Sieci neuronowe

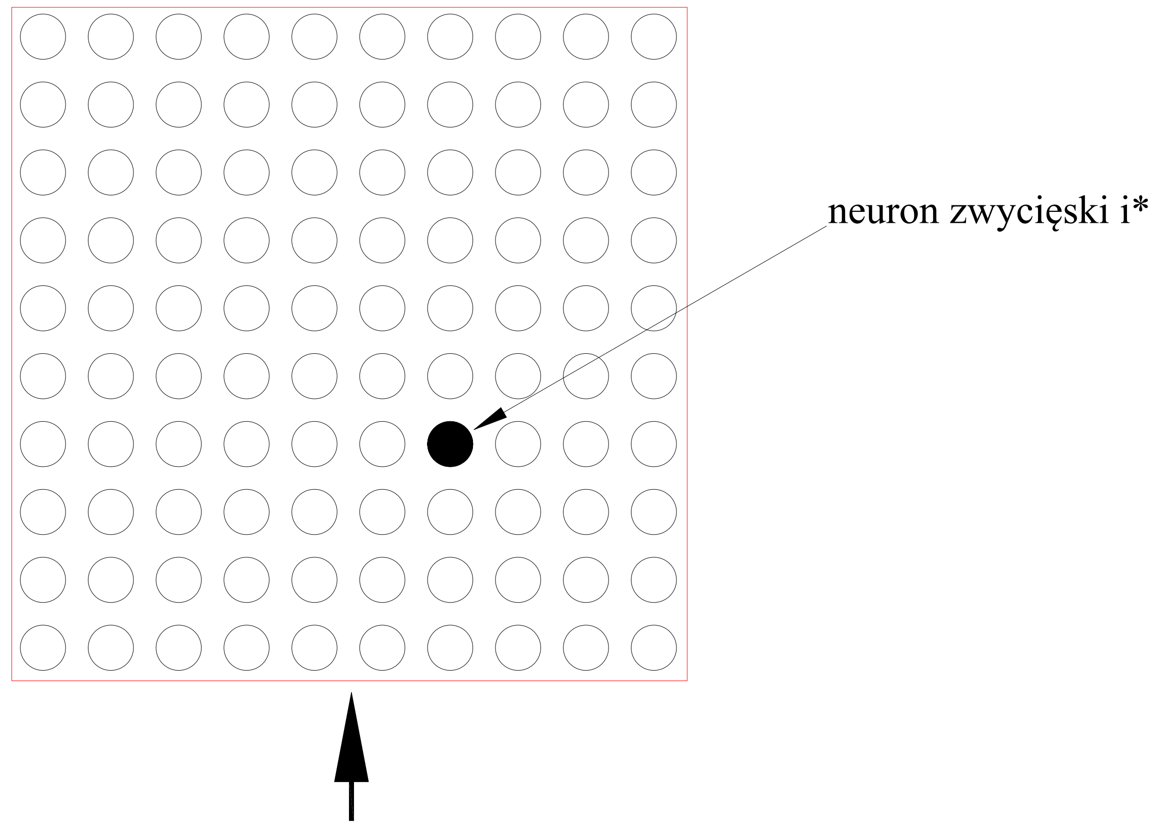
Sieć Kohonena

dr inż. Tomasz Piłot

Literatura

1. Tadeusiewicz R.: **Sieci neuronowe**. Akademicka Oficyna Wydawnicza, Warszawa 1993
2. Osowski S.: **Sieci neuronowe w ujęciu algorytmicznym**. WNT, Warszawa 1996
3. Hertz J., Krogh A., Palmer R. G.: **Wstęp do teorii obliczeń neuronowych**. WNT, Warszawa 1993
4. Masters T.: **Sieci neuronowe w praktyce. Programowanie w C++**. WNT, Warszawa 1996
5. **Zastosowania sztucznej inteligencji w inżynierii produkcji**. Pod Red. R. Knosali, WNT 2002
6. Larose D.T.: Odkrywanie wiedzy z danych. PWN, 2006

Schemat wyjściowej warstwy sieci o algorytmie uczenia o konkurencji ostrej



[wektor cech wejściowych x_j]

Zastosowania

- Kodowanie i kompresja danych
- Aproksymacja funkcji
- Przetwarzanie obrazów
- Inicjalizacja parametrów uczenia innych sieci
- Analiza statystyczna
- …

Efekty działania sieci o algorytmie WTA

- Wyznaczanie kategorii (grupy) na podstawie korelacji w danych wejściowych (zbiorze uczącym),
- Związanie kategorii z pojedynczym neuronem zwycięskim (kwantowanie wektorowe) – wektor wejściowy (przykład uczący) jest reprezentowany przez indeks neuronu zwycięskiego,

Wady działania sieci o algorytmie WTA

- Jedna komórka reprezentuje jedną kategorię, czyli liczba rozpoznawanych kategorii wynosi N (N – liczba neuronów),
- Nie jest odporna na uszkodzenia – utrata danego neuronu oznacza utratę kategorii
- Nie może reprezentować wiedzy wielopoziomowej. Nie ma kategorii wewnątrz kategorii.

Algorytm uczenia Kohonena
Metoda uczenia konkurencyjnego

Modyfikacja wag połączeń sieci:

$$w_{i^*j} = w_{i^*j} + \eta(t)\,(x_j^{\mu} - w_{i^*j}),$$

gdzie: $\eta(t)$ – współczynnik uczenia, t – numer iteracji, x_j^{μ} – wartość j-tej cechy μ-tego wzorca wejściowego, w_{ij} – wartość wagowa połączenia wejściowego węzła j z i-tym neuronem wyjściowym.

Wartość wyjściowa:

$$h_i^{\mu} = \sum_j w_{ij}\, x_j^{\mu} = \mathbf{w}_i \cdot \mathbf{x}^{\mu}$$

Przy normalizacji wekora wag oraz normalizacji wektorów wejściowych spełniona jest zależność:

$$|\mathbf{w}_{i*} - \mathbf{x}^{\mu}| \leq \mathbf{w}_i \cdot \mathbf{x}^{\mu}$$

Dla innych miar:

$$d\,(\mathbf{x}^{\mu},\mathbf{w}_{i*}) = \min d(\mathbf{x}, \mathbf{w}_i), \quad 0 \leq i \leq n$$

Normalizacja wektora wejściowego i wagowego

Sposoby normalizacji można przedstawić następująco:

- redefinicja składowych wektora wejściowego:

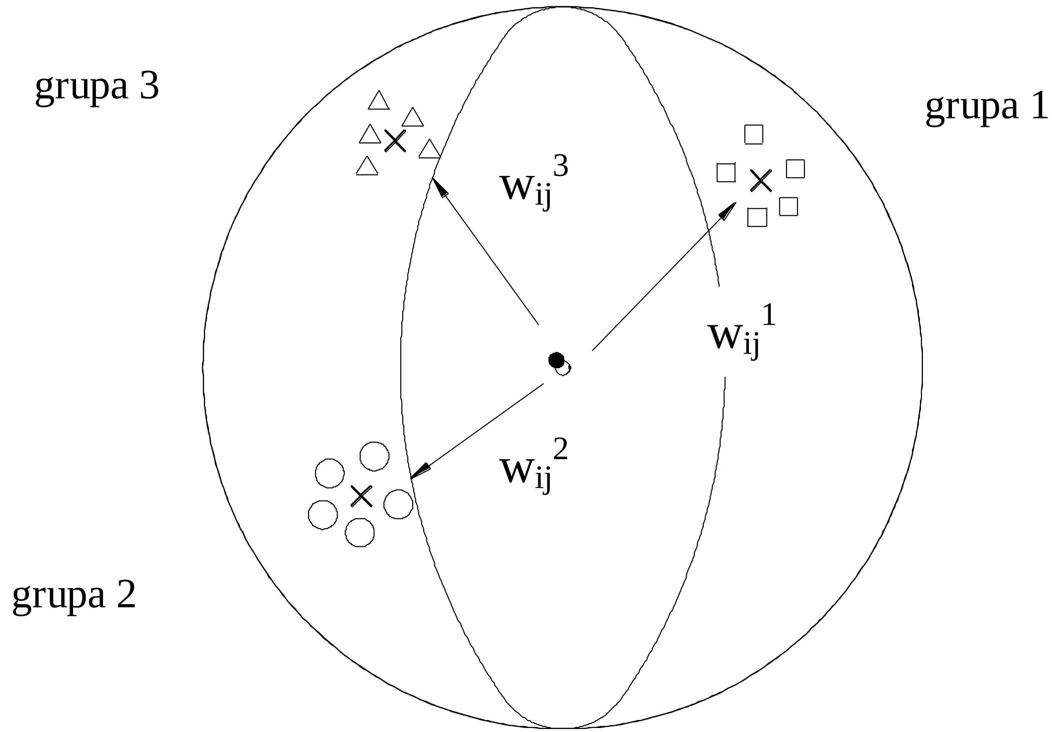
$$x_j \leftarrow \frac{x_j}{\sqrt{\sum_{j=1}^{N} x_j^2}}$$

