Klaudia Pakos

IO, rok 3, 267561

PODSTAWY SZTUCZNEJ INTELIGENCJI

SPRAWOZDANIE NR 6

BUDOWA I DZIAŁANIE SIECI KOHONENA DLA WTM

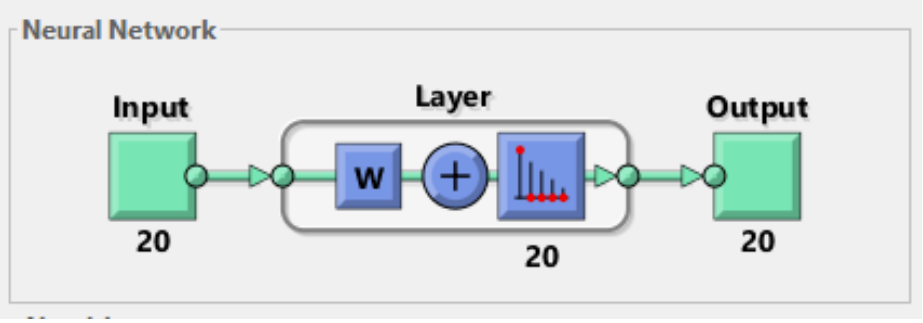
1. Cel ćwiczenia

Celem ćwiczenia jest poznanie budowy i działania sieci Kohonena przy wykorzystaniu reguły WTM do odwzorowywania istotnych cech liter alfabetu.

2. Zadania do wykonania

- Wygenerowanie danych uczących i testujących, zawierających 20 dużych liter dowolnie wybranego alfabetu w postaci dwuwymiarowej tablicy
- Przygotowanie (implementacja lub wykorzystanie gotowych narzędzi) siec Kohonena i algorytmu uczenia opartego o regułę Winner Takes Most (WTM).
- Uczenie sieci dla różnych współczynników uczenia.
- Testowanie sieci.

3. Opis budowy sieci



Działanie sieci Kohonena

Sieci Kohonena są szczególnym przypadkiem algorytmu realizującego uczenie się bez nadzoru. Ich głównym zadaniem jest organizacja wielowymiarowej informacji w taki sposób, żeby można ją było prezentować i analizować w przestrzeni o znacznie mniejszej liczbie wymiarów, czyli mapie.

Warunkiem jest to, że rzuty "podobnych" danych wejściowych powinny być bliskie również na mapie.

Zasady działania sieci Kohonena:

- Wejścia 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.

Mechanizm WTM (Winner Takes Most)

Jest to reguła aktywacji neuronów w sieci neuronowej, która jest oparta na zasadzie działania WTA z tą różnicą, że oprócz zwycięzcy wagi modyfikują również neurony z jego sąsiedztwa, przy czym im dalsza odległość od zwycięzcy, tym mniejsza jest zmiana wartości wag neuronu. Metoda WTA jest metodą słabo zbieżną - w szczególności dla dużej liczby neuronów.

Sąsiedztwo jest pojęciem umownym - można definiować sąsiadów bliższych i dalszych, sąsiedztwo nie oznacza również, że neurony muszą być bezpośrednio połączone ze zwycięzcą.

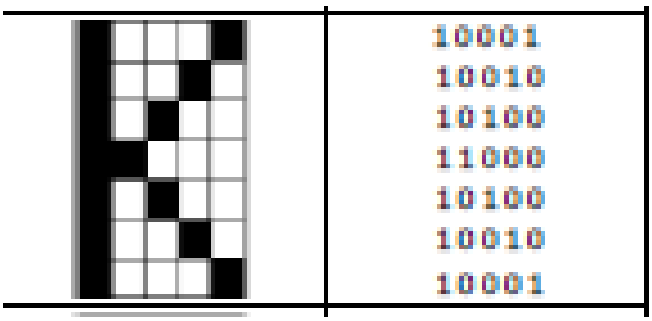
Podobnie jak w metodzie WTA, aby uniknąć tego, że jeden neuron będzie zawsze wygrywał stosuje się mechanizm zmęczenia, który w przypadku, gdy jeden neuron wygrywa zbyt często, to na pewien czas przestaje on być brany pod uwagę w rywalizacji.

Użyty algorytm uczenia sieci

- Generowanie losowo znormalizowanych wektorów wag.
- Losowanie wektora X oraz obliczanie dla niego aktywację Y dla wszystkich neuronów.
- Szukanie neuronu zwycięzcy.
- Modyfikacja wektora wag neuronu zwycięzcy, a następnie jego normalizacja (sprawdzenie warunków WTM).
- Zatrzymanie algorytmu po odpowiednio dużej ilości iteracji.

Dane wejściowe:

Umieszczanie liter w tablicy polegało na zinterpretowaniu danego pola przez sieć. Tablica składa się z 20 pól, które odpowiednio są czarne (1), lub białe (0). Czarne pole oznaczało, że w danym miejscu występował fragment litery, np. dla litery K wygląda to następująco:

