Sprawozdanie Scenariusz 5 Aleksandra Karaś

Budowa i działanie sieci Kohonena dla WTA

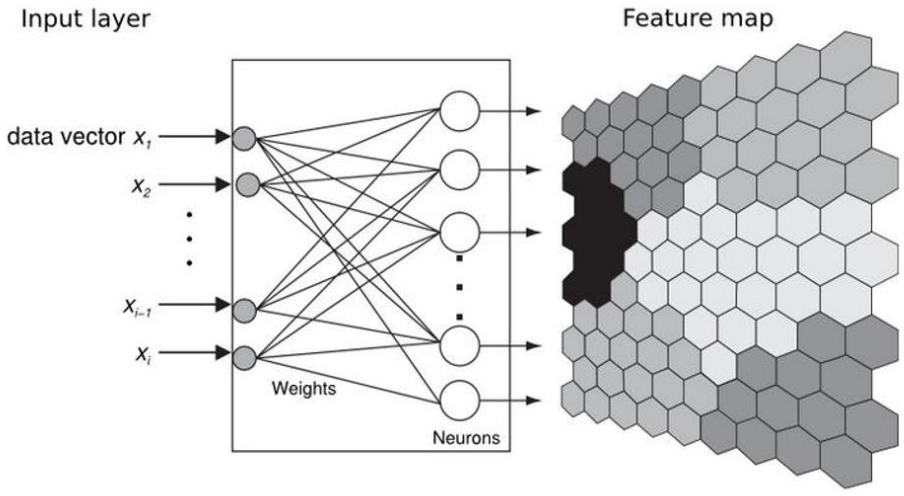
Cel ćwiczenia:

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

Opis budowy użytej sieci i algorytmu uczenia:

Celem zbudowanej sieci jest podział danych uczących na określone grupy i przyporządkowanie danej grupie danego elementu wyjściowego. Podział na grupy polega na tym, żeby elementy w danej grupie były jak najbardziej podobne do siebie jednocześnie będąc zupełnie inne od elementów z innych grup.

Nauka sieci odbywa się za pomocą uczenia rywalizującego (metoda uczenia sieci samoorganizujących). Podczas procesu uczenia neurony są nauczane rozpoznawania danych i zbliżają się do obszarów zajmowanych przez te dane. Po wejściu każdego wektora uczącego wybierany jest tylko jeden neuron (neuron będący najbliższemu prezentowanemu wzorcowi). Wszystkie neurony rywalizują między sobą, gdzie zwycięża ten neuron, którego wartość jest największa. Zwycięski neuron przyjmuje na wyjściu wartość 1, pozostałe 0.



Rys. 1 Sieć samoorganizująca

Proces uczenia przebiega według następującego schematu:

1. Normalizacja wszystkich danych

2. Wybór współczynnika uczenia η z przedziału (0; 1>

3. Wybór początkowych wartości wag z przedziału <0; 1)

4. Dla danego zbioru uczącego obliczamy odpowiedź sieci – dla każdego pojedynczego neuronu obliczana jest suma ilorazów sygnałów wejściowych oraz wag

