

Computer Vision 3

Ș.I. dr. ing. Mihai DOGARIU
www.mdogariu.aimultimedialab.ro

1

Structura cursului



- M1. Introducere
- M2. Fundamentele Învățării Adânci (Deep Learning Fundamentals)
- M3. Învățare Adâncă Supervizată (Supervised Deep Learning)
- M4. Învățare Adâncă Nesupervizată (Unsupervised Deep Learning)
- M5. Învățare Consolidată (Reinforcement Learning)

20.10.2022

Computer Vision 3, Ș.I. Mihai DOGARIU

2

2

M2. Fundamentele Învățării Adânci

- 2.1. Structura unei rețele neuronale
- 2.2. Procesul de bază al învățării
- 2.3. Terminologie
- 2.4. Considerații practice

20.10.2022

Computer Vision 3, Ș.I. Mihai DOGARIU

3

3

M2.1. Structura unei rețele neuronale

20.10.2022

Computer Vision 3, Ș.I. Mihai DOGARIU

4

4

Definiția învățării

Învățare (learning) = procesul de a dobândi noi înțelegeri, cunoștințe, comportamente, aptitudini, valori, atitudini sau preferințe [1].

Învățarea mașinilor (machine learning) = spunem despre un sistem că „învată” din experiența E cu privire la o clasă de sarcini de lucru T și o măsură de performanță P, dacă performanța sa în rezolvarea sarcinilor T, măsurată prin P, crește cu experiența E [2].

Exemplu:

- T = prognoză meteo (plouă/nu plouă)
- P = procentajul de zile prezise corect
- E = datele meteo ale precedentilor 3 ani