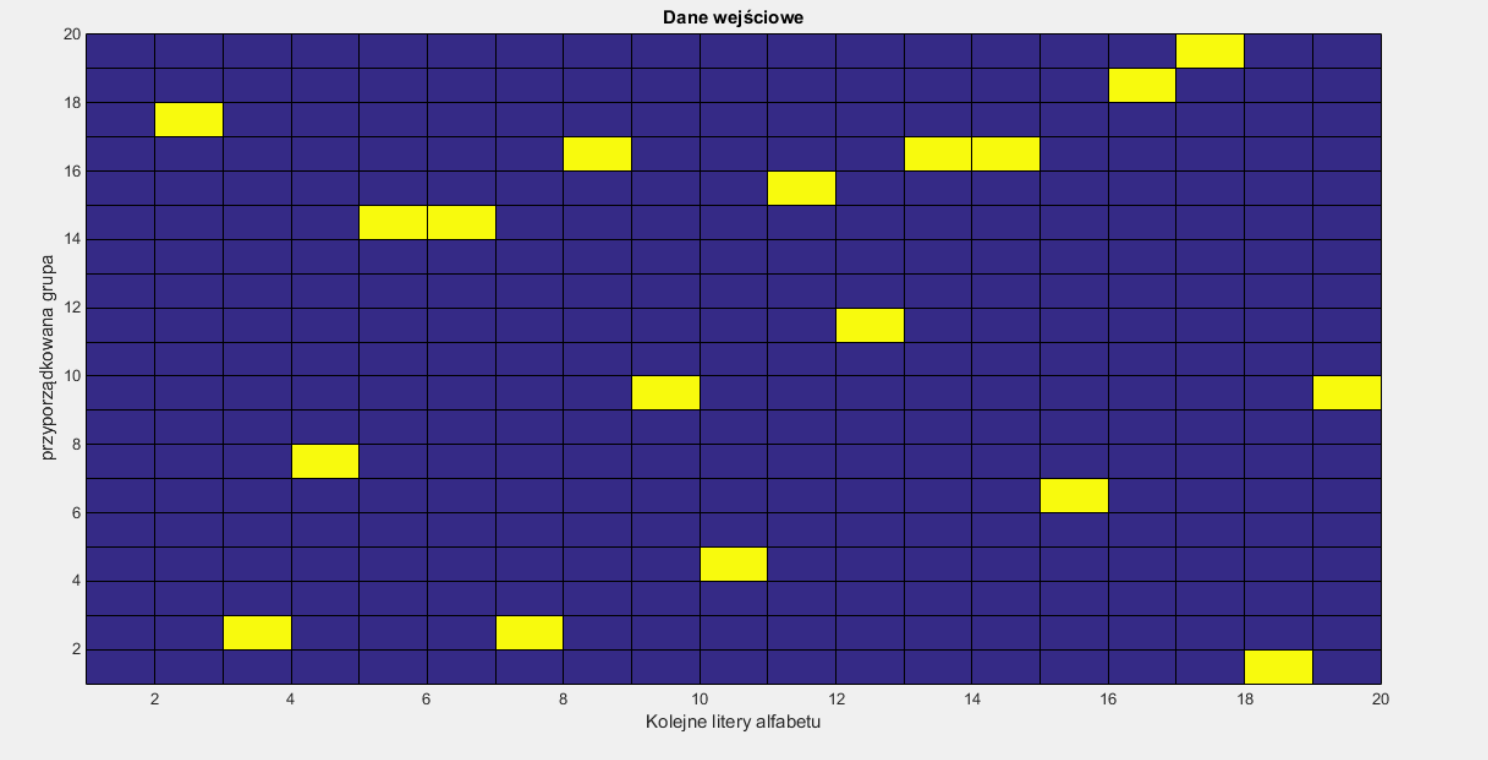


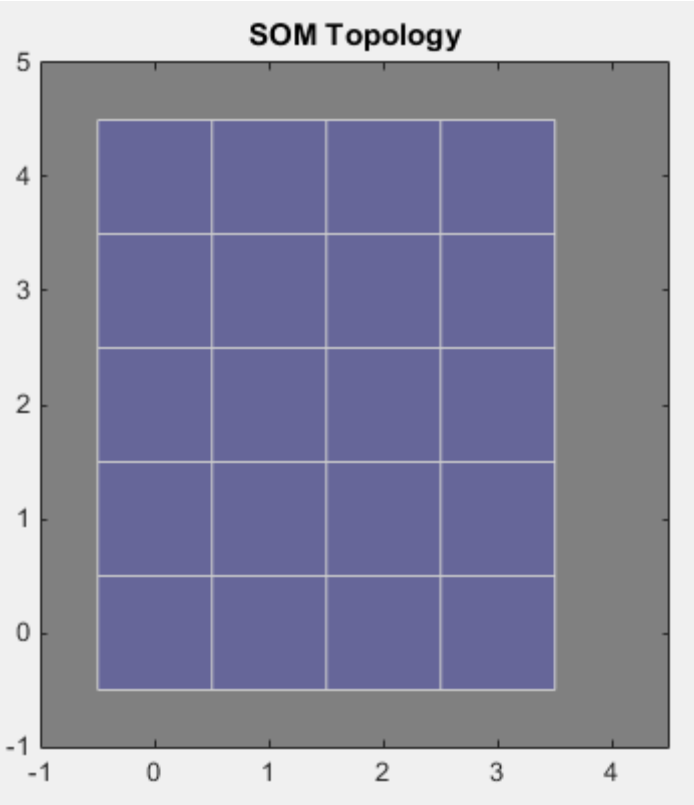
4. Przebieg zadania-listing kodu

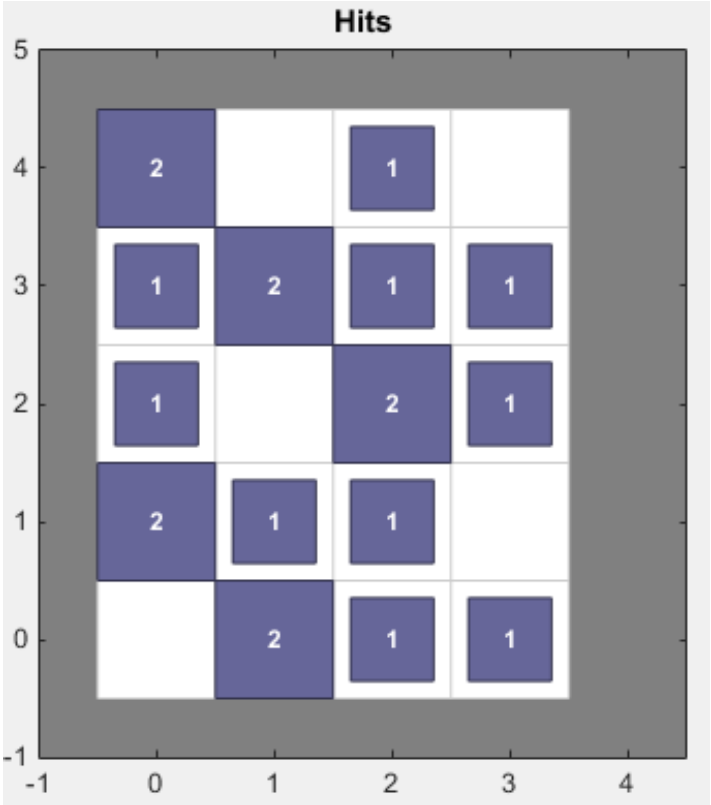
```
close all; clear all; clc;

%kolumnowa reprezentacja binarna pierwszych 20 duzych liter
alfabetu dla tablicy 4x5
%dane wejsciowe:
          %A B C D E F G H I J K L M N O P R S T U
WEJSCIE = [0 1 0 1 1 1 0 1 1 1 1 1 1 1 0 1 1 0 1 1;
           1 1 1 1 1 1 1 0 1 1 0 0 0 0 1 1 1 1 1 0;
           1 1 1 1 1 1 1 0 1 1 0 0 0 0 1 1 1 1 1 0;
           0 0 0 0 1 1 0 1 0 1 1 0 1 1 0 0 0 0 0 1;

           1 1 1 1 1 1 1 1 0 0 1 1 1 1 1 1 1 1 0 1;
           0 0 0 0 0 0 0 0 1 0 0 0 1 0 0 0 0 0 1 0;
           0 0 0 0 0 0 0 0 0 0 1 0 1 0 0 0 0 0 0 0;
           1 1 1 1 0 0 0 1 0 1 0 0 1 1 1 1 1 0 0 1;

           1 1 1 1 1 1 1 1 0 0 1 1 1 1 1 1 1 0 0 1;
           1 1 0 0 1 1 0 1 1 0 1 0 0 1 0 1 1 1 1 0;
           1 1 0 0 1 1 1 1 0 0 0 0 1 0 0 1 1 1 0 0;
           1 0 0 1 0 0 1 1 0 1 0 0 1 1 1 0 0 0 0 1;

           1 1 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 0 0 1;
           0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 1 0;
           0 0 0 0 0 0 0 0 0 0 0 1 0 0 1 0 0 1 0 0 0;
