Wydział Informatyki

Metody sztucznej inteligencji

Laboratorium 03 IUz-22 Urbaniak

Sprawozdanie

Autor: Sergiusz Urbaniak
Grupa: IUz-22
Data: 7 stycznia 2010

Spis treści

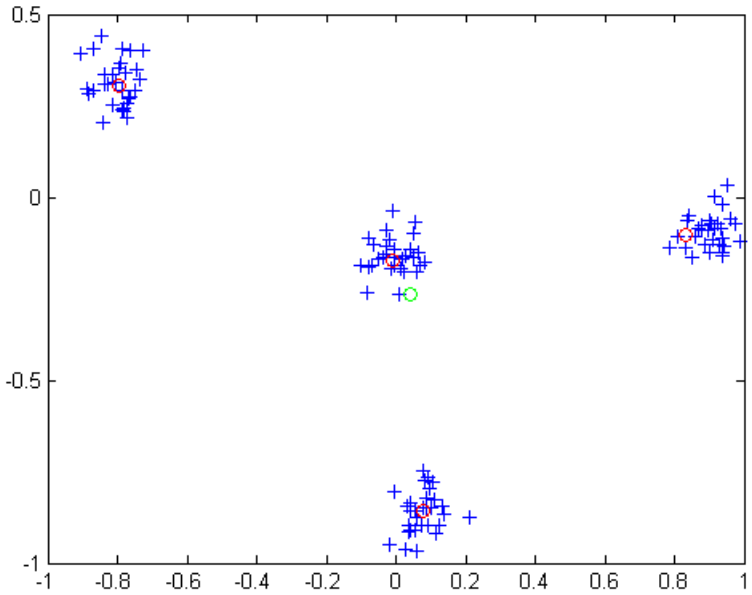
1 Badania plików

W następujących oddziałach są pokazane wyniki uczenia konkurencyjnego podanych plików. Najpierw w tabeli są przedstawione neurony i ich uczone wagi. Potem jest pokazany wynik uczenia.

1.1 `dane1.txt`

Neuron	Wagi
1	0.8566, -0.0946
2	-0.7966, 0.3138
3	-0.0100, -0.1565
4	0.0753, -0.8150

Tablica 1: Wagi uczonych neuronów pliku `dane1.txt`

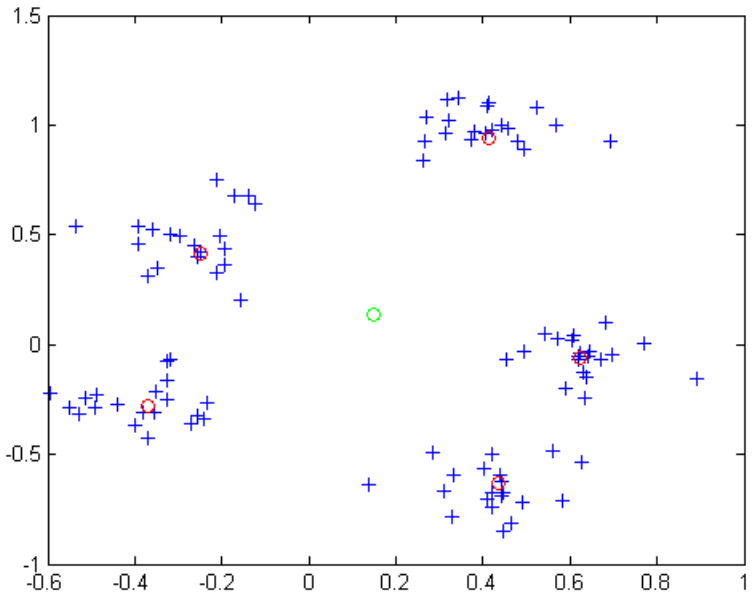


Rysunek 1: Wyniki uczenia

1.2 `dane2.txt`

Neuron	Wagi
1	0.4372, -0.6343
2	-0.3693, -0.2767
3	0.4129, 0.9439
4	-0.2477, 0.4151
5	0.6246, -0.0639

