|  |  |  |
| --- | --- | --- |
| Grupa ćwicz. **2** | Data wykonania 22.11.2017 | Nr. Scenariusza  **4** |
| **Temat ćwiczenia:** Uczenie sieci regułą Hebba. | | |
| Imię i nazwisko  **Kamil Szczurkowski** | | Ocena i Uwagi |

**Cel ćwiczenia:**

Celem ćwiczenia jest poznanie działania reguły Hebba dla sieci jednowarstwowej na przykładzie grupowania liter alfabetu.

**Wykonane zadania:**

1. Wygenerowano dane uczące i testujące, zawierające 20 dużych liter alfabetu łacińskiego w postaci dwuwymiarowej tablicy 7x5, reprezentowanej w kodzie jako jednowymiarowa tablica 35 elementowa
2. Wykorzystano narzedzie Matlab do stworzenia jednowarstwowej sieci (newff) oraz reguły Hebba z i bez współczynnika zapominania. ( learnh ).
3. Uczono sieć dla różnych współczynników uczenia i zapominania.
4. Testowano poprawność działania sieci

**Specyfikacja wykorzystanych funkcji:**

**sim** - Symulacja sieci neuronowej dla zadanych danych wejściowych. SIM Funkcja służy do wyznaczenia wyjść sieci neuronowej dla zadanej macierzy danych wejściowych. Argumenty *Pi*, *Ai*, *Pf*, *Af* są opcjonalne i nie będą używane przez sieci nieliniowe, wykorzystywane w ćwiczeniach

**newff** - Tworzenie wielowarstwowej jednokierunkowej sieci neuronowej, złożonej z neuronów o nieliniowych funkcjach aktywacji. ( wykorzystamy do stworzenia sieci jednowarstwowej)

NEWFF Funkcja tworzy wielowarstwową sieć neuronową; każda warstwa składa się z

zadanej liczby neuronów o nieliniowych funkcjach aktywacji (jakkolwiek funkcje aktywacji w poszczególnych warstwach mogą mieć również postać liniową).

**WEJŚCIE:**

**PR**- macierz o wymiarach *R*x*2*, gdzie *R* jest liczbą wejść sieci (współrzędnych wektorów wejściowych); pierwsza kolumna zawiera minimalne wartości kolejnych współrzędnych wektorów wejściowych, druga kolumna – maksymalne wartości tych współrzędnych

***Si***- liczba neuronów w *i*-tej warstwie sieci; liczba warstw wynosi *N1*

**TFi-** nazwa funkcji aktywacji neuronów w *i*-tej warstwie sieci (zmienna tekstowa);

domyślna = 'tansig' (tangens hiperboliczny); dopuszczalne wartości parametru *TF* to: ‘tansig’ i ‘logsig’ i ‘purelin’

**BTF**- nazwa funkcji, wykorzystywanej do treningu sieci (zmienna tekstowa); domyślnie *BTF* = ‘trainlm’ (metoda Levenberga-Marquardta)

**BLF**- nazwa funkcji, wykorzystywanej do wyznaczania korekcji wag sieci podczas treningu (zmienna tekstowa); domyślnie *BLF* = ‘learngd’; dopuszczalne wartości parametru *BLF* to: ‘learngd’ (gradient prosty) i ‘learngdm’ (gradient prosty z momentum)

**PF**- funkcja wyznaczająca wartość wskaźnika jakości treningu sieci jednokierunkowej (zmienna tekstowa); domyślnie *PF* = ‘mse’ (błąd średniokwadratowy); parametr ten moŜe oznaczać dowolną róŜniczkowalną funkcję błędu, np. ‘msereg’ (suma błędu średniokwadratowego i kwadratów wag sieci – metoda regularyzacji wag) lub ‘sse’ (suma kwadratów błędów)

**WYJŚCIE:**

**NET**- struktura (obiekt) zawierająca opis architektury, metod treningu, wartości liczbowe wag oraz inne parametry wielowarstwowej sieci jednokierunkowej.

