Osnovi Računarske Inteligencije

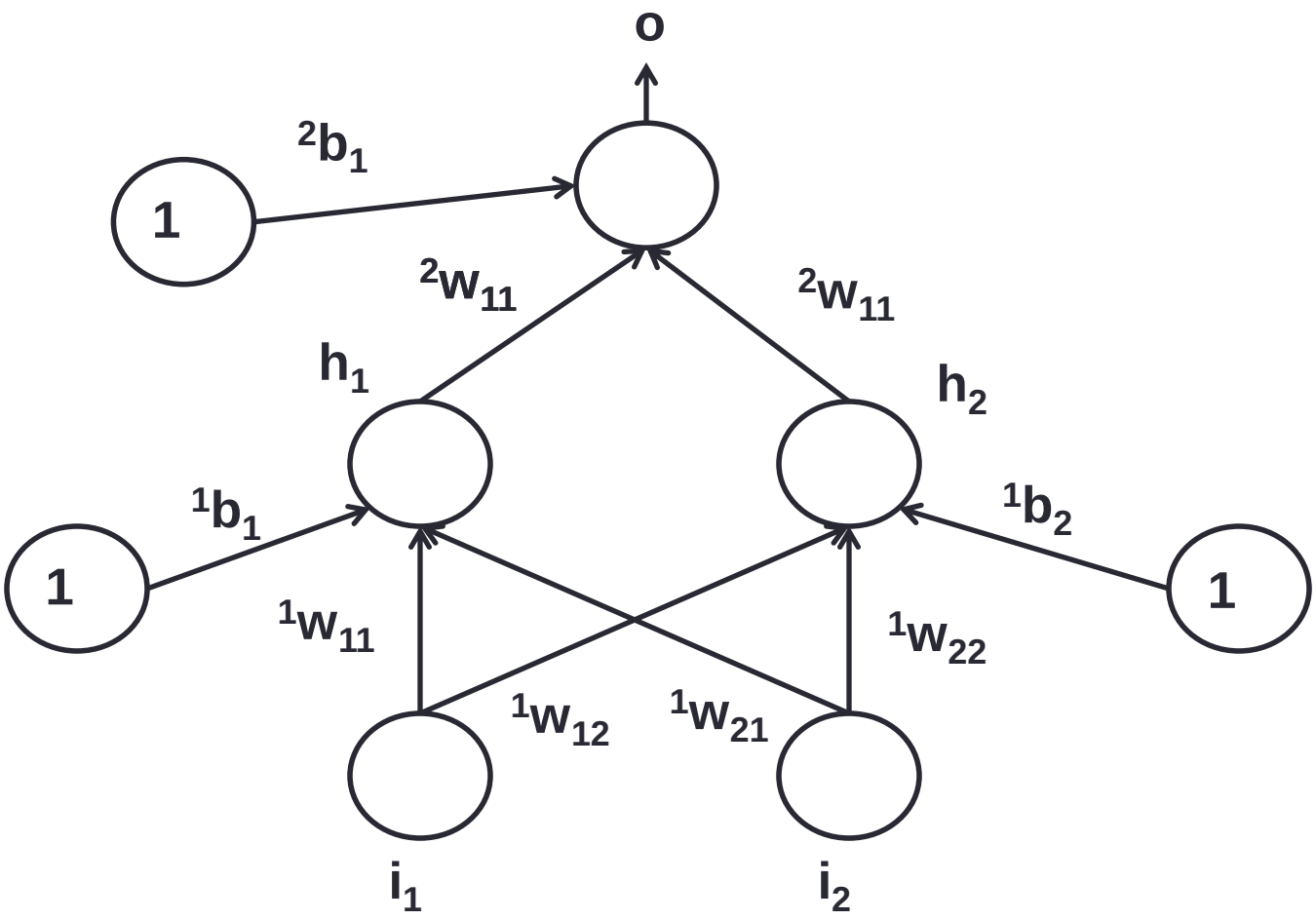
Neuronska Mreža
Primer

predavač: Aleksandar Kovačević

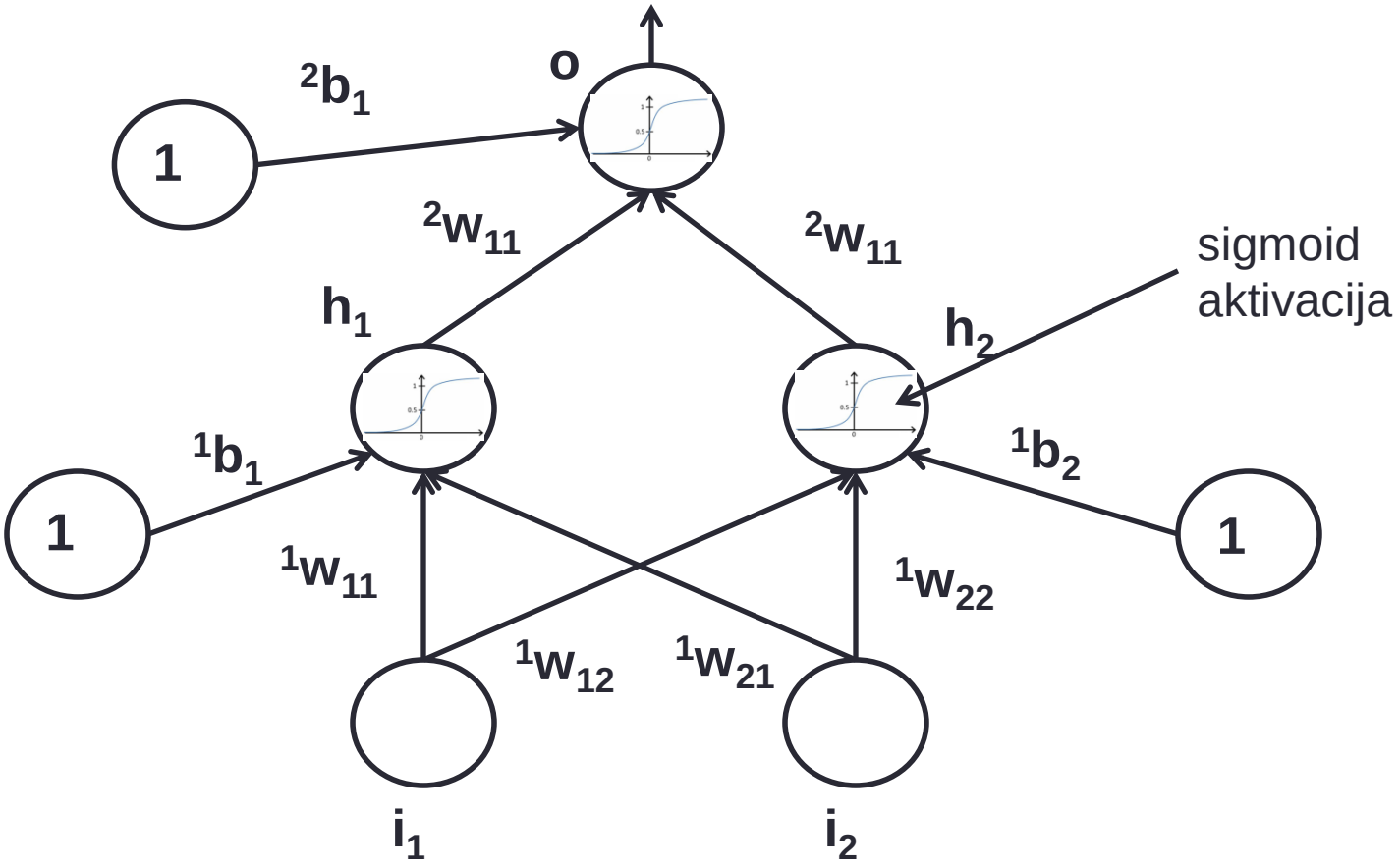
Uvodne napomene

- U nastavku je dat jedan primer obučavanja jednostavne neuronske mreže.
- Arhitektura mreže je odabrana proizvoljno, jedini cilj je da se demostriraju svi koraci obučavanja.
- Skup podataka ima samo jedan primer, vektor [1 1], koji ima zadati izlaz 1.
- Prateći Matlab kod takođe ima za cilj samo da demonstira korake obučavanja. Optimalnost i brzina nisu cilj.

