Klaudia Pakos

IO, rok 3, 267561

PODSTAWY SZTUCZNEJ INTELIGENCJI

SPRAWOZDANIE NR 4

UCZENIE SIECI REGUŁĄ HEBBA

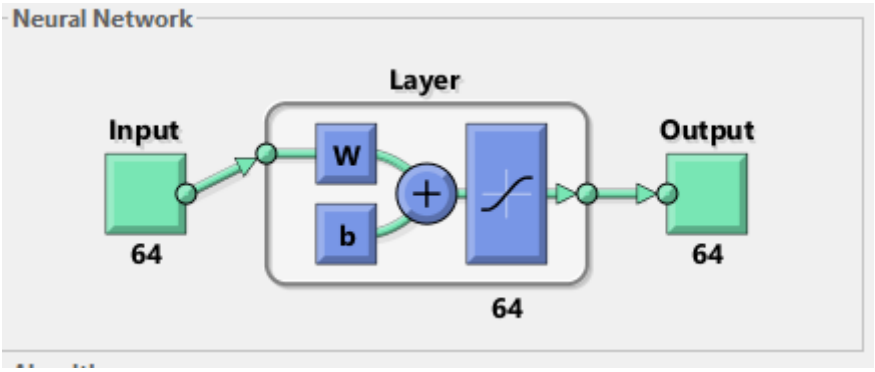
1. Cel ćwiczenia

Celem ćwiczenia jest poznanie działania reguły Hebba na przykładzie rozpoznawania emotikon.

2. Zadania do wykonania

- Wygenerowanie danych uczących i testujących, zawierających 4 różne emotikony np. czarno-białe, wymiar 8x8 pikseli dla jednej emotikony.
- Przygotowanie (implementacja lub wykorzystanie gotowych narzędzi) sieci oraz reguły Hebba w wersji z i bez współczynnika zapominania.
- Uczenie sieci dla różnych współczynników uczenia i zapominania.
- Testowanie sieci.

3. Opis budowy sieci



Zasada działania reguły Hebba polega na interpretacji zachowań aktywnych neuronów. Jeśli aktywny neuron A jest cyklicznie pobudzany przez drugi neuron B, to staje się on jeszcze bardziej czuły na pobudzenie tego neuronu. Reguła Hebba składa się z następujących punktów:

1. Jeżeli połączone synapsą neurony A i B są pobudzane jednocześnie (synchronicznie) to połączenie synaptyczne między nimi jest wzmacniane.
2. Jeżeli neurony A i B połączone synapsą nie są pobudzane jednocześnie (asynchronicznie) to połączenie pomiędzy nimi jest osłabiane.

Reguła Hebba jest jedną z najpopularniejszych metod samouczenia sieci neuronowych. Polega ona na tym, że sieci pokazuje się kolejne przykłady sygnałów wejściowych nie podając żadnych informacji o tym, co z tymi sygnałami należy zrobić. Sieć obserwuje otoczenie i odbiera różne sygnały nie zostają określone jednak jakie znaczenie mają pokazujące się obiekty i jakie są pomiędzy nimi zależności.

Sieć na podstawie obserwacji występujących sygnałów stopniowo sama odkrywa, jakie jest ich znaczenie i również sama ustala zachodzące między sygnałami zależności. Po podaniu do sieci neuronowej każdego kolejnego zestawu sygnałów wejściowych tworzy się w tej sieci pewien rozkład sygnałów wyjściowych - niektóre neurony sieci są pobudzone bardzo silnie inne słabiej, a jeszcze inne mają sygnały wyjściowe wręcz ujemne.