**learnh**- Reguła uczenia wagi hebba

LP.lr – Wskaźnik uczenia się.

LP.dr – Szybkość zaniku.

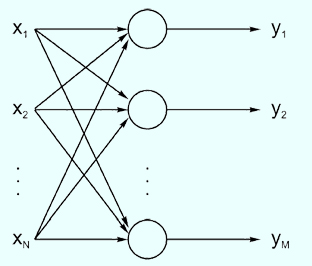
learnh oblicza zmianę wagi dw dla danego neuronu z wejścia P neuronu, wyjście A i szybkość uczenia LR zgodnie z zasadą uczenia Hebb

dw = lr\*a\*p'

**learnhd**- Reguła uczenia hebba z rozkładem wag

learnhd oblicza zmianę wagi dW dla danego neuronu z wejścia P neuronu, wyjście A, szybkość zaniku DR i szybkość uczenia LR zgodnie z zasadą Hebb z rozpadem:

dw = lr\*a\*p' - dr\*w



Rysunek 1- Sieć jednowarstwowa

**Wykonanie zadania:**

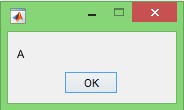
Do wykonania zadania użyłem zastawu danych z 20 literami. W sieci do rozpoznawania liter użyłem następująca ilość neuronów : 35 – 20

**Wyniki:**

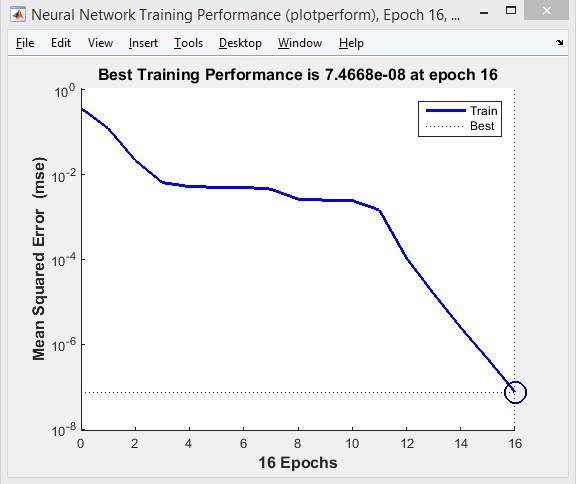
Kilka przykładowych pomiarów przeprowadzonych dla różnych liter oraz rożnej budowy sieci : Dla sieci 35 – 20 z **learnh** oraz **lr =** 0.1;