Tablica 2: Wagi uczonych neuronów pliku `dane2.txt`

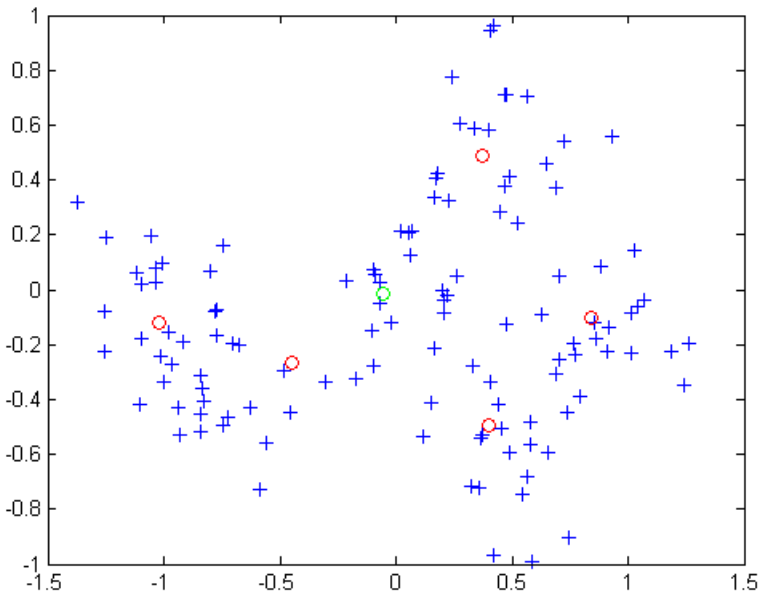


Rysunek 2: Wyniki uczenia

1.3 `dane3.txt`

Neuron	Wagi
1	0.3999, -0.4940
2	-0.4454, -0.2632
3	-1.0240, -0.1175
4	0.8445, -0.1047
5	0.3707, 0.4891

Tablica 3: Wagi uczonych neuronów pliku `dane3.txt`

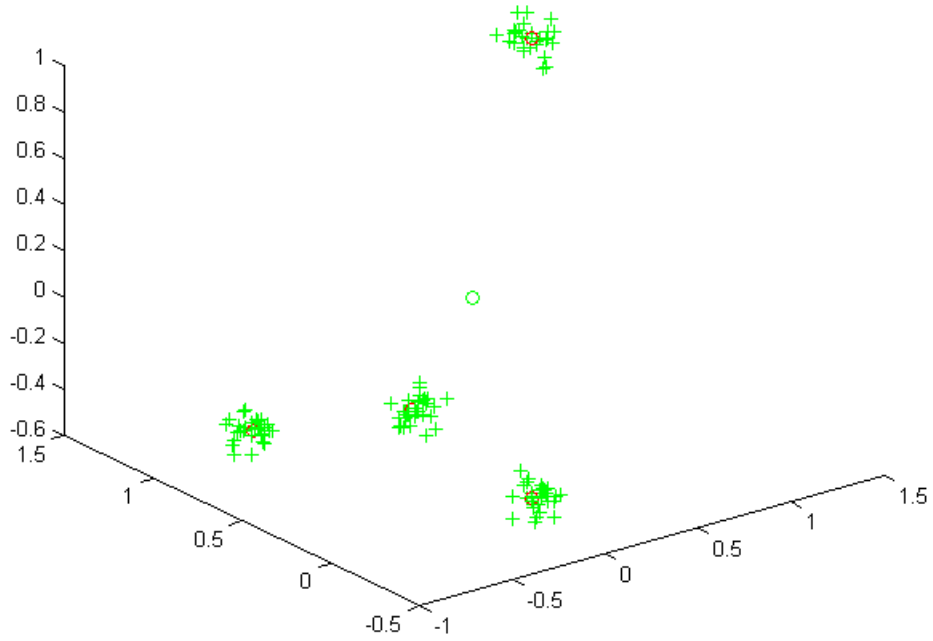


Rysunek 3: Wyniki uczenia

1.4 `dane3d1.txt`

Neuron	Wagi
1	-0.2860, -0.0676, 0.1099
2	0.1101, -0.0131, 0.2516
3	-0.1527, 0.1315, 0.1206
4	-0.0069, 0.1796, 0.2781

