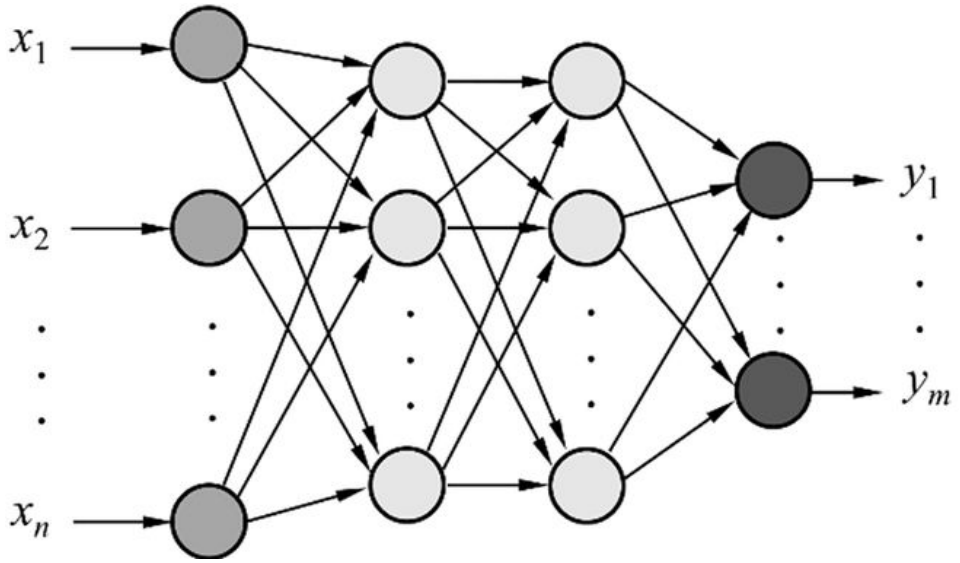
Maciej Stawarz Nr albumu 284310	Inżynieria Obliczeniowa Podstawy Sztucznej Inteligencji
Projekt 4 - Uczenie sieci regułą Hebba - Sprawozdanie	

1) Cel ćwiczenia:
   Celem poniższego ćwiczenia było poznanie działania reguły Hebba na przykładzie rozpoznawania emotikon.

2) Opis użytej struktury i algorytmu uczenia, oraz funkcji przykładowej:
   Sztuczna sieć neuronowa - struktura matematyczno-programowa, realizująca różne funkcje, najczęściej wzorowane na działaniu biologicznego mózgu.

   Sieci typu feedforward - sieci w których połączenia między węzłami nie tworzą pętli ani obiegów. Informacja (sygnał) przepływa tylko w jednym kierunku, z węzłów wejściowych, przez ukryte, do wyjściowych.