Rysunek 2- Testowana litera



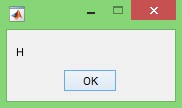
Rysunek 3- Otrzymany wynik



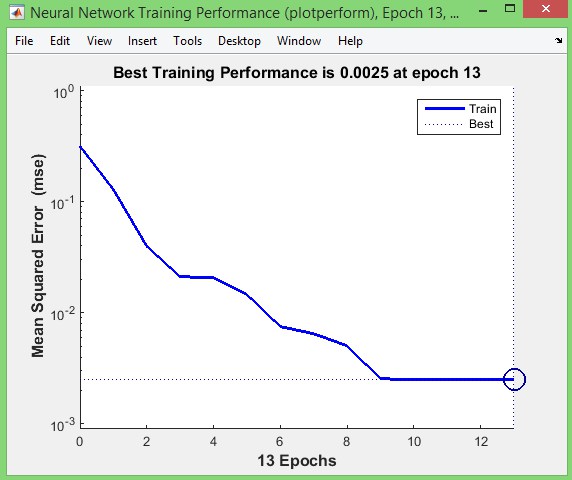
Rysunek 4- Wykres uczenia się

**C:\Users\km64\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Screenshot 2017-11-22 16-45-36.jpg**

Rysunek 5 - Testowana litera



Rysunek 6 - Otrzymany wynik

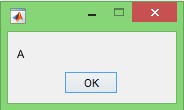


Rysunek 7- Wykres uczenia się

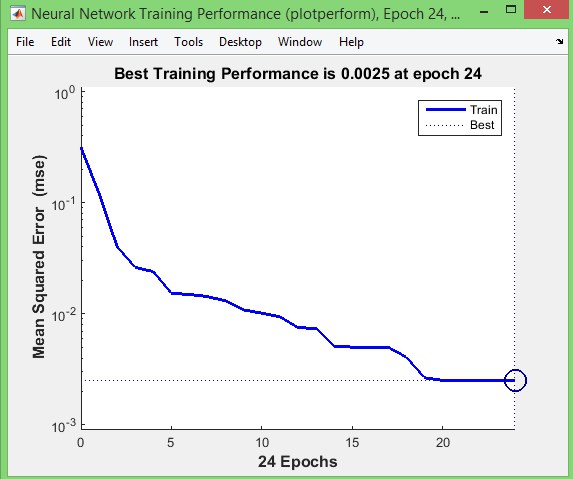
sieci : Dla sieci 35 – 20 z **learnh** oraz **lr =** 0.01;

**C:\Users\km64\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Screenshot 2017-11-22 16-42-05.jpg**

Rysunek 8 - Testowana litera

****

Rysunek 9- Otrzymany wynik

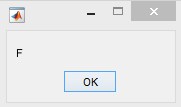


Rysunek 10- Wykres uczenia się

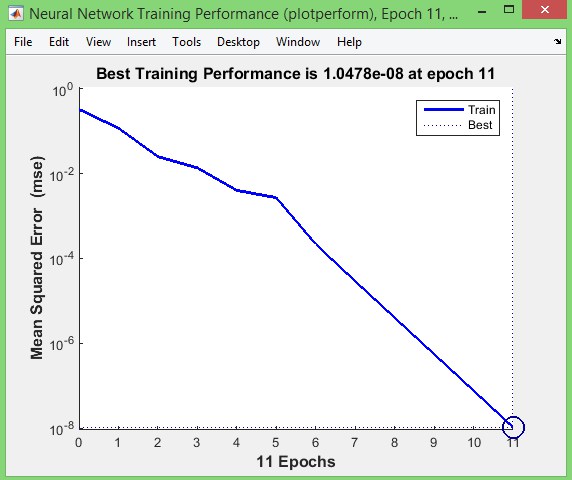
Dla sieci 35 – 20 z **learnhd** , **lr =** 0.1, **dr =** 0.1;



Rysunek 11- Testowana litera



Rysunek 12- Otrzymany wynik

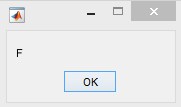


Rysunek 13- Wykres uczenia się

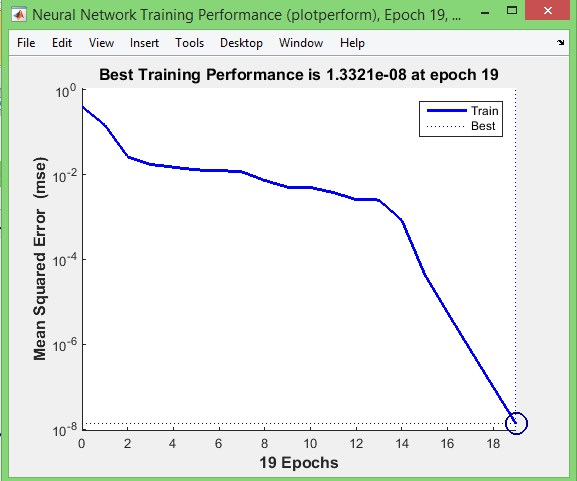
Dla sieci 35 – 20 z **learnhd** , **lr =** 0.01, **dr =** 0.01;