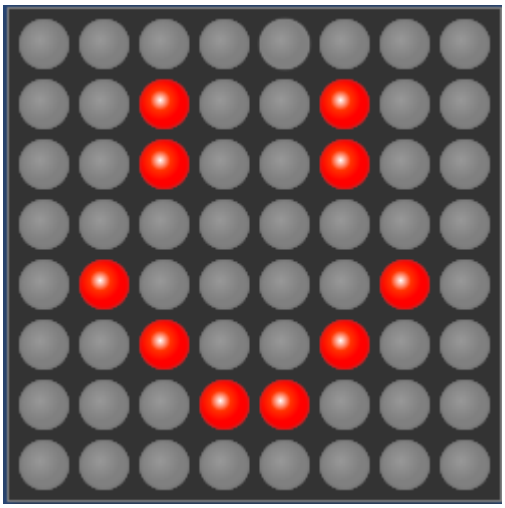
Proces samouczenia ma niestety wady. W porównaniu z procesem uczenia z nauczycielem samouczenie jest zwykle znacznie powolniejsze. Co więcej bez nauczyciela nie można z góry określić, który neuron wyspecjalizuje się w rozpoznawania której klasy sygnałów. Stanowi to pewną trudność przy wykorzystywaniu i interpretacji wyników pracy sieci. Co więcej - nie można określić, czy sieć uczona w ten sposób nauczy się wszystkich prezentowanych jej wzorców. Dlatego sieć przeznaczona do samouczenia musi być większa niż sieć wykonująca to samo zadanie, ale trenowana w sposób klasyczny z udziałem nauczyciela.

Metoda Hebba posiada wiele wad, są nimi m. in.:

- niska efektywność uczenia,
- przemnożony wpływ początkowych wartości wag sieci,
- wagi w przypadku tej reguły mogą rosnąć bez ograniczeń,
- proces uczenia sieci nigdy nie zakończy się samodzielnie,
- nie ma pewności, że jednej klasie wzorców będzie odpowiadał jeden neuron,
- nie ma również gwarancji, że wszystkie klasy wzorców będą miały oddzielne reprezentacje w postaci oddzielnych zbiorów aktywnych neuronów.

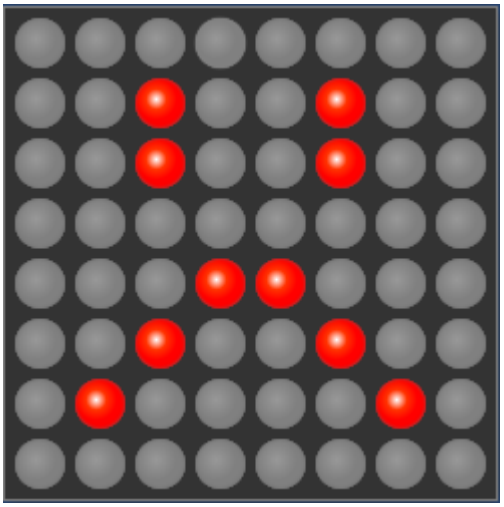
Bardzo istotną kwestią jest wybór początkowych wartości wag neuronów sieci przeznaczonej do samouczenia. Wartości te mają bardzo silny wpływ na ostateczne zachowanie sieci, ponieważ proces uczenia jedynie pogłębia i doskonali pewne tendencje istniejące w sieci od samego początku. Od jakości początkowych właściwości sieci silnie zależy do czego sieć dojdzie na końcu procesu uczenia.

Do sprawdzenia działania sieci wykorzystano 4 wygenerowane emotikony: SMILE, SAD, KISS, CONFUSE przedstawione w formie matryc 8x8 z reprezentacją zero jedynkową.



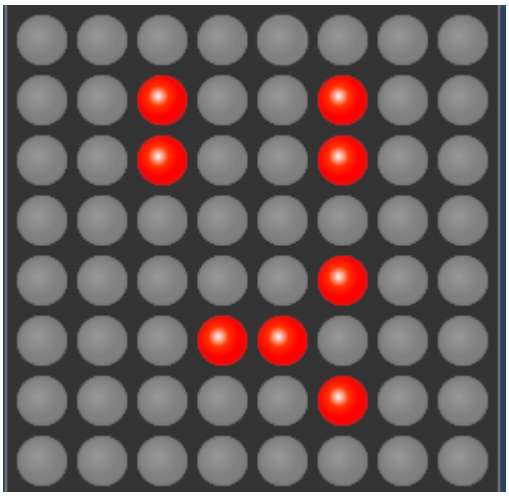
SMILE:

00000000;
00100100;
00100100;
00000000;
01000010;
00100100;
00011000;
00000000



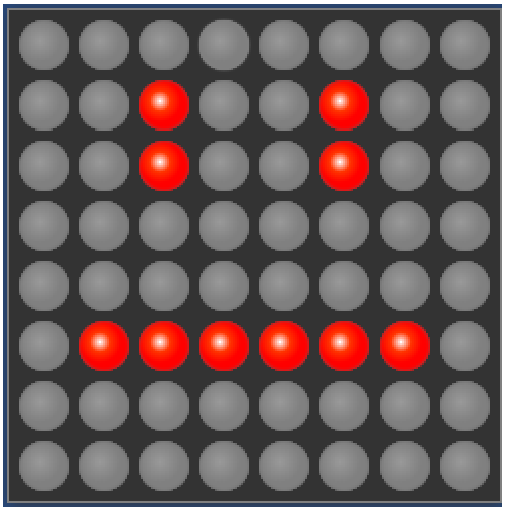
SAD:

00000000;
00100100;
00100100;
00000000;
00011000;
00100100;
01000010;
00000000



KISS:

00000000;
00100100;
00100100;
00000000;
00000100;
00011000;
00000100;
00000000;



CONFUSE:
00000000;
00100100;
00100100;
00000000;
00000000;
01111110;
00000000;
00000000

4. Przebieg zadania-listing kodu

```
%%
close all; clear all; clc;

%wejścia do sieci z min i max wartościami
minmax=[0 1; 0 1; 0 1; 0 1; 0 1; 0 1; 0 1; 0 1; 0 1; 0 1;
    0 1; 0 1; 0 1; 0 1; 0 1; 0 1; 0 1; 0 1; 0 1; 0 1;
    0 1; 0 1; 0 1; 0 1; 0 1; 0 1; 0 1; 0 1; 0 1; 0 1;
    0 1; 0 1; 0 1; 0 1; 0 1; 0 1; 0 1; 0 1; 0 1; 0 1;
    0 1; 0 1; 0 1; 0 1; 0 1; 0 1; 0 1; 0 1; 0 1; 0 1;
    0 1; 0 1; 0 1; 0 1; 0 1; 0 1; 0 1; 0 1; 0 1; 0 1;
    0 1; 0 1; 0 1; 0 1];

%ilość wyjść z sieci
ilosc_wyj = 64;

%użycie funkcji tworzącej sieć
net = newff(minmax, ilosc_wyj,{'tansig'}, 'trainlm',
'learnh');

%kolumnowe wprowadzenie emotikon w formie 0-1
%         SMILE/SAD/KISS/CONFUSE
input   = [0  0  0  0;
           0  0  0  0;
           0  0  0  0;
           0  0  0  0;
           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;
```

```
0 0 0 0;
1 1 1 1;
0 0 0 0;
0 0 0 0;
0 0 0 0;
0 0 0 0;
1 1 1 1;
0 0 0 0;
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;
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 0 0 0;
0 1 0 0;
0 1 0 0;
0 0 1 0;
1 0 0 0;
0 0 0 0;
0 0 0 0;
0 0 0 1;
1 1 0 1;
0 0 1 1;
0 0 1 1;
1 1 0 1;
0 0 0 1;
0 0 0 0;
0 0 0 0;
0 1 0 0;
0 0 0 0;
1 0 0 0;
1 0 0 0;
0 0 1 0;
0 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 0 0;
0 0 0 0;
0 0 0 0;
```