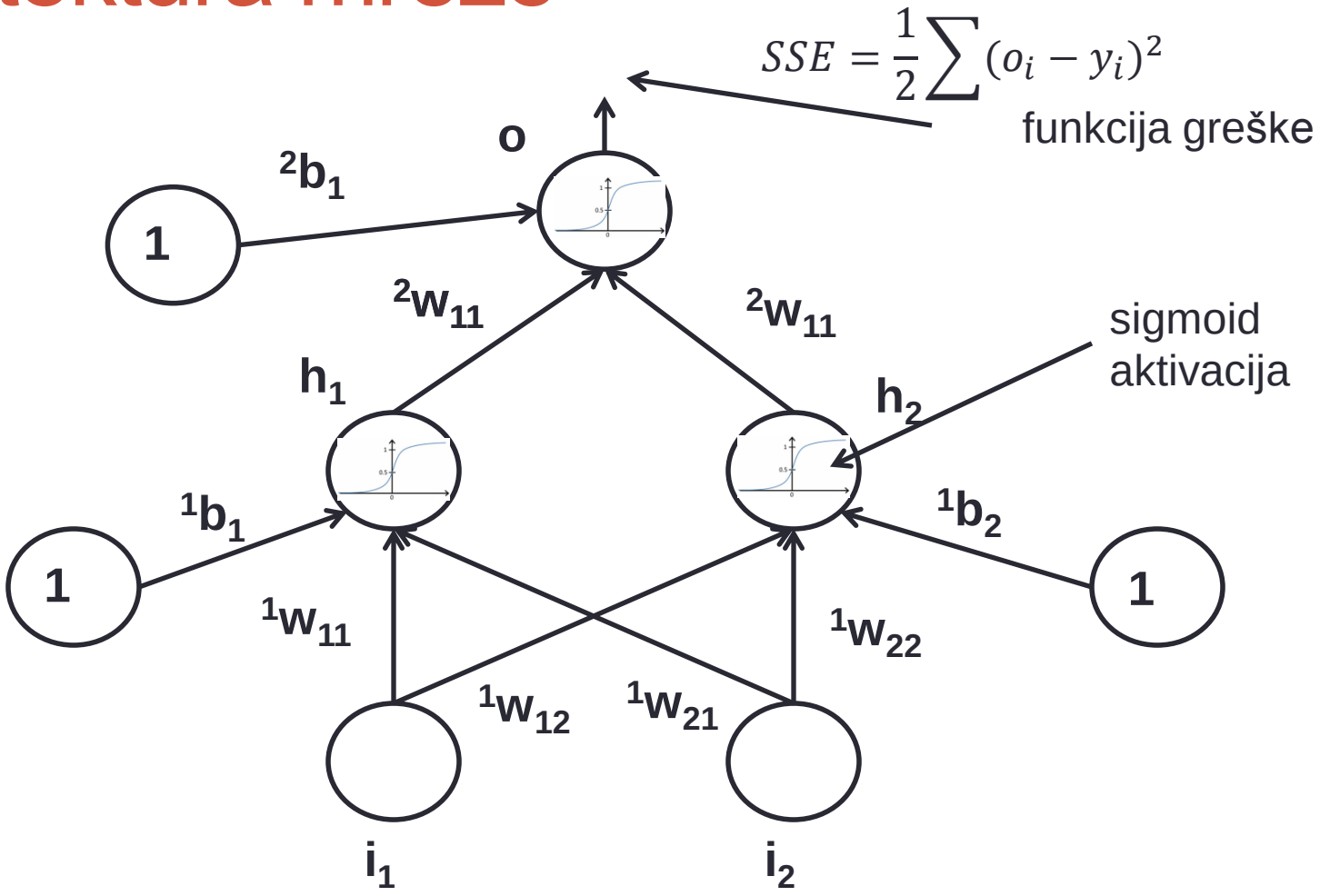
Arhitektura mreže



Arhitektura mreže



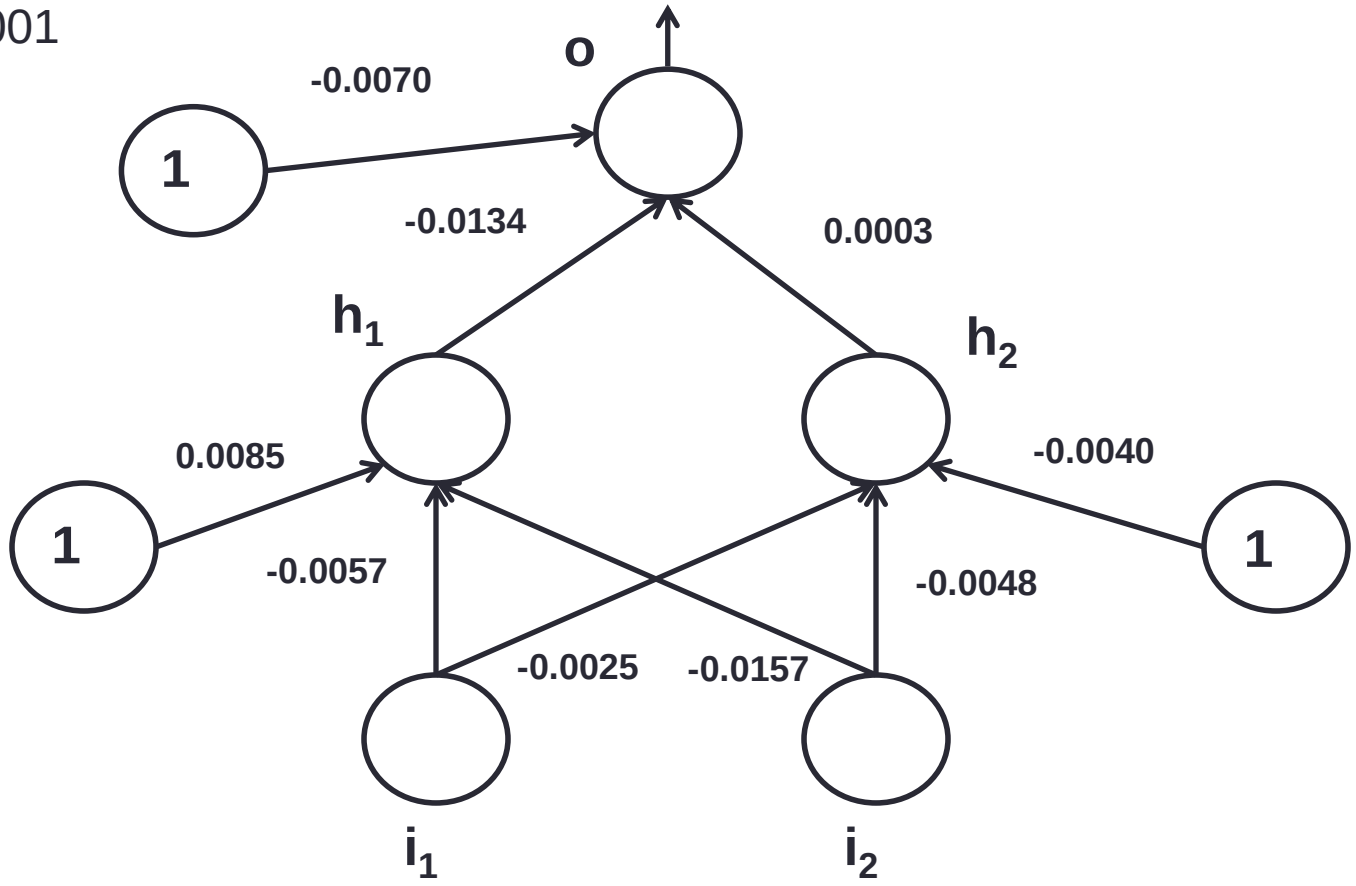
Arhitektura mreže



Inicijalizacija - Matlab

```
W1=randn(2,2)*0.001
W2=randn(1,2)*0.001
%bijas, ulaz,... bice vektori kolone