5. Wybierany jest neuron, dla którego obliczona suma jest największa. Tylko dla tego neuronu następuje aktualizacja wag.

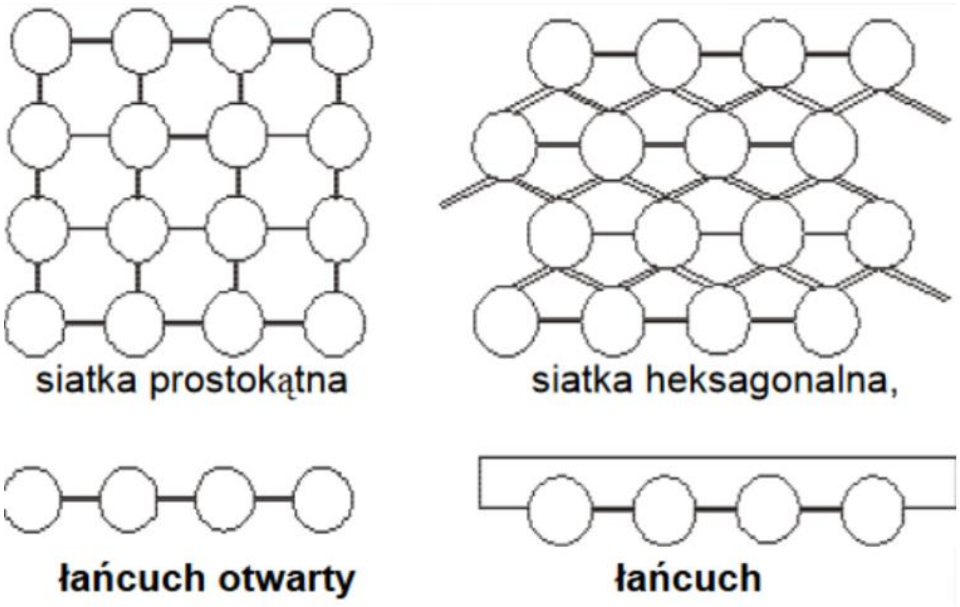
Wzór na zaktualizowanie wagi jest następujący: $wi,j(t + 1) = wi,j(t) + \eta * (xi * wi,j(t))$

6. Znormalizowanie wartości nowego wektora wag

7. Zwycięski neuron daje odpowiedź na swoim wyjściu równą 1, a pozostałe 0.

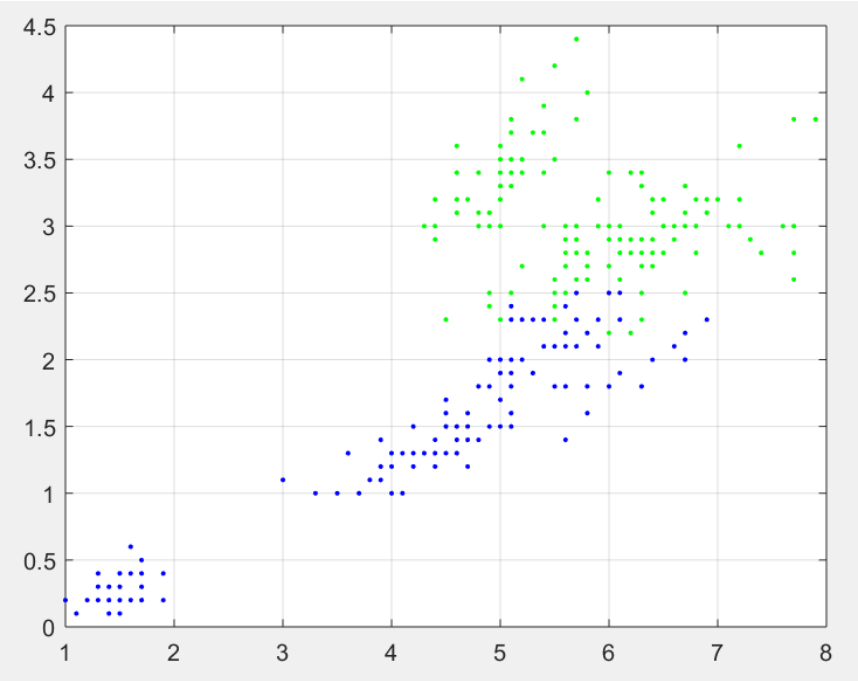
8. Wczytanie kolejnego wektora uczącego.

Topologię sieci Kohonena można określić przez zdefiniowanie sąsiedztwa każdego neuronu. Sieć jest uporządkowana, jeśli topologiczne relacje między sygnałami wejściowymi i ich obrazami są takie same.



Przebieg programu:

Do stworzenia sieci wykorzystałam pakiet MATLAB narzędzie Neural Networking Training Tool. Danymi wejściowymi był wbudowany w pakiecie zbiór danych iris_dataset. Również wygenerowałam dwuwymiarowy wykres dla parametrów płatka i działki kielicha.



Wykorzystaliśmy funkcję do tworzenia map samoorganizacji selforgmap. W ramach wykonywania programu otrzymaliśmy 6 różnych wykresów, które mają swoje znaczenie:

• SOM Topology – jeden kształt symbolizuje jeden neutron. Są one rozmieszczone w siatce o określonych wymiarach. Ułożenie jest nieprzypadkowe – ich sąsiedztwo może wskazywać na ich podobieństwo.