20.10.2022

Computer Vision 3, Ș.I. Mihai DOGARIU

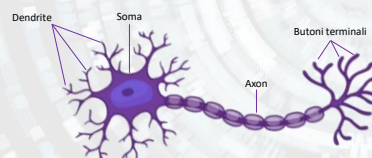
5

5

Neuronul biologic

Sursă imagine: Pinterest

➤ Celula fundamentală a sistemului nervos

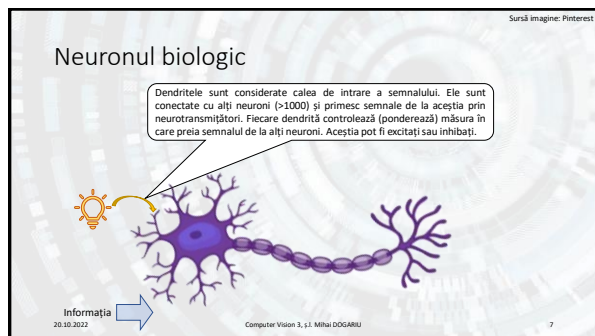


20.10.2022

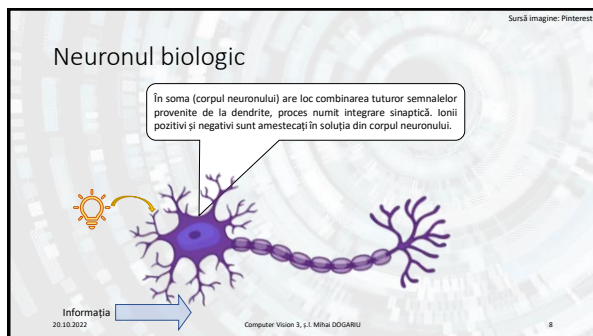
Computer Vision 3, Ș.I. Mihai DOGARIU

6

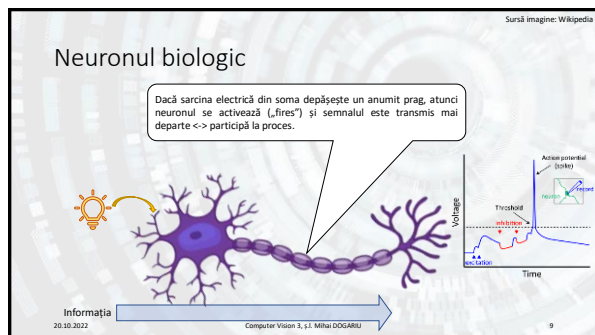
6



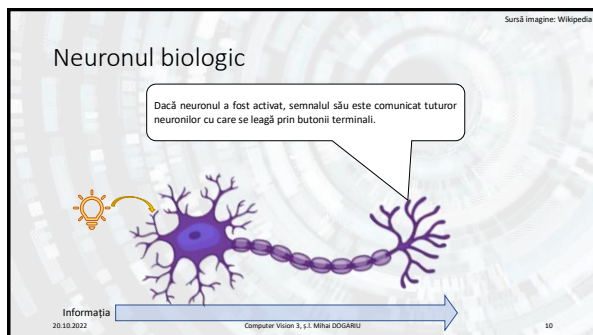
7



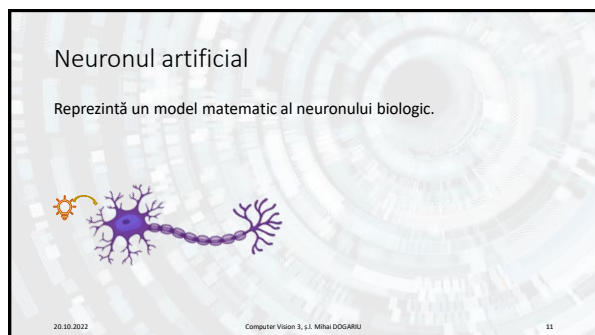
8



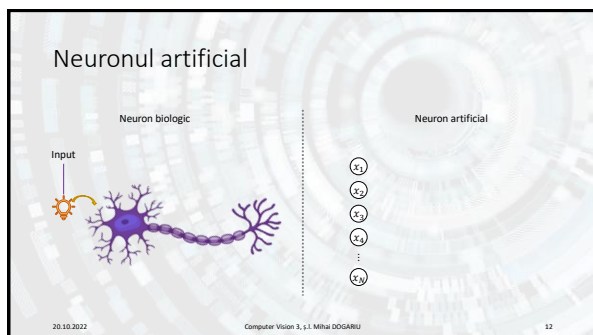
9



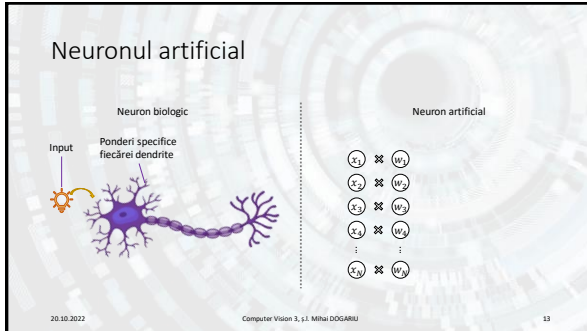
10



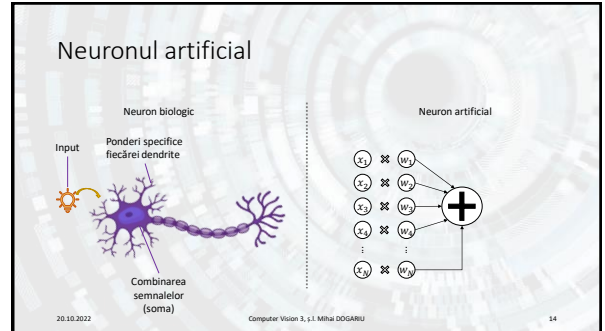
11



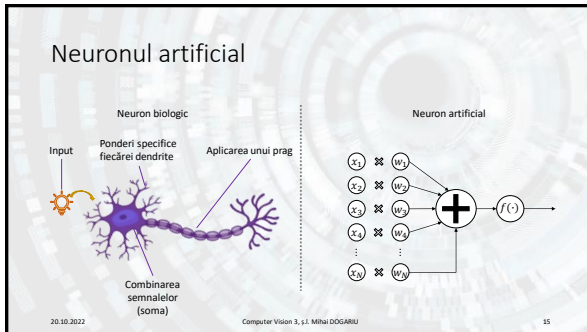
12



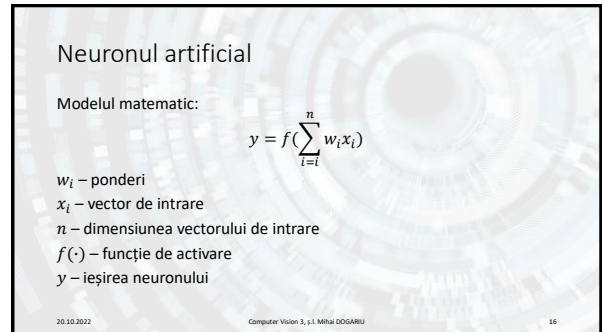
13



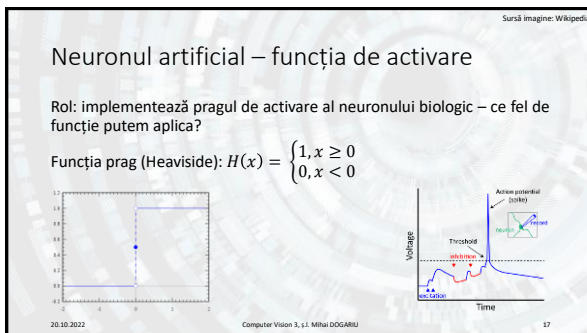
14



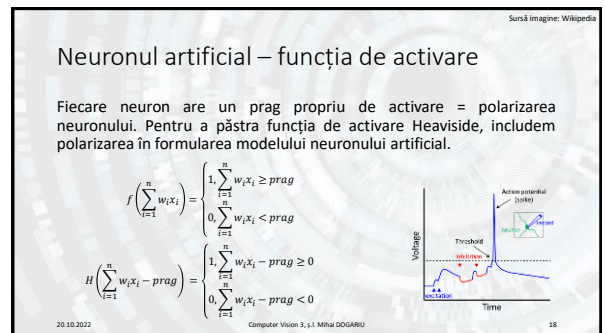
15



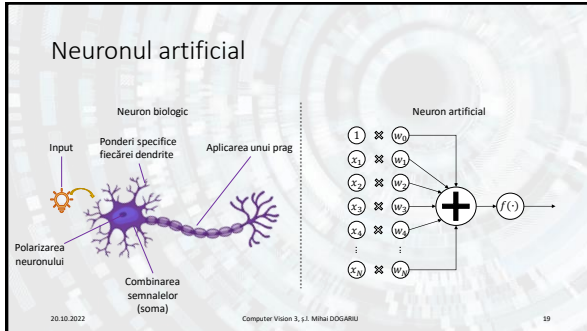
16



17



18



19

Neuronul artificial

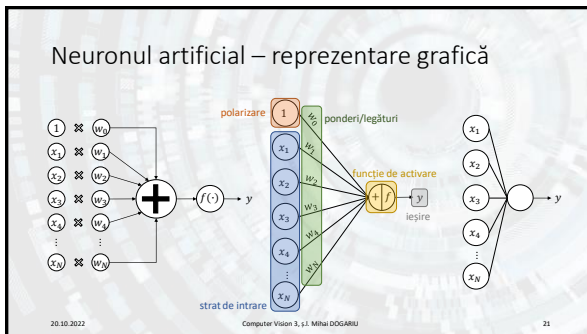
Modelul matematic:

$$y = f\left(\sum_{i=0}^n w_i x_i\right) = f\left(\sum_{i=1}^n w_i x_i + b\right) = f(\mathbf{w} \cdot \mathbf{x} + b)$$

w_i – ponderi; $w_0 \rightarrow b$ - polarizare (bias)
 x_i – vector de intrare; $x_0 = 1$
 n – dimensiunea vectorului de intrare
 $f(\cdot)$ – funcție de activare; $f(\cdot) = H(\cdot) \Rightarrow$ **perceptron (clasificator binar)**
 y – ieșirea neuronului

20.10.2022 Computer Vision 3, 1.1 Mihai DOGARU 20

20



21

Perceptronul

- Caz particular al neuronului artificial
- Folosește funcția de activare Heaviside
- Poate fi folosit pentru a implementa doar modele liniar separabile

20.10.2022 Computer Vision 3, 1.1 Mihai DOGARU 22

22

Spațiu liniar separabil

Un spațiu bidimensional de puncte făcând parte din 2 tipuri (clase) diferite de puncte este liniar separabil dacă există o dreaptă care separă planul în cele 2 clase distincte. Această dreaptă nu este unică.

20.10.2022 Computer Vision 3, 1.1 Mihai DOGARU 23

23

Spațiu liniar separabil

Un spațiu bidimensional de puncte făcând parte din 2 tipuri (clase) diferite de puncte este liniar separabil dacă există o dreaptă care separă planul în cele 2 clase distincte. Această dreaptă nu este unică.

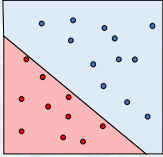
20.10.2022 Computer Vision 3, 1.1 Mihai DOGARU 24

24

Sură imagine: Pinterest

Spațiu liniar separabil

Un spațiu bidimensional de puncte făcând parte din 2 tipuri (clase) diferite de puncte este liniar separabil dacă există o dreaptă care separă planul în cele 2 clase distincte. Această dreaptă nu este unică.



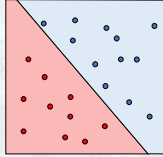
20.10.2022 Computer Vision 3, s.1. Mihai DOGARU 25

25

Sură imagine: Pinterest

Spațiu liniar separabil

Un spațiu bidimensional de puncte făcând parte din 2 tipuri (clase) diferite de puncte este liniar separabil dacă există o dreaptă care separă planul în cele 2 clase distincte. Această dreaptă nu este unică.