           1 1 1 1 0 0 1 1 0 1 0 0 1 1 1 0 0 1 0 1;

           1 1 0 1 1 1 0 1 1 0 1 1 1 1 0 1 1 0 0 0;
           0 1 1 1 1 0 1 0 1 1 0 1 0 0 1 0 0 1 1 1;
           0 1 1 1 1 0 1 0 1 1 0 1 0 0 1 0 0 1 0 1;
           1 0 0 0 1 0 0 1 0 1 1 1 1 1 0 0 1 0 0 0];
          %A B C D E F G H I J K L M N O P R S T U


% PARAMETRY SIECI KOHONENA
dimensions   = [4 5];            %wymiar wektora
coverSteps   = 100;              %liczba kroków szkoleniowych dla
początkowego pokrycia przestrzeni wejściowej
initNeighbor = 1;                %początkowy rozmiar sąsiedztwa
topologyFcn  = 'gridtop';        %funkcja topologii warstw
%topologyFcn  = 'hextop';        %funkcja topologii warstw

distanceFcn  = 'dist';           %funkcja odległości neuronowej

% TWORZENIE SIECI KOHONENA (SOM)
net = selforgmap(dimensions, coverSteps, initNeighbor, topologyFcn,
distanceFcn);
net.trainParam.epochs = 500;   %ustalenie maksymalnej liczby epok
treningowych utworzonej sieci

% TRENING SIECI
[net, tr] = train(net, WEJSCIE);%uczenie sieci
y = net(WEJSCIE);                %sprawdzenie działania sieci
figure, pcolor(y); title('Dane wejściowe'); %wyświetlenie mapy dla
danych początkowych
```