- zwiększenie wymiaru przestrzeni wejściowej o jeden, tzn. z $R^N \rightarrow R^{N+1}$, gdzie:

$$\sum_{j=1}^{N+1} x_j^2 = 1$$

Algorytm uczenia konkurencyjnego WTA

dostosowanie wektorów wag neuronów zwycięskich do reprezentacji grup wzorców wejściowych



Algorytm uczenia Kohonena

Funkcja celu

$$E\{\mathrm{w}_{\mathrm{ij}}\} = \frac{1}{2}\sum_{ijk} \mathrm{M}_{\mathrm{i}}^{\mu}\left(\mathrm{x}_{\mathrm{j}} - \mathrm{w}_{\mathrm{ij}}\right)^2 = \frac{1}{2}\sum_{\mu}\left|\mathbf{x}^{\mu} - \mathbf{w}_{i*}\right|$$

$$\mathrm{M}_{\mathrm{i}}^{\mu} = \begin{cases} 1 & \text{jeżeże } \mathrm{i} = \mathrm{i}^{*}{}_{\mu} \\ 0 & \end{cases}$$

$$\left\langle \Delta \mathrm{w}_{\mathrm{ij}} \right\rangle = -\eta \frac{\partial E}{\partial \mathrm{w}_{\mathrm{ij}}} = \eta \sum_{\mu} \mathrm{M}_{\mathrm{i}}^{\mu}\left(\mathrm{x}_{\mathrm{j}} - \mathrm{w}_{\mathrm{ij}}\right)$$

Problem zmęczenia neuronów - zapobieganie

– podstawienie jako początkowego wektora wag $\mathbf{w}^0_i$ równego samemu wektorowi wejściowemu,

– aktualizację wag wszystkich tego typu neuronów tak jak zwycięzców, przy zmniejszonym współczynniku uczenia $\eta_i(t)$, tzw. nieszczelne uczenie się,

– aktualizację wag sąsiednich neuronów jako zwycięskich (odwzorowanie cech istotnych),

– dodanie szumu do wektorów wejściowych, dzięki czemu otrzymuje się tzw. rozmyte wektory,
– stopniowe włączanie w proces uczenia wzorców wejściowych [1].

[1] Hecht-Nielsen R.: **Counter-propagation networks**, Applied Optics 26, s. 4979-4984

Algorytm uczenia Kohonena

Funkcja celu Rittera i Schultena

$$E\{\mathrm{w}_{\mathrm{ij}}\} = \frac{1}{2}\sum_{ijk} \mathrm{M}_{\mathrm{i}}^{\mu} \Lambda(i,k)\left(\mathrm{x}_{\mathrm{j}} - \mathrm{w}_{\mathrm{ij}}\right)^2 = \frac{1}{2}\sum_{\mu} \Lambda(i,i^*)\left|\mathbf{x}^{\mu} - \mathbf{w}_{i*}\right|$$

Algorytm uczenia Kohonena

Metoda uczenia konkurencyjnego z funkcją sąsiedztwa

Modyfikacja wartości wag połączeń sieci:

$$\mathrm{w}_{\mathrm{ij}}(t) = \mathrm{w}_{\mathrm{ij}}(t-1) + \eta(t)\Lambda\left(i, i^*\right)\left(\mathrm{x}_{\mathrm{j}} - \mathrm{w}_{\mathrm{ij}}(t-1)\right)$$

Określenie węzła o maksymalnej wartości wyjściowej, gdy $|\mathbf{w}_{\mathrm{i}}| = 1$:

$$|\mathbf{w}_{\mathrm{i}*} - \mathbf{x}^{\mu}| \leq \mathbf{w}_{\mathrm{i}} \cdot \mathbf{x}^{\mu}$$

Obliczenie wartości funkcji sąsiedztwa w otoczeniu węzła zwycięskiego i^*:

$$\Lambda\left(i, i^*\right) = \exp\left(-\frac{\mathrm{d}^2\left(i, i^*\right)}{2\lambda(t)^2}\right)$$

Obliczenie nowych wartości funkcji promienia sąsiedztwa $\lambda(t)$ i współczynnika uczenia $\eta(t)$.

Algorytm uczenia Kohonena

Funkcje sąsiedztwa

• typu prostokąta:

$$\Lambda\left(i,i^{*}\right)=\begin{cases}1 & \text{dla}\,\mathrm{d(i,w)}\leq\lambda(t)\\ 0 & \text{dla innych}\end{cases}$$

• typu Gaussa:

$$\Lambda\left(i,i^{*}\right)=\exp\left(-\frac{\mathrm{d}^{2}\left(i,i^{*}\right)}{2\lambda(t)^{2}}\right),$$

gdzie:

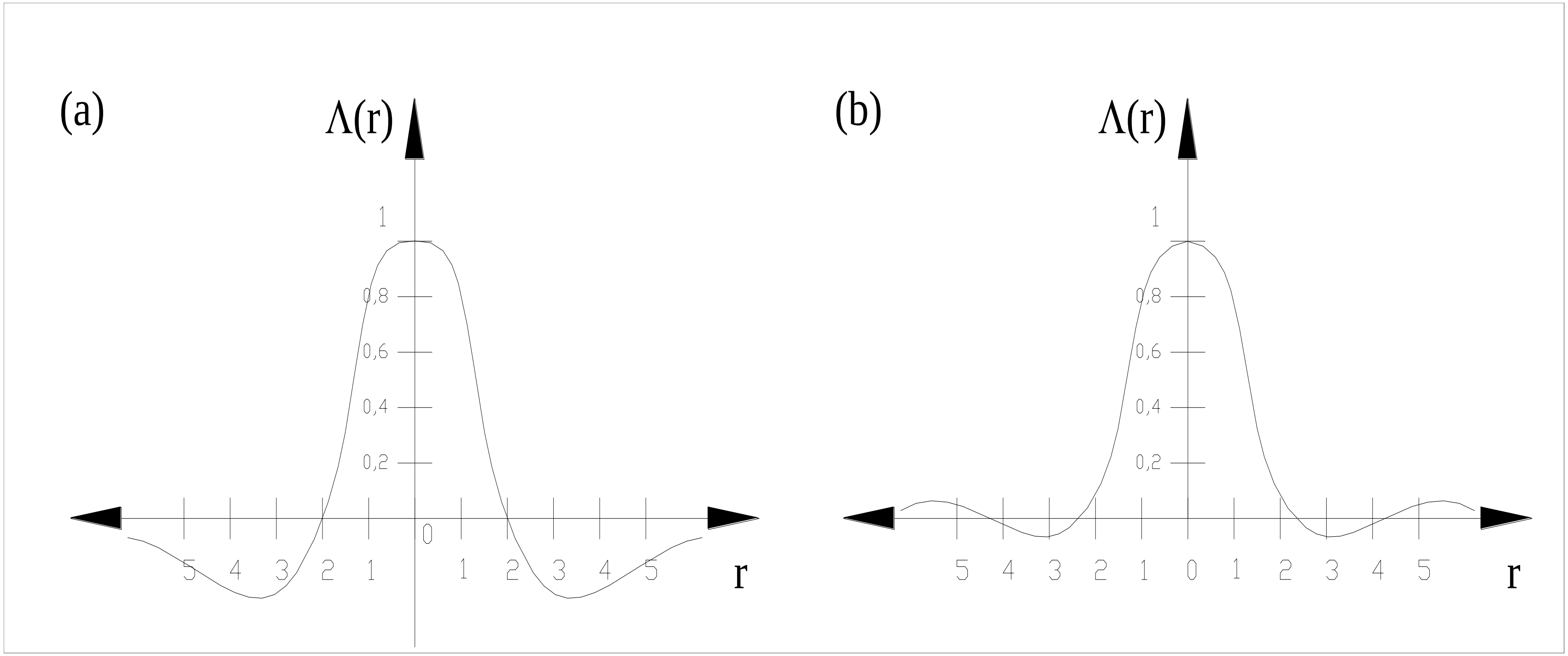
λ(t) - promień sąsiedztwa o wartości malejącej wraz z czasem uczenia *t*,

$d(i,i^*)$ - odległość w mierze Euklidesa węzła zwycięskiego i^* i *i*-tego w przestrzeni wyjściowej sieci.

• typu „Mexican hat" (kapelusz meksykański), w którym realizowana jest zasada hamowania obocznego, czyli ujemnego wpływu na węzły najdalej położone w stosunku do węzła zwycięskiego:

$$\Lambda\left(i,i^{*}\right)=\frac{1-s(t)\mathrm{d}_{ii^{*}}^{2}}{\left(1+\mathrm{d}_{ii^{*}}^{2}\right)^{2}} \qquad \mathrm{d}_{ii^{*}}=\left|r_{i}-r_{i^{*}}\right| \qquad s(t)=\tanh\left(\frac{t}{T_{11}}\right)$$

Funkcje sąsiedztwa



Algorytm uczenia Kohonena

Funkcje promienia sąsiedztwa

Jako funkcje promienia sąsiedztwa $\lambda(t)$ stosuje się zwykle następujące:

- funkcję liniową $\lambda_1(t)$:

$$\lambda_1(t) = \lambda_0 \left(1 - \frac{t}{T_1}\right) + 1$$

gdzie: T_1 określa maksymalny okres,
t określa numer iteracji,
λ_0 wartość początkowa promienia sąsiedztwa.

