Patryk Kurzeja

Inżynieria Obliczeniowa

Nr albumu : 286112

Podstawy sztucznej inteligencji

Sprawozdanie : Budowa i działanie sieci Kohonena dla WTM [Scenariusz 6] :

1. Cel ćwiczenia :

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

2. Pojęcia i algorytmy :

- **Sieci Kohonena** - Sieci Kohonena są jednym z podstawowych typów sieci samoorganizujących się. Właśnie dzięki zdolności samoorganizacji otwierają się zupełnie nowe możliwości - adaptacja do wcześniej nieznanych danych wejściowych, o których bardzo niewiele wiadomo. Wydaje się to naturalnym sposobem uczenia, który jest używany chociażby w naszych mózgach, którym nikt nie definiuje żadnych wzorców, tylko muszą się one krystalizować w trakcie procesu uczenia, połączonego z normalnym funkcjonowaniem. Sieci Kohonena stanowią synonim całej grupy sieci, w których uczenie odbywa się metodą samoorganizującą typu konkurencyjnego. Polega ona na podawaniu na wejścia sieci sygnałów, a następnie wybraniu w drodze konkurencji zwycięskiego neuronu, który najlepiej odpowiada wektorowi wejściowemu.

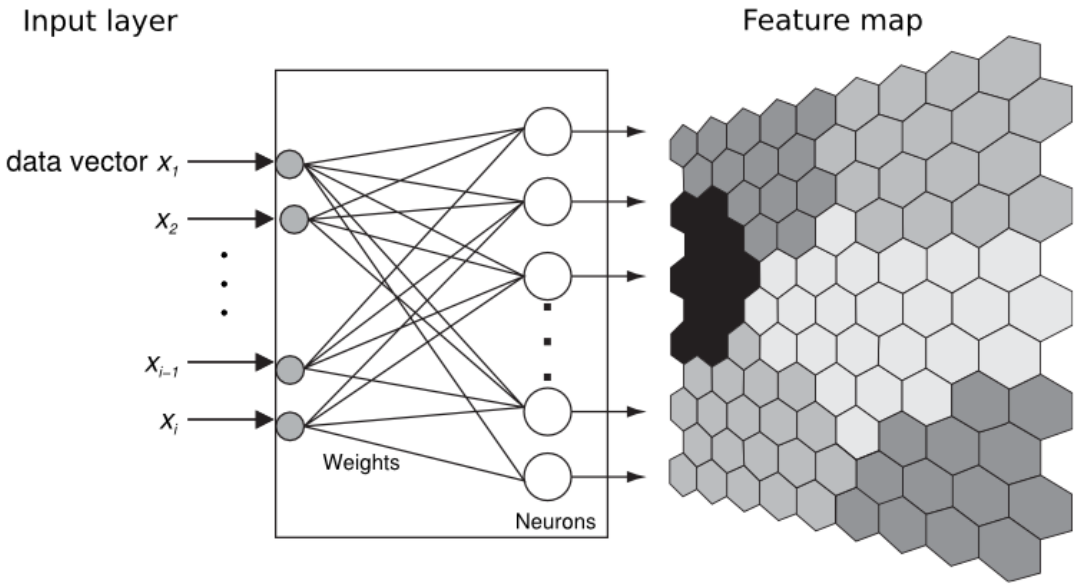
  Funkcjonowanie samoorganizujących się sieci neuronowych odbywa się w trzech etapach:

  - konstrukcja
  - uczenie
  - rozpoznawanie

  Potrzebna nam jest zatem macierz neuronów pobudzanych przez sygnały wejściowe. Sygnały te powinny opisywać pewne charakterystyczne cechy zjawisk zachodzących w otoczeniu, tak, aby na ich postawie sieć była w stanie je pogrupować. Zbiór sygnałów przekazywanych do każdego neuronu nie musi być identyczny.
  Muszą one tylko spełniać jeden warunek, a mianowicie jednoznacznie określać dane zdarzenia.

  Uczenie jest mechanizmem który dla każdego neuronu określa stopień podobieństwa jego wag do danego sygnału wejściowego oraz wyznacza jednostkę z największym dopasowaniem – czyli zwycięzcę.

Wtedy sieć powinna zaadaptować wartości wag neuronu zwycięzcy i jego sąsiadów w zależności od siły, z jaką odpowiedział on na dane wejście. SOM jest siecią uczącą się bez nadzoru. Oznacza to, że w trakcie uczenia opierają się wyłącznie na obserwacji danych wejściowych, nikt im nie mówi, co z tych danych wejściowych powinno wynikać, sama ma to ustalić.



