Akademia Górniczo-Hutnicza
WIMIiP, Inżynieria Obliczeniowa
grupa laboratoryjna 01
Agnieszka Kępka

28.12.2018,Kraków

Podstawy Sztucznej Inteligencji

Sprawozdanie numer 5

Budowa i działanie sieci Kohonena dla WTA

1. Cel ćwiczenia:

Celem ćwiczenia jest poznanie budowy i działania sieci Kohonena przy wykorzystaniu reguły WTA do odwzorowywania istotnych cech kwiatów.

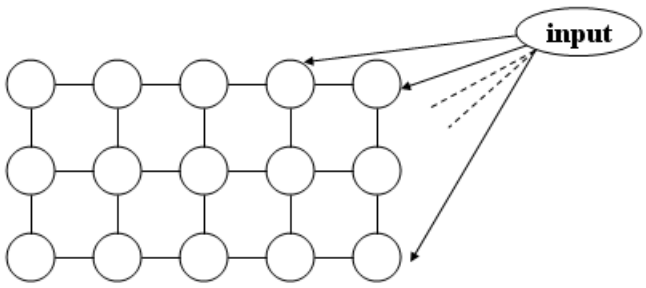
2. Zadania do wykonania:

- Przygotowanie danych uczących zawierających numeryczny opis cech kwiatów.
- Przygotowanie sieci Kohonena i algorytmu uczenia opartego o regułę Winner Takes All(WTA).
- Uczenie sieci dla różnych współczynników uczenia.
- Testowanie sieci.

3. Zagadnienia teoretyczne:

Sieci Kohonena są szczególnym przypadkiem algorytmu realizującego uczenie się bez nadzoru. Ich głównym zadaniem jest organizacja wielowymiarowej informacji np. obiektów opisanych 50 parametrami w taki sposób, żeby można ją było prezentować i analizować w przestrzeni o znacznie mniejszej liczbie wymiarów, czyli mapie (np. na dwuwymiarowym ekranie).
Warunek: rzuty "podobnych" danych wejściowych powinny być bliskie również na mapie.
Sieci Kohonena znane są też pod nazwami **Self-Organizing Maps (SOM).**

Topologia sieci Kohonena odpowiada topologii docelowej przestrzeni. Jeśli np. chcemy prezentować wynik na ekranie, rozsądnym modelem jest prostokątna siatka węzłów (im więcej, tym wyższą rozdzielczość będzie miała mapa):



Zasady działania sieci Kohonena:

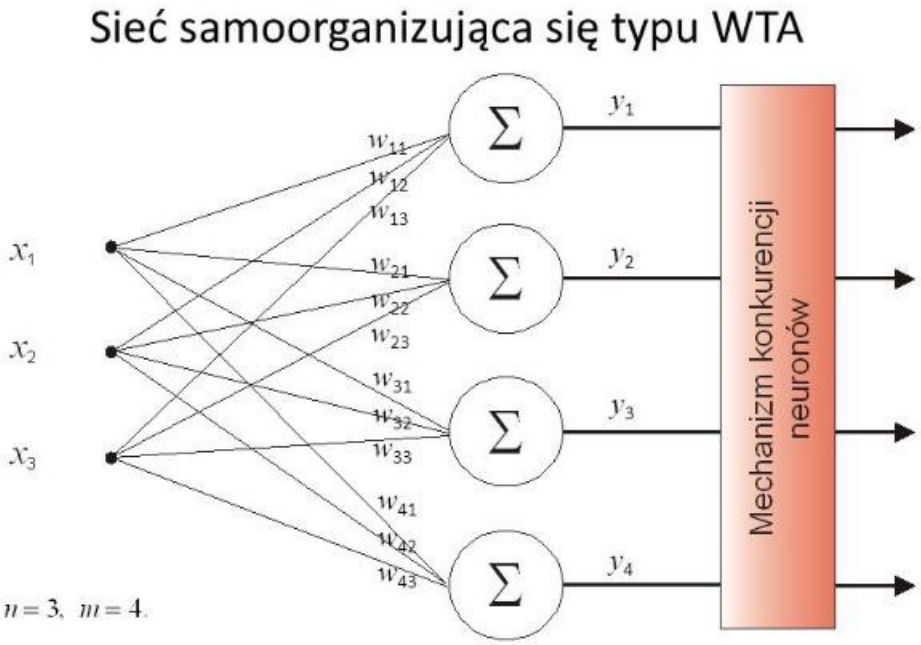
- Wejścia (tyle, iloma parametrami opisano obiekty) połączone są ze wszystkimi węzłami sieci
- Każdy węzeł przechowuje wektor wag o wymiarze identycznym z wektorami wejściowymi
- Każdy węzeł oblicza swój poziom aktywacji jako iloczyn skalarny wektora wag i wektora wejściowego (podobnie jak w zwykłym neuronie)
- Ten węzeł, który dla danego wektora wejściowego ma najwyższy poziom aktywacji, zostaje zwycięzcą i jest uaktywniony
- Wzmacniamy podobieństwo węzła-zwycięzcy do aktualnych danych wejściowych poprzez dodanie do wektora wag wektora wejściowego (z pewnym współczynnikiem uczenia)
- Każdy węzeł może być stowarzyszony z pewnymi innymi, sąsiednimi węzłami - wówczas te węzły również zostają zmodyfikowane, jednak w mniejszym stopniu.