- funkcję wykładniczą:

$$\lambda_2(t) = \lambda_0 \left(\frac{\lambda_{min}}{\lambda_0}\right)^{\frac{t}{T_{max}}}$$

gdzie: T_{max} - parametr określający maksymalną wartość iteracji,
λ_{min} - wartość przejściowa promienia sąsiedztwa, przy którym funkcja ta przechodzi przez punkt o współrzędnych (T_{max}, λ_{min}).

Algorytm uczenia Kohonena

Funkcje współczynnika uczenia

Funkcja liniowa

$$\eta(t) = \eta_0 \left(1 - \frac{t}{T_1} \right)$$

Funkcja potęgowa

$$\eta(t) = \eta_0 \alpha t^{-\alpha}$$

Funkcja wykładnicza

$$\eta_2(t) = \eta_0 \left(\frac{\eta_{\min}}{\eta_0} \right)^{\frac{t}{T_{\max}}}$$

Funkcje współczynnika uczenia sieci η(t) c.d.

Funkcja pulsacyjna [1]:

$$\eta_3(t) = \eta_0 \left| \, e^{-at} \cos\big(b(t)\, t \big) \right|$$

gdzie:

$$b(t) = \frac{\pi}{2T_z} K(t)$$

$$a(t) = W\, b(t)$$

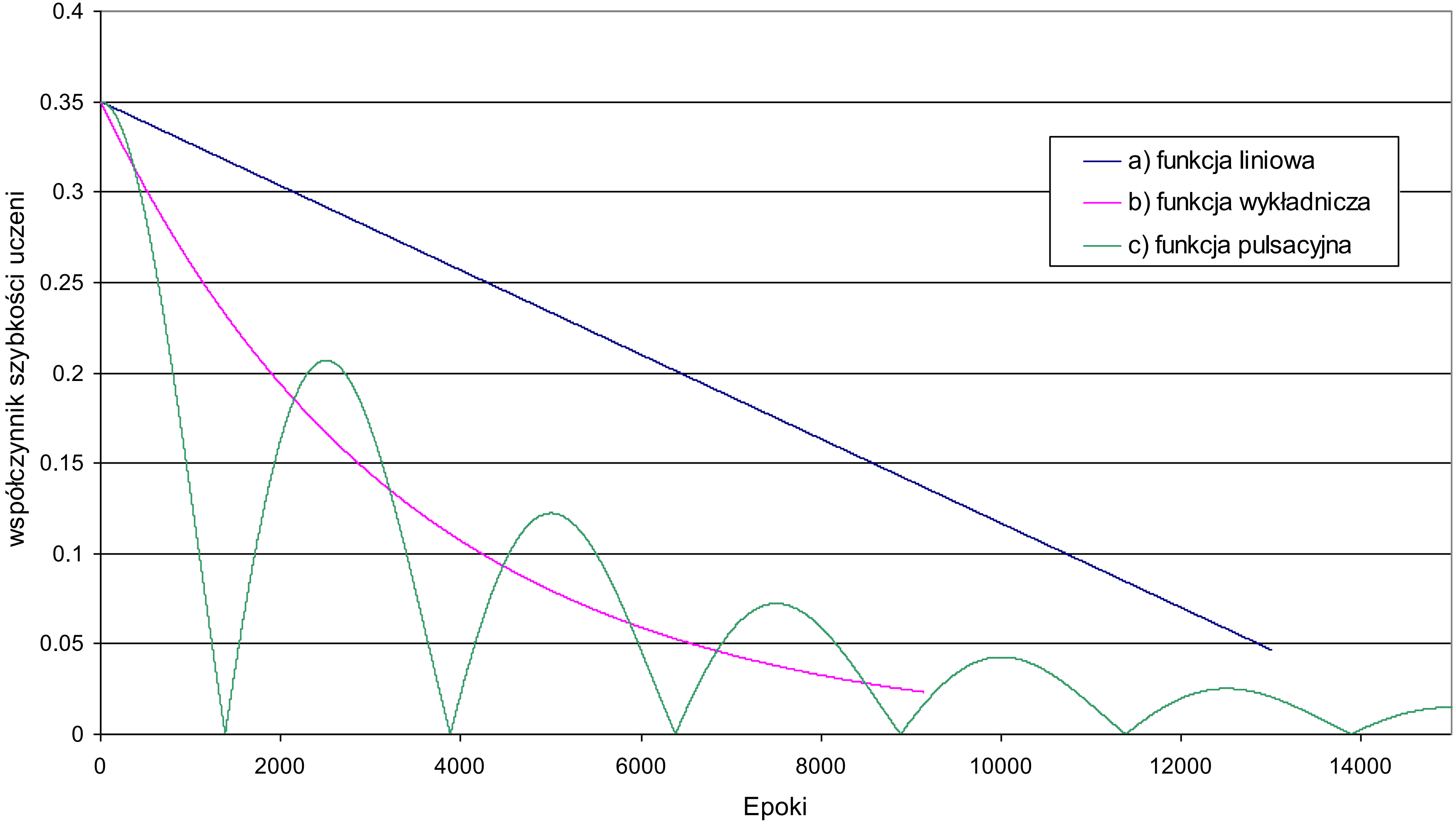
gdzie T_z jest numerem epoki dla pierwszego miejsca zerowego funkcji η_3,
W jest współczynnikiem wielkości spadku wartości funkcji η_3.

Natomiast *K(t)* określa położenie drugiego miejsca zerowego funkcji η_3:

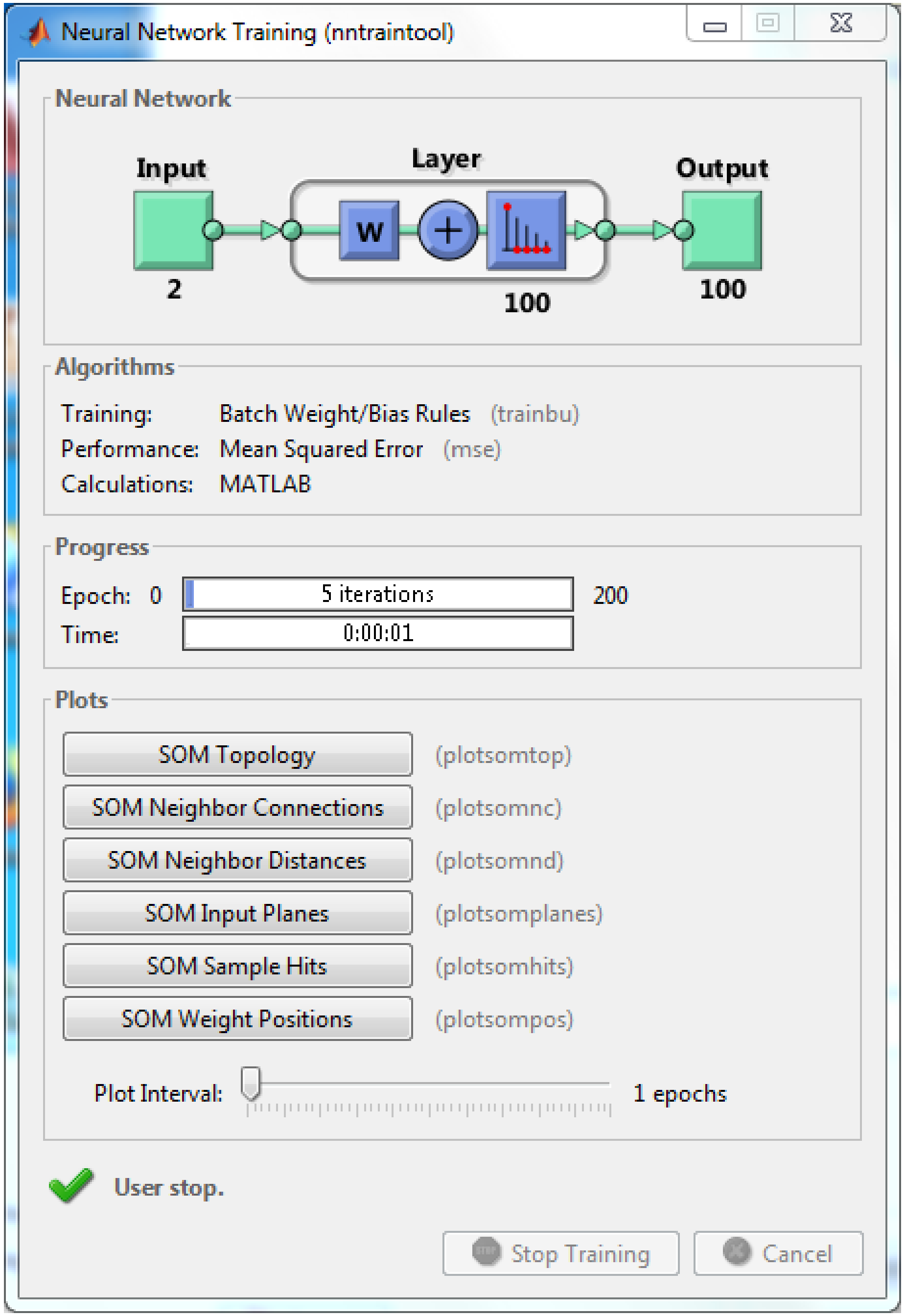
$$K(t) = e^{\left(\frac{t - T_z}{T_z N k} \right)}$$

