare neuron al rețelei

Se ajustează ponderile

Se stabilește și se propagă eroarea înapoi

Se stabilesc erorile corespunzătoare neuronilor din stratul de ieșire

$\delta_r^d = t_r^d - o_r^d$ sau $\delta_r^d = o_r^d(1 - o_r^d)(t_r^d - o_r^d)$, cu $r = 1, 2, \dots, R$
Se modifică ponderile între nodurile de pe ultimul strat ascuns și stratul de ieșire

$$w_{h_p r}^{p+1} = w_{h_p r}^{p+1} + \eta \delta_r^d o_{h_p}^d, \text{ unde } h_p = 1, 2, \dots, H_p \text{ și } r = 1, 2, \dots, R$$

Sisteme inteligente – SIS – RNA

- Proiectare → Antrenarea RNA → Cum învață o întreagă RNA?
 - Algoritmul backpropagation pentru RNA_p

Se inițializează ponderile $w_{ih_1}^1, w_{h_1h_2}^2, \dots, w_{h_{p-1}h_p}^p, w_{h_p r}^{p+1}$

Cât timp nu este îndeplinită condiția de oprire

Pentru fiecare exemplu (x^d, t^d)

Se activează fiecare neuron al rețelei

Se ajustează ponderile

Se stabilește și se propagă eroarea înapoi

Se stabilesc erorile corespunzătoare neuronilor din stratul de ieșire

Se modifică ponderile între nodurile de pe ultimul strat ascuns și stratul de ieșire

Se propagă (pe starturi) aceste erori înapoi în toată rețeaua → se distribuie erorile pe toate conexiunile existente în rețea proporțional cu valorile ponderilor asociate acestor conexiuni și se modifică ponderile corespunzătoare

$$\delta_{h_p}^d = \sum_{r=1}^R w_{h_p r}^{p+1} \delta_r^d \text{ sau } \delta_{h_p}^d = o_{h_p}^d (1 - o_{h_p}^d) \sum_{r=1}^R w_{h_p r}^{p+1} \delta_r^d$$

$$w_{h_p r}^{p+1} = w_{h_p r}^{p+1} + \eta \delta_r^d o_{h_p}^d, \text{ unde } h_p = 1, 2, \dots, H_p \text{ și } r = 1, 2, \dots, R$$

$$\delta_{h_{p-1}}^d = \sum_{h_p=1}^{H_p} w_{h_{p-1}h_p}^p \delta_{h_p}^d \text{ sau } \delta_{h_{p-1}}^d = o_{h_{p-1}}^d (1 - o_{h_{p-1}}^d) \sum_{h_p=1}^{H_p} w_{h_{p-1}h_p}^p \delta_{h_p}^d$$

$$w_{h_{p-1}h_p}^p = w_{h_{p-1}h_p}^p + \eta \delta_{h_p}^d o_{h_{p-1}}^d, \text{ unde } h_{p-1} = 1, 2, \dots, H_{p-1} \text{ și } h_p = 1, 2, \dots, H_p$$

...

$$\delta_{h_1}^d = \sum_{h_2=1}^{H_2} w_{h_1h_2}^2 \delta_{h_2}^d \text{ sau } \delta_{h_1}^d = o_{h_1}^d (1 - o_{h_1}^d) \sum_{h_2=1}^{H_2} w_{h_1h_2}^2 \delta_{h_2}^d$$

$$w_{ih_1}^1 = w_{ih_1}^1 + \eta \delta_{h_1}^d x_i^d, \text{ unde } i = 1, 2, \dots, m \text{ și } h_1 = 1, 2, \dots, H_1$$

Inteligență artificială - sisteme inteligente (RNA)

Sisteme inteligente – SIS – RNA

□ Cum se masoara calitatea algoritmului de invatare?