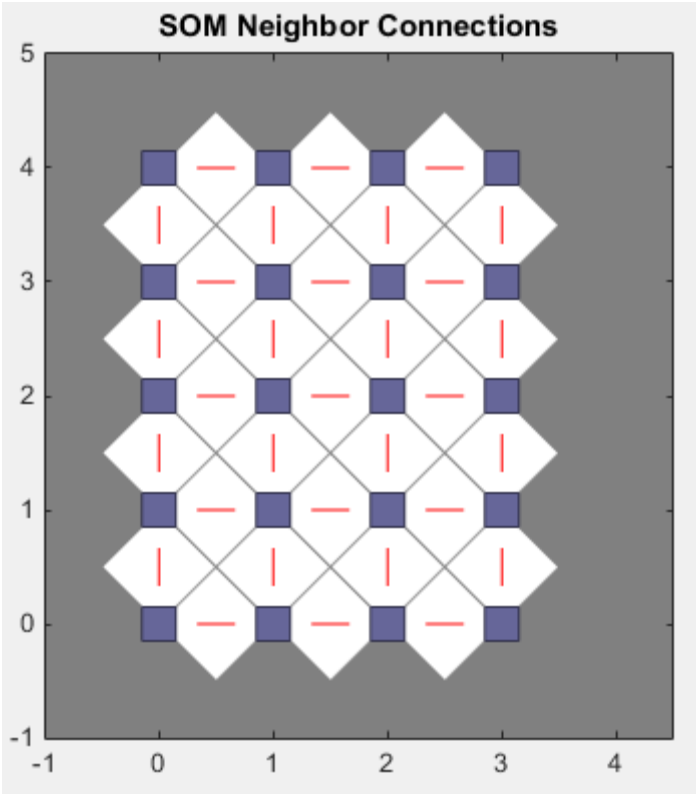
5. Przedstawienie wyników

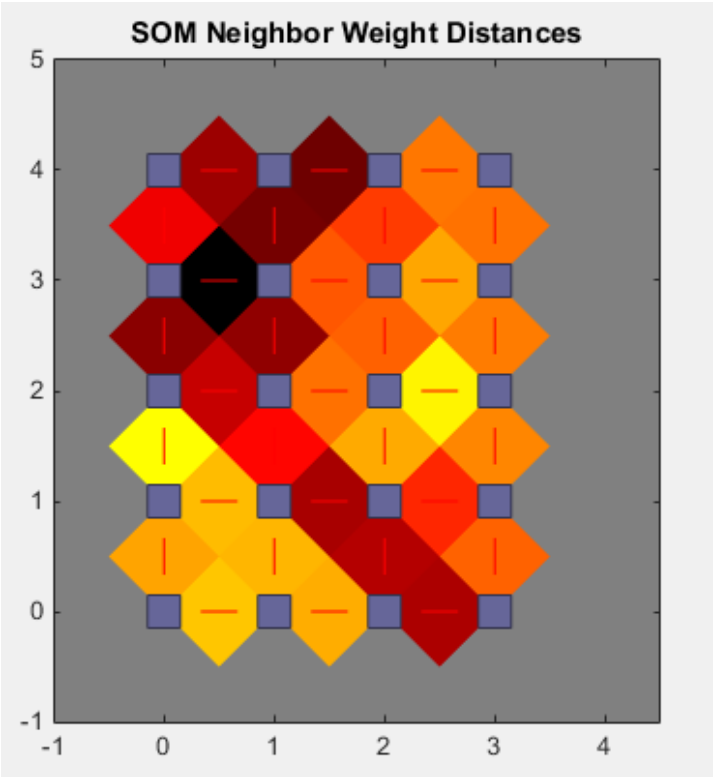
Sieć prostokątna

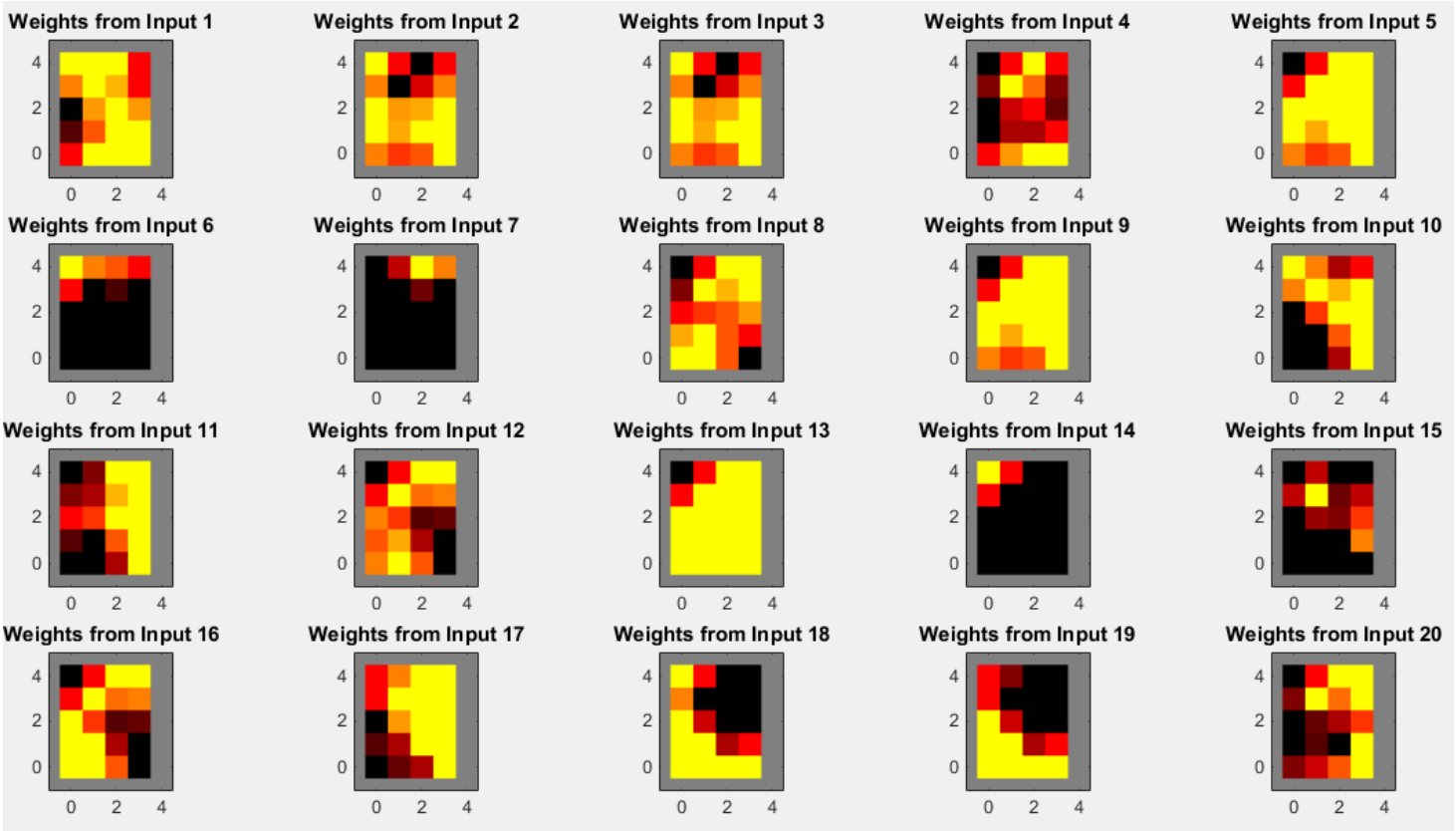


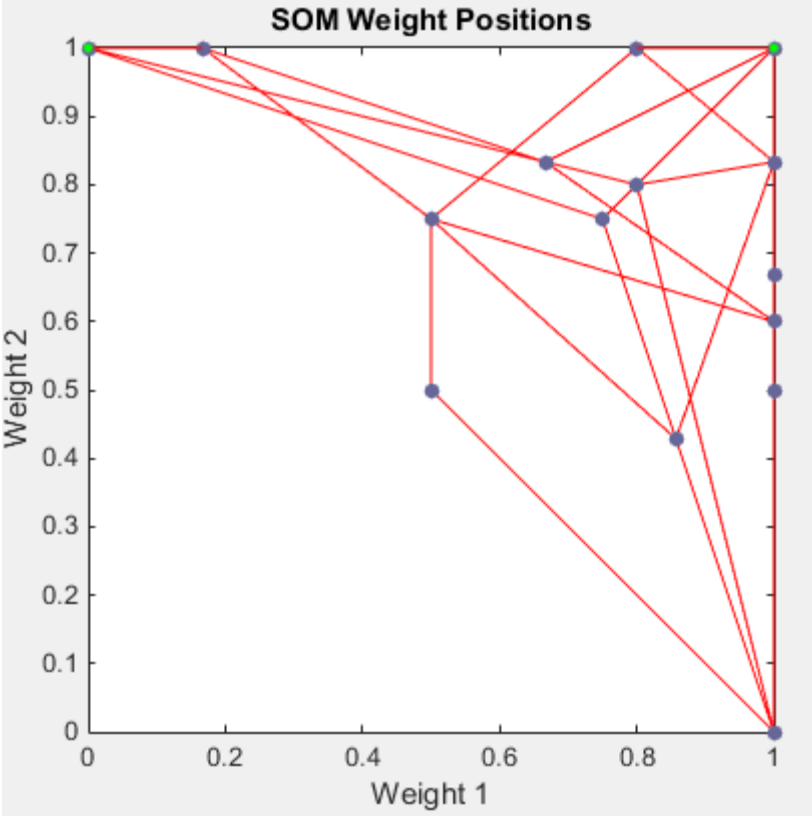




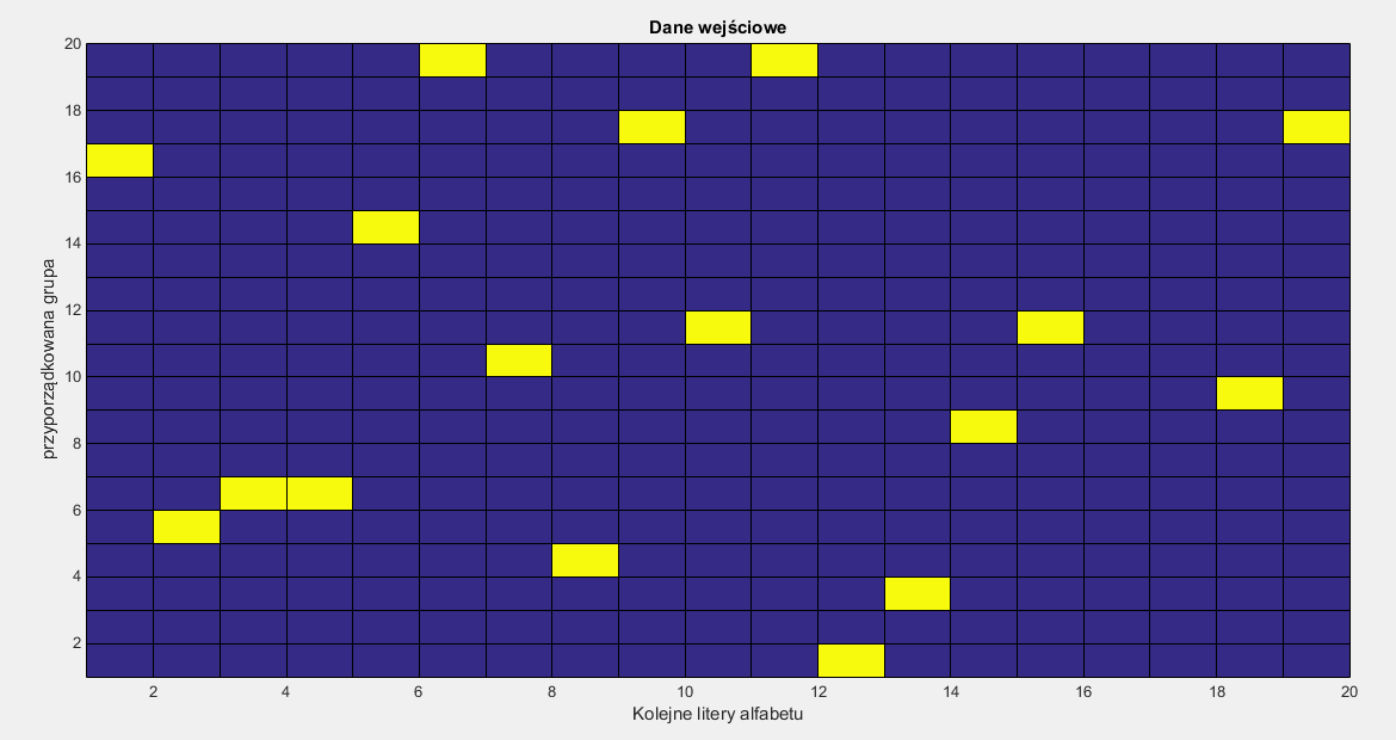

SOM Neighbor Connections