[1] Zastosowania sztucznej inteligencji w inżynierii produkcji.
Pod Red. R. Knosali, WNT 2002

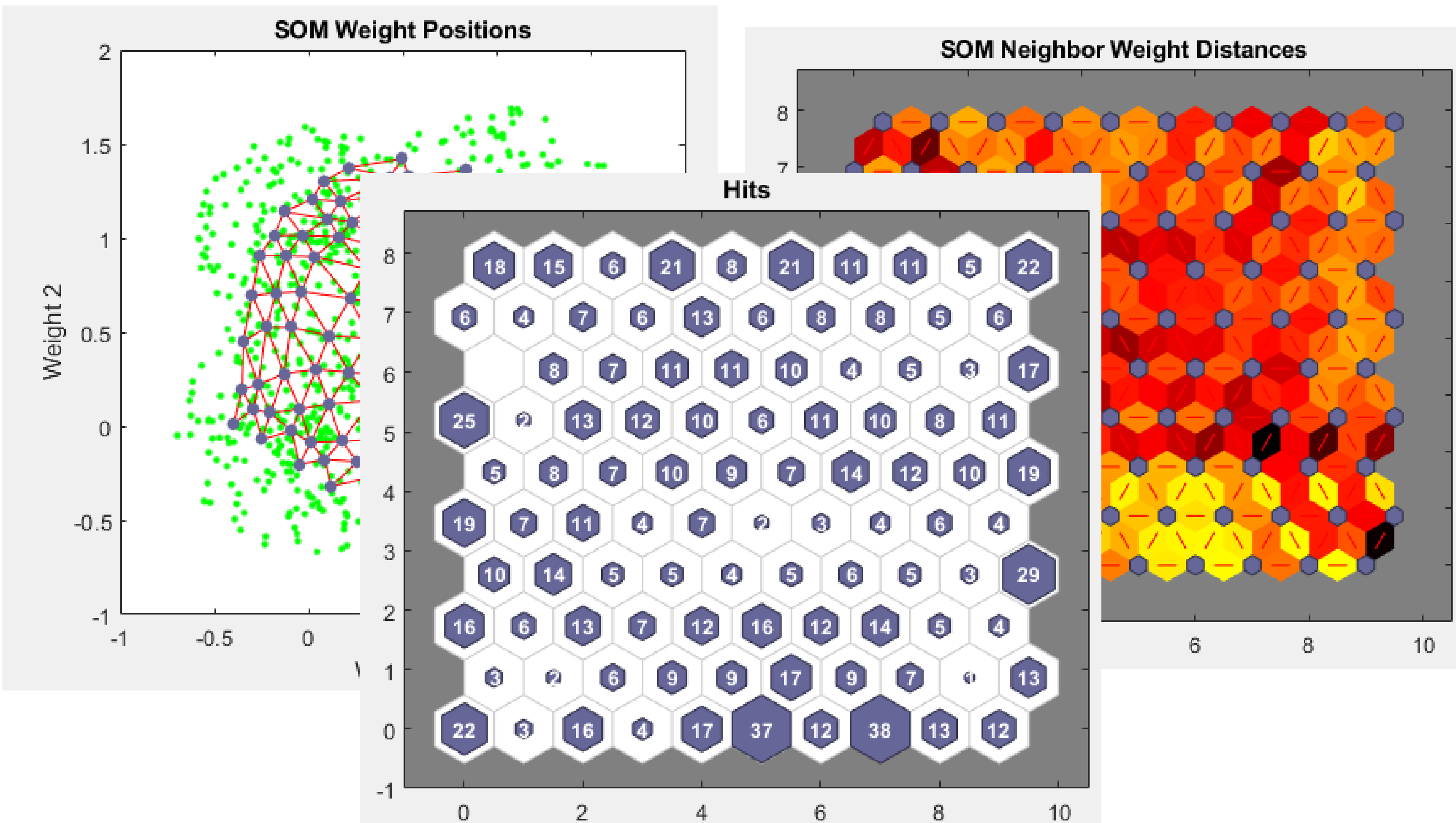
Porównanie funkcji współczynnika uczenia