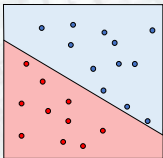
20.10.2022 Computer Vision 3, s.1. Mihai DOGARU 26

26

Sură imagine: Pinterest

Spațiu liniar separabil

Un spațiu bidimensional de puncte făcând parte din 2 tipuri (clase) diferite de puncte este liniar separabil dacă există o dreaptă care separă planul în cele 2 clase distincte. Această dreaptă nu este unică.



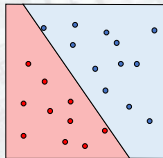
20.10.2022 Computer Vision 3, s.1. Mihai DOGARU 27

27

Sură imagine: Pinterest

Spațiu liniar separabil

Un spațiu bidimensional de puncte făcând parte din 2 tipuri (clase) diferite de puncte este liniar separabil dacă există o dreaptă care separă planul în cele 2 clase distincte. Această dreaptă nu este unică.




20.10.2022 Computer Vision 3, s.1. Mihai DOGARU 28

28

Sură imagine: veryicon

Învățarea unui perceptron

Învățarea = procesul iterativ prin care sunt găsite valorile tuturor parametrilor perceptronului astfel încât modelul final să clasifice corect în mod binar datele de intrare.



Ce parametri sunt fiși?

- input
- output
- funcția de transfer

Ce parametri pot fi ajustați/învățați?

- ponderile
- polarizarea

20.10.2022 Computer Vision 3, s.1. Mihai DOGARU 29

29

Sură imagine: veryicon

Învățarea unui perceptron

Notății:

- $D = \{(x_1, \hat{y}_1), (x_2, \hat{y}_2), \dots, (x_n, \hat{y}_n)\}$ – setul de date de antrenare;
- x_j – al j -lea vector de intrare n -dimensional;
- $x_{j,i}$ – a j -a componentă a celui de-al j -lea vector de intrare;
- \hat{y}_j – a j -a ieșire dorită, corespunzând lui $x_j \leftrightarrow$ groundtruth;
- $y = f(x)$ – ieșirea sistemului pentru vectorul de intrare x ;
- α – rata de învățare, ce va avea o valoare între 0 și 1;
- w_i – ponderea de pe poziția i , care se va înmulți cu componenta i a vectorului de intrare;
- $x_{j,0} = 1, \forall j$;
- w_0 – polarizarea;
- $w_i(t)$ – valoarea ponderii w_i la momentul de timp t .

20.10.2022 Computer Vision 3, s.1. Mihai DOGARU 30

30

Sursă imagine: vericon

Învățarea unui perceptron

Algoritm:

1. inițializarea ponderilor la o valoare mică.
2. pentru fiecare pereche (x_j, y_j) din setul de date de antrenare:
 - 2.1. calculăm ieșirea sistemului:

$$y_j(t) = f\left(\sum_{i=0}^n w_i x_{j,i}\right)$$
 - 2.2. actualizăm ponderile, $\forall i, 0 \leq i \leq n$:

$$w_i(t+1) = w_i(t) + \alpha * (y_j - y_j(t)) * x_{j,i}$$
3. repetăm pasul 2 până când eroarea iteratiei este mai mică decât un prag prestabilit sau până când s-au rulat un anumit număr de iterații.

20.10.2022 Computer Vision 3, I.I. Mihai DOGARU 31

31

Sursă imagine: vericon

Învățarea unui perceptron

Aplicație

Să se realizeze un perceptron care implementează funcția logică AND.

AND	0	1
0	0	0
1	0	1

	Input	Output
j=1	0 0	0
j=2	0 1	0
j=3	1 0	0
j=4	1 1	1

$D = \{(0,0,0), (0,1,0), (1,0,0), (1,1,1)\}$
 $\alpha = 1$

20.10.2022 Computer Vision 3, I.I. Mihai DOGARU 32

32

Sursă imagine: vericon

Învățarea unui perceptron

1. Inițializăm toate ponderile (o singură dată):

$$w_0 = w_1 = w_2 = 0$$
- 2.1. calculăm ieșirea sistemului:

$$y_1(0) = f\left(\sum_{i=0}^n w_i x_{1,i}\right)$$

$$= f(w_0 x_{1,0} + w_1 x_{1,1} + w_2 x_{1,2})$$

$$= f(0 * 1 + 0 * 0 + 0 * 0)$$

$$= f(0)$$

$$= 1$$

$f(x) = \begin{cases} 1, x \geq 0 \\ 0, x < 0 \end{cases}$
 $t = 0 \quad j = 1$
 $w_0 = 0; x_{1,0} = 1$
 $w_1 = 0; x_{1,1} = 0$
 $w_2 = 0; x_{1,2} = 0$
 $\bar{y}_1 = 0$

20.10.2022 Computer Vision 3, I.I. Mihai DOGARU 33

33

Sursă imagine: vericon

Învățarea unui perceptron

- 2.2. actualizăm ponderile, $\forall i, 0 \leq i \leq n$:

$$w_0(1) = w_0(0) + \alpha * (\bar{y}_1 - y_1(0)) * x_{1,0}$$

$$w_1(1) = w_1(0) + \alpha * (\bar{y}_1 - y_1(0)) * x_{1,1}$$

$$w_2(1) = w_2(0) + \alpha * (\bar{y}_1 - y_1(0)) * x_{1,2}$$

$$w_0(1) = 0 + 1 * (0 - 1) * 1 = -1$$

$$w_1(1) = 0 + 1 * (0 - 1) * 0 = 0$$

$$w_2(1) = 0 + 1 * (0 - 1) * 0 = 0$$

$f(x) = \begin{cases} 1, x \geq 0 \\ 0, x < 0 \end{cases}$
 $t = 0 \quad j = 1$
 $w_0 = 0; x_{1,0} = 1$
 $w_1 = 0; x_{1,1} = 0$
 $w_2 = 0; x_{1,2} = 0$
 $\bar{y}_1 = 0$

20.10.2022 Computer Vision 3, I.I. Mihai DOGARU 34

34

Sursă imagine: vericon

Învățarea unui perceptron

- 2.1. calculăm ieșirea sistemului:

$$y_2(1) = f\left(\sum_{i=0}^n w_i x_{2,i}\right)$$

$$= f(w_0 x_{2,0} + w_1 x_{2,1} + w_2 x_{2,2})$$

$$= f(-1 * 1 + 0 * 0 + 0 * 1)$$

$$= f(-1)$$

$$= 0$$

$f(x) = \begin{cases} 1, x \geq 0 \\ 0, x < 0 \end{cases}$
 $t = 1 \quad j = 2$
 $w_0 = -1; x_{2,0} = 1$
 $w_1 = 0; x_{2,1} = 0$
 $w_2 = 0; x_{2,2} = 1$
 $\bar{y}_2 = 0$