Inicjalizacja wag sieci Kohonena jest losowa. Wektory wejściowe stanowią próbę uczącą, podobnie jak w przypadku zwykłych sieci rozpatrywaną w pętli podczas budowy mapy. Wykorzystanie utworzonej w ten sposób mapy polega na tym, że zbiór obiektów umieszczamy na wejściu sieci i obserwujemy, które węzły sieci się uaktywniają. Obiekty podobne powinny trafiać w podobne miejsca mapy.

Algorytm WTA

Algorytm WTA polega na obliczaniu aktywacji każdego neuronu, a następnie wyborze zwycięzcy o największym sygnale wyjściowym.

Sieć WTA jest siecią jednowarstwową należącą do grupy sieci samoorganizujących się z konkurencją. Ciąg uczący się składa się jedynie z wektorów wejściowych (sieć bez nauczyciela). Istotą działania sieci jest współzawodnictwo neuronów ze sobą – jeden z nich wygrywa przy prezentacji pojedynczego wektora – stąd akronim WTA (ang. Winner Takes All – „zwycięzca bierze wszystko"). Reguła „wygrywający bierze wszystko" przyznaje więc prawo do korekcji wag tylko temu neuronowi, który jest najbardziej wrażliwy na aktualny wektor wejściowy. W efekcie współzawodnictwa następuje samoorganizacja procesu uczenia. Neurony dopasowują swoje wagi w taki sposób, że dla grupy wektorów bliskich sobie (wg pewnej miary odległości – szczegóły dalej) zwycięża zawsze ten sam neuron. Zatem układy tego typu mogą być stosowane do klasyfikacji wektorów.



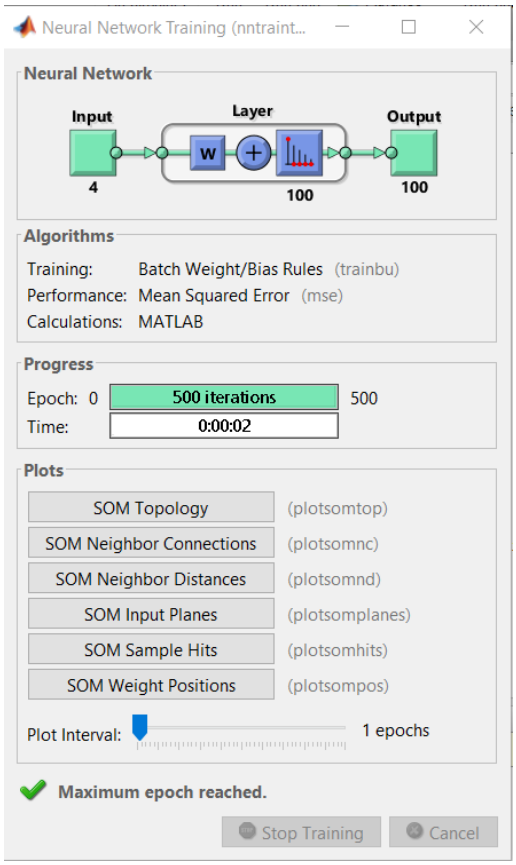
Po podaniu na wyjściu wektora uczącego x=(x1,x2,x3) mechanizm konkurencji wybierze tylko jeden z czterech pokazanych neuronów(zwycięzcę). Tylko jego wagi będą modyfikowane (w tym cyklu, gdyż dla następnego wektora zwycięzca może być inny).