C:\Users\km64\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Screenshot 2017-11-22 17-03-12.jpg

Rysunek 14- Testowana litera



Rysunek 15- Otrzymany wynik



Rysunek 16- Wykres uczenia się

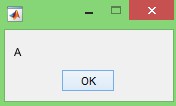
Testowanie sieci dla pozmienianych liter :



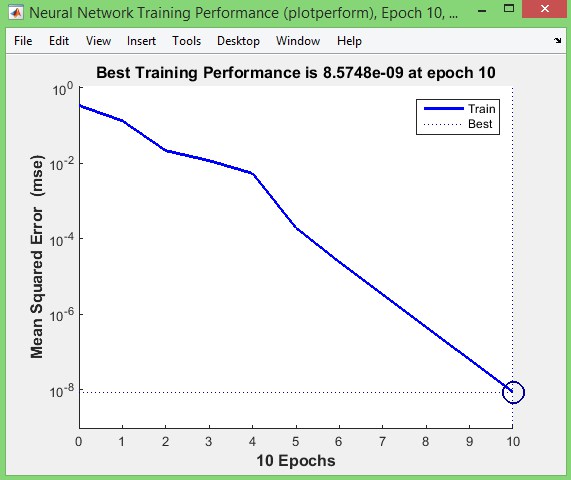
Rysunek - Zmieniona literka



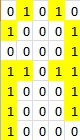
Rysunek - Testowana litera



Rysunek - Otrzymany wynik



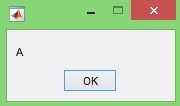
Rysunek - Wykres uczenia się



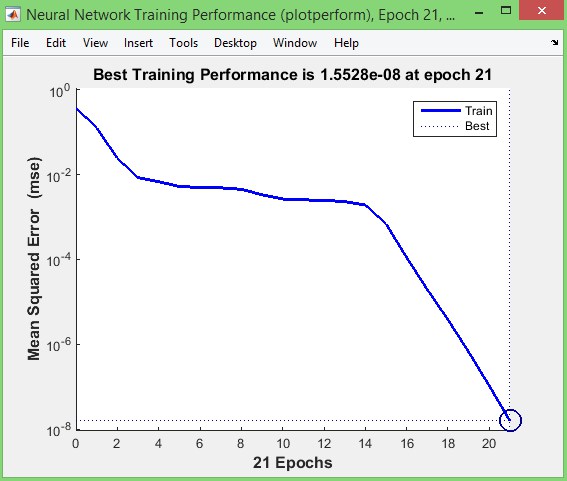
Rysunek - Zmieniona literka

C:\Users\km64\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Screenshot 2017-11-23 16-22-06.jpg

Rysunek - Testowana litera



Rysunek - Otrzymany wynik



Rysunek - Wykres uczenia się

**Analiza:**

Proces uczenia odbywał się dla trainlm z learnhd oraz learnh. Można było zauważyć ze learnh z parametrem lr =0.1 wykonuje się w małej liczbie epok i jego wyniki zmierzają do 1 nie było nigdy 1 przez co nie nauczył się w 100% danej literki natomiast podczas stosowania learnh z parametrem lr =0.01 były jeszcze bliższe jedynki lecz liczba epok zwiększyła się o polowe. Przeprowadziłem również testy dla learnhd z roznymi parametrami lr oraz dr wyniki nieznacznie się różniły liczbą epok .

Podając mu uszkodzone literki dalej potrafił wskazać poprawna i grupował je sobie w odpowiednie miejsca. Dla 2 przykładowych A grupował je pod litry A natomiast jestli z przykładu 2 usunęliśmy cala góre to od razu przypisał ja do grupy H.

Obserwując procesy uczenia można było stwierdzić ze obie metody działają podobnie dopiero przy zmianie parametrów lr i dr można było zauwazyc wzrost liczby epok i minimalny wzrost precyzji.