• SOM Neighbor Distances – Na tej mapie otrzymamy możliwość sprawdzenia, jak bardzo połączone są ze sobą poszczególne neutrony – czyli jak silne podobieństwo one wykazują. Im jaśniejsze połączenie, tym bardziej te dane są do siebie podobne. Ciemne linie mogą zatem oznaczać granice klas.

• SOM Neighbor Connections – na tym wykresie umieszczone są połączenia między sąsiadami. Sąsiadami są zwykle próbki do siebie podobne.

• SOM Weight Planes – jest to zestaw wykresów, który wskazuje na rozkład wag poszczególnych neutronów w zależności od cechy. Im ciemniejszy kolor, tym większą wagę dany neuron skupia. Im więcej neuronów o podobnych kolorach w danym sąsiedztwie, tym te neurony są bardziej ze sobą skorelowane

• SOM Sample Hits – ten wykres pokazuje nam, ile podobnych wyników otrzymaliśmy dla danej klasy– im wyższa liczba, tym więcej obiektów o podobnych cechach można wykazać

• SOM Weight Positions – zielone kropki oznaczają dane wejściowe, a linie je łączące wskazują na korelacje pomiędzy poszczególnymi neuronami.

iris_dataset;

danymi wejściowymi jest zbiór danych, gdzie w I i II kolumnie mamy dane o długości i szerokości działki kielicha, a w III i IV kolumnie o długości i szerokości płatka (wszystkie wymiary podane są w cm)

plot(in_value(1, :) ,in_value(2, :), 'b.', in_value(3, :), in_value(4, :), 'g.');

narysowanie dwuwymiarowego wykresu zależności długości od szerokości

dimensions = [12 12]; wymiary wektora

coverstep = 50; etapy szkolenia w celu pokrycia przestrzeni wejściowej

initNeighbor = 0; wejściowy rozmiar sąsiedztwa

topologyFcn = 'hextop'; funkcja topologiczna

distanceFcn = 'dist';

funkcja dystansu neronów - miara euklidesowa (znormalizowana) domyślnym parametrem jest odległość między neuronami warstwy z uwzględnieniem ich położenia

net = selforgmap(dimensions, coverstep, initNeighbor, topologyFcn, distanceFcn);

tworzenie mapy samoorganizacji

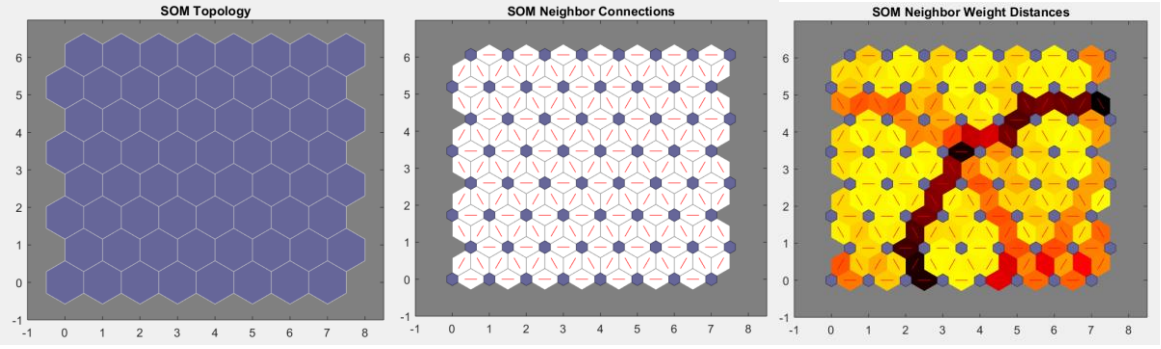
[net, tr] = train(net, in_value); trening sieci