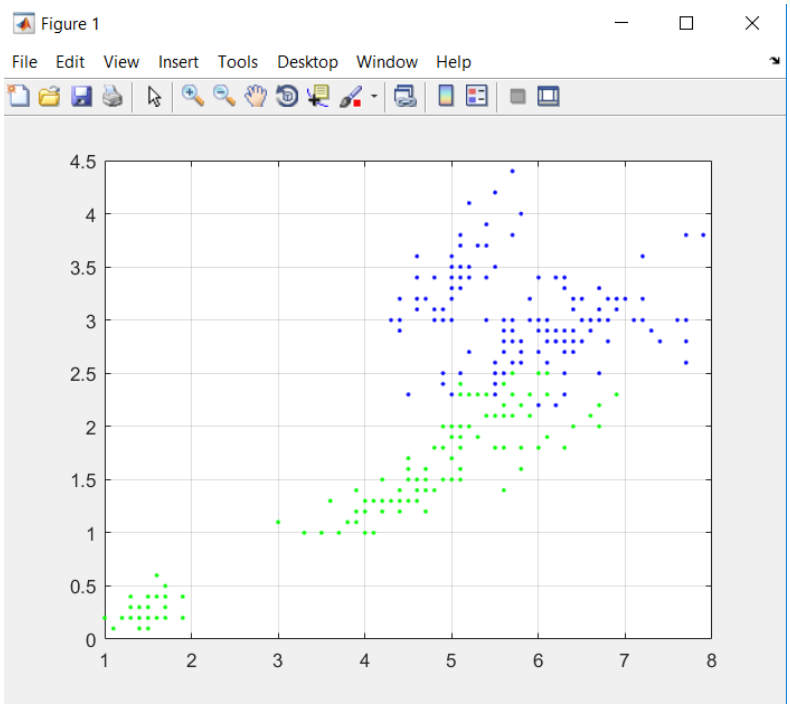
Dlaczego WTA nie wystarcza?
Taka reguła adaptacji powoduje, że algorytm słabo zbiega. Reguła, w której modyfikuje się nie tylko wagi zwycięzcy, ale również jego sąsiadów, jest zwana regułą WTM (Winer Takes Most). W praktyce reguła WTA została zastąpiona regułą WTM.

4. Wyniki przeprowadzonego ćwiczenia:

W programie Matlab, za pomocą biblioteki Neural Network Toolbox, zaimplementowałam sztuczną sieć neuronową. Danymi wejściowymi jest zestaw zaimplementowany w oprogramowaniu MATLAB o nazwie iris_dataset. Zawiera on opis 4 cech kwiatów irysa, tj. długość i szerokość płatka oraz długość i szerokość działki kielicha.

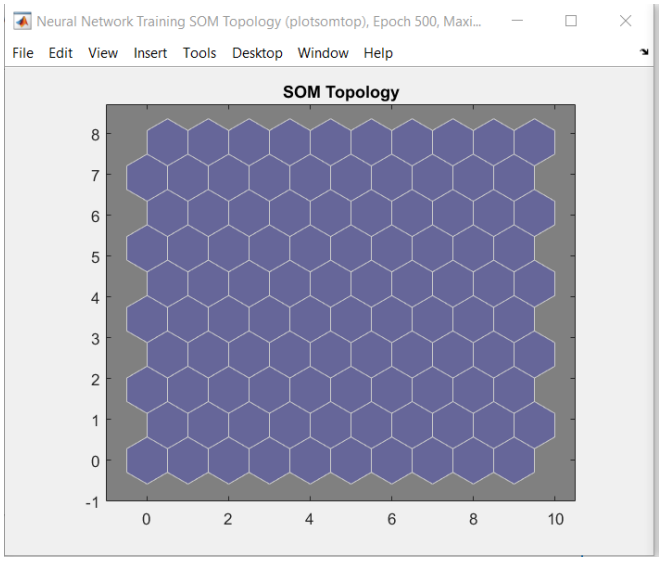
Wykorzystałam heksagonalną siatkę neuronów. Uczenie sieci nastąpiło wg reguły Kohonena oraz WTA. Program odwzorowuje istotne cech kwiatów irysa na podstawie danych wejściowych.