Zmieniając wskaźnik uczenia się lr otrzymywaliśmy trochę lepsze przybliżenie wyników ale w efekcie siec musiała się uczyć dłużej. Były to różnice kilku epok a wzrost wydajności minimalny.

**Wnioski:**

Na podstawie otrzymanych wyników można było zauważyć ze dobór lr ma nie tak duże znaczenie na otrzymywanie wyniki. Jeśli chodzi o funkcję które uczą nam siec to otrzymaliśmy podobny efekt z learnh i learnhd . Natomiast przez mniejszy lr wzrosła o połowę liczba epok natomiast wyniki stały się odrobine precyzyjniejsze chodź przy tak małym przybliżeniu nie opłaca się mniejszych lr.

Zmiana dr zmniejsza liczbę epok przez to ze wyniki wag są zapominane a nie zapamiętywane.  
Tworząc tego typu sieci można wybrać jedna z 2 reguł uczenia wagi hebba, trzeba pamiętać o doborze odpowiednich trenerów oraz parametrów lr. Często większe przybliżenie może nas kosztować wiele epok a nie da nam dużo lepszego przybliżenia.

**Bibliografia:**

<https://www.mathworks.com/help/nnet/ref/learnhd.html>

<https://www.mathworks.com/help/nnet/ref/learnh.html>

<http://galaxy.agh.edu.pl/~vlsi/AI/koho_t/>

**Listing:** close all; clear all; clc;

A=[0 1 1 1 0 1 0 0 0 1 1 0 0 0 1 1 1 1 1 1 1 0 0 0 1 1 0 0 0 1 1 0 0 0 1];

B=[1 1 1 1 0 1 0 0 0 1 1 0 0 0 1 1 1 1 1 0 1 0 0 0 1 1 0 0 0 1 1 1 1 1 0];

C=[0 1 1 1 0 1 0 0 0 1 1 0 0 0 0 1 0 0 0 0 1 0 0 0 0 1 0 0 0 1 0 1 1 1 0];

D=[1 1 1 1 0 1 0 0 0 1 1 0 0 0 1 1 0 0 0 1 1 0 0 0 1 1 0 0 0 1 1 1 1 1 0];

E=[1 1 1 1 1 1 0 0 0 0 1 0 0 0 0 1 1 1 1 0 1 0 0 0 0 1 0 0 0 0 1 1 1 1 1];

F=[1 1 1 1 1 1 0 0 0 0 1 0 0 0 0 1 1 1 1 0 1 0 0 0 0 1 0 0 0 0 1 0 0 0 0];

G=[0 1 1 1 0 1 0 0 0 1 1 0 0 0 0 1 0 1 1 1 1 0 0 0 1 1 0 0 0 1 0 1 1 1 0];

H=[1 0 0 0 1 1 0 0 0 1 1 0 0 0 1 1 1 1 1 1 1 0 0 0 1 1 0 0 0 1 1 0 0 0 1];

I=[0 1 1 1 0 0 0 1 0 0 0 0 1 0 0 0 0 1 0 0 0 0 1 0 0 0 0 1 0 0 0 1 1 1 0];

J=[1 1 1 1 1 0 0 0 0 1 0 0 0 0 1 0 0 0 0 1 0 0 0 0 1 1 0 0 0 1 0 1 1 1 0];

K=[1 0 0 0 1 1 0 0 1 0 1 0 1 0 0 1 1 0 0 0 1 0 1 0 0 1 0 0 1 0 1 0 0 0 1];

L=[1 0 0 0 0 1 0 0 0 0 1 0 0 0 0 1 0 0 0 0 1 0 0 0 0 1 0 0 0 0 1 1 1 1 1];

M=[1 0 0 0 1 1 1 0 1 1 1 0 1 0 1 1 0 0 0 1 1 0 0 0 1 1 0 0 0 1 1 0 0 0 1];