Tablica 4: Wagi uczonych neuronów pliku `dane3d1.txt`

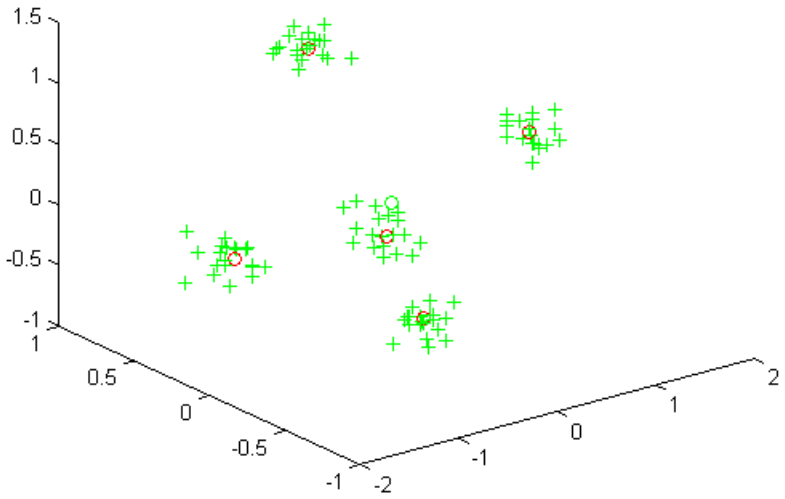


Rysunek 4: Wyniki uczenia

1.5 `dane3d2.txt`

Neuron	Wagi
1	0.2261, 0.7919, 0.9134
2	-0.7728, 0.6324, -0.5033
3	-0.8225, -0.4064, 0.2913
4	0.9081, -0.2220, 0.6549
5	-0.9577, -0.7378, -0.1668

Tablica 5: Wagi uczonych neuronów pliku `dane3d2.txt`

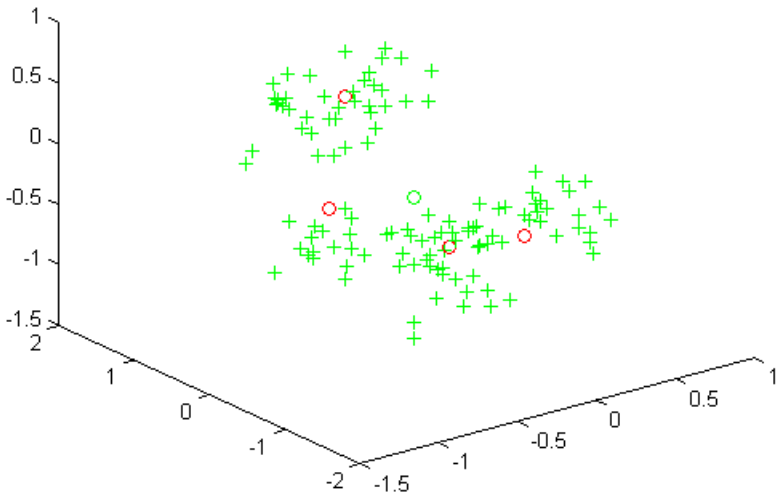


Rysunek 5: Wyniki uczenia

1.6 `dane3d3.txt`

Neuron	Wagi
1	0.3386, -0.3440, -0.7325
2	-0.2812, 0.7484, 0.3229
3	-0.3490, -0.7794, -0.4610
4	-0.7397, -0.0117, -0.2279