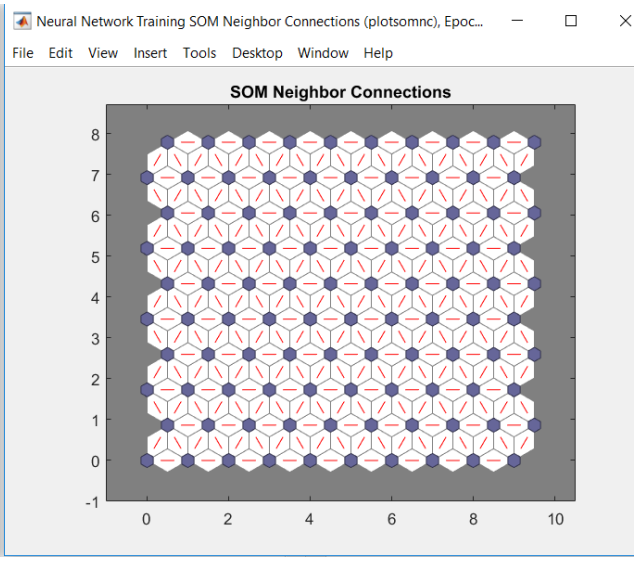




Szkolenie obejmuje maksymalną liczbę epok, która wynosi 500, czas wykonania zadania to 0:00:02.

Wykres obok przedstawia długość i szerokość płatka (fioletowe kropki) oraz długość i szerokość kielicha (zielone kropki) wszystkich kwiatów znajdujących się w danych uczących.



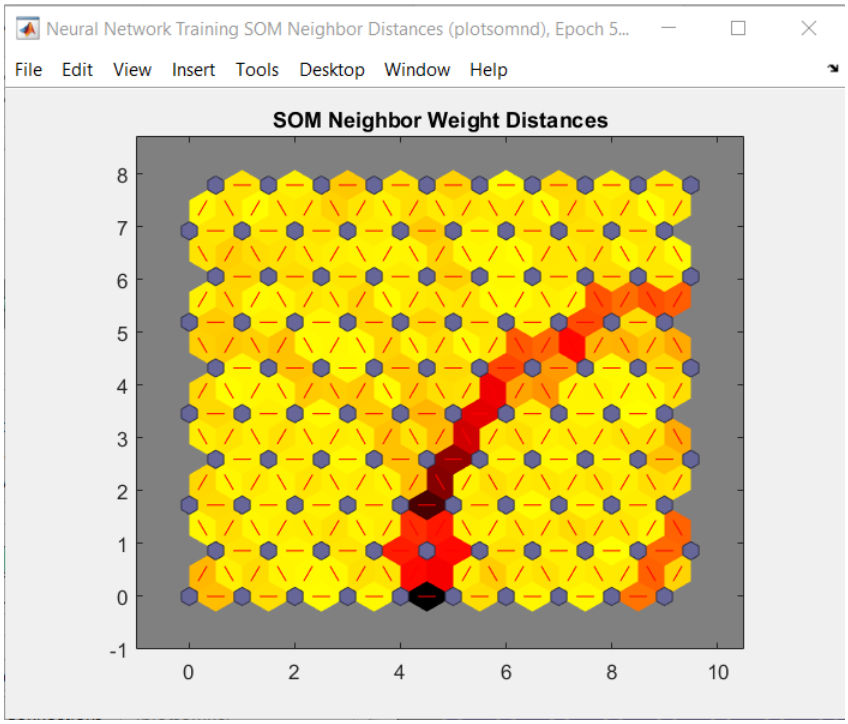


SOM Topology:

Na tej figurze każdy z sześciokątów reprezentuje neuron. W każdym wektorze wejściowym znajdują się cztery elementy, więc przestrzeń wejściowa jest czterowymiarowa. Wektory masy (centra skupień) mieszczą się w tej przestrzeni.