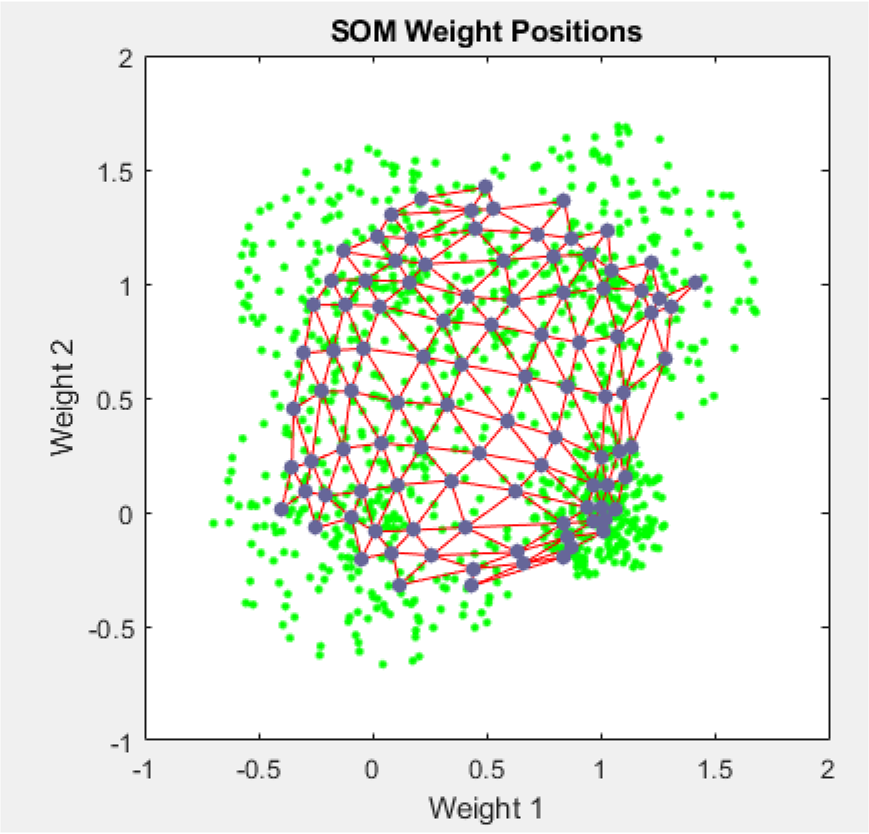
Przykład grupowania



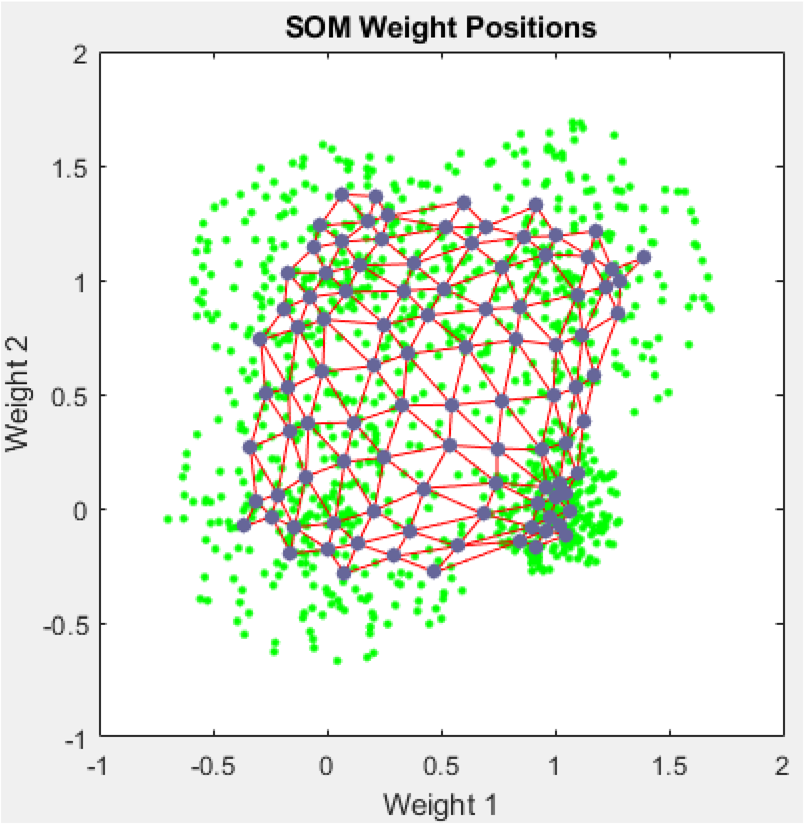
Po 5 iteracjach



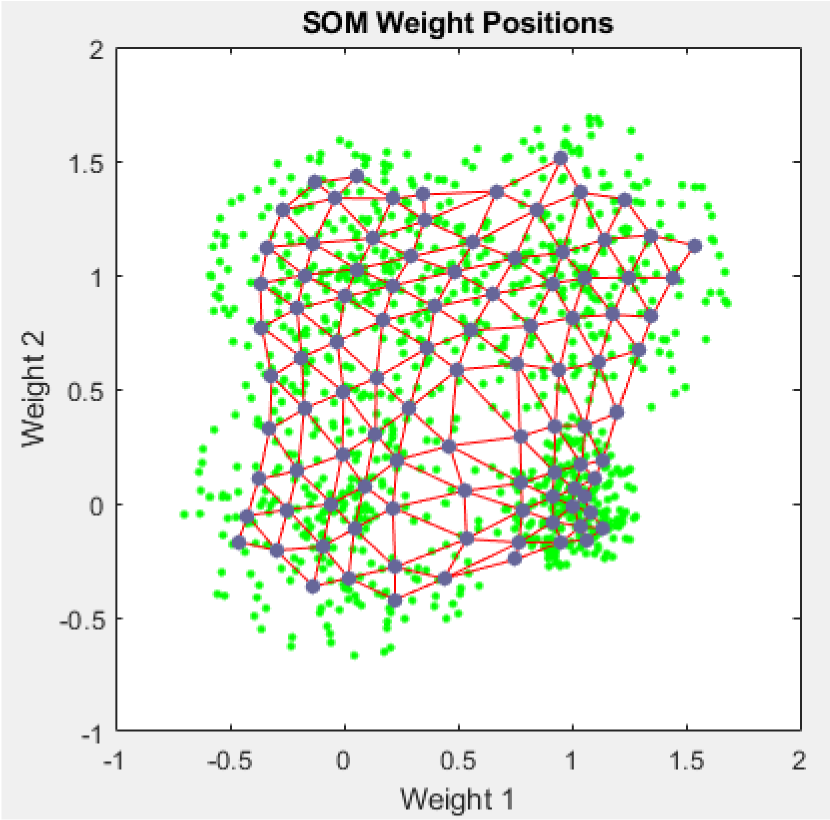
Pozycje neuronów w trakcie uczenia – 5 iteracji



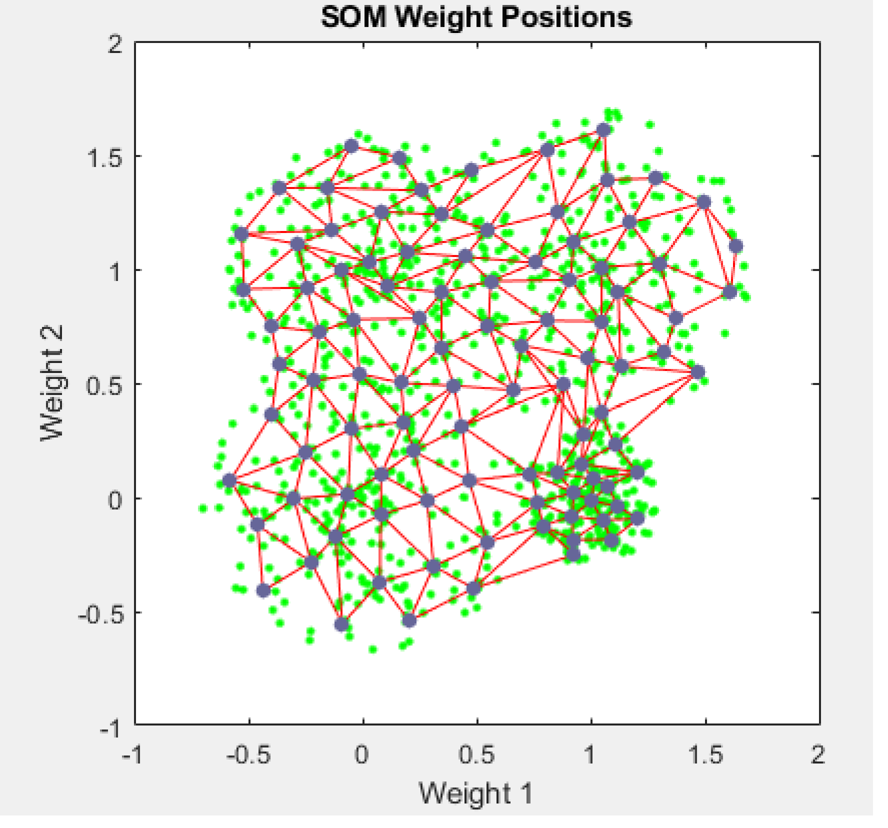
Pozycje neuronów w trakcie uczenia – 32 iteracja



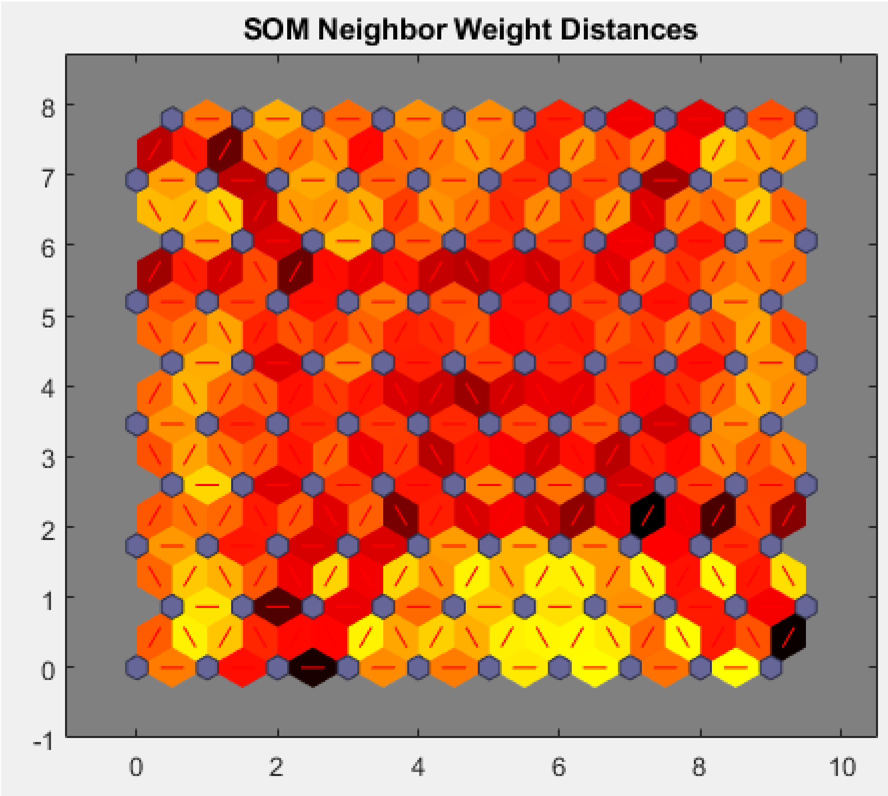
Pozycje neuronów w trakcie uczenia – 63 iteracja