b1=randn(2,1)*0.001
b2=randn(1,1)*0.001
```

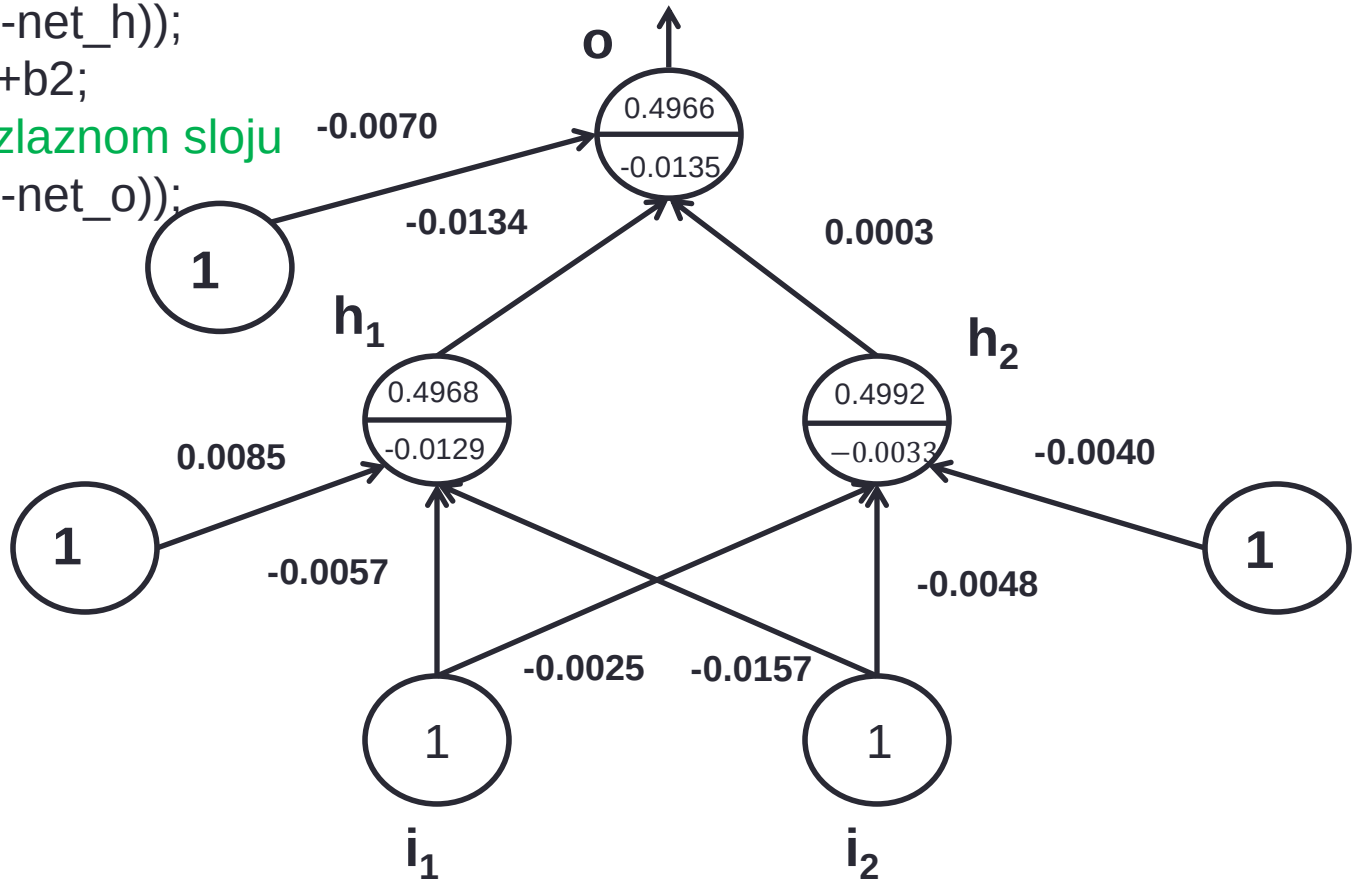
```
W1 =

   -0.0057   -0.0157
   -0.0025   -0.0048

W2 =

   -0.0134    0.0003

b1 =

    0.0085
    0.0040

b2 =

   -0.0070
```