20.10.2022 Computer Vision 3, I.I. Mihai DOGARU 35

35

Sursă imagine: vericon

Învățarea unui perceptron

- 2.2. actualizăm ponderile, $\forall i, 0 \leq i \leq n$:

$$w_0(2) = w_0(1) + \alpha * (\bar{y}_2 - y_2(1)) * x_{2,0}$$

$$w_1(2) = w_1(1) + \alpha * (\bar{y}_2 - y_2(1)) * x_{2,1}$$

$$w_2(2) = w_2(1) + \alpha * (\bar{y}_2 - y_2(1)) * x_{2,2}$$

$$w_0(2) = -1 + 1 * (0 - 0) * 1 = -1$$

$$w_1(2) = 0 + 1 * (0 - 0) * 0 = 0$$

$$w_2(2) = 0 + 1 * (0 - 0) * 1 = 0$$

$f(x) = \begin{cases} 1, x \geq 0 \\ 0, x < 0 \end{cases}$
 $t = 1 \quad j = 2$
 $w_0 = -1; x_{2,0} = 1$
 $w_1 = 0; x_{2,1} = 0$
 $w_2 = 0; x_{2,2} = 1$
 $\bar{y}_2 = 0$

20.10.2022 Computer Vision 3, I.I. Mihai DOGARU 36

36

Învățarea unui perceptron

2.1. calculăm ieșirea sistemului:

$$y_3(2) = f\left(\sum_{i=0}^n w_i x_{3,i}\right)$$

$$= f(w_0 x_{3,0} + w_1 x_{3,1} + w_2 x_{3,2})$$

$$= f(-1 * 1 + 0 * 1 + 0 * 0)$$

$$= f(-1)$$

$$= 0$$

$f(x) = \begin{cases} 1, x \geq 0 \\ 0, x < 0 \end{cases}$
 $t = 2 \quad j = 3$
 $w_0 = -1; \quad x_{3,0} = 1$
 $w_1 = 0; \quad x_{3,1} = 1$
 $w_2 = 0; \quad x_{3,2} = 0$
 $\bar{y}_3 = 0$

20.10.2022 Computer Vision 3, 1.1. Mihai DOGARU 37

37

Învățarea unui perceptron

2.2. actualizăm ponderile, $\forall i, 0 \leq i \leq n$:

$$w_0(3) = w_0(2) + \alpha * (\bar{y}_3 - y_3(2)) * x_{3,0}$$

$$w_1(3) = w_1(2) + \alpha * (\bar{y}_3 - y_3(2)) * x_{3,1}$$

$$w_2(3) = w_2(2) + \alpha * (\bar{y}_3 - y_3(2)) * x_{3,2}$$

$$w_0(3) = -1 + 1 * (0 - 0) * 1 = -1$$

$$w_1(3) = 0 + 1 * (0 - 0) * 1 = 0$$

$$w_2(3) = 0 + 1 * (0 - 0) * 0 = 0$$

$f(x) = \begin{cases} 1, x \geq 0 \\ 0, x < 0 \end{cases}$
 $t = 2 \quad j = 3$
 $w_0 = -1; \quad x_{3,0} = 1$
 $w_1 = 0; \quad x_{3,1} = 1$
 $w_2 = 0; \quad x_{3,2} = 0$
 $\bar{y}_3 = 0$

20.10.2022 Computer Vision 3, 1.1. Mihai DOGARU 38

38

Învățarea unui perceptron

2.1. calculăm ieșirea sistemului:

$$y_4(3) = f\left(\sum_{i=0}^n w_i x_{4,i}\right)$$

$$= f(w_0 x_{4,0} + w_1 x_{4,1} + w_2 x_{4,2})$$

$$= f(-1 * 1 + 0 * 1 + 0 * 1)$$

$$= f(-1)$$

$$= 0$$

$f(x) = \begin{cases} 1, x \geq 0 \\ 0, x < 0 \end{cases}$
 $t = 3 \quad j = 4$
 $w_0 = -1; \quad x_{4,0} = 1$
 $w_1 = 0; \quad x_{4,1} = 1$
 $w_2 = 0; \quad x_{4,2} = 1$
 $\bar{y}_4 = 1$

20.10.2022 Computer Vision 3, 1.1. Mihai DOGARU 39

39

Învățarea unui perceptron

2.2. actualizăm ponderile, $\forall i, 0 \leq i \leq n$:

$$w_0(4) = w_0(3) + \alpha * (\bar{y}_4 - y_4(3)) * x_{4,0}$$

$$w_1(4) = w_1(3) + \alpha * (\bar{y}_4 - y_4(3)) * x_{4,1}$$

$$w_2(4) = w_2(3) + \alpha * (\bar{y}_4 - y_4(3)) * x_{4,2}$$

$$w_0(4) = -1 + 1 * (1 - 0) * 1 = 0$$

$$w_1(4) = 0 + 1 * (1 - 0) * 1 = 1$$

$$w_2(4) = 0 + 1 * (1 - 0) * 1 = 1$$

$f(x) = \begin{cases} 1, x \geq 0 \\ 0, x < 0 \end{cases}$
 $t = 3 \quad j = 4$
 $w_0 = -1; \quad x_{4,0} = 1$
 $w_1 = 0; \quad x_{4,1} = 1$
 $w_2 = 0; \quad x_{4,2} = 1$
 $\bar{y}_4 = 1$

20.10.2022 Computer Vision 3, 1.1. Mihai DOGARU 40

40

Învățarea unui perceptron

2.1. calculăm ieșirea sistemului:

$$y_1(4) = f\left(\sum_{i=0}^n w_i x_{1,i}\right)$$

$$= f(w_0 x_{1,0} + w_1 x_{1,1} + w_2 x_{1,2})$$

$$= f(0 * 1 + 1 * 0 + 1 * 0)$$

$$= f(0)$$

$$= 1$$

$f(x) = \begin{cases} 1, x \geq 0 \\ 0, x < 0 \end{cases}$
 $t = 4 \quad j = 1$
 $w_0 = 0; \quad x_{1,0} = 1$
 $w_1 = 1; \quad x_{1,1} = 0$
 $w_2 = 1; \quad x_{1,2} = 0$
 $\bar{y}_1 = 0$