Dane wejściowe trafiają do neuronów a następnie są odwzorowywane na warstwę topologiczną, co daje siatkę neuronów z efektem działania sieci. Neurony mogą być ułożone w siatkę heksagonalną lub prostokątną.

- **Reguła Winner Takes Most (WTM):**
  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.

  Algorytm WTM :
  1. Generujemy losowo znormalizowane wektory wag
  2. Losujemy wektor x oraz liczymy dla niego aktywację y dla wszystkich neuronów
  3. Szukamy neuronu zwycięzcy

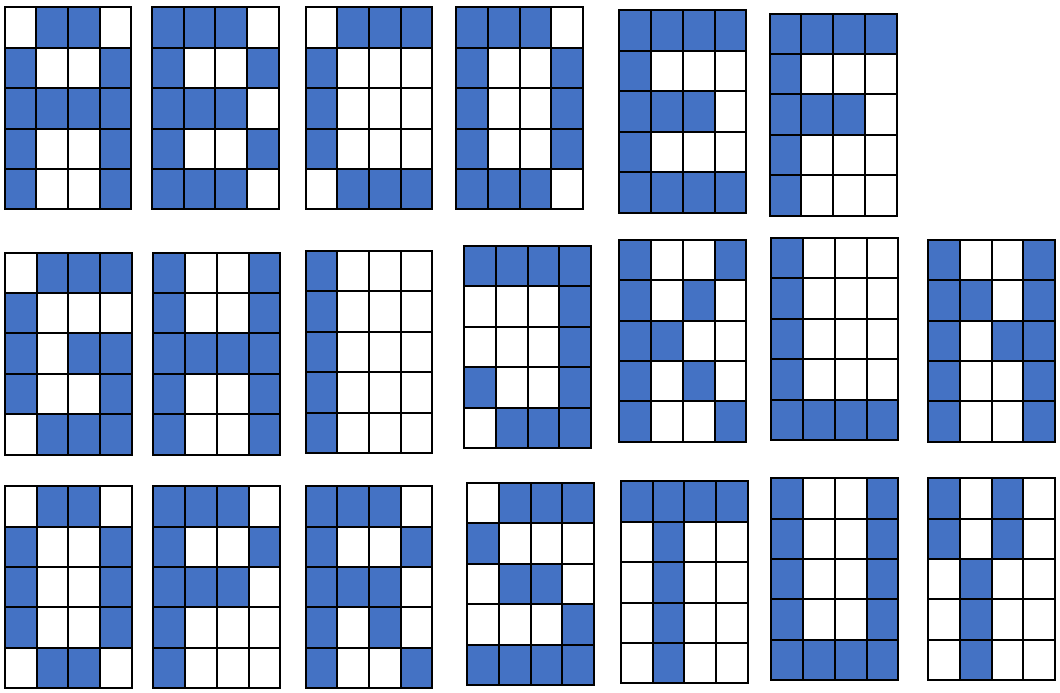
4. Modyfikujemy wektory wag zwycięzcy oraz sąsiedztwa a następnie normalizujemy zwycięzcę (sprawdzamy czy nie wygrywa zbyt często, jeśli tak to jest na chwilę usypiany)
5. Zatrzymujemy algorytm po odpowiednio dużej ilości iteracji

3. Wykonanie :

W programie wykorzystałem prostokątną siatkę neuronów, uczenie wg reguły Kohonena i metodę WTM. Program odwzorowuje istotne cechy liter alfabetu na podstawie otrzymanych danych. Stworzyłem sieć o wymiarze 4x5, czyli odpowiadającym rozmiarowi liter. Sąsiedztwo wynosi 1 a więc sąsiadami są tylko i wyłącznie neurony graniczące bezpośrednio ze zwycięzcą.

Dane wejściowe :

20 dużych liter alfabetu : A, B, C, D, E, F, G, H, I, J, K, L, N, O, P, R, S, T, U, Y zapisanych w postaci tablic 4x5 w postaci binarnej :



Procedura przekształcania liter do postaci binarnej :

- Wybieramy literę : np. A
- Tabelę 4x5 wypełniam 0 i 1
- A = [0; 1; 1; 0; 1; 0; 0; 1; 1; 1; 1; 1; 1; 0; 0; 1; 1; 0; 0; 1];
- Każdą literę zapisuję w postaci 20 znakowego ciągu