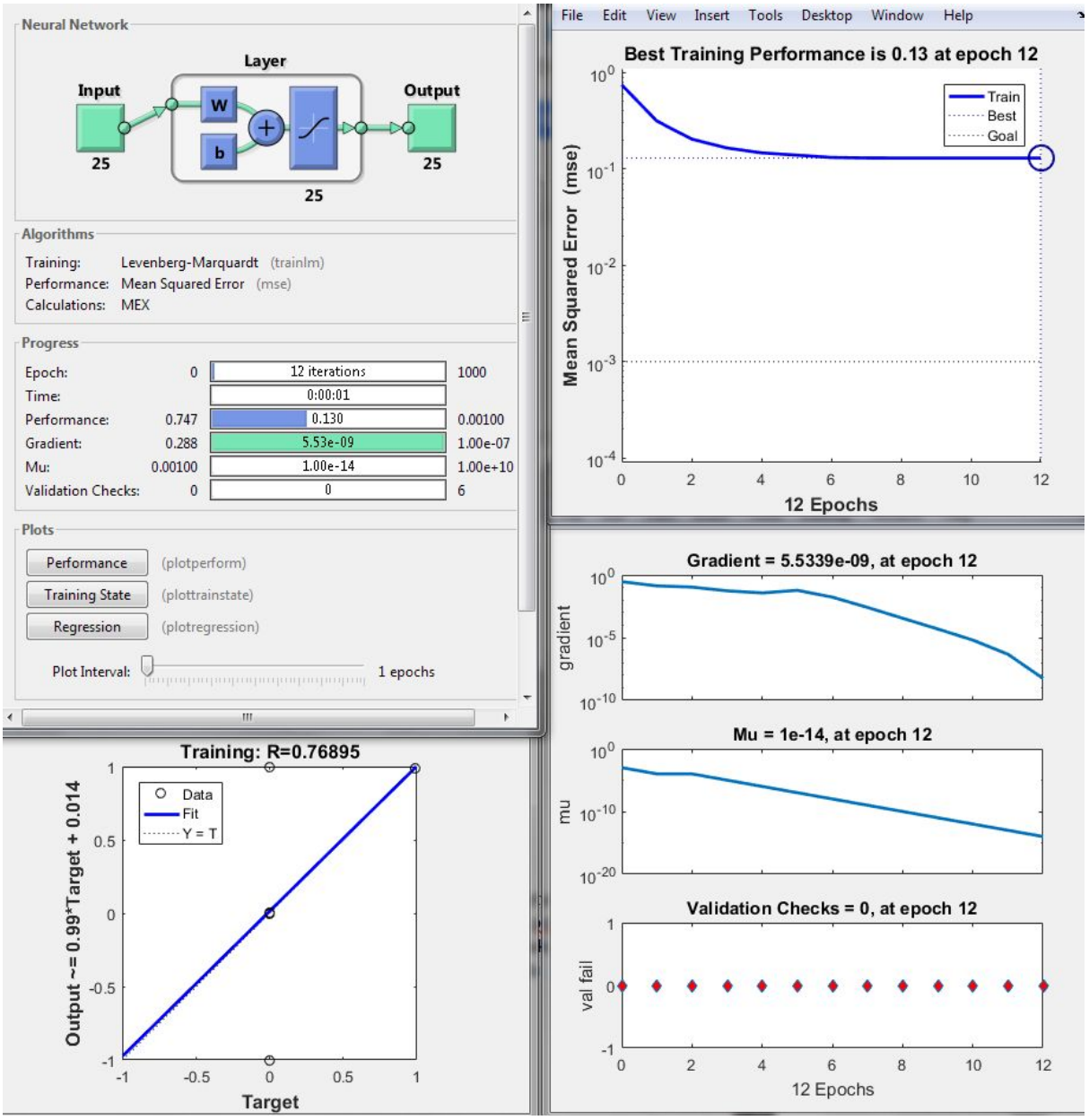
   Hebb zauważył podczas badań działania komórek nerwowych, że połączenie między dwiema komórkami jest wzmacniane, jeśli w danej chwili obie są aktywne. Zaproponował algorytm, w którym modyfikację wag przeprowadza się według wzoru:

$$w_i(t+1) = w_i(t) + nyx_i$$

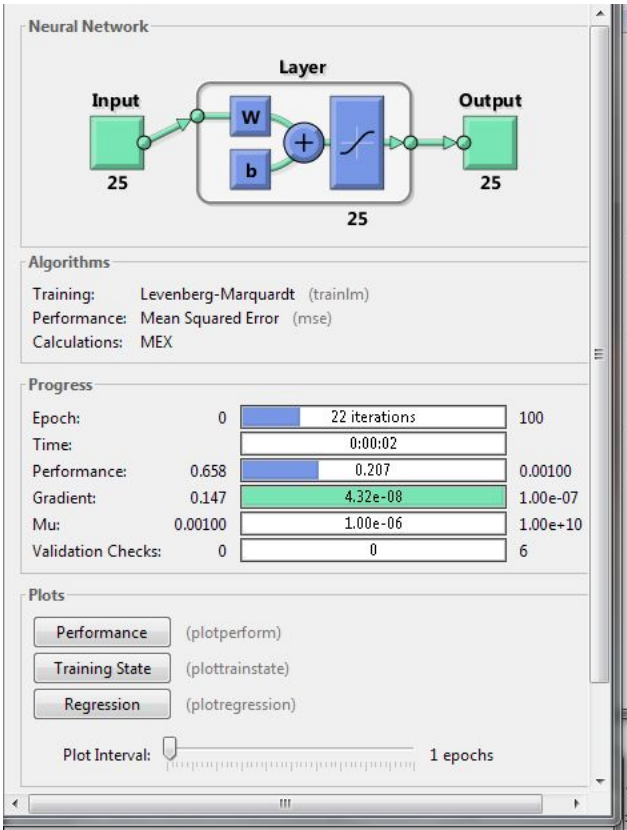
   w - waga neuronu
   t - numer iteracji w epoce
   y - sygnał wyjściowy neuronu
   x - sygnał wejściowy neuronu
   n - współczynnik uczenia