■ Probleme de regresie

- Stratul de iesire are un nr de neuroni egal cu nr de variabile care trebuie prezise

- Identificarea erorii: pentru fiecare neuron de output

- Suma diferentelor intre valorile prezise si cele reale

- Diferentele in valoare absoluta (L1, MAE)

- Robustete la valori extreme (outliers)

- Dar, gradienti mari \leftrightarrow eroare mica

- Solutii: rata de invatare dinamica sau MSE

- Patrutul diferentelor (L2, MSE)

- Stabilitatea solutiilor

- Dar sensibilitate la valori extreme

- Huber

$$L_{\delta}(y, f(x)) = \begin{cases} \frac{1}{2}(y - f(x))^2 & \text{for } |y - f(x)| \leq \delta, \\ \delta |y - f(x)| - \frac{1}{2}\delta^2 & \text{otherwise.} \end{cases}$$

- Pentru

- Un exemplu din date

- Pentru mai multe exemple din date (batch)





- Functia de cost (loss) = eroarea

Sisteme inteligente – SIS – RNA





□ Cum se masoara calitatea algoritmului de invatare?

■ Probleme de clasificare – clasificarea fructelor in imagini




□ Binara

examples	labels
	fruit
	non-fruit
	non-fruit
	fruit





□ Multi-clasa

examples	labels
	apple
	pear
	grappe
	pear

□ Multi-label

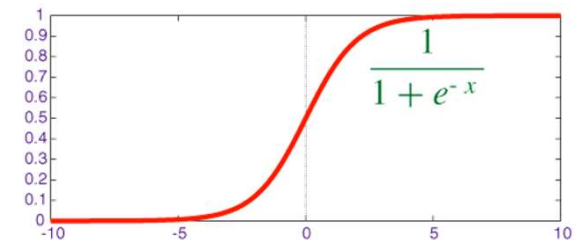
examples	labels [apple? pear? grappes?]
	[1 1 0]
	[1 0 1]
	[1 1 1]

Sisteme inteligente – SIS – RNA

examples	labels
	fruit
	non-fruit
	non-fruit
	fruit

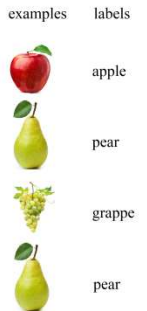
- Cum se masoara calitatea algoritmului de invatare? → Problem clasificare – clasificarea fructelor in imagini → Binara

- Stratul de iesire are un singur neuron cu functie de activare sigmoid
 - Outputul neuronului este transformat prin sigmoid → o valoare intre 0 si 1 → probabilitatea apartenentei la una din cele 2 clase (→ acuratetea, precizia, rapelul)



- Functia de cost != acuratete (precizie, recall)
 - Functia de cost = functia de loss = Cross-entropy loss (Logistic Loss or Multinomial Logistic Loss)
 - Cross Entropy pentru 2 distributii p si q
 - $CE(p,q) = -\sum(p_i \log_2(q_i))$, $i = 1, 2, \dots$
 - Cross-entropy pt clasificare binara
 - Pentru fiecare exemplu din date
 - Se transforma fiecare eticheta reala intr-o lista cu 2 probabilitati → **p**
[1-eticheta_clasa, eticheta_clasa]
 - se transforma outputul sigmoidat al neuronului de iesire intr-o lista de 2 probabilitati → **q**
[1 - out_sigm, out_sigm]
 - se calculeaza entropia conditionata intre **p** si **q** : $CE(\text{exemplul crt})$
 - Se calculeaza media entropiilor conditionate pt fiecare exemplu

Sisteme inteligente – SIS – RNA



- Cum se masoara calitatea algoritmului de invatare? → Probleme de clasificare – clasificarea fructelor in imagini → multi-clasa

- Abordarea one-vs-all

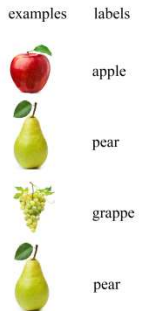
- O problema cu C-clase → C probleme de clasificare binara
 - Apple vs. others, Pear vs. others, Grape vs others
- Se antreneaza C modele de clasificare binara (neuronul de output are activare sigmoidala)
- La testare, se aplica toate cele C modele antrenate si se alege clasa indicata de cel cu outputul cel mai mare

- Abordarea soft-max

- O retea cu C neuroni in stratul de iesire si activare softmax

$$out'(neuron_i) = \frac{e^{out(neuron_i)}}{\sum_{i=1}^C e^{out(neuron_i)}}$$

Sisteme inteligente – SIS – RNA



- Cum se masoara calitatea algoritmului de invatare? → Probleme de clasificare – clasificare fructelor in imagini → multi-clasa → Abordarea soft-max