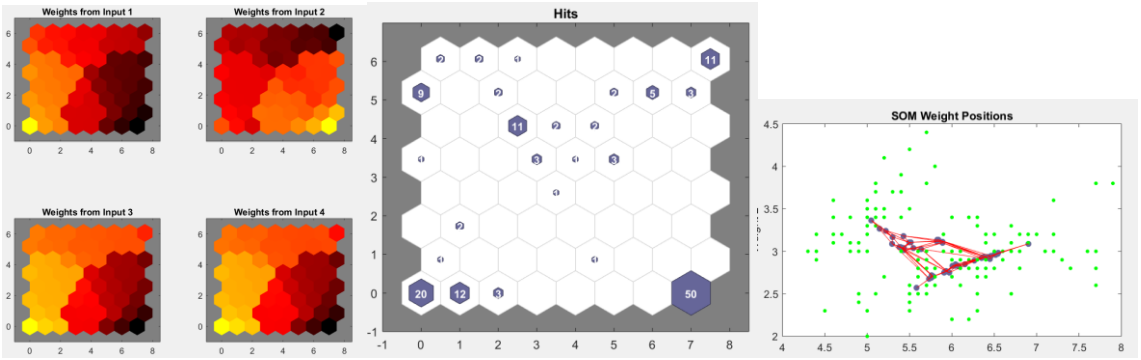
y = net(in_value); testowanie i zapis wyników osiągniętych przez sieć

Wyniki:

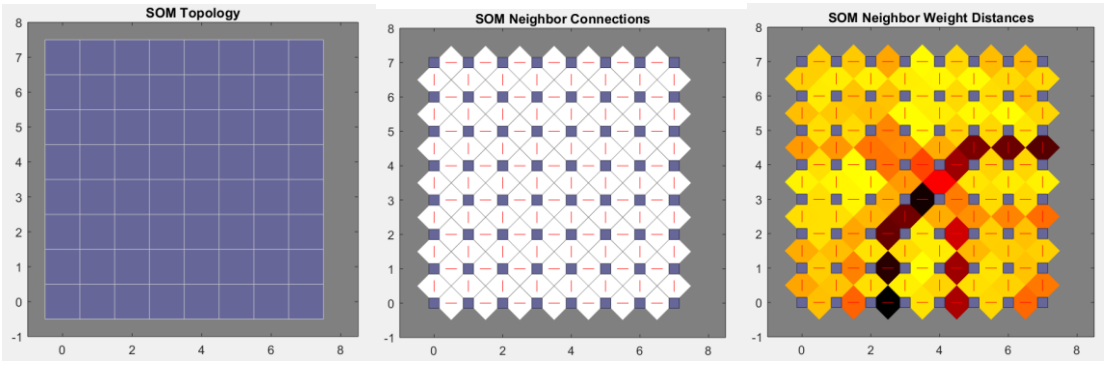
Pakiet MATLAB daje podgląd tylko dla topologii heksagonalnej i kwadratowej. Przyjęłam, wymiary siatki neuronów 8x8 oraz maksymalną liczbę epok uczenia równą 500.

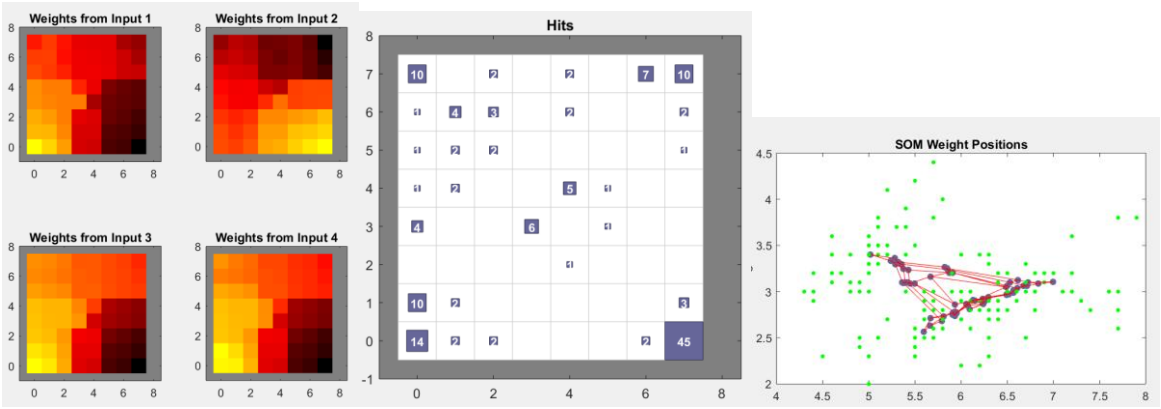
Wykresy dla topologii heksagonalnej (wsp. uczenia = 0,5)