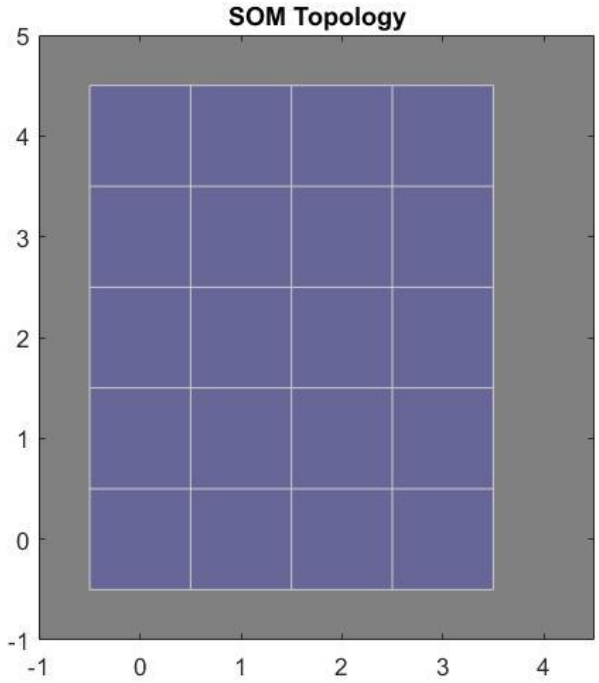
Listing kodu programu :

```
close all; clear all; clc;

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

% Parametry sieci Kohonena :
dimensions = [4 5];
coverSteps = 100;
initNeighbor = 1;
topologyFcn = 'gridtop';
distanceFcn = 'dist';

% Tworzenie SOM :
net = selforgmap(dimensions,coverSteps,initNeighbor,topologyFcn,distanceFcn);
net.trainParam.epochs = 2000;

% Trenowanie sieci :
[net,tr] = train(net,INPUT);
y = net(INPUT);
classes = vec2ind(y);
```

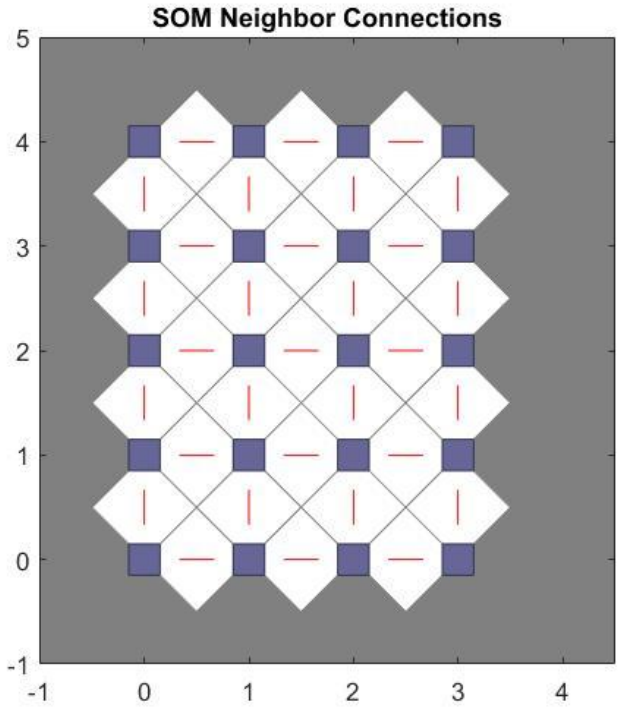
Opis zmiennych oraz funkcji :

- net.trainParam.epochs – maksymalna liczba epok podczas treningu sieci
- Selforgmap – tworzy mapę neuronów za pomocą algorytmu Kohonena z odpowiednimi parametrami :
    - Dimensions – wymiar sieci (4x5)
    - InitNeighbor – ilość neutronów tworzących sąsiedztwo (1)
    - TopologyFcn – funkcja topologii :
        - Gridtop – przy użyciu prostokątów
        - Hextop – przy użyciu sześciokątów
        - Randtop – neutrony rozrzucane losowo
    - DistanceFcn – oblicza dystans pomiędzy neuronami :
        - Dist – odległość Euklidesowa
        - Linkdist – kilka możliwych sposobów obliczenia, wybór zależny od ilości wprowadzonych wektorów.
- Vec2ind – konwertowanie wektorów uczonej sieci na indeksy
- coverSteps – liczba kroków szkoleniowych dla początkowego pokrycia przestrzeni wejściowej (100)