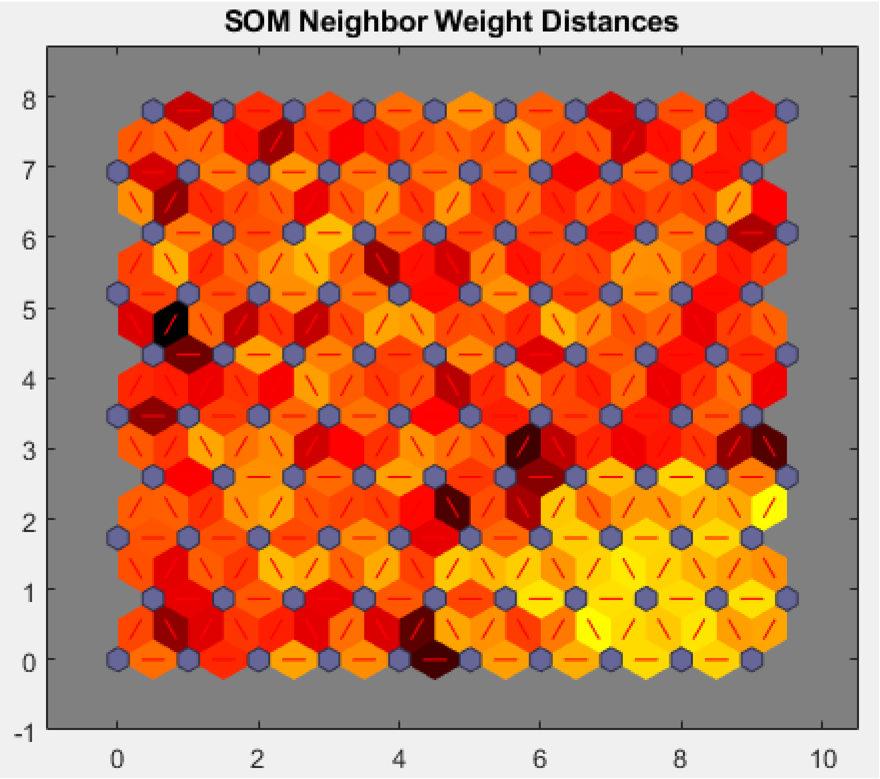


Pozycje neuronów w trakcie uczenia – 200 iteracja

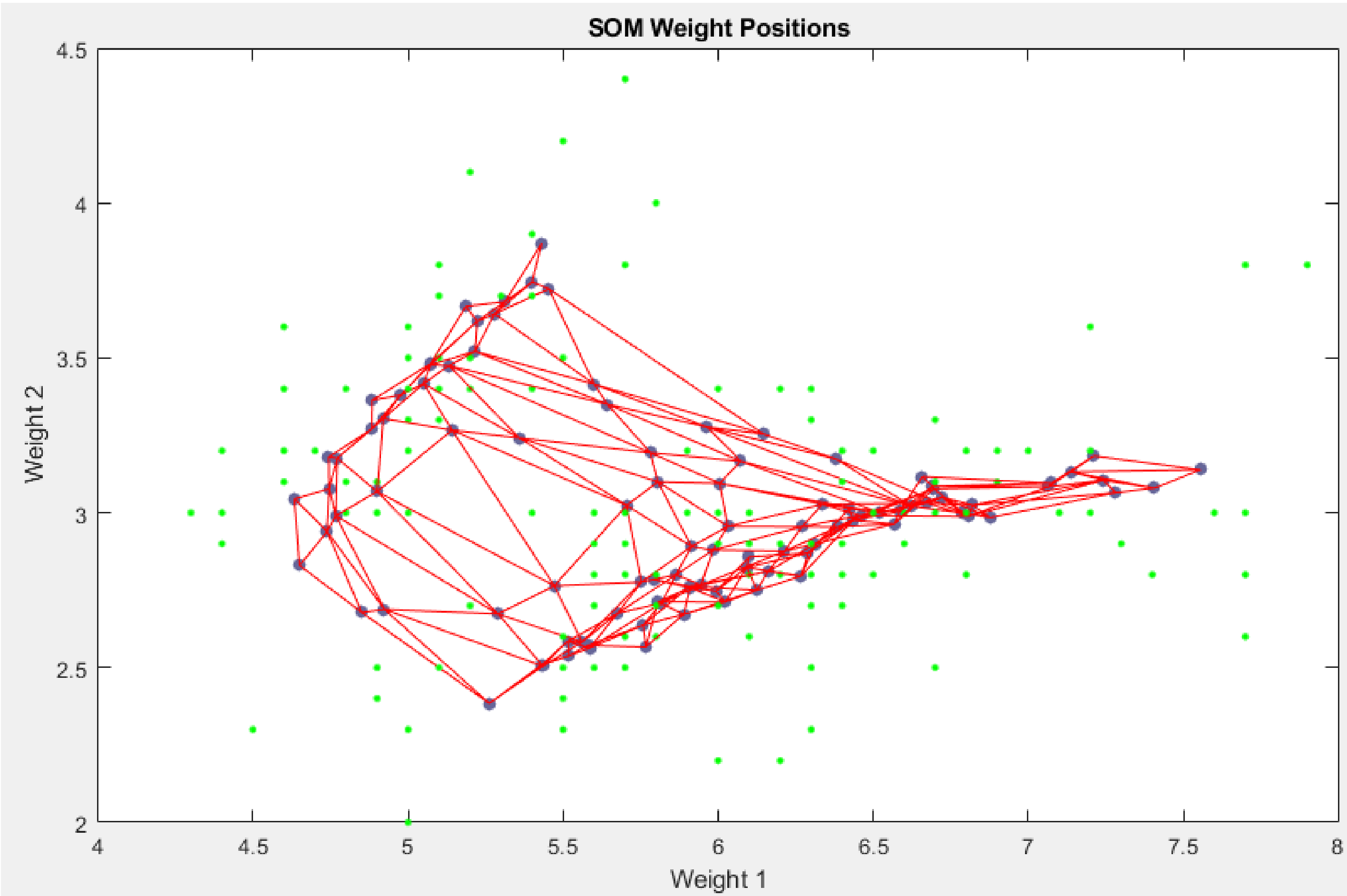


Odległości między neuronami

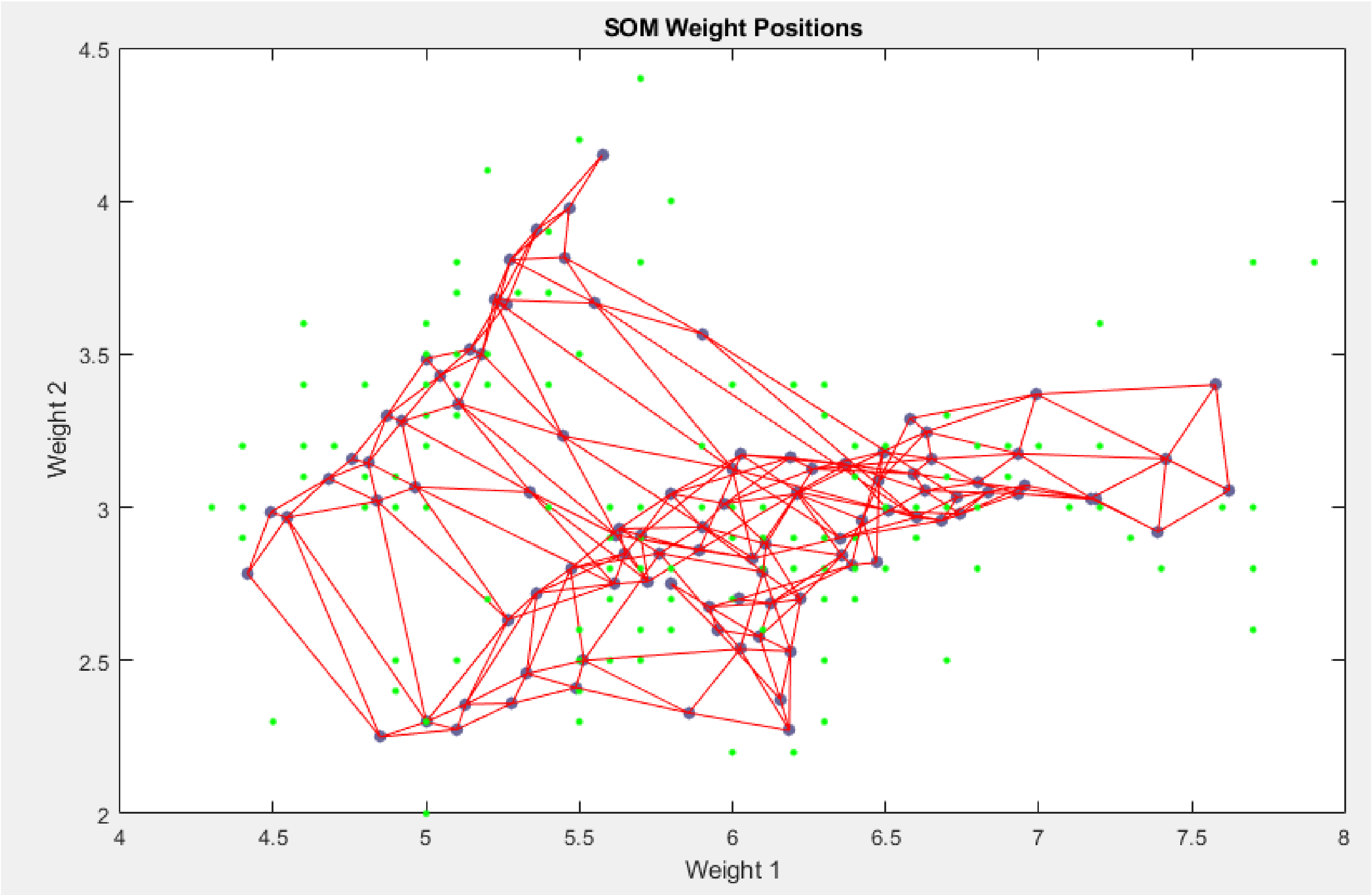




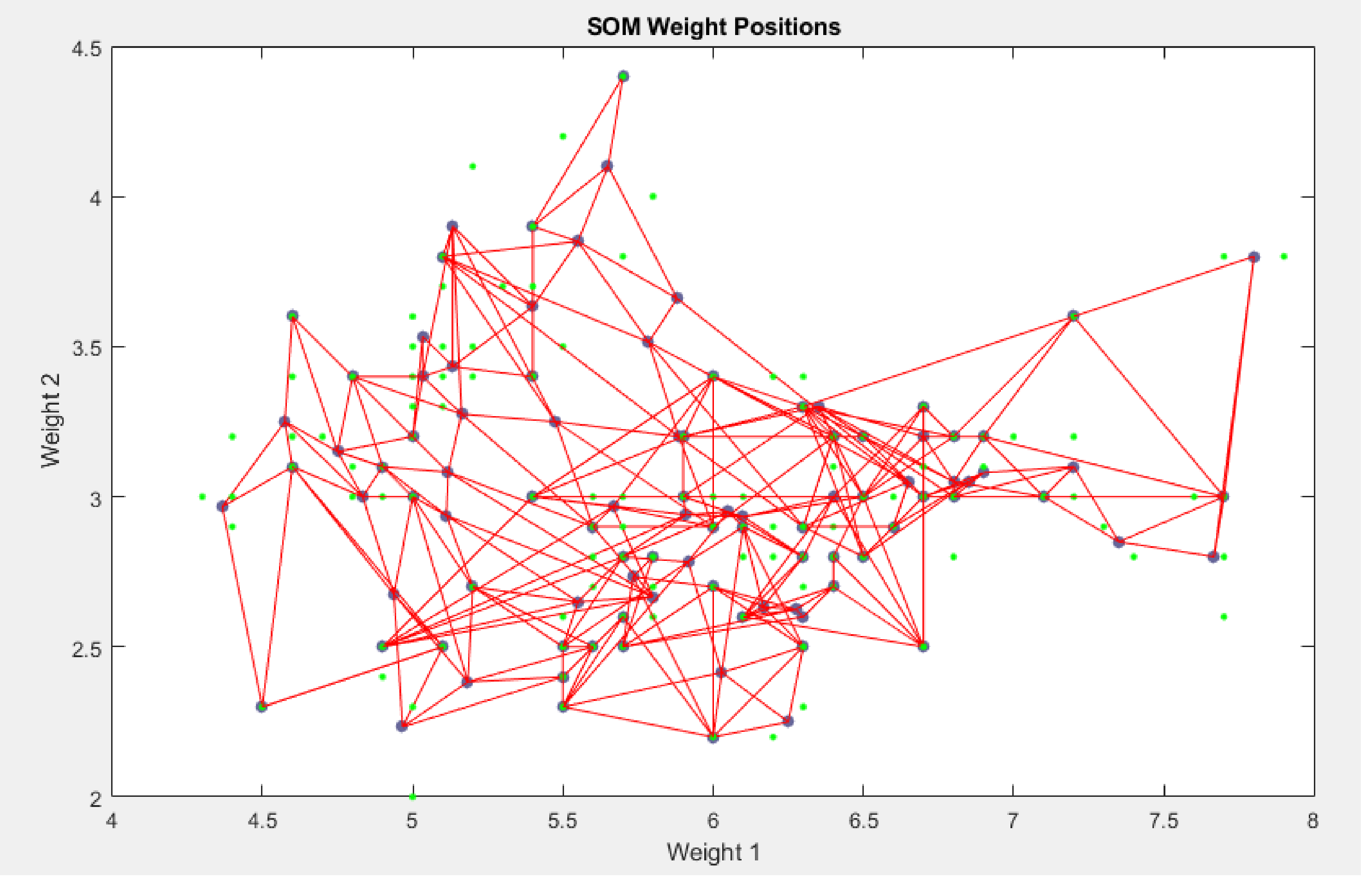
Przykład iris – stan początkowy



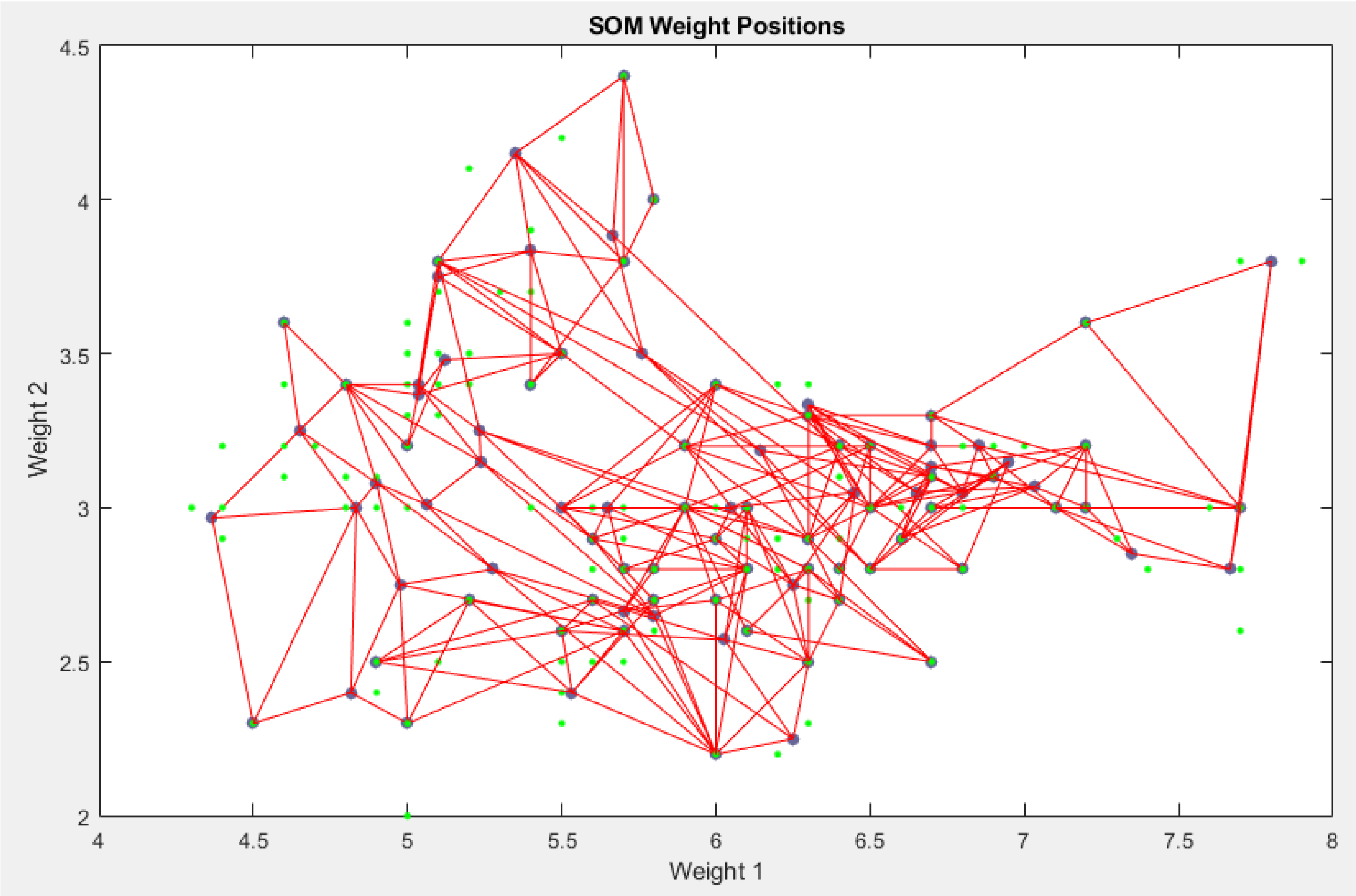
Przykład iris



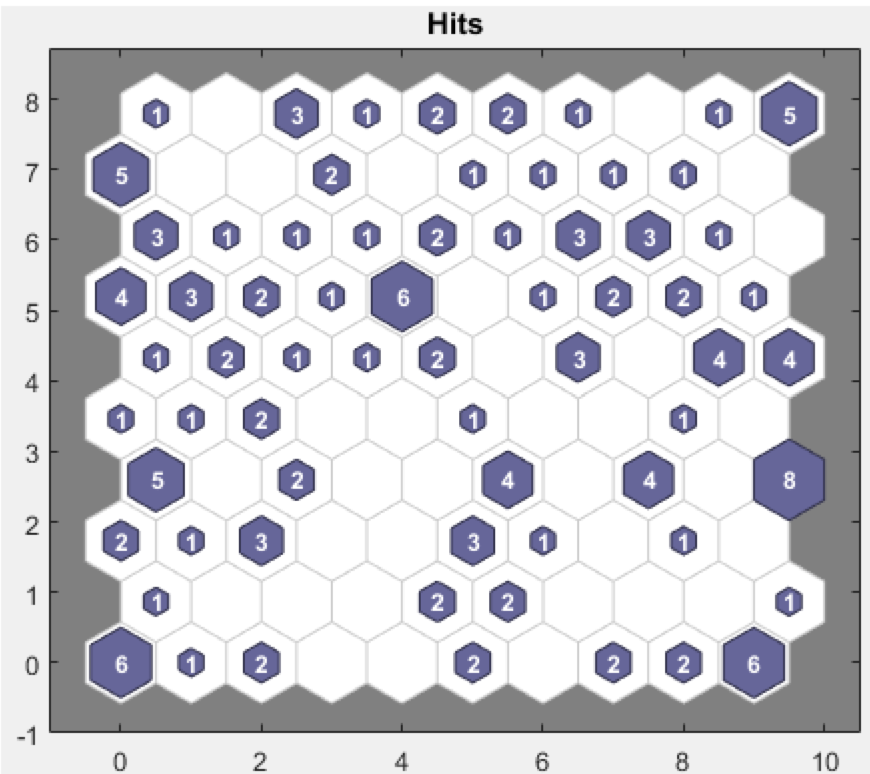
Przykład iris

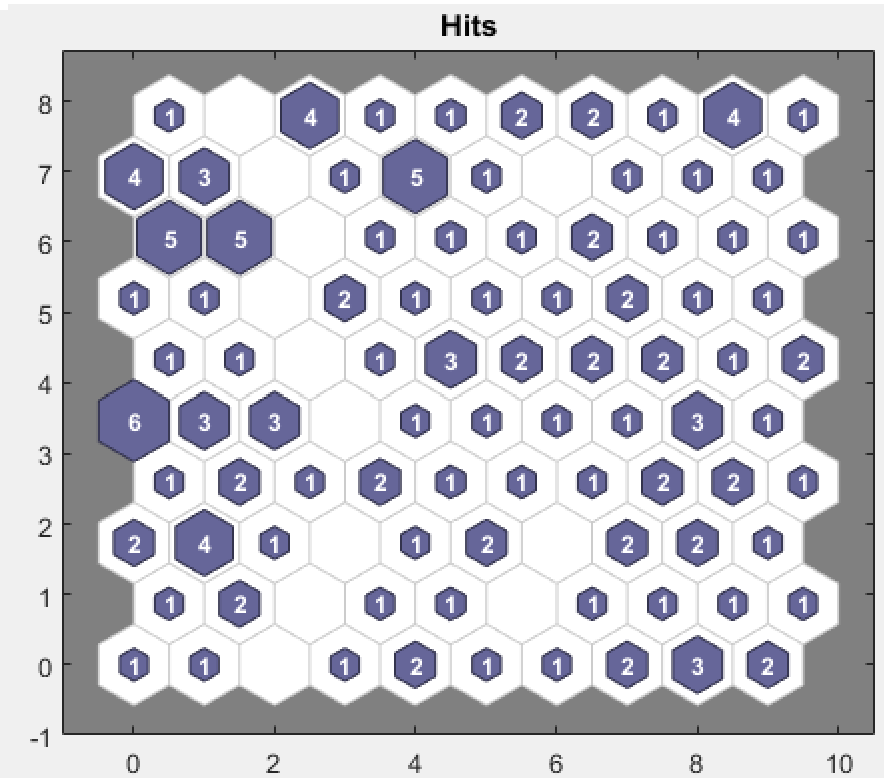


Przykład iris



Iris – aktywność neuronów





Algorytm „gazu neuronowego” cz.I