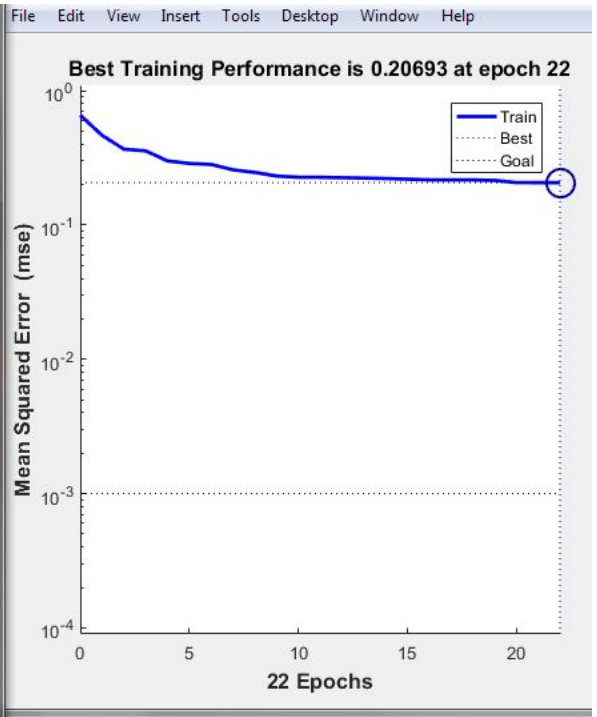
3) Przykładowe wyniki:

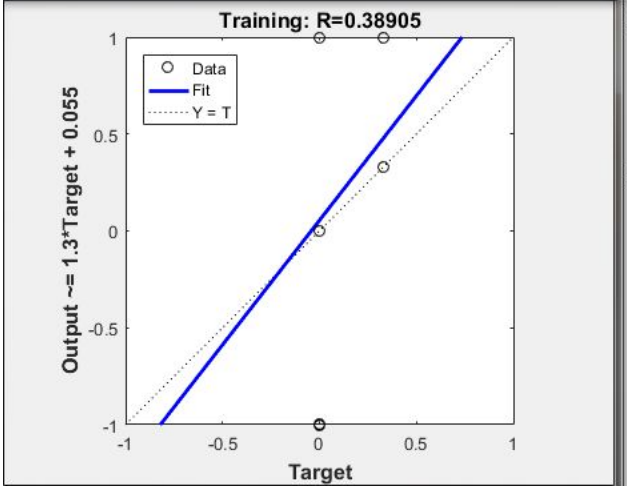
```
%PARAMETRY ALGORYTMU HEBBA
lp.dr = 0.5;%wspolczynnik zapominania
lp.lr = 0.99;%wspolczynnik uczenia

%PARAMETRY TRENINGU SIECI:
net.trainParam.epochs = 100;%maksymalna ilosc epok
net.trainParam.goal = 0.001;%cel wydajnosci
net.trainParam.lr=0.5;%wspolczynnik uczenia
```



```
%PARAMETRY ALGORYTMU HEBBA
lp.dr = 0;%wspolczynnik zapominania
lp.lr = 0.33;%wspolczynnik uczenia

%PARAMETRY TRENINGU SIECI:
net.trainParam.epochs = 100;%maksymalna ilosc epok
net.trainParam.goal = 0.001;%cel wydajnosci
net.trainParam.lr=0.1;%wspolczynnik uczenia
```