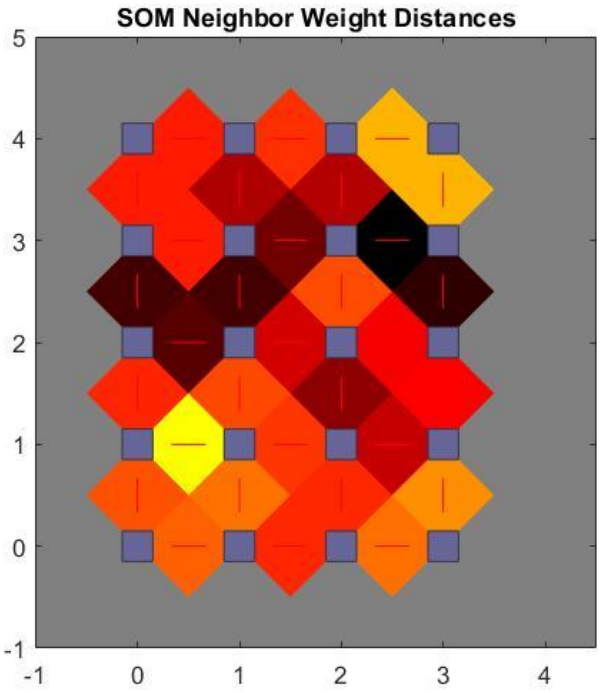
4. Wyniki działania :



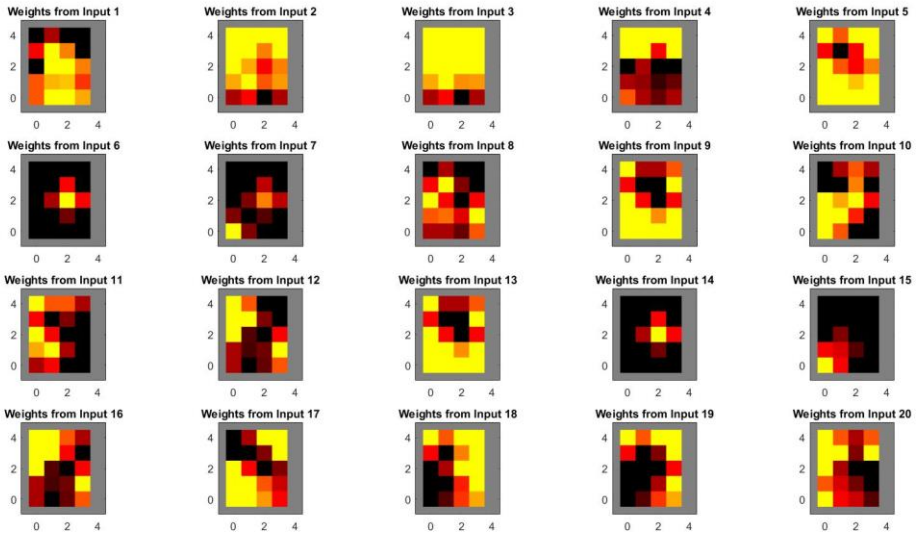
Topologia sieci Kohonena (siatka neuronów 4x5)



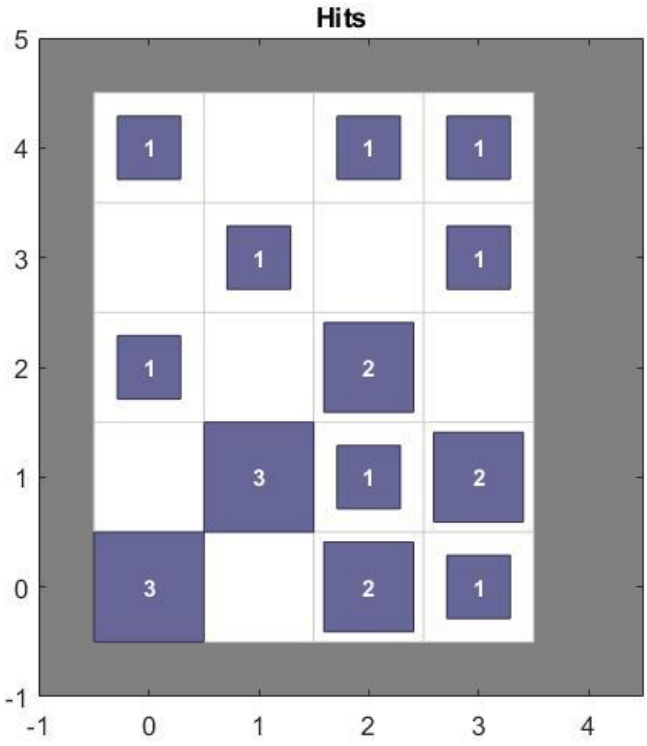
Połączenia pomiędzy poszczególnymi neuronami
(sąsiedztwo = 1)



