**Scenariusz projektu nr 5**

Monika Pawlik, gr 03

**Temat ćwiczenia:**   
Budowa i działanie sieci Kohonena dla WTA

**Cel ćwiczenia:**  
Celem ćwiczenia jest poznanie budowy i działania sieci Kohonena przy wykorzystaniu reguły WTA do odwzorowania istotnych cech kwiatów.

1. **Opis budowy sieci Kohonena**

**Sieć Kohonena** – [sieć neuronowa](https://pl.wikipedia.org/wiki/Sie%C4%87_neuronowa) uczona w trybie bez nauczyciela w celu wytworzenia niskowymiarowej (przeważnie dwuwymiarowej) [zdyskretyzowanej](https://pl.wikipedia.org/wiki/Matematyka_dyskretna) reprezentacji przestrzeni wejściowej. Sieć Kohonena wyróżnia się tym od innych sieci, że zachowuje odwzorowanie sąsiedztwa przestrzeni wejściowej.

**Zasady działania sieci Kohonena:**

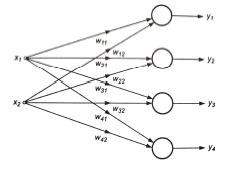
* Wejścia (tyle, iloma parametrami opisano obiekty) połączone są ze wszystkimi węzłami sieci
* Każdy węzeł przechowuje wektor wag o wymiarze identycznym z wektorami wejściowymi
* Każdy węzeł oblicza swój poziom aktywacji jako iloczyn skalarny wektora wag i wektora wejściowego (podobnie jak w zwykłym 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.

1. **Uczenie sieci przy wykorzystaniu reguły WTA**

**WTA – Winner Takes All** – Zwycięzca bierze wszystko. Po przedstawieniu sieci wektora wejściowego, neuron najbardziej podobny do elementu prezentowanego (którego wagi są najbardziej podobne składowym wektora wejściowego) zostaje zmodyfikowany zgodnie z funkcją f tak aby jego wagi były jak najbardziej zbliżone do wektora wejściowego.

Neuron typu WTA – mają stopień wejściowy w postaci standardowego sumatora dokonującego sumowania sygnałów wejściowych z odpowiednimi wagami.

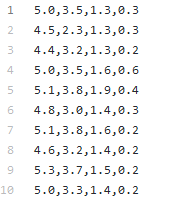


**Rys. Sieć neuronowa typu WTA**

1. **Dane uczące i testujące**

W programie zestaw danych uczących stanowi zbiór danych zawierający 30 instancji opisujących kwiaty irysa. Kwiaty są określone przy pomocy 4 atrybutów numerycznych opisujących długości i szerokości płatków kwiatu sepal i petal. (Do implementacji programu zostały pozyskane dane ze zbioru Iris Data Set utworzonego przez genetyka Ronalda Fishera. )

Przykład kilku rekordów pliku zawierającego dane:



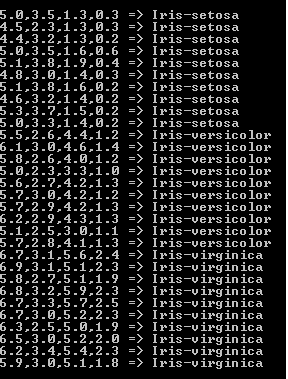
Opis: 1 wers zawiera 4 wartości pomiarów długości i szerokości płatków

Sepal.Length Sepal.Width Petal.Lenght Petal.Width

|  |
| --- |
| 5.0 3.5 1.3 0.3 |
|  |

1. **Analiza i zestawienie otrzymanych wyników**

Implementacja aplikacji rozwiązuje problem jakim jest określenie gatunku dla nowego kwiatu irysa w zależności od wielkości jego płatków. A więc napisany program na podstawie 4 podanych parametrów poda właściwą klasę obiektu – jedną z 3 dostępnych : setosa, versicolor i virginicacs.   
Screen konsoli:



Screen konsoli przedstawia wynik działania programu. Ostatnia kolumna to kolumna klasy obiektu pokazująca rozwiązanie. Sieć została nauczona poprawnie, wszystkie cechy zostały prawidłowo rozpoznane.