Tablica 6: Wagi uczonych neuronów pliku `dane3d3.txt`

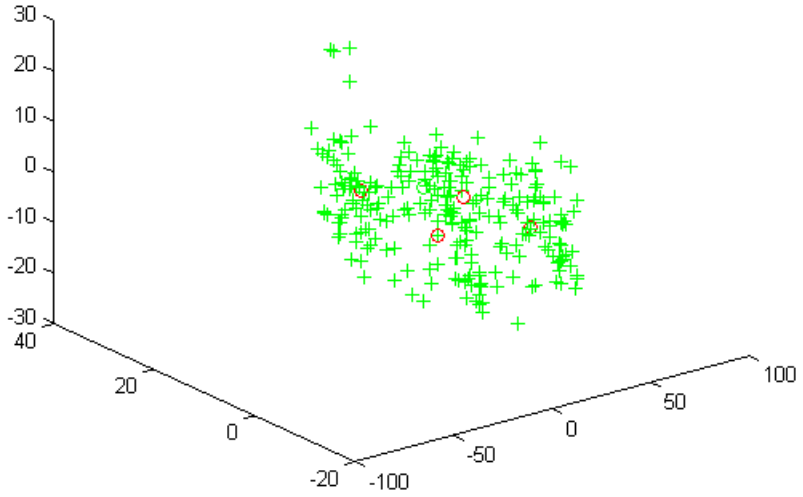


Rysunek 6: Wyniki uczenia

1.7 `kapitan_i.txt`

Neuron	Wagi
1	38.8117, -0.5899, -7.1230
2	-43.3287, 0.8705, 8.1441
3	6.4899, 0.2057, 2.1501
4	-14.0384, -2.8818, -1.9704

Tablica 7: Wagi uczonych neuronów pliku `kapitan_i.txt`



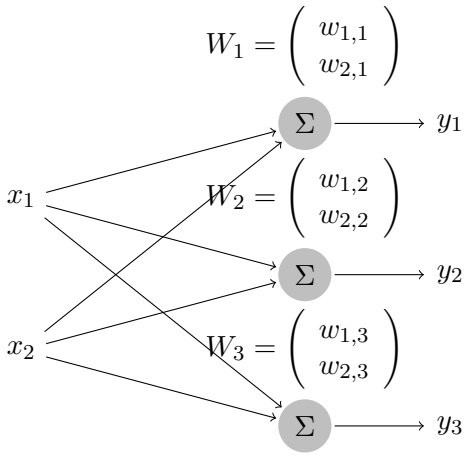
Rysunek 7: Wyniki uczenia

2 Własna funkcja uczenia konkurencyjnego

2.1 Wzory

Przy uczeniu konkurencyjnym neurony nie posiadają funkcji aktywacji. Wejścia neuronów bezpośrednio są przeliczane za pomocą wag według wzoru 1.

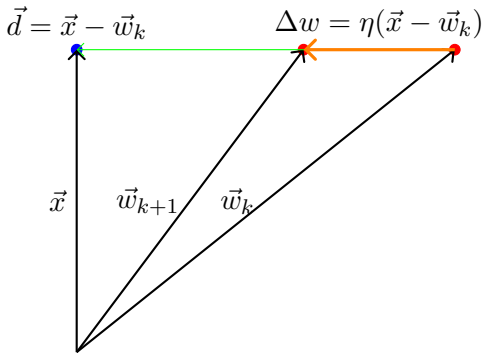
$$y_n = W_n^T X = \sum_{i=1}^{n} w_{i,n}\, x_i \tag{1}$$



Rysunek 8: Sieć neuronów konkurencyjnych

Wagi W_n danej iteracji są korygowane tylko dla tego neuronu, który najmocniej zostaje pobudzony. Pobudzenie jest mierzone odległością między neuronem a punktem wejściowym wektorem d według wzoru 2.

$$\begin{aligned} \vec{d} &= \vec{x} - \vec{w}_k \\ |\vec{d}| &= \sqrt{(w_1 - x_1)^2 + (w_2 - x_2)^2 + \ldots + (w_n - x_n)^2} \end{aligned} \quad \text{gdzie odległośc jest} \tag{2}$$



Rysunek 9: Wektorowa reprezentacja iteracji uczenia

Największe pobudzenie jest wtedy osiągane jeżeli znajdziemy neuron z najmniejszą odległością $|\vec{d}|_{min}$ dla danej iteracji. Wtedy korygowane są wagi W_{k+1} danego neuronu według wzoru 3.