N=[1 0 0 0 1 1 0 0 0 1 1 1 0 0 1 1 0 1 0 1 1 0 0 1 1 1 0 0 0 1 1 0 0 0 1];

O=[0 1 1 1 0 1 0 0 0 1 1 0 0 0 1 1 0 0 0 1 1 0 0 0 1 1 0 0 0 1 0 1 1 1 0];

P=[1 1 1 1 0 1 0 0 0 1 1 0 0 0 1 1 1 1 1 0 1 0 0 0 0 1 0 0 0 0 1 0 0 0 0];

Q=[0 1 1 1 0 1 0 0 0 1 1 0 0 0 1 1 0 0 0 1 1 0 1 0 1 1 0 0 1 1 0 1 1 1 1];

R=[1 1 1 1 0 1 0 0 0 1 1 0 0 0 1 1 1 1 1 0 1 0 1 0 0 1 0 0 1 0 1 0 0 0 1];

S=[0 1 1 1 0 1 0 0 0 1 1 0 0 0 0 0 1 1 1 0 0 0 0 0 1 1 0 0 0 1 0 1 1 1 0];

T=[1 1 1 1 1 0 0 1 0 0 0 0 1 0 0 0 0 1 0 0 0 0 1 0 0 0 0 1 0 0 0 0 1 0 0];

%A1=[1 1 1 1 0 1 0 0 0 1 1 0 0 0 1 1 1 1 1 1 1 0 0 0 1 1 0 0 0 1 0 0 0 0 1];

%A2=[0 1 1 1 1 1 0 0 0 1 1 0 0 0 1 1 1 1 1 1 1 0 0 0 1 1 0 0 0 1 1 0 0 0 1];

%A3=[0 1 1 1 0 1 0 0 0 1 1 0 0 0 1 1 0 1 1 0 1 0 0 0 1 1 0 0 0 1 1 0 0 0 1];

wyjscie = eye(20);

litery=[A;B;C;D;E;F;G;H;I;J;K;L;M;N;O;P;Q;R;S;T];

% litery=[A;B;C;D;E;F;G;H;I;J;K;L;M;N;O;P;Q;A1;A2;A3];

litery=litery';

testA=[0; 1; 1; 1; 0; 1; 0; 0; 0; 1; 1; 0; 0; 0; 1; 1 ;1 ;1; 1; 1; 1 ;0; 0; 0; 1; 1 ;0; 0; 0; 1; 1 ;0; 0; 0; 1;];

testB=[1; 1; 1; 0; 1; 0; 0; 0; 1; 1; 0; 0; 0; 1; 1; 1; 1; 1; 0; 1; 0; 0; 0; 1; 1; 0; 0; 0; 1; 1; 1; 1; 1; 0; 0;];

testC=[0; 1; 1; 1; 0; 1; 0; 0; 0; 1; 1; 0; 0; 0; 0; 1; 0; 0; 0; 0; 1; 0; 0; 0; 0; 1; 0; 0; 0; 1; 0; 1; 1; 1; 0;];

testD=[1; 1; 1; 1; 0; 1; 0; 0; 0; 1; 1; 0; 0; 0; 1; 1; 0; 0; 0; 1; 1; 0; 0; 0; 1; 1; 0; 0; 0; 1; 1; 1; 1; 1; 0;];

testE=[1; 1; 1; 1; 1; 1; 0; 0; 0; 0; 1; 0; 0; 0; 0; 1; 1; 1; 1; 0; 1; 0; 0; 0; 0; 1; 0; 0; 0; 0; 1; 1; 1; 1; 1;];

testF=[1; 1; 1; 1; 1; 1; 0; 0; 0; 0; 1; 0; 0; 0; 0; 1; 1; 1; 1; 0; 1; 0; 0; 0; 0; 1; 0; 0; 0; 0; 1; 0; 0; 0; 0;];