**Analiza ze względu na współczynnik uczenia**

Uczenie zostało przeprowadzone dla różnych współczynników uczenia. Współczynnik uczenia może przyjmować wartości z przedziału (0,1).

Im mniejszy współczynnik uczenia, tym mniejsza liczba iteracji, co ma wpływ na działanie programu, ponieważ zdarzały się błędne wyniki. Zmiany wag neuronów nie były bardzo skokowe Zredukowanie błędnych wyników można uzyskać poprzez zwiększenie liczby iteracji.

Wysoki współczynnik uczenia, np. 0.8, spowodował że wagi neuronów w dużym stopniu upodobniają się do prezentowanych wzorców.

1. **Wnioski**

Rozwiązanie zadania jest wykonane poprzez wykorzystanie algorytmu uczącego jakim jest WTA z siecią Kohonena. Algorytm uczący na podstawie zbioru uczącego tworzy reprezentacje rozwiązania w postaci modelu.

Uczenie się sieci ma charakter iteracyjny, co oznacza, że zbiór danych wejściowych jest wielokrotnie prezentowany podczas kolejnych epok trenowania.

Dla sieci samoorganizującej Kohonena ilość danych uczących jest ważna – przy zbyt małej można otrzymać błędne wyniki. W programie zbiór 30 wersów danych okazał się wystarczający.

Współczynnik uczenia oddziałuje na zmiany wag, które mają bezpośredni wpływ na działanie neuronów w sieci, w zależności od ich wartości zmienia się poprawność wyników i liczba iteracji.

Dla zasady WTA bardzo ważne jest określenie sąsiedztwa. Jest to położenie neuronu względem zwycięzcy.

Listing kodu:

var synaptic = require('synaptic');

var fs = require('fs');

var Layer = synaptic.Layer,

Network = synaptic.Network;

// set up neural network

var inputLayer = new Layer(4);

var hiddenLayer = new Layer(5);

var outputLayer = new Layer(3);

inputLayer.project(hiddenLayer);

hiddenLayer.project(outputLayer);

var network = new Network({

input: inputLayer,

hidden: [hiddenLayer],

output: outputLayer

});

var som = require('node-som');

var somInstance = new som({

inputLength: 4,

maxClusters: 10,

loggingEnabled:true,

scale:{

min:0,

max:10

}

});

somInstance.train(trainingData);

var sample = [5.9,3.0,5.1,1.8];

var group = somInstance.classify(sample);

console.log(getFlowerName2(group));

var trainingData = [];

fs.readFile('iris.txt','utf8',function(err, data){

if(err) throw err;

var lines = data.split("\n");

for(var i = 0; i < lines.length; i++){

var line = lines[i].trim();

var splitLine = line.split(",");

var input = splitLine.slice(0, 4);

var output = splitLine[4]=='Iris-virginica' ? [0,0,1]

: splitLine[4] == 'Iris-versicolor' ? [0,1,0]

: [1,0,0];

trainingData.push({

input: input,

output: output

});

}

var learningRate = .01;

for (var i = 1; i <= 10000; i++)

{

for(var j = 0; j < trainingData.length; j++){

network.activate(trainingData[j].input);

network.propagate(learningRate, trainingData[j].output);

}

}

fs.readFile('testing\_data.txt','utf8',function(err, data){

if(err) throw err;

var lines = data.split("\n");

for(var i = 0; i < lines.length; i++){

var input = lines[i].trim().split(",");

var result = getFlowerName(network.activate(input));

console.log(lines[i].trim()+" => "+result);

}

});

});

function getLargestIndex(arr){

var result = 0;

for(var i = 1; i < arr.length; i++)

if(arr[i] > arr[result])

result = i;

return result;

}

function getFlowerName(arr){

var index = getLargestIndex(arr);

if(index == 0)

return "Iris-setosa";

if(index == 1)

return "Iris-versicolor";

return "Iris-virginica";

}

function getFlowerName2(index){

if(index < 51)

return "Iris-setosa";

if(index < 101)

return "Iris-versicolor";

return "Iris-virginica";

}