S-a efectuat o trecere completă prin baza de date

20.10.2022 Computer Vision 3, 1.1. Mihai DOGARU 41

41

Învățarea unui perceptron

2.2. actualizăm ponderile, $\forall i, 0 \leq i \leq n$:

$$w_0(5) = w_0(4) + \alpha * (\bar{y}_1 - y_1(4)) * x_{1,0}$$

$$w_1(5) = w_1(4) + \alpha * (\bar{y}_1 - y_1(4)) * x_{1,1}$$

$$w_2(5) = w_2(4) + \alpha * (\bar{y}_1 - y_1(4)) * x_{1,2}$$

$$w_0(5) = 0 + 1 * (0 - 1) * 1 = -1$$

$$w_1(5) = 1 + 1 * (0 - 1) * 0 = 1$$

$$w_2(5) = 1 + 1 * (0 - 1) * 0 = 1$$

$f(x) = \begin{cases} 1, x \geq 0 \\ 0, x < 0 \end{cases}$
 $t = 4 \quad j = 1$
 $w_0 = 0; \quad x_{1,0} = 1$
 $w_1 = 1; \quad x_{1,1} = 0$
 $w_2 = 1; \quad x_{1,2} = 0$
 $\bar{y}_1 = 0$

20.10.2022 Computer Vision 3, 1.1. Mihai DOGARU 42

42

Învățarea unui perceptron

Sursă imagine: verrycon

	$x_{j,0}$	$x_{j,1}$	$x_{j,2}$	\hat{y}_j	y_j	Δy	W_0	W_1	W_2
Iterația #1	j=1	1	0	0	0	1	-1	-1	0
	j=2	1	0	1	0	0	0	-1	0
	j=3	1	1	0	0	0	0	-1	0
	j=4	1	1	1	1	0	1	0	1
Iterația #2	j=1	1	0	0	0	1	-1	-1	1
	j=2	1	0	1	0	1	-1	-2	1
	j=3	1	1	0	0	0	0	-2	1
	j=4	1	1	1	1	0	1	-1	2

20.10.2022 Computer Vision 3, 1.1. Mihai DOGARU 43

43

Învățarea unui perceptron

Sursă imagine: verrycon

	$x_{j,0}$	$x_{j,1}$	$x_{j,2}$	\hat{y}_j	y_j	Δy	W_0	W_1	W_2
Iterația #3	j=1	1	0	0	0	0	0	-1	2
	j=2	1	0	1	0	1	-1	-2	2
	j=3	1	1	0	0	1	-1	-3	1
	j=4	1	1	1	1	0	1	-2	2
Iterația #4	j=1	1	0	0	0	0	0	-2	2
	j=2	1	0	1	0	0	0	-2	2
	j=3	1	1	0	0	1	-1	-3	1
	j=4	1	1	1	1	0	1	-2	2

20.10.2022 Computer Vision 3, 1.1. Mihai DOGARU 44

44

Învățarea unui perceptron

Sursă imagine: verrycon

	$x_{j,0}$	$x_{j,1}$	$x_{j,2}$	\hat{y}_j	y_j	Δy	W_0	W_1	W_2
Iterația #5	j=1	1	0	0	0	0	-2	2	2
	j=2	1	0	1	0	1	-1	-3	2
	j=3	1	1	0	0	0	0	-3	2
	j=4	1	1	1	1	1	0	-3	2
Iterația #6	j=1	1	0	0	0	0	0	-3	2
	j=2	1	0	1	0	0	0	-3	2
	j=3	1	1	0	0	0	0	-3	2
	j=4	1	1	1	1	1	0	-3	2

20.10.2022 Computer Vision 3, 1.1. Mihai DOGARU 45

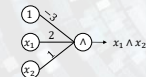
45

Învățarea unui perceptron

Sursă imagine: verrycon

	$x_{j,0}$	$x_{j,1}$	$x_{j,2}$	\hat{y}_j	y_j	Δy	W_0	W_1	W_2
Iterația #6	j=1	1	0	0	0	0	0	-3	2
	j=2	1	0	1	0	0	0	-3	2
	j=3	1	1	0	0	0	0	-3	2
	j=4	1	1	1	1	1	0	-3	2

Dacă nu apare nicio eroare pe parcursul unei iterații => s-a ajuns la convergența algoritmului



20.10.2022 Computer Vision 3, 1.1. Mihai DOGARU 46

46

Concluzii perceptron

- Clasificator liniar binar ⇔ împarte planul datelor cu un hiperplan conform ecuației unei drepte de ecuație $\mathbf{w} \cdot \mathbf{x} = b$;
- Funcție de activare neliniară;
- Pentru a obține moduri de separare mai complexe se pot folosi alte tipuri de ecuații neliniare.

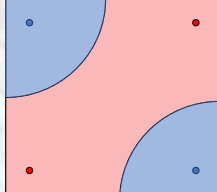
20.10.2022 Computer Vision 3, 1.1. Mihai DOGARU 47

47

Spațiu liniar separabil

Cum procedăm dacă spațiul nu este liniar separabil (e.g. funcția XOR)?

1. Combinăm mai multe funcții liniare în cascadă?
2. Aplicăm non-liniarități?
3. Combinăm mai multe funcții non-liniare în cascadă?



20.10.2022 Computer Vision 3, 1.1. Mihai DOGARU 48

48

Sursă imagine: Santiago Ramón y Cajal

Rețeaua de neuroni unistrat

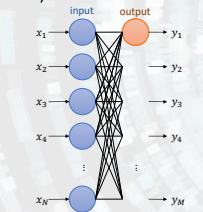
Sistemul nervos nu este format dintr-un singur neuron, ci dintr-o rețea. Numărul total de neuroni nu este cunoscut exact. Se estimează că ar fi 86.1 ± 8.1 miliarde de neuroni în corpul uman [3].



20.10.2022 Computer Vision 3, 1.1. Mihai DOGARU 49

49

Rețeaua de neuroni unistrat



Un singur perceptron poate clasifica informația în doar 2 clase. Mai mult de atât, clasificarea se face „hard” (nu există valori intermediare)

O soluție parțială: Rețea formată din mai mulți perceptroni pe același nivel

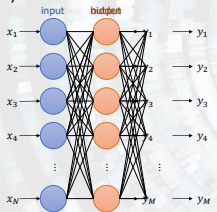
- + Se adaptează mai multor clase;
- În continuare, nu se pot modela spații neliniar separabile;
- Toate ieșirile sunt binare => răspunde unei game restrânse de nevoi

Nu trebuie să fie neapărat egal cu M (în practică, cele două sunt egale foarte rar)

20.10.2022 Computer Vision 3, 1.1. Mihai DOGARU 50

50

Rețeaua de neuroni multistrat



+ O rețea multistrat de perceptroni poate clasifica modele mai complexe de date, e.g. funcția XOR.

- Toate ieșirile sunt binare => răspunde unei game restrânse de nevoi

Nu trebuie să fie neapărat egal cu M (în practică, cele două sunt egale foarte rar)

20.10.2022 Computer Vision 3, 1.1. Mihai DOGARU 51

51

M2.2. Procesul de bază al învățării

20.10.2022 Computer Vision 3, 1.1. Mihai DOGARU 52

52

Procesul de învățare

⁴ **Învățarea mașinilor (machine learning)** – slide #4 = spunem despre un sistem că „învăță” din experiența E cu privire la o clasă de sarcini de lucru T și o măsură de performanță P, dacă performanța sa în rezolvarea sarcinilor T, măsurată prin P, crește cu experiența E [2].