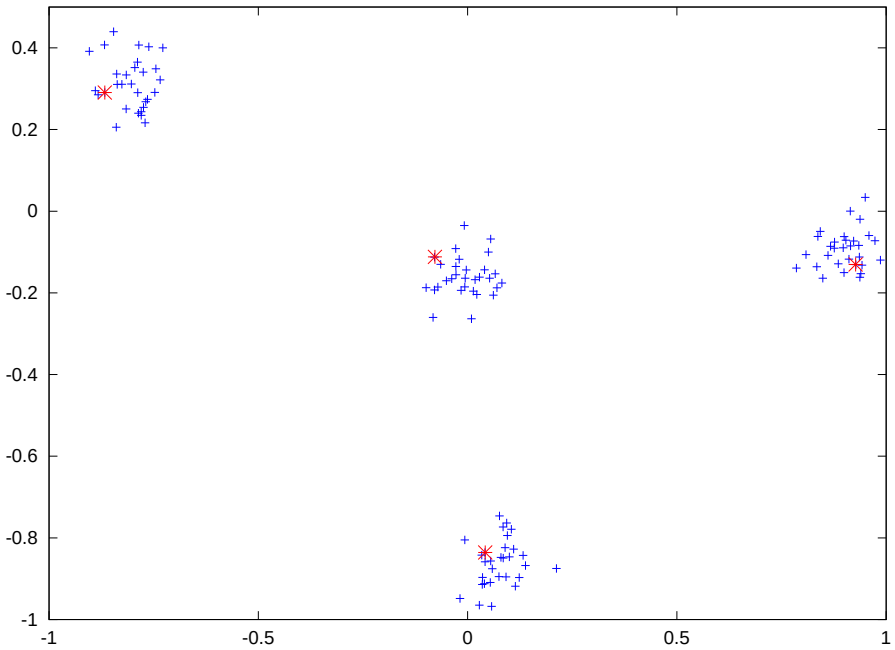
$$\begin{array}{rcll} W_{k+1} & = & \Delta w + W_k & \text{gdzie} \\ \Delta w & = & \eta(X - W_k) & \end{array} \tag{3}$$

W następnym rozdziale są przedstawiane wyniki badan z zaimplementowanym algorytmem. Kod algorytmu jest widoczny w listingu 1 w rozdziale 2.9. Tradycyjnie kod został napisany w Octave pod systemem Linux.

2.2 `dane1.txt`

Neuron	Wagi
1	0.042189, -0.835570
2	0.927179, -0.130506
3	-0.866712, 0.290581
4	-0.078092, -0.111721