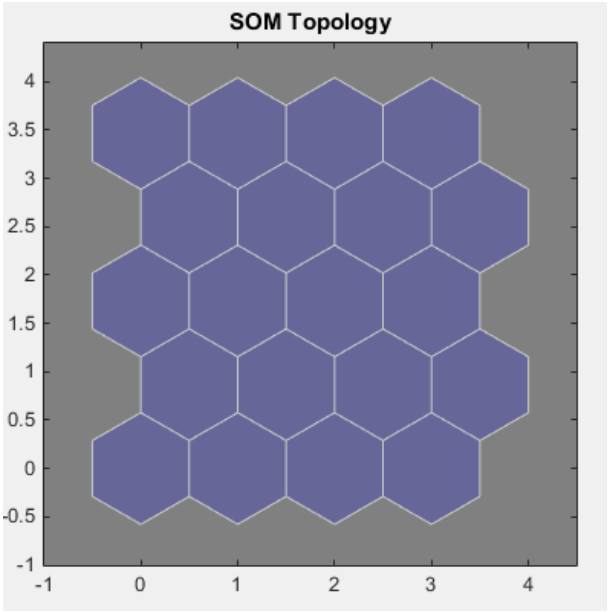

SOM Neighbor Weight Distances

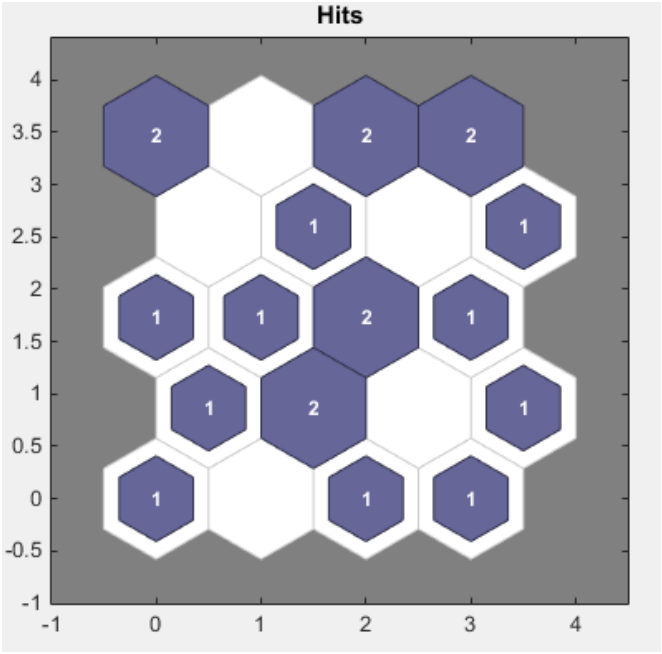

Weights from Input 1
Weights from Input 2
Weights from Input 3
Weights from Input 4
Weights from Input 5
Weights from Input 6
Weights from Input 7
Weights from Input 8
Weights from Input 9
Weights from Input 10
Weights from Input 11
Weights from Input 12
Weights from Input 13
Weights from Input 14
Weights from Input 15
Weights from Input 16
Weights from Input 17
Weights from Input 18
Weights from Input 19
Weights from Input 20