Scopul: găsirea unei funcții care asociază un set de date de intrare cu ieșirea lor corectă, în cel mai bun mod posibil.

Constă în 2 etape:

1. Propagarea înainte (forward propagation);
2. Propagare înapoi (backpropagation).

20.10.2022 Computer Vision 3, 1.1. Mihai DOGARU 53

53

Propagarea înainte

➤ **Scopul:** obținerea ieșirii $y = M(x)$;

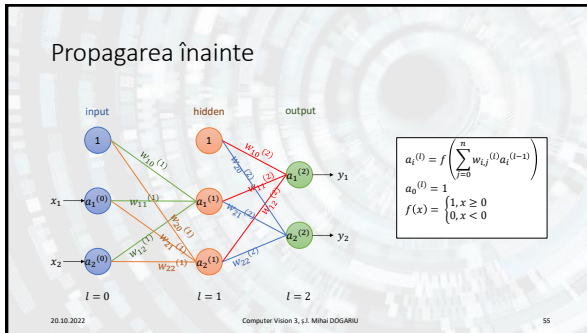
➤ M = funcția de transfer a rețelei neuronale, obținută prin compunerea tuturor funcțiilor de transfer asociate straturilor rețelei.

➤ Presupune parcurgerea rețelei de neuroni de la intrare către ieșire.

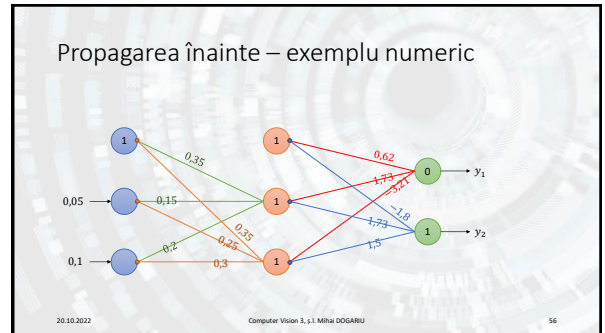
➤ Se reduce la a calcula activările pentru fiecare neuron din rețea, de la straturile inferioare (mai apropiate de input) către cele superioare (mai apropiate de output).

20.10.2022 Computer Vision 3, 1.1. Mihai DOGARU 54

54



55



56

Backpropagation

➤ **Scopul:** învățarea reprezentării interne a unei rețele neuronale. Presupune aflarea tuturor ponderilor (și polarizărilor) care aduc ieșirea modelului la o formă cât mai apropiată, ideal identică, cu ieșirea așteptată/reală.

➤ Cum măsurăm ce înseamnă „cât mai apropiată”?

- R: folosind o funcție de cost, e.g. „eroarea pătratică medie” (mean square error).

➤ Ce regulă folosim pentru a modifica ponderile?

- R: scăderea gradientului (gradient descent).

20.10.2022 Computer Vision 3, p.1 Mihai DOGARU 57

57



58



59



60



61



62

Gradient descent

➤ Metodă de a calcula gradientul parametrilor unei rețele neuronale.

$$y = f(\mathbf{w} \cdot \mathbf{x} + b) = f(\mathbf{w}^T \mathbf{x} + b)$$

Funcție de pierdere (se calculează pe o singură pereche de eșantioane)

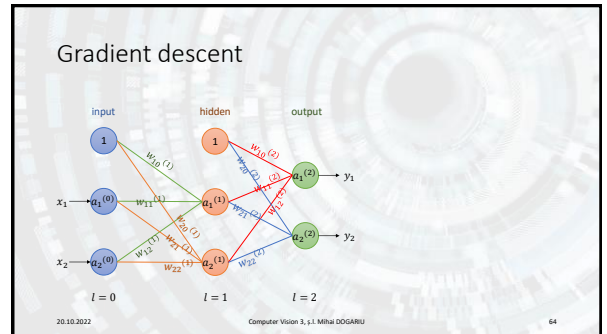
$$\mathcal{L}(y, \hat{y}) \stackrel{e.g.}{\Rightarrow} \mathcal{L}(y, \hat{y}) = (y - \hat{y})^2$$

Funcție de cost (se calculează pe un set de perechi de eșantioane)

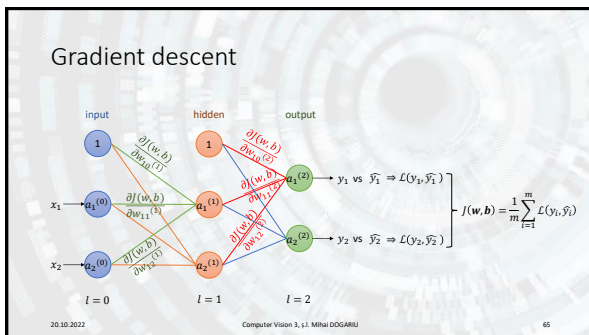
$$J(\mathbf{w}, \mathbf{b}) = \frac{1}{m} \sum_{l=1}^m \mathcal{L}(y_l, \hat{y}_l)$$

20.10.2022 Computer Vision 3, p.1 Mihai DOĞARU 63

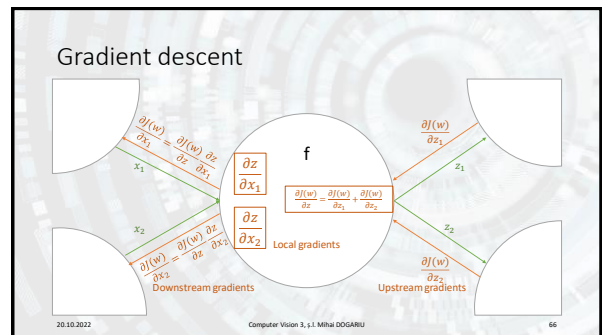
63



64



65



66

Gradient descent

➤ Actualizarea ponderilor se face după regula:

$$w^{(t+1)} = w^{(t)} - \alpha \frac{\partial J(w)}{\partial w^{(t)}}$$

Scriere simplificată:

$$w := w - \alpha \frac{\partial J(w)}{\partial w}$$

➤ Pentru ponderi din interiorul rețelei se aplică regula înlănțuirii derivatelor parțiale:

$$\frac{\partial z}{\partial x} = \frac{\partial z}{\partial y} \frac{\partial y}{\partial x}$$

20.10.2022

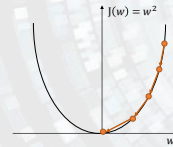
Computer Vision 3, I.I. Mihai DOGARU

67

67

Gradient descent

➤ Exemplu grafic: considerăm cazul simplificat în care funcția de cost depinde de o singură variabilă și alegem funcția pătratică.