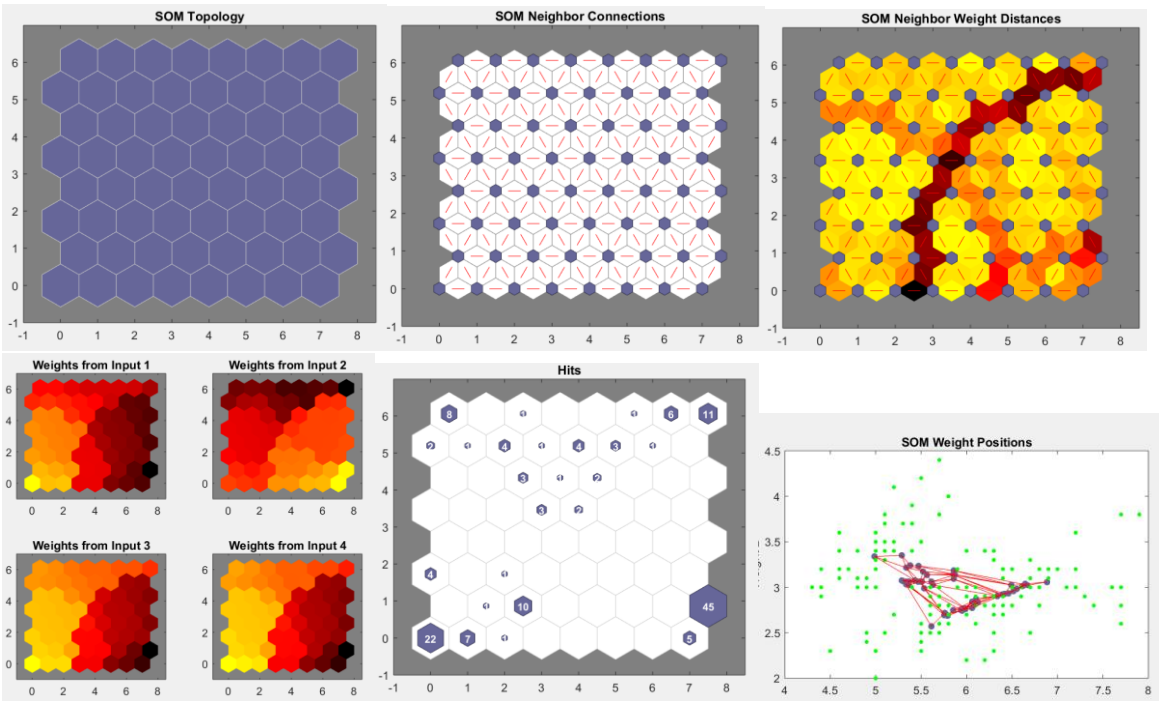
Wykresy dla topologii kwadratowej (wsp. uczenia = 0,5)