- Stratul de iesire are C neuroni (C – nr de clase)

- Outputurile neuronilor (s) se transforma prin softmax → o valoare intre 0 si 1 → probabilitatea apartenentei la una din cele C clase (→ acuratetea, precizia, rapelul)

$$q = \left[\frac{e^{s_1}}{\sum_{j=1}^C e^{s_j}}, \frac{e^{s_2}}{\sum_{j=1}^C e^{s_j}}, \dots, \frac{e^{s_C}}{\sum_{j=1}^C e^{s_j}} \right]$$

- Functia de cost != acuratete (precizie, recall)

- Functia de cost = functia de loss = categorical cross-entropy loss (Softmax Loss)

- Cross Entropy pentru 2 distributii p si q

- $CE(p,q) = -\sum (p_i \log_2(q_i))$, $i = 1, 2, \dots$

- Cross-entropy pt clasificare multi-clasa

- Pentru fiecare exemplu din date

- Se transforma fiecare eticheta reala intr-o lista cu C probabilitati, probabilitatea clasei corecte fiind 1, restul 0 (one-hot encoding) → **p**

$$\mathbf{p} = [1, 0, 0], \mathbf{p} = [0, 1, 0], \mathbf{p} = [0, 0, 1]. \mathbf{p} = [0, 1, 0]$$




- Pentru fiecare exemplu din date, se transforma outputurile neuronilor cu softmax → **q**

- Se calculeaza entropia conditionata intre **p** si **q** : $CE(\text{exemplul crt})$

$$CE = -\sum_{i=1}^C p_i \log(q_i) = -\log(q_{\text{correctClass}})$$

- Se calculeaza media entropiilor conditionate pt fiecare exemplu

Sisteme inteligente – SIS – RNA

examples	labels [apple? pear? grapes?]
	[1 1 0]
	[1 0 1]
	[1 1 1]

- Cum se masoara calitatea algoritmului de invatare? → Probleme de clasificare – clasificarea fructelor in imagini → multi-label

- Stratul de iesire are C neuroni (C – nr de clase)

- Outputurile neuronilor (s) se transforma prin sigmoid → o valoare intre 0 si 1 → probabilitatea apartenentei la una din cele C clase (→ acuratetea, precizia, rapelul)

$$\left[\frac{1}{1+e^{-s_1}}, \frac{1}{1+e^{-s_2}}, \dots, \frac{1}{1+e^{-s_C}} \right]$$

- Functia de cost != acuratete (precizie, recall)

- Functia de cost = functia de loss = Binary cross entropy (sigmoid cross entropy) pt fiecare eticheta posibila

- Cross Entropy pentru 2 distributii p si q

- $CE(p,q) = -\sum (p_i \log_2(q_i))$, $i = 1, 2, \dots$

- Cross-entropy pt clasificare multi-label

- Pentru fiecare exemplu din date

- Se transforma fiecare eticheta reala intr-o lista cu C probabilitati, probabilitatea clasei corecte fiind 1, restul 0 (one-hot encoding) → **p**

- $\mathbf{p} = [1, 0, 0]$, $\mathbf{p} = [0, 1, 0]$, $\mathbf{p} = [0, 0, 1]$. $\mathbf{p} = [0, 1, 0]$

- Pentru fiecare exemplu din date, se transforma outputurile neuronilor cu sigmoid → **q**

- $q_i = \text{sigm}(s_i)$, $i = 1, 2, \dots, C$, $\mathbf{q} = [q_i, 1 - q_i]$

- Se calculeaza entropia conditionata intre **p** si **q** : CE(exemplul crt)

$$CE_i = -p_i \log(q_i) - (1 - p_i) \log(1 - q_i)$$

$$CE = \sum_{i=1}^C CE_i$$

- Se calculeaza media entropiilor conditionate pt fiecare exemplu

Sisteme inteligente – SIS – RNA

- Cum se masoara calitatea algoritmului de invatare? → Probleme de clasificare – clasificarea fructelor in imagini → multi-class & multi-label