$$w := w - \alpha \frac{\partial J(w)}{\partial w}$$

În cazul optim găsim w a.i. $J'(w) = 0$, însă, în practică, acest lucru nu se întâmplă aproape niciodată

20.10.2022

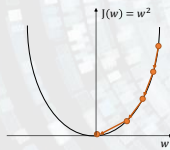
Computer Vision 3, I.I. Mihai DOGARU

68

68

Gradient descent

➤ Exemplu grafic: considerăm cazul simplificat în care funcția de cost depinde de o singură variabilă și alegem funcția pătratică.



$$w := w - \alpha \frac{\partial J(w)}{\partial w}$$

- $\frac{\partial J(w)}{\partial w}$ ne indică direcția în care trebuie făcută modificarea
- α ne indică amplitudinea modificării

20.10.2022

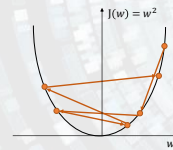
Computer Vision 3, I.I. Mihai DOGARU

69

69

Gradient descent – impactul ratei de învățare

➤ Același exemplu ca mai devreme, însă avem rată de învățare **mare**.



$$w := w - \alpha \frac{\partial J(w)}{\partial w}$$

- Algoritmul poate rata punctul de optim, deoarece fiecare iterație aduce un deplasament mare.
- + Datorită „pașilor” mari, deplasarea se face mai rapid.

20.10.2022

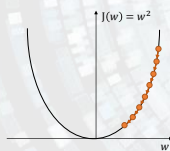
Computer Vision 3, I.I. Mihai DOGARU

70

70

Gradient descent – impactul ratei de învățare

➤ Același exemplu ca mai devreme, însă avem rată de învățare **mică**.



$$w := w - \alpha \frac{\partial J(w)}{\partial w}$$

- Algoritmul ajustează ponderile foarte lent.
- + Datorită „pașilor” mici, sunt șanse mai mici să se rateze punctul optim.
- De ce nu folosim mereu rată de învățare mică?

20.10.2022

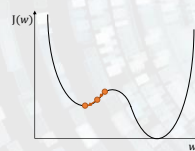
Computer Vision 3, I.I. Mihai DOGARU

71

71

Gradient descent – impactul ratei de învățare

➤ Funcție de cost neconvexă (un minim local nu este neapărat și minimul global), rată de învățare **mică**.



$$w := w - \alpha \frac{\partial J(w)}{\partial w}$$

- Există posibilitatea ca algoritmul să se blocheze într-un minim local.

20.10.2022

Computer Vision 3, I.I. Mihai DOGARU

72

72

Pașii generali pentru învățarea unei rețele

1. Inițializarea parametrilor rețelei
2. Încărcarea datelor de intrare
3. Forward propagation
4. Calcularea funcției de cost prin compararea datelor de ieșire cu ieșirea reală (groundtruth)
5. Backpropagation + actualizarea parametrilor
6. Repetarea pașilor 3 – 5 până la convergență/optimizare.

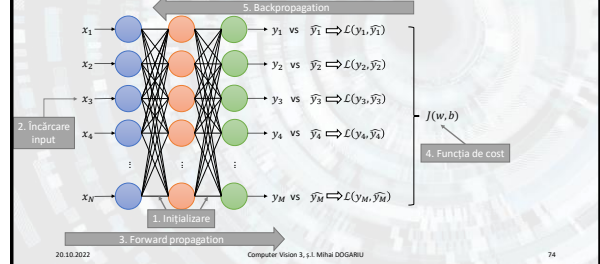
20.10.2022

Computer Vision 3, s.l. Mihai DOGARU

73

73

Pașii generali pentru învățarea unei rețele



20.10.2022

Computer Vision 3, s.l. Mihai DOGARU

74

74

M2.3. Terminologie

20.10.2022

Computer Vision 3, s.l. Mihai DOGARU

75

75

Terminologie

1. Funcții de activare
2. Funcții de cost
3. Rata de învățare (learning rate)
4. Optimizatori
5. Parametri vs hiperparametri
6. Baze de date
7. Regularizare
8. Straturi populare

20.10.2022

Computer Vision 3, s.l. Mihai DOGARU

76

76

1. Funcții de activare

Funcție de activare = operator diferențiabil care transformă semnalul de intrare în cel de ieșire.

- Funcția de activare decide dacă un neuron va fi activat sau nu.
- Poate fi liniară sau neliniară.
- Se alege în funcție de aplicația vizată.

20.10.2022

Computer Vision 3, s.l. Mihai DOGARU

77

77

Funcții de activare liniare

Sunt echivalente cu funcțiile de gradul 1. Reprezintă ecuația unei drepte:

$$f(x) = ax + b$$



➤ $\frac{\partial f}{\partial x} = a \leftarrow \text{constantă};$

- + poate produce rezultate de orice valoare, codomeniul ei putând fi chiar \mathbb{R} ;
- + se folosesc pentru a rezolva probleme simple, precum regresia liniară;
- nu se pretează pentru sarcini mai dificile;
- compunerea mai multor funcții liniare este tot o funcție liniară.

20.10.2022

Computer Vision 3, s.l. Mihai DOGARU

78

78

Funcții de activare liniare

Prezența funcțiilor liniare în toate straturile unei rețele de neuroni anulează prezența mai multor straturi. O astfel de rețea poate fi echivalată cu un singur strat.

$$\begin{aligned}y &= \mathbf{w}\mathbf{x} + b \\f(x) &= ax + ct \\f(y) &= a(\mathbf{w}\mathbf{x} + b) + ct \\&= a\mathbf{w}\mathbf{x} + ab + ct \\&= (a\mathbf{w})\mathbf{x} + (ab + ct) \\&= \mathbf{a}'\mathbf{x} + ct'\end{aligned}$$

20.10.2022

Computer Vision 3, 1.1. Mihai DOGARU

79

79

Funcții de activare neliniare

Corpul uman răspunde neliniar la stimuli. Exemplu – servirea felului de mâncare preferat.

Scenariu:

1. felul de mâncare preferat, în compania persoanei preferate, în mediul preferat => neuronii sunt intens activați, senzație puternică.
2. Cazul (1) + ați servit cu o oră mai devreme o pizza mare, deci sunteți sățuli => neuronii sunt mai puțin activați, senzație mai slabă.
3. Cazul (2) + dar nu ați mai mâncat felul preferat de mai mult de o lună => neuronii sunt mai puternic activați decât la (2), dar mai slab decât în cazul (1)

Preferăm utilizarea funcțiilor neliniare pentru a modela sarcini mai complexe!

20.10.2022

Computer Vision 3, 1.1. Mihai DOGARU

80

80

Funcții de activare neliniare

Sunt funcții cu comportament nelinier, capabile să modeleze spații de date mai complexe. Cu ajutorul lor, se pune în valoare capacitatea de a generaliza a rețelelor neuronale. Cele mai utilizate:

1. treaptă (perceptron);
2. logistică (sigmoid);
3. softplus.
4. tangentă hiperbolică (tanh);
5. ReLU și variantele sale.