Feed Forward – Izračunavanje unapred

```
%ulaz
input=[1;1]';
%ukupan ulaz u neurone u skrivenom sloju
net_h=W1*input+b;
%aktivacija neurona u skrivenom sloju
activation_h=1./(1+exp(-net_h));
net_o=W2*activation_h+b2;
%aktivacija neurona u izlaznom sloju
activation_o=1./(1+exp(-net_o));
```

```
net_h =
   -0.0129
   -0.0033

activation_h =
    0.4968
    0.4992

net_o =
   -0.0135

activation_o =
    0.4966
```



Feed Forward – Izračunavanje unapred

$$net_{h1} = i_1{}^1w_{11} + i_2{}^1w_{12} + {}^1b_1 = 1 * -0.0057 + 1 * -0.0157 + 0.0085 = -0.0129$$

$$a_{h1} = \frac{1}{1 + e^{-net_{h1}}} = 0.4968$$

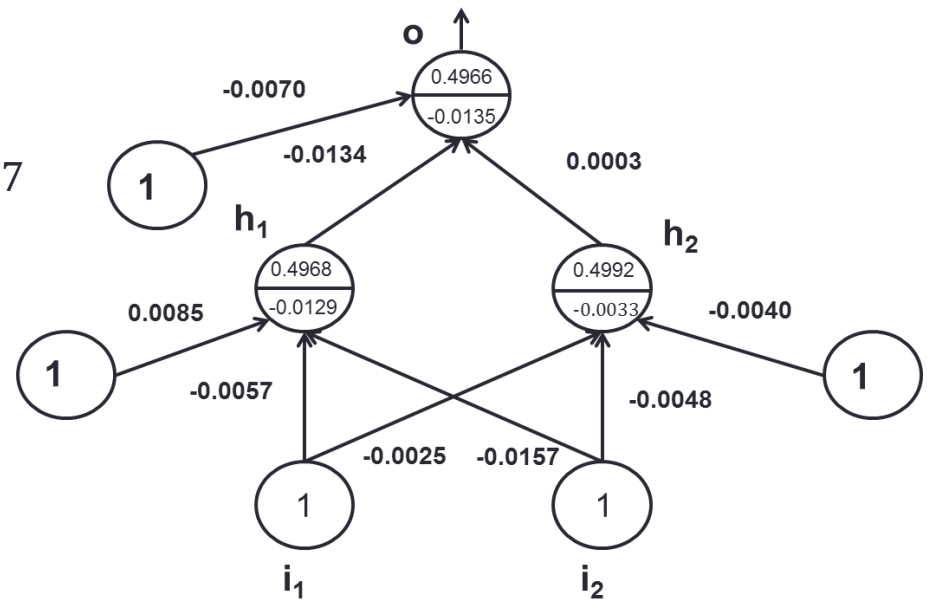
$$net_{h2} = i_1{}^1w_{21} + i_2{}^1w_{22} + {}^1b_1 = 1 * -0.0157 + 1 * -0.0048 - 0.0040 = -0.0033$$

$$a_{h2} = \frac{1}{1 + e^{-net_{h2}}} = 0.4992$$

$$net_0 = h_1{}^2w_{11} + h_2{}^2w_{12} + {}^2b_1 = -0.00225 * -0.0134 + -0.0048 * -0.0003 * -0.0040 = -0.0135$$