testG=[0; 1; 1; 1; 0; 1; 0; 0; 0; 1; 1; 0; 0; 0; 0; 1; 0; 1; 1; 1; 1; 0; 0; 0; 1; 1; 0; 0; 0; 1; 0; 1; 1; 1; 0;];

testH=[1; 0; 0; 0; 1; 1; 0; 0; 0; 1; 1; 0; 0; 0; 1; 1; 1; 1; 1; 1; 1; 0; 0; 0; 1; 1; 0; 0; 0; 1; 1; 0; 0; 0; 1;];

testI=[0; 1; 1; 1; 0; 0; 0; 1; 0; 0; 0; 0; 1; 0; 0; 0; 0; 1; 0; 0; 0; 0; 1; 0; 0; 0; 0; 1; 0; 0; 0; 1; 1; 1; 0;];

testJ=[1; 1; 1; 1; 1; 0; 0; 0; 0; 1; 0; 0; 0; 0; 1; 0; 0; 0; 0; 1; 0; 0; 0; 0; 1; 1; 0; 0; 0; 1; 0; 1; 1; 1; 0;];

testK=[1; 0; 0; 0; 1; 1; 0; 0; 1; 0; 1; 0; 1; 0; 0; 1; 1; 0; 0; 0; 1; 0; 1; 0; 0; 1; 0; 0; 1; 0; 1; 0; 0; 0; 1;];

testL=[1; 0; 0; 0; 0; 1; 0; 0; 0; 0; 1; 0; 0; 0; 0; 1; 0; 0; 0; 0; 1; 0; 0; 0; 0; 1; 0; 0; 0; 0; 1; 1; 1; 1; 1;];

testM=[1; 0; 0; 0; 1; 1; 1; 0; 1; 1; 1; 0; 1; 0; 1; 1; 0; 0; 0; 1; 1; 0; 0; 0; 1; 1; 0; 0; 0; 1; 1; 0; 0; 0; 1;];

testN=[1; 0; 0; 0; 1; 1; 0; 0; 0; 1; 1; 1; 0; 0; 1; 1; 0; 1; 0; 1; 1; 0; 0; 1; 1; 1; 0; 0; 0; 1; 1; 0; 0; 0; 1;];

testO=[0; 1; 1; 1; 0; 1; 0; 0; 0; 1; 1; 0; 0; 0; 1; 1; 0; 0; 0; 1; 1; 0; 0; 0; 1; 1; 0; 0; 0; 1; 0; 1; 1; 1; 0;];

testP=[1; 1; 1; 1; 0; 1; 0; 0; 0; 1; 1; 0; 0; 0; 1; 1; 1; 1; 1; 0; 1; 0; 0; 0; 0; 1; 0; 0; 0; 0; 1; 0; 0; 0; 0;];

testQ=[0; 1; 1; 1; 0; 1; 0; 0; 0; 1; 1; 0; 0; 0; 1; 1; 0; 0; 0; 1; 1; 0; 1; 0; 1; 1; 0; 0; 1; 1; 0; 1; 1; 1; 1;];

testR=[1; 1; 1; 1; 0; 1; 0; 0; 0; 1; 1; 0; 0; 0; 1; 1; 1; 1; 1; 0; 1; 0; 1; 0; 0; 1; 0; 0; 1; 0; 1; 0; 0; 0; 1;];

testS=[0; 1; 1; 1; 0; 1; 0; 0; 0; 1; 1; 0; 0; 0; 0; 0; 1; 1; 1; 0; 0; 0; 0; 0; 1; 1; 0; 0; 0; 1; 0; 1; 1; 1; 0;];

testT=[1; 1; 1; 1; 1; 0; 0; 1; 0; 0; 0; 0; 1; 0; 0; 0; 0; 1; 0; 0; 0; 0; 1; 0; 0; 0; 0; 1; 0; 0; 0; 0; 1; 0; 0;];