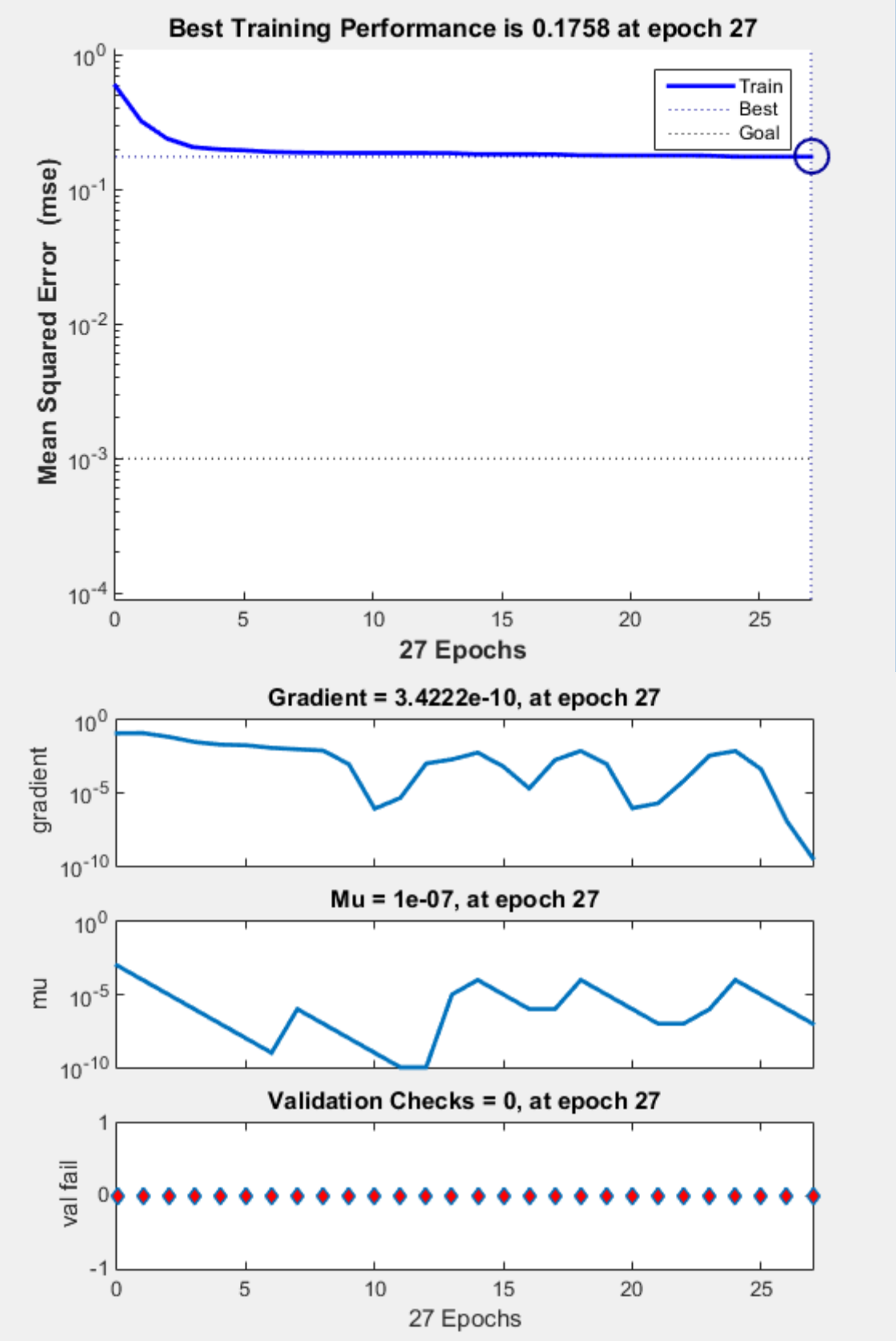
```
          0 0 0 0;
          ];

%zmienna zawierająca 1 gdy trafimy w emotikon i 0 gdy
chybimy
output = [ 1 0 0 0   %SMILE
           0 1 0 0   %SAD
           0 0 1 0   %KISS
           0 0 0 1]; %CONFUSE

%parametry reguły Hebba
lp.dr = 0.1; %wsp. zapominania
lp.lr = 0.1; %wsp. uczenia

%użycie reguły Hebba
hebb = learnh( [], input, [], [], output, [], [], [], [],
[], lp, []);
heb=hebb';

net.trainParam.epochs = 1000;
net.trainParam.goal = 0.001;
net.trainParam.lr=0.5;

%trenowanie sieci z użyciem reguły Hebba
net = train(net, input, heb);


%DANE TESTUJACE
smile = [0 0 0 0 0 0 0 0;  0 0 1 0 0 1 0 0;  0 0 1 0 0 1 0
0;  0 0 0 0 0 0 0 0;  0 1 0 0 0 0 1 0;  0 0 1 0 0 1 0 0;
0 0 0 1 1 0 0 0;  0 0 0 0 0 0 0 0 ];
sad = [0 0 0 0 0 0 0 0;  0 0 1 0 0 1 0 0;  0 0 1 0 0 1 0
0;  0 0 0 0 0 0  0 0;  0 0 0 1 1 0 0 0;  0 0 1 0 0 1 0 0;
0 1 0 0 0 0 1 0;  0 0 0 0 0 0 0 0 ];
kiss=[0 0 0 0 0 0 0 0;  0 0 1 0 0 1 0 0;  0 0 1 0 0 1 0 0;
0 0 0 0 0 0 0 0;  0 0 0 0 0 1 0 0;  0 0 0 1 1 0 0 0;  0 0
0 0 0 1 0 0;  0 0 0 0 0 0 0 0 ];
confuse = [  0 0 0 0 0 0 0 0;  0 0 1 0 0 1 0 0;  0 0 1 0 0
1 0 0;  0 0 0 0 0 0 0 0;  0 0 0 0 0 0 0 0;  0 1 1 1 1 1 1
0;  0 0 0 0 0 0 0 0;  0 0 0 0 0 0 0 0 ];

%sprawdzenie poprawności wytrenowanej sieci
test = sim(net, smile);
test1 = sim(net, sad);
test2 = sim(net, kiss);
test3 = sim(net, confuse);



%wypisanie wartości
disp('Wartości')
```