Tablica 8: Wagi uczonych neuronów pliku `dane1.txt`

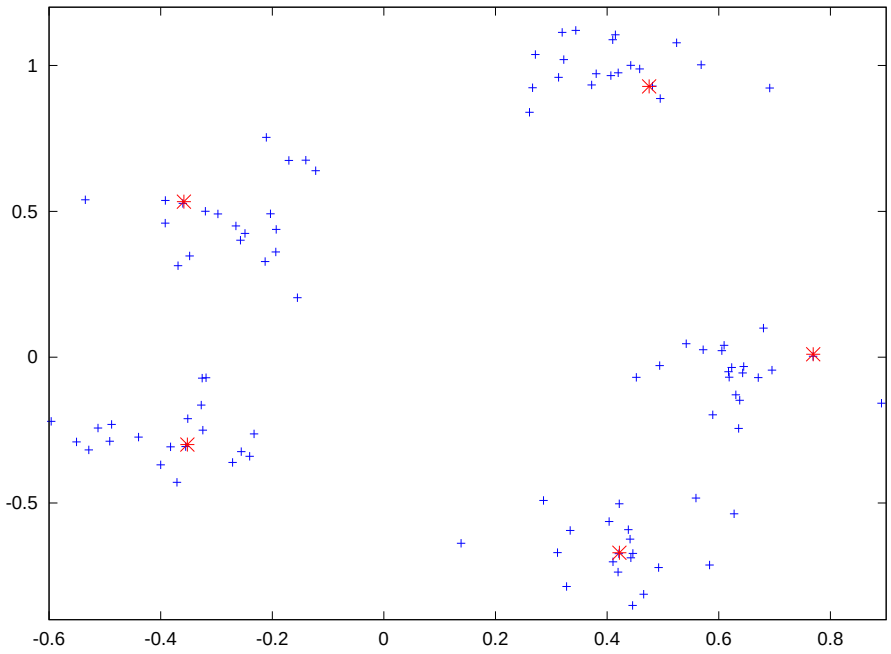


Rysunek 10: Wyniki uczenia

2.3 `dane2.txt`

Neuron	Wagi
1	0.4753834, 0.9282058
2	-0.3521456, -0.2996429
3	-0.3582966, 0.5332628
4	0.7693442, 0.0098722
5	0.4220855, -0.6704579

Tablica 9: Wagi uczonych neuronów pliku `dane2.txt`

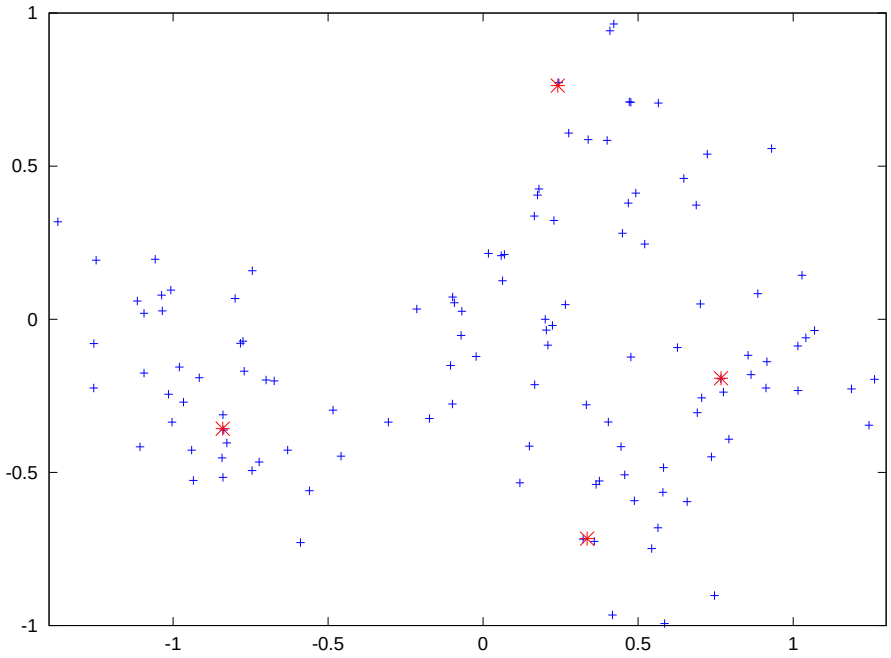


Rysunek 11: Wyniki uczenia

2.4 `dane3.txt`

Neuron	Wagi
1	0.33542, -0.71606
2	-0.84005, -0.35707
3	0.24094, 0.76322
4	0.76744, -0.19235