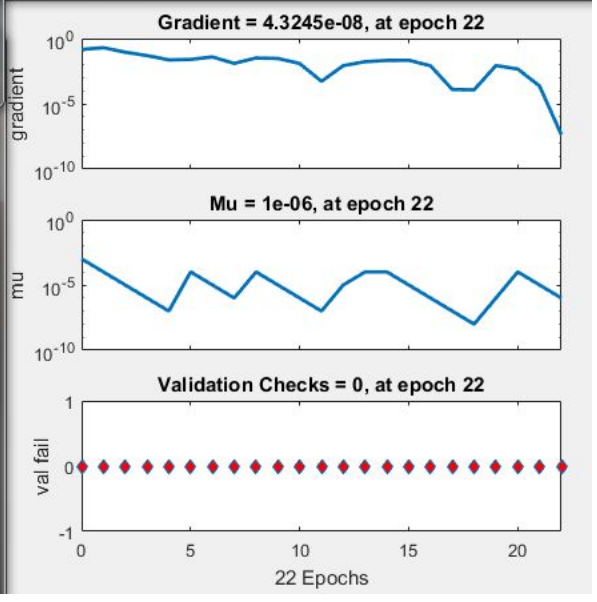


Tabela wyników w zależności od kombinacji współczynników Hebba

wspł. zapominania	wspł. uczenia	ilość epok	:)	:O	:(	:|
0	0.1	15	-4.4409e-16	1.7764e-15	1	-1.0000
	0.33	22	0	1	-2.2204e-16	5.6843e-14
	0.99	12	0	2.2204e-16	-1	-0.7764e-15
0.001	0.1	11	-1	2.2449e-13	-9.5457e-13	-2.2204e-16
	0.33	28	0	-1.0000	0	2.2204e-16
	0.99	20	1.1369e-13	-2.2204e-16	0	-4.5714e-09
0.5	0.1	27	-2.2204e-16	0	-2.2204e-16	-2.2204e-16
	0.33	27	6.6613e-16	0	1	0
	0.99	12	3.7748e-15	-2.2204e-16	-2.2204e-16	-2.2204e-16

4) Wnioski:

Im większy współczynnik uczenia tym szybszy wzrost wag. Zły współczynnik powoduje błędy w treningu.
Metoda Hebba jest metodą uczenia bez nauczyciela, więc sieć potrzebuje więcej czasu na naukę. Fluktuacje wag zależą mocno od wartości współczynnika zapominania - bez niego wagi nie są stabilizowane (ciągle rosną), natomiast jeśli jest on zbyt duży, sieć zapomina większość informacji i nie uczy się odpowiednio.

5) Listing kodu:

```
close all; clear all; clc;

start=[zeros(25,1),ones(25,1)];
n_out=25;
net = newff(start, n_out, {'tansig'}, 'trainlm', 'learnh');

%reprezentacja binarna 4 emotikon dla tablicy 8x4
      %:):O:(:|
IN =  [ 0 0 0 0
        0 0 0 0
        0 0 0 0
        0 0 0 0
        0 0 0 0
        0 0 0 0
        1 1 1 1
```

```
        0 0 0 0
        1 1 1 1
        0 0 0 0
        0 0 0 0
        0 0 0 0
        0 0 0 0
        0 0 0 0
        0 0 0 0
        1 0 0 0
        0 1 1 1
        0 1 1 1
        0 1 1 1
        1 0 0 0
        0 0 1 0
        1 1 0 0
        1 1 0 0
        1 1 0 0
        0 0 1 0
        ];
%zmienna okreslajaca trafienie w danego emotikona
OUT =[1 0 0 0 ;  %:)
        0 1 0 0 ;  %:O
        0 0 1 0 ;  %:(
        0 0 0 1 ]; %:|


%PARAMETRY ALGORYTMU HEBBA
lp.dr = 0;%wspolczynnik zapominania
lp.lr = 0.33;%wspolczynnik uczenia

%PARAMETRY TRENINGU SIECI:
net.trainParam.epochs = 100;%maksymalna ilosc epok
net.trainParam.goal = 0.001;%cel wydajnosci
net.trainParam.lr=0.1;%wspolczynnik uczenia

%dostosowanie parametrów sieci do metody Hebba
wHebb = learnh([], IN, [], [], OUT, [], [], [], [], [], lp, []);

net = train(net, IN, wHebb');

%dane testowe
a_test=[0;0;0;0;0;
        0;1;0;1;0;
        0;0;0;0;0;
        1;0;0;0;1;
        0;1;1;1;0];
```

```
b_test=[0;0;0;0;0;
        0;1;0;1;0;
        0;0;0;0;0;
        0;1;1;1;0;
        0;1;1;1;0;];

c_test=[0;0;0;0;0;
        0;1;0;1;0;
        0;0;0;0;0;
        0;1;1;1;0;
        1;0;0;0;1;];

d_test=[0;0;0;0;0;
        0;1;0;1;0;
        0;0;0;0;0;
        0;1;1;1;0;
        0;0;0;0;0;];

F1 = wHebb;
%symulacja
F2 = sim(net, a_test);

disp(':) = '), disp(F2(1));
disp(':O = '), disp(F2(2));
disp(':( = '), disp(F2(3));
disp(':| = '), disp(F2(4));
```