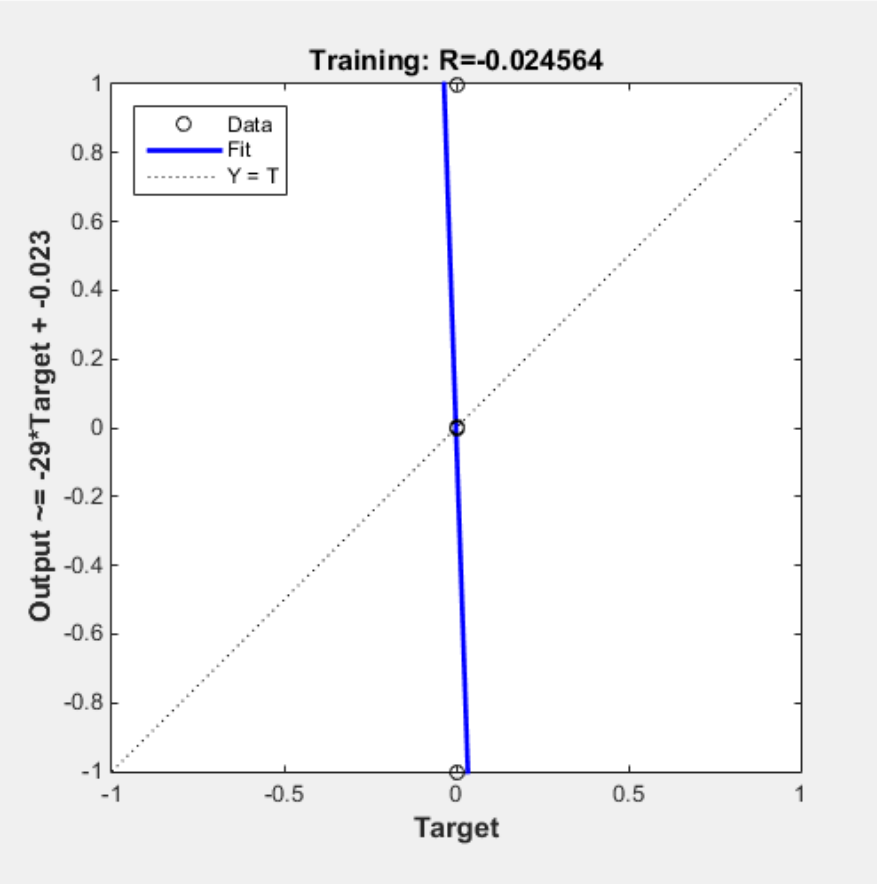
```
disp('SMILE ='), disp(test(1));
disp('SAD ='), disp(test1(2));
disp('KISS ='), disp(test2(3));
disp('CONFUSE ='), disp(test3(4));
```