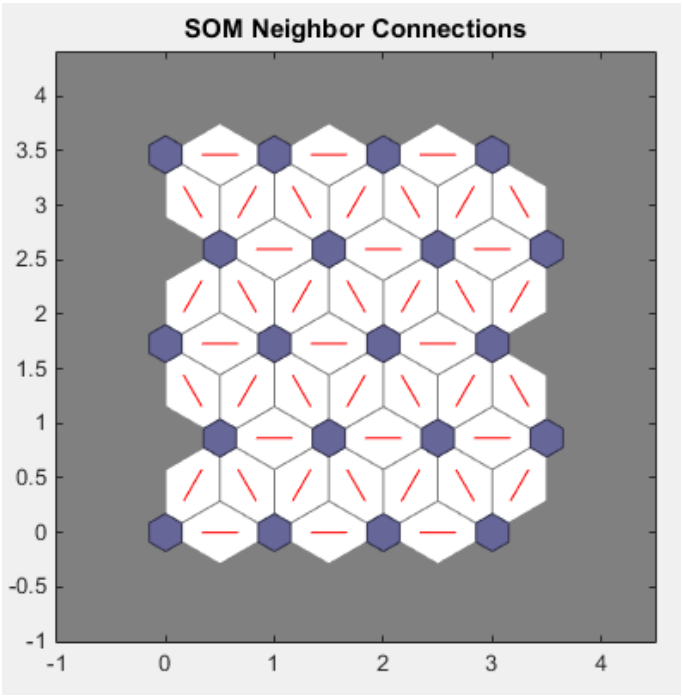


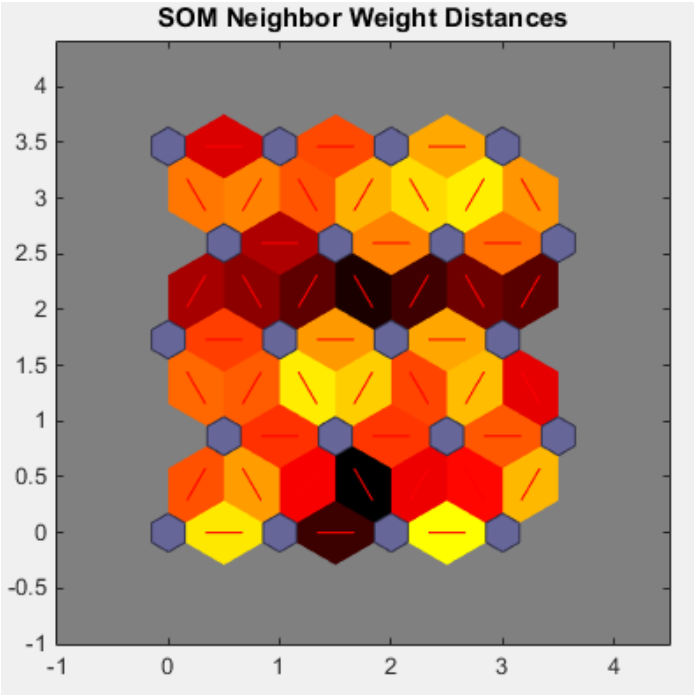
Sieć heksagonalna

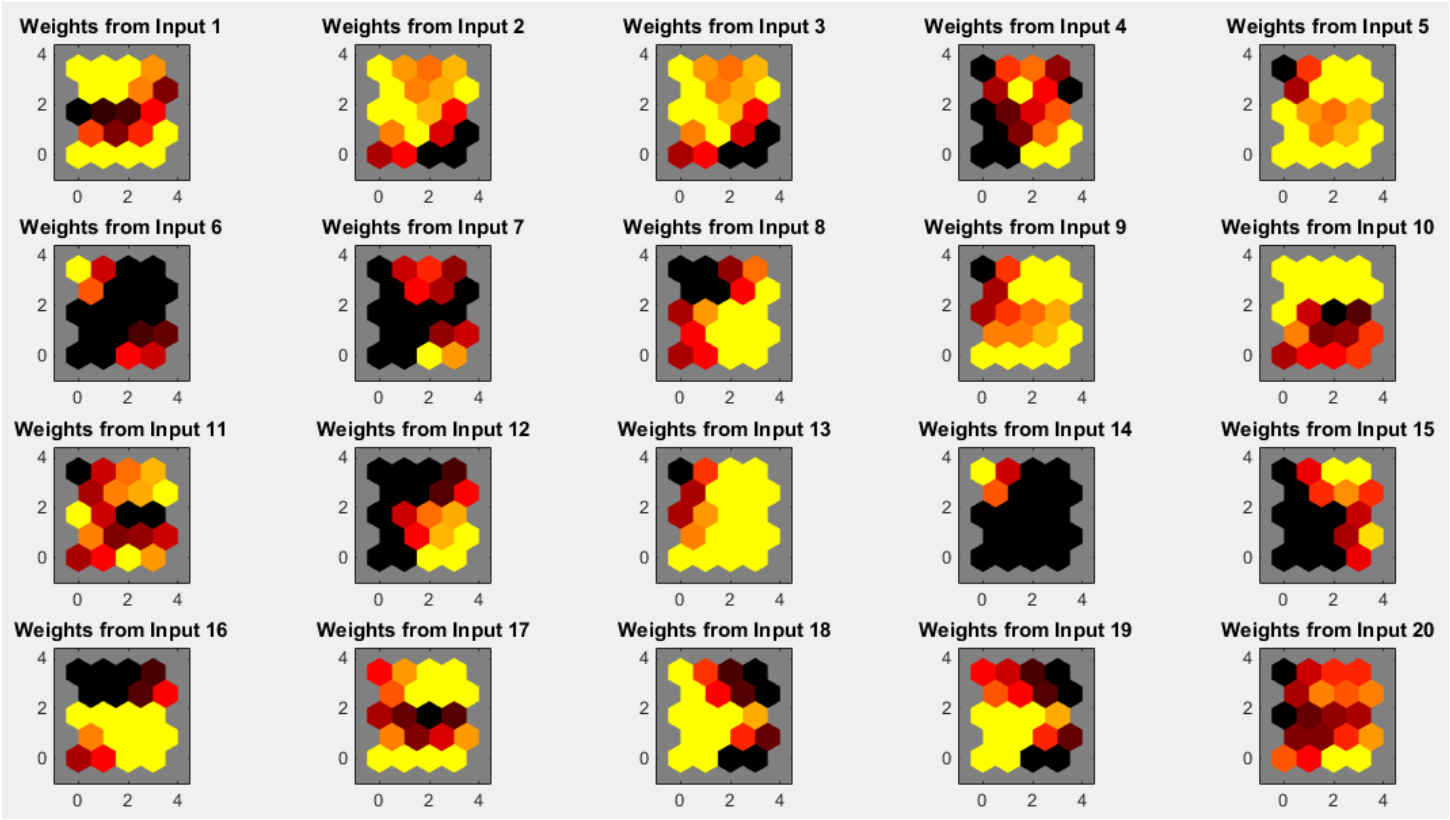


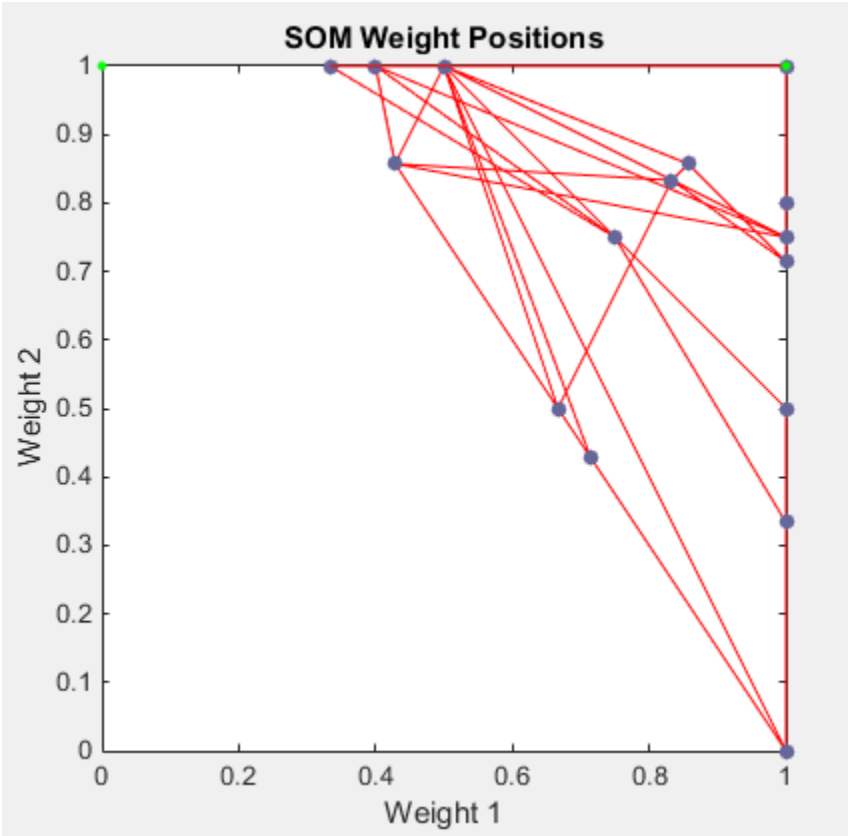

SOM Topology
4
3.5
3
2.5
2
1.5
1
0.5
0
-0.5
-1
-1
0
1
2
3
4


Hits
2
2
2
1
1
1
1
2
1
1
2
1
1
1
1
4
3.5
3
2.5
2
1.5
1
0.5
0
-0.5
-1
-1
0
1
2
3
4


SOM Neighbor Connections
4
3.5
3
2.5
2
1.5
1
0.5
0
-0.5
-1
-1
0
1
2
3
4


SOM Neighbor Weight Distances
4
3.5
3
2.5
2
1.5
1
0.5
0
-0.5
-1
-1
0
1
2
3
4


Weights from Input 1
Weights from Input 2
Weights from Input 3
Weights from Input 4
Weights from Input 5
Weights from Input 6
Weights from Input 7
Weights from Input 8
Weights from Input 9
Weights from Input 10
Weights from Input 11
Weights from Input 12
Weights from Input 13
Weights from Input 14
Weights from Input 15
Weights from Input 16
Weights from Input 17
Weights from Input 18
Weights from Input 19
Weights from Input 20


SOM Weight Positions
Weight 2
Weight 1
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
0.2
0.4
0.6
0.8

6. Wnioski