Tablica 10: Wagi uczonych neuronów pliku `dane3.txt`

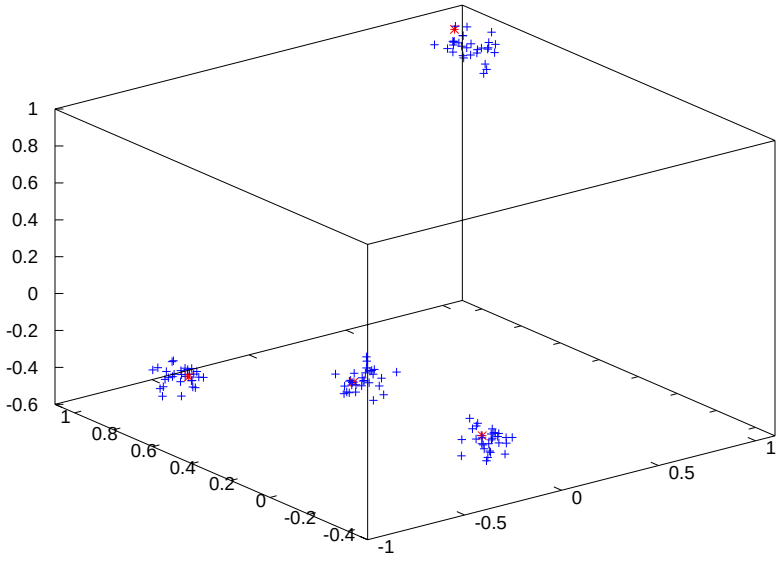


Rysunek 12: Wyniki uczenia

2.5 `dane3d1.txt`

Neuron	Wagi
1	-0.878974, -0.309702, 0.133947
2	-0.013616, -0.107567, -0.480594
3	0.932130, 0.973650, 0.971466
4	-0.723387, 0.691054, -0.337821

Tablica 11: Wagi uczonych neuronów pliku `dane3d1.txt`

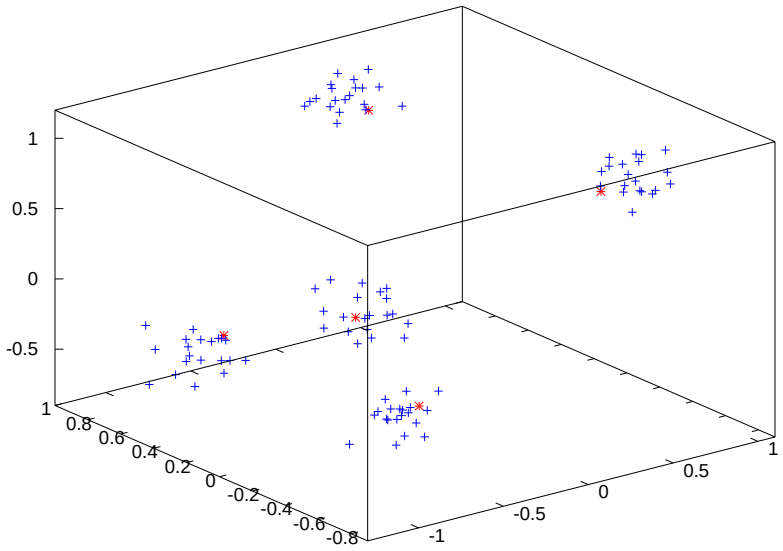


Rysunek 13: Wyniki uczenia

2.6 `dane3d2.txt`

Neuron	Wagi
1	0.80463, -0.14690, 0.55540
2	-0.90488, -0.80573, -0.11126
3	0.19803, 0.63894, 0.92120
4	-0.52862, 0.76973, -0.52255
5	0.36165, 0.88769, -0.72851

Tablica 12: Wagi uczonych neuronów pliku `dane3d2.txt`

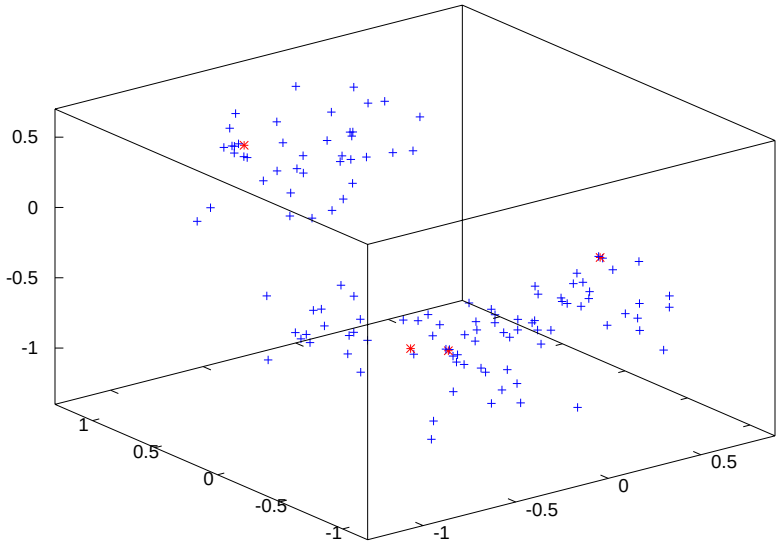


Rysunek 14: Wyniki uczenia

2.7 `dane3d3.txt`

Neuron	Wagi
1	-0.55551, 0.89028, 0.34978
2	-0.42394, -0.24565, -0.70310
3	-0.74934, -1.03103, -0.30708
4	0.61840, -0.21862, -0.41426