5. Przedstawienie wyników

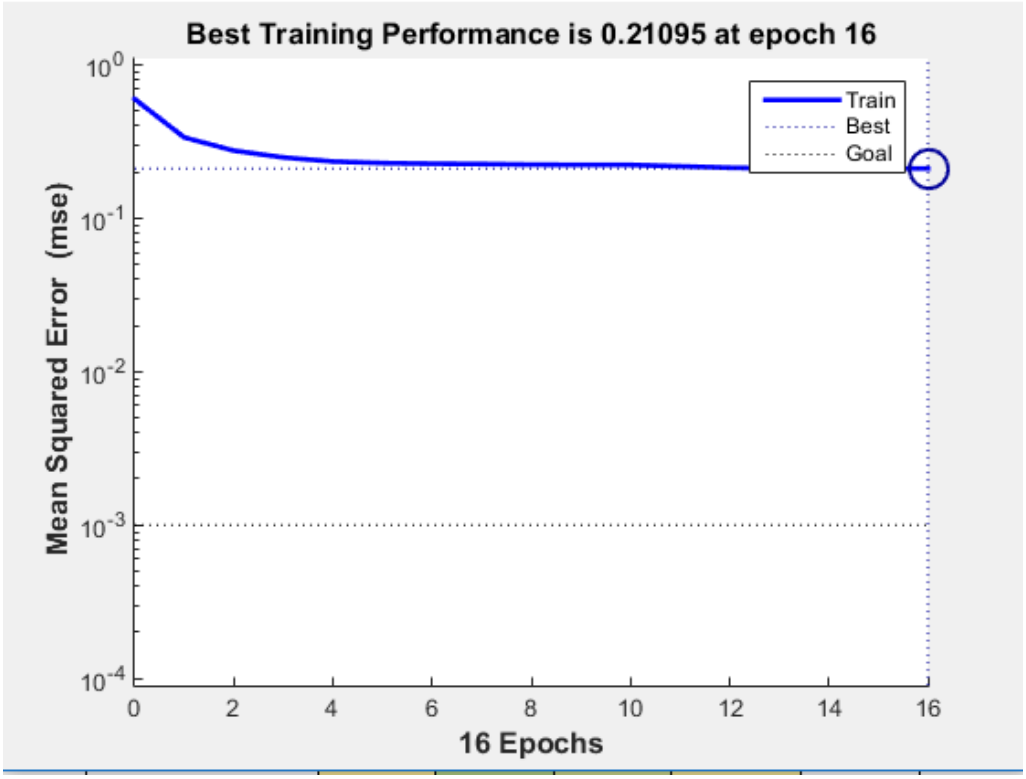
współczynniki uczenia/zapominania	SMILE	SAD	KISS	CONFUSE	liczba epok
0.001/0	1	1	0,2939	0,8609	27
0.01/0	-0,9895	1	-0,408	0,5841	18
0.1/0	1	-1	-0,8761	-1	18
0.5/0	-0,6431	0,3738	-0,6805	0,9318	21
0.99/0	1	-0,0856	-1	-1	20
0.001/0.01	0,402	0,2239	1	1	16
0.01/0.01	0,7671	0,9252	0,1817	-0,0268	16
0.1/0.01	-0,8291	0,1203	0,6097	1	16
0.5/0.01	0,1672	-0,8006	-0,7775	-0,1785	12
0.99/0.01	1	-0,5259	-0,9954	-0,3377	18
0.001/0.1	-0,9185	-1	-0,064	1	14
0.01/0.1	0,926	-0,9357	1	-0,6084	14
0.1/0.1	-1	-0,0807	-0,4434	-0,9151	12
0.5/0.1	-0,691	-1	0,9542	1	29
0.99/0.1	-0,8955	-0,8	-1	0,1264	18
0.001/0.5	1	1	-0,6735	1	14
0.01/0.5	-1	-1	0,8896	-0,961	14
0.1/0.5	-0,2012	1	1	0,5373	16
0.5/0.5	0,7985	-0,8795	-1	-0,9768	15
0.99/0.5	0,7855	-1	-1	0,8822	13