$$o = \frac{1}{1 + e^{-net_o}} = 0.4966$$

$$SSE = \frac{1}{2}\sum(o_i - y_i)^2 = 0.5(1 - 0.4966)^2 = 0.1267$$



Backpropagation – Propagacija unazad

```
%tacna klasa je 1
error = 0.5*(activation_o-1)^2
%backpropagation
do = (activation_o-1);
do_activation = (1-activation_o)*activation_o*do;
dw2 = do_activation*activation_h';
db2 = do_activation;
dh = W2'*do_activation;
dh_akt = (1-activation_h).*activation_h.*dh;
dw1 = dh_akt*input'
db1 = dh_akt;
```

$$SSE = \frac{1}{2}\sum (o_i - y_i)^2$$

error = 0.1267

dw2 =

-0.0625 -0.0628

db2 = -0.1258

dh =
0.0017
-0.0000

dh_akt =
1.0e-03 *

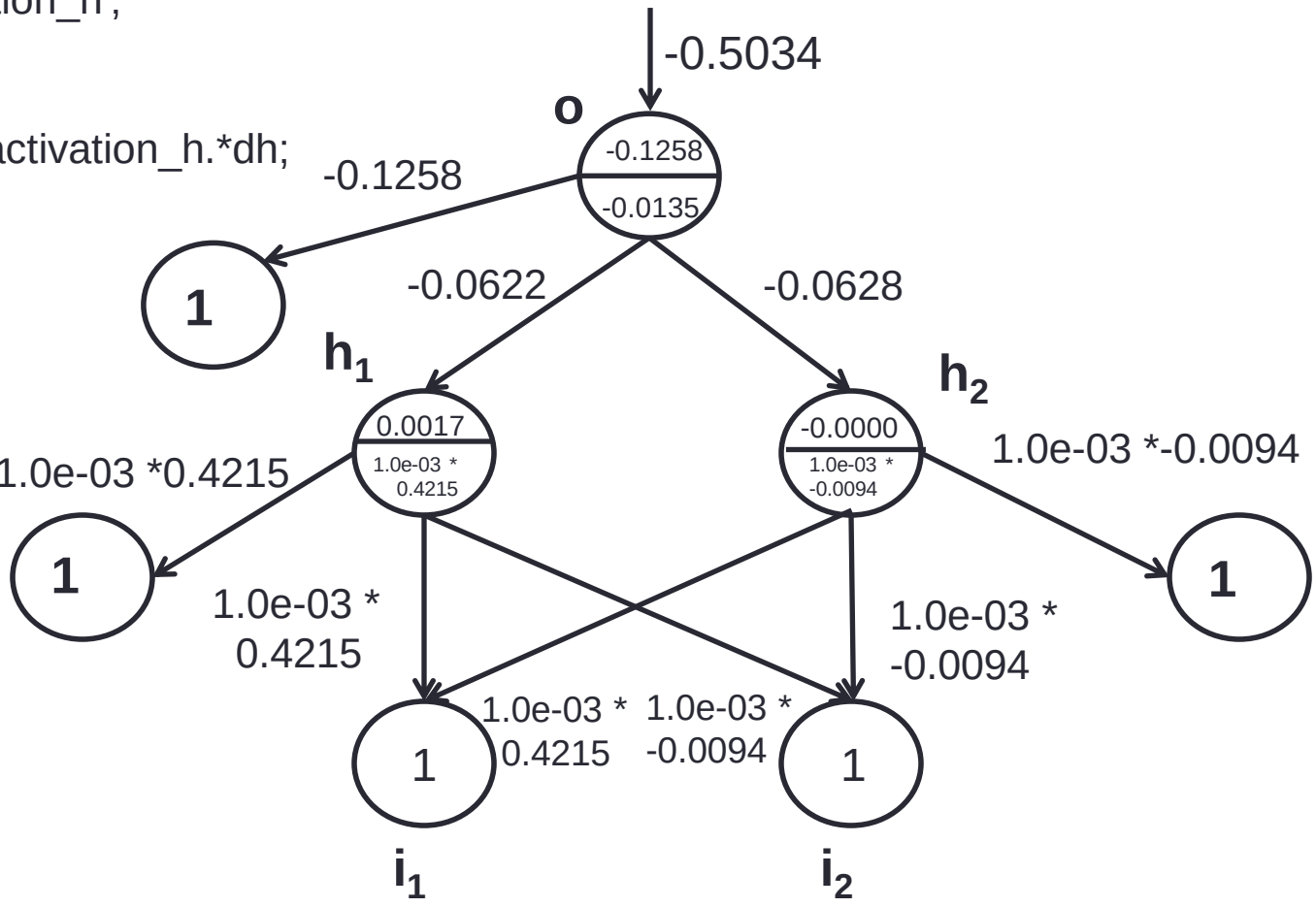
0.4215
-0.0094

dw1 =
1.0e-03 *

0.4215 0.4215
-0.0094 -0.0094

db1 =
1.0e-03 *

0.4215
-0.0094



Backpropagation – Propagacija unazad

$$derror = (o_i - yi) = 0.4966 - 1 = -0.5034$$

izvod SSE

$$da_o = (1\text{-}a_o) * a_o * derror = (1\text{-}0.4966) * 0.4966 * (-0.5034) = -0.1258$$

izvod sigmoid funkcije

Pogledati naredna dva slajda kao podsetnik.