Prostokątna siatka neuronów umożliwia utworzonej sieci stworzenie bezpośrednich powiązań pomiędzy najbliższymi neuronami (sieć co prawda ma mniej możliwości w doborze odpowiednich wag, ale za to ogranicza sąsiedztwo).

Algorytm Winner Takes Most pokazywał prawie równomierne rozłożenie zwycięstw na całej wygenerowanej wcześniej sieci. W poprzednim scenariuszu metoda Winner Takes All koncentrowała wygrane neurony głównie w jednym miejscu sieci heksagonalnej. Rozkład zwycięstw metody WTM zwiększała poprawność działania sieci ze względu na równomierny rozkład zwycięstw.

Na podstawie wykresu pokazującego rozkład sił neuronów można zauważyć, że sieć korzystała z mechanizmu WTM (wykres pokazuje prawie równomierny rozkład zwycięstw neuronów - z pewnością otrzymane wyniki mniej koncentrują się na jednym punkcie sieci - jak w przypadku WTA).

Otrzymane wyniki dla prostokątnej siatki neuronów, na 20 testowanych liter przyporządkowała dobrze 15 liter. Do tych samych grup przyporządkowała (HMN), (EF), (IU),(CG). Natomiast sieć heksagonalna przyporządkowała prawidłowo 16 liter, a pomyliła pary (FK), (IU), (JO), (CD).