- Cross-entropy

- Classic loss

$$CE = -\sum_{i=1}^C p_i \log(q_i)$$

- Focal loss

$$CE = -\sum_{i=1}^C (1 - q_i)^{\gamma} p_i \log(q_i)$$

Sisteme inteligente – SIS – RNA

- Proiectare → Antrenarea RNA → Cum învață o întreagă RNA?
 - Algoritmul backpropagation
 - Condiții de oprire
 - S-a ajuns la eroare 0
 - S-au efectuat un anumit număr de iterații
 - La o iterație se procesează un singur exemplu
 - n iterații = o epocă

Sisteme inteligente – SIS – RNA

□ Proiectare

- Construirea RNA pentru rezolvarea problemei P
- Inițializarea parametrilor RNA
- Antrenarea RNA
- **Testarea RNA**

Sisteme inteligente – SIS – RNA

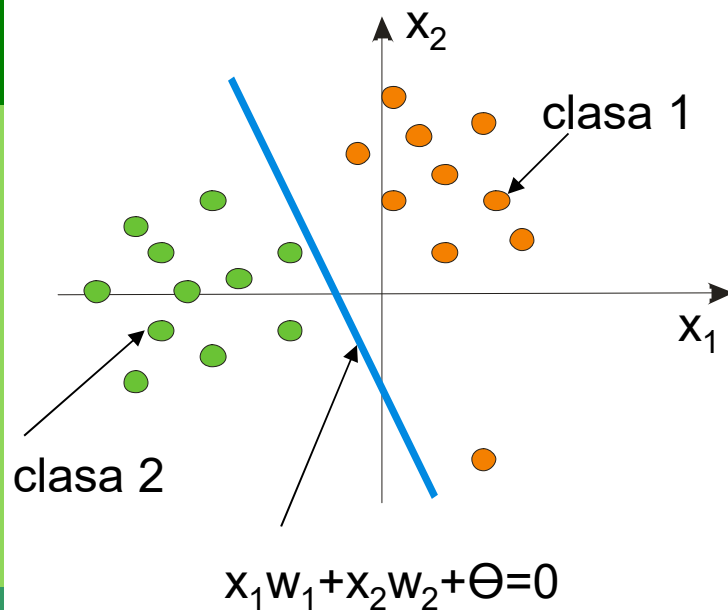
□ Proiectare

■ Testarea RNA

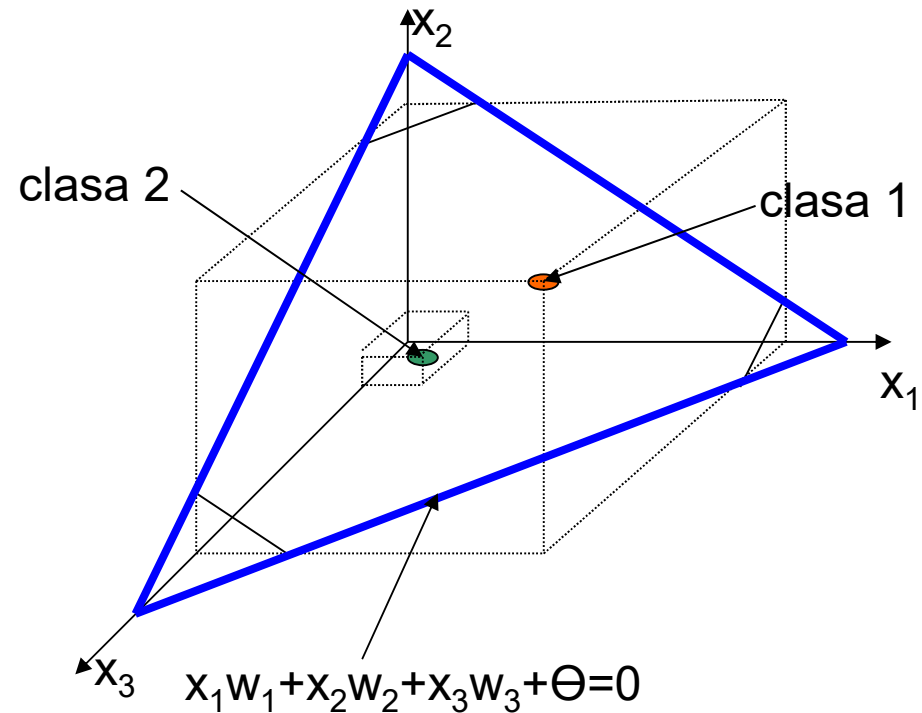
- Se decodifică modelul învățat de RNA
 - prin combinarea ponderilor cu intrările
 - ținând cont de funcțiile de activare a neuronilor și de structura rețelei