Tablica 13: Wagi uczonych neuronów pliku `dane3d3.txt`

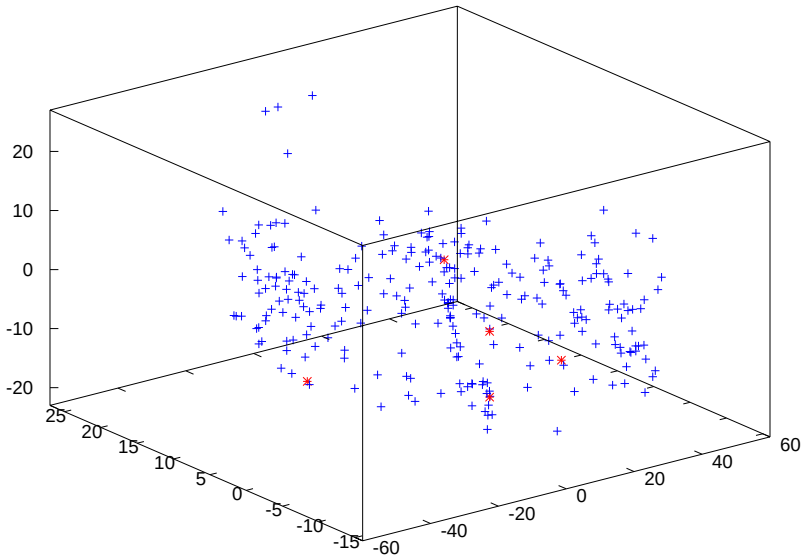


Rysunek 15: Wyniki uczenia

2.8 `kapitan_i.txt`

Neuron	Wagi
1	-28.56914, 6.30167, -12.52388
2	-5.16547, -7.86742, -11.05208
3	46.48602, 6.37457, -19.95036
4	34.82169, 10.80715, -15.72390
5	0.24909, 0.91634, 6.73213

Tablica 14: Wagi uczonych neuronów pliku `kapitan_i.txt`



Rysunek 16: Wyniki uczenia

2.9 Kod uczenia `nadzor.m`

```
function y = nadzor(in, clusters, epochs)
    eta = 0.1

    assert(epochs > 0);
    assert(clusters > 0);

    neurons = clusters;

    mx = min_max(in);
    range = (-mx(:,1) + mx(:,2))';
    for i=1:neurons
        initial_w = mx(:,1)' + (rand(1, size(in, 1)) .* range);
        % initial_w = zeros(1, size(in, 1));
        neuron(i).w = initial_w;
        neuron(i).y = initial_w';
    endfor

    for epoch=1:epochs
        for n=1:neurons
            for x=1:size(in,2)
                X = in(:,x)';

                % calculate distanc for each neuron
                for i=1:neurons
                    d = X - neuron(i).w;
                    d = d .^ 2;
                    d = sqrt(sum(d));
                    distance(i) = d;
                endfor

                % get neuron with smalles distance
                [distance, winner] = min(distance);

                % if the current neuron of interest (n) is the winner,
                % then correct its weights. if not, try another input
                if winner == n
                    delta_w = eta * (X - neuron(winner).w);
                    w_new = neuron(winner).w + delta_w;

                    neuron(winner).w = w_new;
                    neuron(winner).y = w_new';
                    break;
                endif
            endfor
        endfor
    endfor

    y = [ neuron.y ];
endfunction
```

Listing 1: Kod uczenia konkurencyjnego