$$dw^2 = dao * ah^T = (-0.1258) * [0.4944 \quad 0.4988] = [-0.0622 \quad -0.0627] \qquad db^2 = da_o$$

$$dh = (w^2)^T * dao = \begin{bmatrix} -0.0134 \\ 0.0003 \end{bmatrix} * (-0.1258) = \begin{bmatrix} 0.0017 \\ -0.0000 \end{bmatrix}$$

$$da_h = (1 - a_h) * a_h * dh = (1 - 0.4944) * 0.4944 * \begin{bmatrix} 0.0017 \\ -0.0000 \end{bmatrix} = \begin{bmatrix} 0.4215 \\ -0.0094 \end{bmatrix} * 10^{-3}$$

$$dw^1 = dah \begin{bmatrix} i_1 \\ i_2 \end{bmatrix}^T = \begin{bmatrix} 0.4215 \\ -0.0094 \end{bmatrix} * 10^{-3} * [1 \quad 1] = \begin{bmatrix} 0.4215 & 0.4215 \\ -0.0094 & -0.0094 \end{bmatrix} * 10^{-3}$$

$$db^1 = da_h = \begin{bmatrix} 0.4215 \\ -0.0094 \end{bmatrix} * 10^{-3}$$

Primer sa vektorima

$$f(x, W) = ||W \cdot x||^2 = \sum_{i=1}^{n} (W \cdot x)_i^2$$

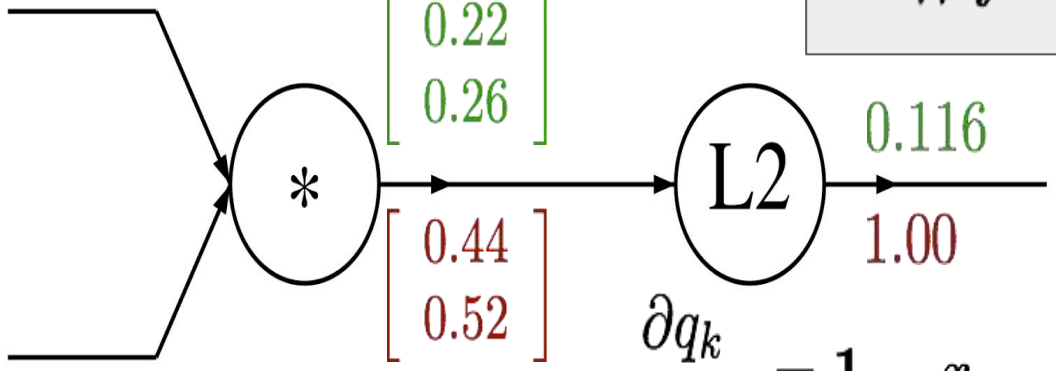
$$\begin{bmatrix} 0.1 & 0.5 \\ -0.3 & 0.8 \end{bmatrix}$$

W

$$\begin{bmatrix} 0.088 & 0.176 \\ 0.104 & 0.208 \end{bmatrix}$$

$$\begin{bmatrix} 0.2 \\ 0.4 \end{bmatrix}$$

x



$$\begin{bmatrix} 0.22 \\ 0.26 \end{bmatrix}$$

$$\begin{bmatrix} 0.44 \\ 0.52 \end{bmatrix}$$

0.116

1.00

$$\nabla_W f = 2q \cdot x^T$$

$$q = W \cdot x = \begin{pmatrix} W_{1,1}x_1 + \cdots + W_{1,n}x_n \\ \vdots \\ W_{n,1}x_1 + \cdots + W_{n,n}x_n \end{pmatrix}$$