Sisteme inteligente – SIS – RNA

□ Exemplu



Clasificare binară cu $m=2$ intrări



Clasificare binară cu $m=3$ intrări

Sisteme inteligente – SIS – RNA

□ Exemplu

■ Perceptron pentru rezolvarea problemei *ȘI logic*

Epoch	Inputs		Desired output Y_d	Initial weights		Actual output Y	Error e	Final weights	
	x_1	x_2		w_1	w_2			w_1	w_2
1	0	0	0	0.3	-0.1	0	0	0.3	-0.1
	0	1	0	0.3	-0.1	0	0	0.3	-0.1
	1	0	0	0.3	-0.1	1	-1	0.2	-0.1
	1	1	1	0.2	-0.1	0	1	0.3	0.0
2	0	0	0	0.3	0.0	0	0	0.3	0.0
	0	1	0	0.3	0.0	0	0	0.3	0.0
	1	0	0	0.3	0.0	1	-1	0.2	0.0
	1	1	1	0.2	0.0	1	0	0.2	0.0
3	0	0	0	0.2	0.0	0	0	0.2	0.0
	0	1	0	0.2	0.0	0	0	0.2	0.0
	1	0	0	0.2	0.0	1	-1	0.1	0.0
	1	1	1	0.1	0.0	0	1	0.2	0.1
4	0	0	0	0.2	0.1	0	0	0.2	0.1
	0	1	0	0.2	0.1	0	0	0.2	0.1
	1	0	0	0.2	0.1	1	-1	0.1	0.1
	1	1	1	0.1	0.1	1	0	0.1	0.1
5	0	0	0	0.1	0.1	0	0	0.1	0.1
	0	1	0	0.1	0.1	0	0	0.1	0.1
	1	0	0	0.1	0.1	0	0	0.1	0.1
	1	1	1	0.1	0.1	1	0	0.1	0.1

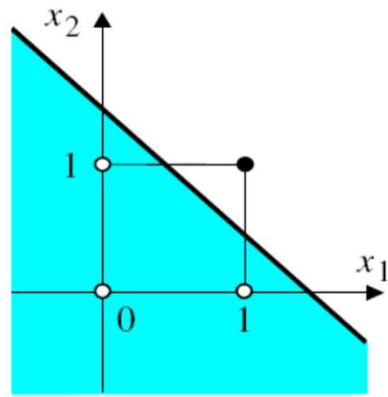
Threshold: $\theta = 0.2$; learning rate: $\alpha = 0.1$

Sisteme inteligente – SIS – RNA

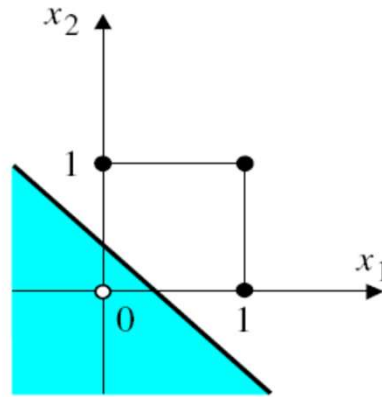
□ Exemplu

■ Perceptron - limitări

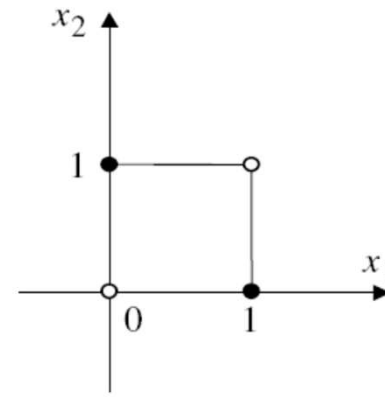
- Un perceptron poate învăța operațiile AND și OR, dar nu poate învăța operația XOR (nu e liniar separabilă)