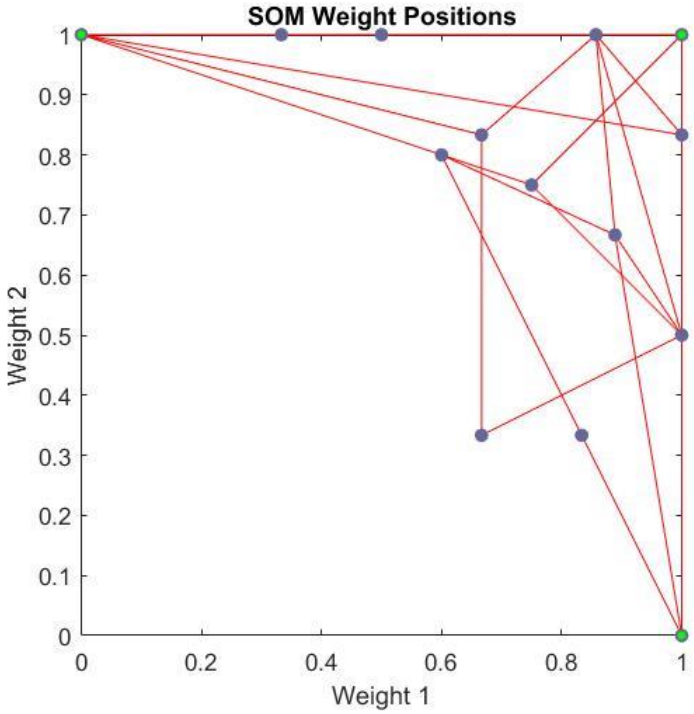
Dystans pomiędzy wagami – ciemniejszy kolor oznacza większy dystans



Rozkład wag dla każdego z wejść, ciemniejszy kolor oznacza wyższe wagi



Ilość zwycięstw poszczególnych neuronów podczas rywalizacji
w ramach Winner Takes Most



Efekt końcowy :
Niebieskie punkty - neurony
Czerwone linie - połączenia

-	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	0	0	0	0	0	0	0	1	0	0	1	0	1	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	1	0	0	0	0
7	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	0	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
9	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1
12	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
13	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
14	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
16	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
17	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
18	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
19	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
20	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Tabela przedstawiająca tablicę wyjścia dla dwudziestu liter. 1 oznacza trafienie czyli najbardziej typową kratkę dla danej litery. Przykładowo dla A zauważamy, że jest to kratka 9.

4. Wnioski :

Zauważamy, że metoda Winner Takes Most (WTM) skutkowała prawie równomiernym rozłożeniem zwycięstw na całej wygenerowanej sieci.
W przeciwieństwie do Winner Takes All gdzie neurony wygrywające były zebrane mniej więcej w jednej okolicy WTM rozłożył zwycięzców nieco bardziej „sprawiedliwie”, co w efekcie mogło sprzyjać poprawności działania sieci gdyż nie skupiała się ona na jednym, konkretnym swoim fragmencie tylko brała podczas działania pod uwagę całą sieć.

- ➔ WTM nie pozwala więc na dominację małej liczby neuronów w sieci.
- ➔ Współczynnik uczenia sieci kontroluje przydział wag dla neuronów.
- ➔ Sąsiedztwo ma istotny wpływ na efekt działania sieci – zbyt duże w małej sieci będzie skutkowało niepoprawnym rozłożeniem wag.
- ➔ Wagi poszczególnych neuronów są rozłożone w zależności od ilości neuronów w sieci.
- ➔ Zwiększanie liczby neuronów wpływało na czas obliczeń powodując jego wydłużenie.

Zauważamy również, że zwiększanie liczby sąsiedztwa powodowało błędy w działaniu sieci neuronowej.

- ➔ Wielkość sąsiedztwa jest uzależniona od wielkości sieci - jeśli oba te parametry rosną jednocześnie wtedy działanie programu jest poprawne.

Na podstawie wykresu pokazującego rozkład sił neuronów zauważamy, że sieć korzystała z algorytmu WTM (wykres pokazuje prawie równomierny rozkład zwycięstw)
Pomimo treningu bez nadzoru sieć Kohonena z algorytmem WMT w sposób prawidłowy dokonała odwzorowania cech typowych dla wybranej litery przy niewielkiej liczbie ustawionych iteracji treningu.
Bardzo ważnym elementem przy tworzeniu sieci tego typu jest prawidłowy dobór liczby neuronów - dla małej liczby wzrasta ryzyko wystąpienia błędu, natomiast zbyt duża liczba neuronów znacznie wydłuża czas potrzebny na naukę.
Dostrzegamy że zastosowanie algorytmu WTM daje zauważalnie lepsze rezultaty od algorytmu WTA w przypadku użycia sieci z dużą liczbą neuronów.