testA1=[0; 1; 0; 1; 0; 1; 0; 0; 0; 1; 1; 0; 0; 0; 1; 1 ;1 ;1; 1; 1; 1 ;0; 0; 0; 1; 1 ;0; 0; 0; 1; 1 ;0; 0; 0; 1;];

testA2=[0; 1; 0; 1; 0; 1; 0; 0; 0; 1; 0; 0; 0; 0; 1; 1 ;1 ;0; 1; 1; 1 ;0; 0; 0; 1; 1 ;0; 0; 0; 1; 1 ;0; 0; 0; 0;];

testA3=[0; 0; 0; 0; 0; 1; 0; 0; 0; 1; 1; 0; 0; 0; 1; 1 ;1 ;0; 1; 1; 1 ;0; 0; 0; 1; 1 ;0; 0; 0; 1; 1 ;0; 0; 0; 1;];

%net=newff( minmax(litery), 20, {'logsig', 'purelin', 'train', 'learnh'});

net=newff( minmax(litery), 20, {'logsig', 'purelin', 'train', 'learnhd'});

net.trainParam.epochs = 100;

net.trainParam.goal = 0.001; %Cel wydajności

net.trainParam.lr=0.08; % wskażnik uczenia sie

lp.dr = 0.01;

lp.lr = 0.01;

%learnh([],litery,[],[],wyjscie,[],[],[],[],[],lp,[])

learnhd([0],litery,[0],[0],wyjscie,[0],[0],[0],[0],[0],lp,[0]);

wynik=sim(net, litery);

litera=input('podaj litere do rozpoznania : ', 's');

net.trainFcn = 'trainlm';

net.adaptFcn = 'adaptwb';

%net.inputWeights{:,:}.learnFcn = 'learnh';

%net.layerWeights {:,:} .learnFcn = 'learnh';

net.inputWeights{:,:}.learnFcn = 'learnhd';

net.layerWeights {:,:} .learnFcn = 'learnhd';

net.trainParam.epochs = 30;

net=train(net, litery, wyjscie);

wynik=sim(net, litery);

switch litera % Sprawdzenie czy jest taka literka ktora podalismy

case 'A'

test=testA;

case 'B'

test=testB;

case 'C'

test=testC;

case 'D'

test=testD;

case 'E'

test=testE;

case 'F'

test=testF;

case 'G'

test=testG;

case 'H'

test=testH;

case 'I'

test=testI;

case 'J'

test=testJ;

case 'K'

test=testK;

case 'L'

test=testL;

case 'M'

test=testM;

case 'N'

test=testN;

case 'O'

test=testO;

case 'P'

test=testP;

case 'Q'

test=testQ;

case 'R'

test=testR;

%test=testA1;

case 'S'

test=testS;

%test=testA2;

case 'T'

test=testT;

%test=testA3;

case 'A1'

test=testA1;

case 'A2'

test=testA2;

case 'A3'

test=testA3;

otherwise

disp('nie ma takiej litery')

end

test1=sim(net,test)

for i=1:20

if(max(test1)== test1(i)) % Sprawdzenie czy nauczyła sie podanej przez nas literki

if(i==1)

msgbox('A');

end

if(i==2)

msgbox('B');

end

if(i==3)

msgbox('C');

end

if(i==4)

msgbox('D');

end

if(i==5)

msgbox('E');

end

if(i==6)

msgbox('F');

end

if(i==7)

msgbox('G');

end

if(i==8)

msgbox('H');

end

if(i==9)

msgbox('I');

end

if(i==10)

msgbox('J');

end

if(i==11)

msgbox('K');

end

if(i==12)

msgbox('L');

end

if(i==13)

msgbox('M');

end

if(i==14)

msgbox('N');

end

if(i==15)

msgbox('O');

end

if(i==16)

msgbox('P');

end

if(i==17)

msgbox('Q');

end

if(i==18)

msgbox('R');

end

if(i==19)

msgbox('S');

end

if(i==20)

msgbox('T');

end

end

end