$$f(q) = ||q||^2 = q_1^2 + \cdots + q_n^2$$

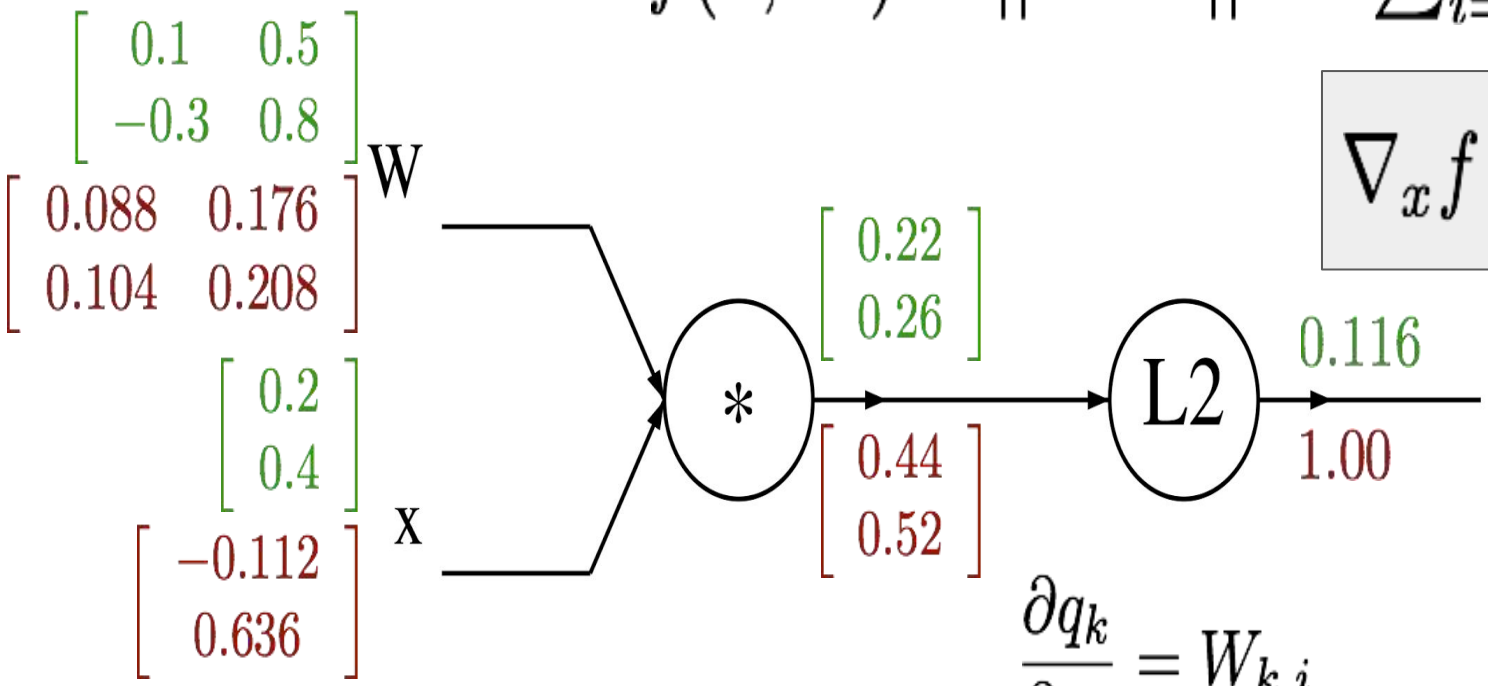
$$\frac{\partial q_k}{\partial W_{i,j}} = 1_{k=i} x_j$$

$$\begin{aligned} \frac{\partial f}{\partial W_{i,j}} &= \sum_k \frac{\partial f}{\partial q_k} \frac{\partial q_k}{\partial W_{i,j}} \\ &= \sum (2q_k)(1_{k=i} x_j) \\ &= 2q_i x_j \end{aligned}$$

Primer sa vektorima

$$f(x, W) = ||W \cdot x||^2 = \sum_{i=1}^{n} (W \cdot x)_i^2$$

$$\nabla_x f = 2W^T \cdot q$$



$$q = W \cdot x = \begin{pmatrix} W_{1,1}x_1 + \cdots + W_{1,n}x_n \\ \vdots \\ W_{n,1}x_1 + \cdots + W_{n,n}x_n \end{pmatrix}$$

$$f(q) = ||q||^2 = q_1^2 + \cdots + q_n^2$$

$$\frac{\partial q_k}{\partial x_i} = W_{k,i}$$

$$\frac{\partial f}{\partial x_i} = \sum_k \frac{\partial f}{\partial q_k} \frac{\partial q_k}{\partial x_i}$$

$$\frac{\partial f}{\partial x_i} = \sum_k 2q_k W_{k,i}$$

Promena vrednosti težina i bijasa – Gradijentni Spust

*nova_vrednost_parametra = stara_vrednost – learning_rate*gradijent_parametra*

- *Learning rate* je 0.1

```
W2 = W2 - 0.1*dw2;
b2 = b2 - 0.1*db2;
W1 = W1 - 0.1*dw1;
b1 = b1 - 0.1*db1;
```

Promena vrednosti težina i bijasa – Gradijentni Spust

- Nakon jedne korekcije parametara dobijamo novi izlaz za ulaz [1 1], kao i novu grešku:

activation_o = 0.5013
error = 0.1243

- Vidimo da je vrednost izlaza veća (cilj nam je da bude 1), a vrednost greške manja (cilj nam je da bude 0).

- Ako prethodno opisani postupak obučavanja mreže ponovimo određeni broj itreacija greška će vremenom pasti na 0 ili veoma malu vrednost.