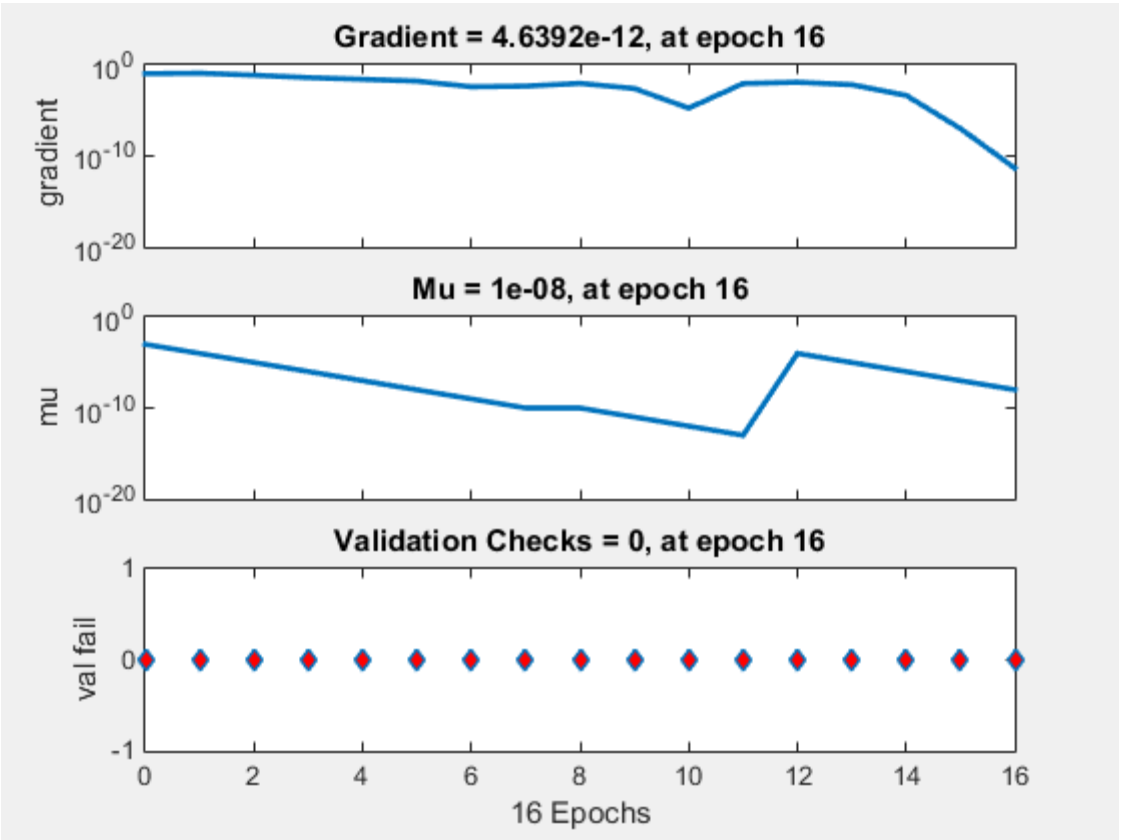
Dla współczynników uczenia/zapominania 0.001/0:

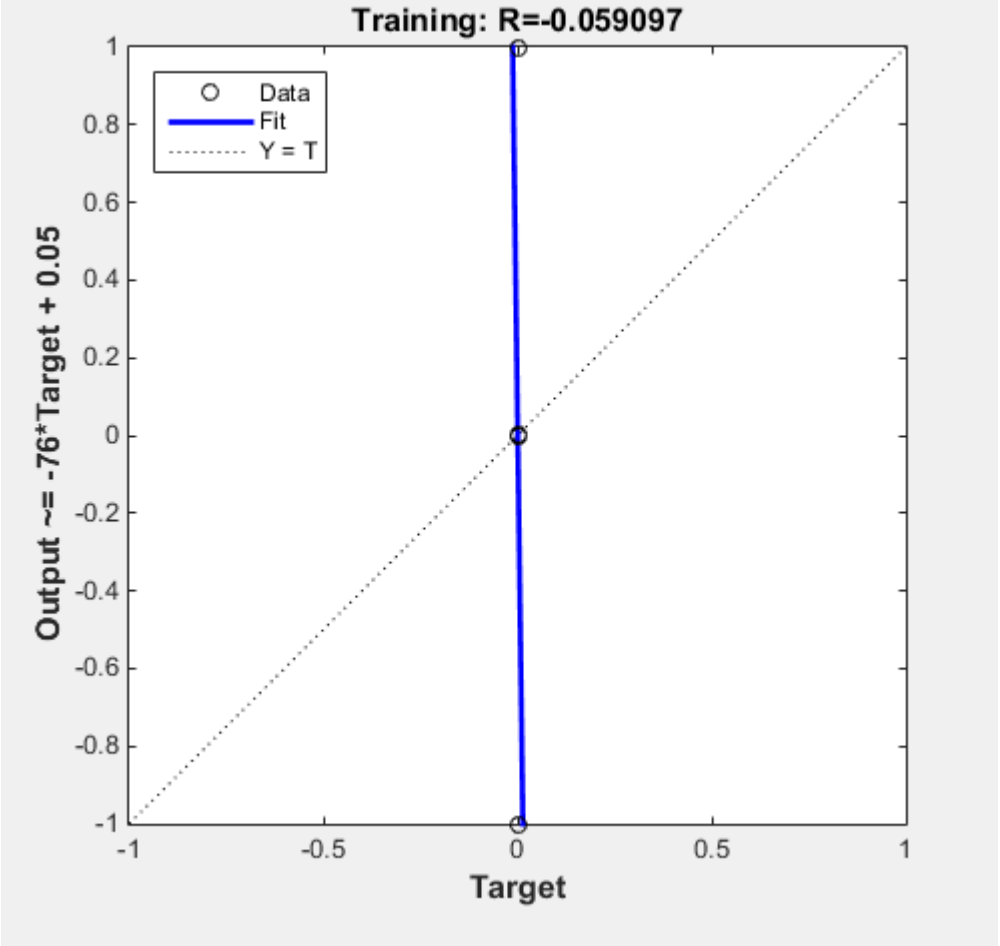




Dla współczynników uczenia/zapominania 0.001/0.01:




Gradient = 4.6392e-12, at epoch 16
gradient
Mu = 1e-08, at epoch 16
mu
Validation Checks = 0, at epoch 16
val fail
16 Epochs


Training: R=-0.059097
Data
Fit
Y = T
Output ~= -.76*Target + 0.05
Target

6. Wnioski

Proces samouczenia z wykorzystaniem algorytmu Hebba wymaga dużo więcej czasu od trenowania sieci neuronowej z nauczycielem. Bardzo istotną kwestią jest wybór początkowych wartości wag neuronów sieci przeznaczonej do samouczenia. Wartości te mają bardzo silny wpływ na ostateczne zachowanie sieci. Dlatego uruchamianie ponownej nauki sieci z tymi samymi parametrami uczenia się i zapominania daje za każdym razem inne efekty, w zależności od początkowych wag neuronów. Ciężko wyciągnąć zatem jednoznaczne wnioski o poprawności, ponieważ końcowe wzmocnienia/osłabienia sygnałów bardzo się różnią. Jednak na podstawie przeprowadzonego testowania sieci najlepsze wyniki osiągnięto dla małych współczynników uczenia (0,001) i braku współczynnika zapominania, oraz współczynnika uczenia (0,001) i współczynnika zapominania 0,01. Wtedy sieci udało się trafiać we wszystkie emotikony. Podczas analizowania wydajności można zauważyć również miejsce, w którym jakość treningu przestaje znacząco wzrastać - może to świadczyć o tym, że jest to punkt, w którym sieć przestaje się uczyć bez większego progresu oraz wzrasta prawdopodobieństwo na zjawisko przeuczenia sieci. Reguła Hebba jest dobrą alternatywą w porównaniu do poprzednich metod, ponieważ pozawala na uczenie sieci bez nauczyciela (jest jedną z najbardziej zbliżonych metod do prawdziwej, biologicznej sieci neuronowej).