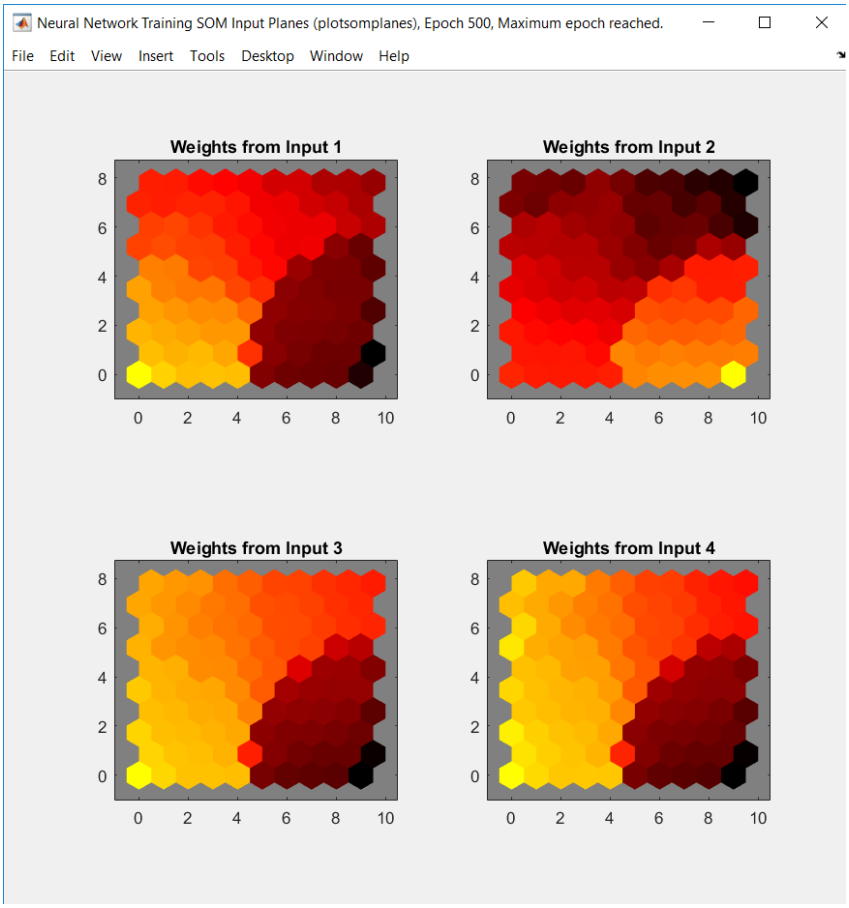
SOM Neighbor Connections:

Mapa ta przedstawia powiązania sąsiedzkie pomiędzy poszczególnymi neuronami.



SOM Neighbor Distance:

Na tej figurze niebieskie sześciokąty reprezentują neurony. Czerwone linie łączą sąsiednie neurony. Kolory w obszarach zawierających czerwone linie wskazują odległości między neuronami. Ciemniejsze kolory reprezentują większe odległości, a jaśniejsze kolory oznaczają mniejsze odległości. Pasmo ciemnych segmentów przechodzi od środkowego dolnego regionu do środkowego prawego regionu.