(a) *AND* ($x_1 \cap x_2$)



(b) *OR* ($x_1 \cup x_2$)



(c) *Exclusive-OR*
($x_1 \oplus x_2$)

- Nu poate clasifica date non-liniar separabile

- soluții
 - Neuron cu un prag continuu
 - Mai mulți neuroni

Sisteme inteligente – SIS – RNA

□ Tipologie

■ RNA feed-forward

- Informația se procesează și circulă de pe un strat pe altul
- Conexiunile între noduri nu formează cicluri
- Se folosesc în special pentru învățarea supervizată
- Funcțiile de activare a nodurilor → liniare, sigmoidale, gaussiene

■ RNA recurente (cu feedback)

- Pot conține conexiuni între noduri de pe același strat
 - Conexiunile între noduri pot forma cicluri
 - RNA de tip Jordan
 - RNA de tip Elman
 - RNA de tip Hopfield
 - RNA auto-organizate → pentru învățarea nesupervizată
 - De tip Hebbian
 - De tip Kohonen (*Self organised maps*)
- } pentru învățarea supervizată

Sisteme inteligente – SIS – RNA

□ Avantaje

- Pot rezolva atât probleme de învățare super-vizată, cât și nesupervizată
- Pot identifica relații dinamice și neliniare între date
- Pot rezolva probleme de clasificare cu oricâte clase (multi-clasă)
- Se pot efectua calcule foarte rapid (în paralel și distribuit)

□ Dificultăți și limite

- RNA se confruntă cu problema overfitting-ului chiar și când modelul se învață prin validare încrucișată
- RNA pot găsi (uneori) doar optimele locale (fără să identifice optimul global)



Recapitulare

□ Sisteme care învață singure (SIS)

■ Rețele neuronale artificiale

- Modele computaționale inspirate de rețelele neuronale artificiale
- Grafe speciale cu noduri așezate pe straturi
 - Strat de intrare → citește datele de intrare ale problemei de rezolvat
 - Strat de ieșire → furnizează rezultate problemei date
 - Strat(uri) ascunse → efectuează calcule
- Nodurile (neuronii)
 - Au intrări ponderate
 - Au funcții de activare (liniare, sigmoidale, etc)
 - necesită antrenare → prin algoritmi precum:
 - Perceptron
 - Scădere după gradient
- Algoritm de antrenare a întregii RNA → Backpropagation
 - Informația utilă se propagă înainte
 - Eroarea se propagă înapoi

Cursul următor

A. Scurtă introducere în Inteligența Artificială (IA)

B. Rezolvarea problemelor prin căutare

- Definirea problemelor de căutare
- Strategii de căutare
 - Strategii de căutare neinformate
 - Strategii de căutare informate
 - Strategii de căutare locale (Hill Climbing, Simulated Annealing, Tabu Search, Algoritmi evolutivi, PSO, ACO)
 - Strategii de căutare adversială

C. Sisteme inteligente

- Sisteme care învață singure
 - Arbori de decizie
 - Rețele neuronale artificiale
 - Algoritmi evolutivi
- Sisteme bazate pe reguli
- Sisteme hibride

Cursul următor – Materiale de citit și legături utile

- ❑ capitolul 15 din *C. Groșan, A. Abraham, Intelligent Systems: A Modern Approach, Springer, 2011*
- ❑ Capitolul 9 din *T. M. Mitchell, Machine Learning, McGraw-Hill Science, 1997*
- ❑ Documentele din directorul *12_svm* și *13_GP*

□ Informațiile prezentate au fost colectate din diferite surse de pe internet, precum și din cursurile de inteligență artificială ținute în anii anteriori de către:

■ Conf. Dr. Mihai Oltean –
www.cs.ubbcluj.ro/~moltean

■ Lect. Dr. Crina Groșan -
www.cs.ubbcluj.ro/~cgrosan

■ Prof. Dr. Horia F. Pop -
www.cs.ubbcluj.ro/~hfpop