20.10.2022

Computer Vision 3, 1.1. Mihai DOGARU

81

81

Funcția treaptă

$$f(x) = \begin{cases} 1, & x \geq 0 \\ 0, & x < 0 \end{cases}$$



$$\frac{\partial f}{\partial x} = \begin{cases} 0, & x \neq 0 \\ \text{nedefinit}, & x = 0 \end{cases}$$

- + cea mai simplă cale de a implementa pragul de activare al neuronului;
- derivata sa nu participă la procesul de antrenare (vezi slide 67).

20.10.2022

Computer Vision 3, 1.1. Mihai DOGARU

82

82

Funcția logistică (sigmoid)

$$f(x) = \sigma(x) = \frac{1}{1 + e^{-x}}$$



$$\frac{\partial f}{\partial x} = f(x)(1 - f(x));$$

➤ Se mai numește și funcție de strivire, deoarece comprimă valorile lui x din $(-\infty, +\infty)$ în $(0, 1)$.

+ Este una dintre cele mai populare funcții de activare, folosită mai ales atunci când ne dorim ca ieșirea să reprezinte o probabilitate.

- argumentele foarte mari sau foarte mici sunt reduse la aceeași valoare;

20.10.2022

Computer Vision 3, 1.1. Mihai DOGARU

83

83

Funcția softplus

$$f(x) = \sigma(x) = \ln(1 + e^x)$$



$$\frac{\partial f}{\partial x} = \frac{1}{1 + e^{-x}};$$

➤ Derivata softplus este chiar funcția logistică.

+ codomeniul este format numai din valori pozitive ⇔ sparsity.

+ utilizată pentru modelarea piețelor financiare.

- complexă din punct de vedere computațional.

20.10.2022

Computer Vision 3, 1.1. Mihai DOGARU

84

84

Funcția tangentă hiperbolică

$$f(x) = \tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$$

- $\frac{\partial f}{\partial x} = 1 - f(x)^2$;
- Comprimă valorile asemănător funcției logistice, însă în intervalul $(-1,1)$.
- + spre deosebire de sigmoid, poate produce și valori negative.
- + de obicei, converge mai repede decât sigmoid.

20.10.2022

Computer Vision 3, I.I. Mihai DOGARU

85

85

Funcția Parametric Rectified Linear Unit (PReLU)

$$f(x) = \begin{cases} x, & x \geq 0 \\ \alpha x, & x < 0 \end{cases}$$

- $\frac{\partial f}{\partial x} = \begin{cases} 1, & x > 0 \\ \alpha, & x < 0; \\ \text{nedefinit}, & x = 0 \end{cases} \quad \alpha = \begin{cases} 0 & \Rightarrow \text{ReLU} \\ 0.01 & \Rightarrow \text{Leaky ReLU} \\ \text{else} & \Rightarrow \text{PReLU} \end{cases}$
- Cea mai populară funcție de activare.
- + complexitate de calcul redusă;

20.10.2022

Computer Vision 3, I.I. Mihai DOGARU

86

86

Funcția softmax

$$f(x_i) = \frac{e^{x_i}}{\sum_{i=1}^V e^{x_i}}$$

- Nu se aplică pe o singură valoare, ci pe un vector de valori.
- Este utilizată pentru probleme de clasificări multi-clasă.
- Este folosită în ultimul strat al rețelei.
- Transformă ieșirea într-o distribuție de probabilități.

20.10.2022

Computer Vision 3, I.I. Mihai DOGARU

87

87

Funcții de activare - probleme



1. Gradient dispărând (vanishing gradient)
 - Derivata funcției de activare la valori mici, propagarea înapoi în rețea devine din ce în ce mai mică, straturile de început ale rețelei ajung să nu se mai modifice pe parcursul antrenării.
 - Funcții afectate: treaptă, sigmoid, tanh (funcții cu codomeniu limitat la ambele capete).
 - Soluții: batch normalization, gradient clipping, greedy layer-wise pre-training, rețele reziduale, inițializare mai bună, alte funcții (e.g. ReLU)

20.10.2022

Computer Vision 3, I.I. Mihai DOGARU

88

88

Funcții de activare - probleme



2. Gradient explodând (exploding gradient)
 - Acumularea unor gradienti de valori mari duce la modificări foarte mari ale parametrilor între iterații succesive => algoritmul de antrenare nu mai converge niciodată.
 - Funcții afectate: toate. Nu depinde de funcții, ci de valorile inițiale ale ponderilor.
 - Soluții: gradient clipping, inițializări mai bune, batch normalization.

20.10.2022

Computer Vision 3, I.I. Mihai DOGARU

89

89

Funcții de activare - probleme



3. Dying ReLU
 - ReLU transformă orice valoare negativă în 0 => derivata pentru valorile negative va fi mereu 0 => există posibilitatea ca o anumită cale a rețelei să fie mereu anulată (moartă).
 - Funcții afectate: ReLU.
 - Soluții: scăderea ratei de învățare, inițializare mai potrivită, Leaky ReLU

20.10.2022

Computer Vision 3, I.I. Mihai DOGARU

90

90

Funcții de activare – ce/când folosim?



- Folosim sigmoid pentru **regresie logistică** și **clasificare binară**;
- În general, tanh este mai bună decât sigmoid fiind centrată în 0;
- Sigmoid și tanh ar trebui evitate datorită **gradientului dispărând**;
- Folosim softmax pentru **clasificări multi-clasă**;
- Rețelele **recurente** folosesc tanh, sigmoid;
- Rețelele convoluționale și complet conectate folosesc ReLU în straturile **ascunse**;
- Dacă anumiți neuroni nu sunt activați niciodată, înlocuim ReLU cu Leaky ReLU. Dacă nici atunci nu sunt activați, folosim Parametric ReLU.

20.10.2022

Computer Vision 3, s.1. Mihai DOGARU

91

Sfârșit M2

20.10.2022

Computer Vision 3, s.1. Mihai DOGARU

92

Bibliografie

- [1] Gross, R. (2015). Psychology: The science of mind and behaviour 7th edition. Hodder Education.
- [2] Mitchell, T. M. (1997). Machine learning (Vol. 1). McGraw-hill New York.
- [3] Azevedo, F. A., Carvalho, L. R., Grinberg, L. T., Farfel, J. M., Ferretti, R. E., Leite, R. E., Jacob Filho, W., Lent, R., & Herculano-Houzel, S. (2009). Equal numbers of neuronal and nonneuronal cells make the human brain an isometrically scaled-up primate brain. The Journal of comparative neurology.

20.10.2022

Computer Vision 3, s.1. Mihai DOGARU

93