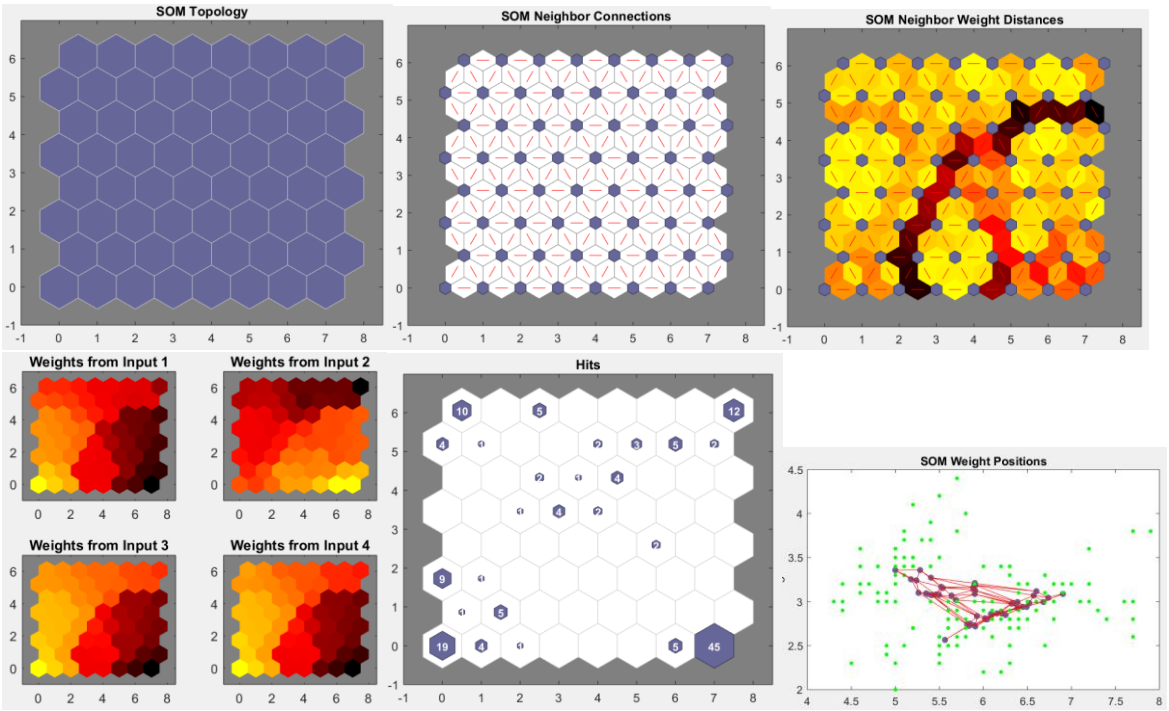


Ponieważ pewne cechy i tendencje lepiej są przedstawione przez topologię heksagonalną. Moje dalsze rozważania będę prowadzić względem tej właśnie topologii. Jednym z parametrów uczenia, od którego zależy algorytm uczenia jest współczynnik uczenia. Przedstawię jeszcze dwa warianty współczynnika uczenia (0.25 oraz 0.75)

Wykresy dla topologii heksagonalnej (wsp. uczenia = 0,25)



Wykresy dla topologii heksagonalnej (wsp. uczenia = 0,75)



Wnioski:

Sieć Kohonena cechuje umiejętność podziału danych, które posiadają różne wartości dla poszczególnych cech, ponieważ jest to sieć samoorganizująca. Powoduje to, że odpowiedni podział na grupy może być wykonywane bez podawania wartości oczekiwanych (uczenie bez nauczyciela). Cały proces uczenia (jego efektywność) zależy od współczynnika uczenia. Im większy jest ten współczynnik, tym sieć uczy się szybciej. Jednakże wzrost efektywności nie jest wprost proporcjonalny do współczynnika uczenia. Dla wzrostu przy małych współczynnikach uczenia następuje większa różnica w ilości potrzebnych epok aniżeli dla wzrostu przy dużych współczynnikach uczenia – występuje stabilizacja uczenia. Ponadto, sieci samoorganizujące potrafią się uczyć o wiele sprawniej od sieci podstawowych, co spowodowane jest typem danych uczących. Sieć dla odpowiednich współczynników uczenia pozwala uzyskać 100% poprawnych odpowiedzi, co powoduje, że sieć Kohonena sprawdza się przy dużej ilości różniących się pomiędzy sobą danych.

Listing kodu:

```
close all; clear all; clc;

data = iris_dataset;

plot(data(1,:),data(2,:),'g.', data(3,:),data(4,:),'b.');
hold on; grid on;

dimensions = [8 8];
coverSteps = 100;
initNeighbor = 0;
%topologyFcn = 'gridtop';
topologyFcn = 'hextop';
distanceFcn = 'dist';
net = selforgmap(dimensions,coverSteps,initNeighbor,topologyFcn,distanceFcn);
net.trainFcn = 'trainbu';
net.trainParam.epochs = 500;
net.trainParam.lr=0.5;

[net,tr] = train(net,data);
y = net(data);
```