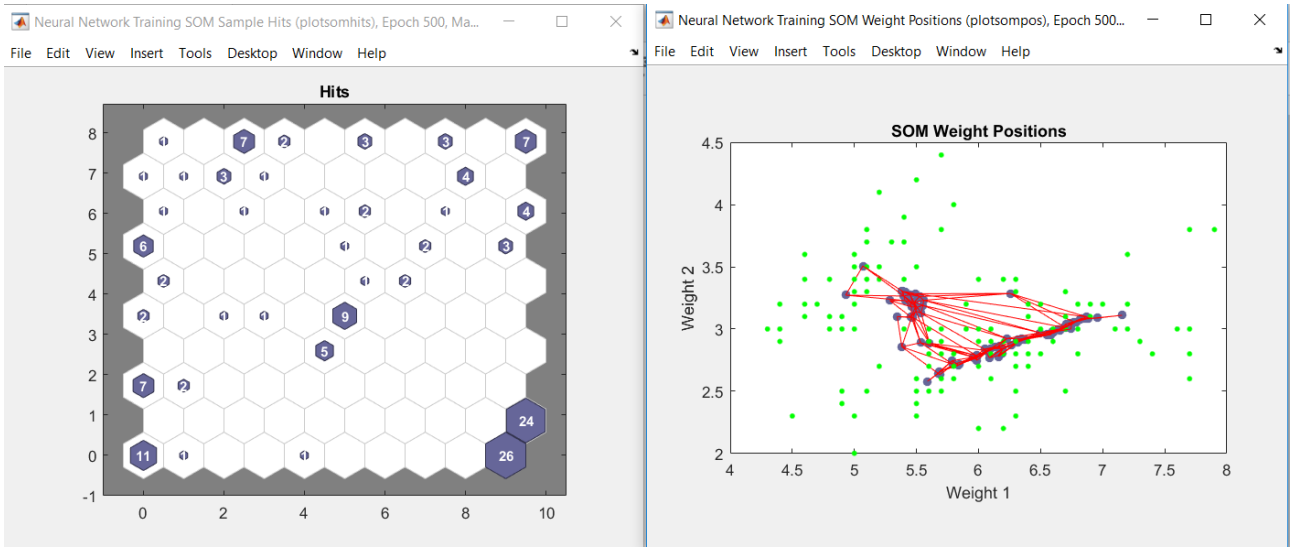


Rozkład wag dla każdego z wejść

SOM Input Planes:

Ta figura przedstawia płaszczyznę ciężkości dla każdego elementu wektora wejściowego (w tym przypadku cztery). Są to wizualizacje wag, które łączą każde wejście z każdym z neuronów. (**Ciemniejsze kolory oznaczają większe wagi**.) Jeśli schematy połączeń dwóch wejść były bardzo podobne, można założyć, że dane wejściowe są wysoce skorelowane.

Wejścia to:
1. długość płatka
2. szerokość płatka
3. długość kielicha
4. szerokość kielicha



SOM Sample Hits:

wektor wagowy związany z każdym neuronem przesuwa się, aby stać się centrum skupienia wektorów wejściowych. Ponadto, neurony, które sąsiadują ze sobą w topologii, powinny również poruszać się blisko siebie w przestrzeni wejściowej, dlatego możliwe jest zwizualizowanie wielowymiarowej przestrzeni wejściowej w dwóch wymiarach topologii sieci.
Zdjęcie to przedstawia ile razy dany neuron zwyciężył podczas rywalizacji.

SOM Weight Positions:

Efekt końcowy – kropki zielone przedstawiają poszczególne kwiaty, a niebieskie to kwiaty, które zawierają typowe cechy irysa

Wnioski:

- Celem sieci SOM jest klasyfikowanie wielowymiarowych danych wejściowych - obiektów opisanych dużą ilością parametrów – w taki sposób, by przedstawić reprezentację tych danych w mniejszej ilości wymiarów, przeważnie dwóch, przy możliwie jak najwierniejszym odwzorowaniu struktury wewnętrznej wektora wejściowego.

- Sieci te przydatne są przy wizualizacjach skomplikowanych struktur, a przetworzone przez nie dane mogą stanowić podstawę do diagramów wyświetlanych na ekranie, jak również znajdują zastosowanie wszędzie tam, gdzie istotna jest redukcja rozmiarów danych wejściowych, ze względu na zdolność kompresji samoorganizującej sieci Kohonena.

- Należy zwrócić szczególną uwagę przy doborze ilości neuronów, ponieważ WTA powoduje konkurencję między neuronami.

- Na podstawie zdobytych informacji przez sieć można domyślić się, że kwiat irysa posiada długie i szerokie liście oraz długi i szeroki kielich. Informacje te pokrywają się z wyglądem kwiatu irysa.

- Analizując SOM weight positions można wywnioskować, że sieć w poprawny sposób wytypowała typowe cechy irysa. Kwiaty znajdujące się pośrodku wykresu miały przeważnie parametry w okolicach ich średnich. Sugeruje, to że były zbliżone do kwiatu irysa.

- Liczba epok 500 jest optymalna do poprawnego działania programu.

Kod programu:

```
close all; clear all; clc;

WE = iris_dataset;
size(WE);
plot(WE(1,:),WE(2,:),'b.',WE(3,:),WE(4,:),'g.');
hold on; grid on;

% Parametry sieci:
dimensions   = [10 10];
coverSteps   = 100;
initNeighbor = 0;
topologyFcn  = 'hextop';
distanceFcn  = 'dist';

% Tworzenie SOM:
net =
selforgmap(dimensions,coverSteps,initNeighbor,topologyFcn,distance
Fcn);
net.trainParam.epochs = 500;

% Trenowanie sieci:
[net,tr] = train(net,WE);
y = net(WE);

grid on
```