1. Sortowanie węzłów wyjściowych ze względu na odległość wzorca wejściowego, w mierze Euklidesa, od macierzy wag, związanej z danym węzłem:

$$d_0 < d_1 < \ldots < d_{n-1}$$

gdzie:

n – liczba neuronów wyjściowych,

m=0,1,...,n-1 – określa pozycję węzła *i*-tego w wektorze odległości,

$\mathrm{d_m}$ – oznacza odległość na *m*-tej pozycji *i*-tego węzła wyjściowego w uporządkowanym wektorze odległości, która wyraża się zależnością:

$$\mathrm{d_m} = \left| \mathbf{x} - \mathbf{w}_i \right|$$

gdzie:

x - macierz wzorca wejściowego,

$\mathbf{w}_\mathrm{i}$ **-** macierz wag połączeń i-tego węzła wyjściowego i macierzy wejściowej **x**.

Algorytm „gazu neuronowego” cz.II

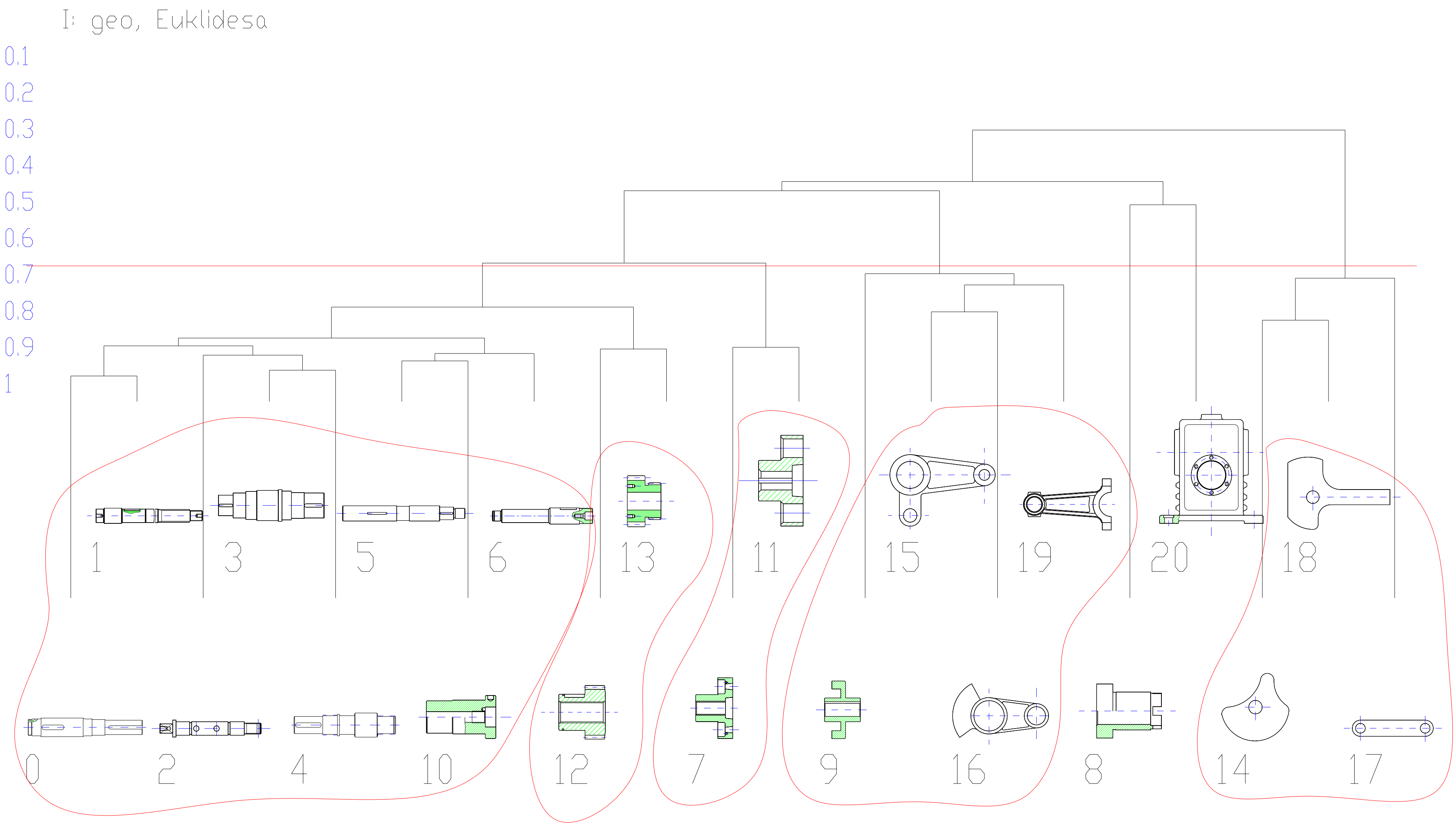
2. Określenie wartości funkcji sąsiedztwa na węzłach uporządkowanych wg odległości d_m:

$$\Lambda\left(\mathrm{i}, \mathrm{i}^*\right) = e^{\frac{\lambda(t)}{\mathrm{d}_\mathrm{m}^2}}$$

gdzie $\lambda(t)$ jest funkcją promienia sąsiedztwa.

Funkcja celu stowarzyszona z funkcją sąsiedztwa charakteryzuje się, przy wartościach $\lambda \to 0$, przejściem w algorytm uczenia konkurencji ostrej. Postać funkcji przybiera kształt paraboli. W przypadku jednak, gdy $\lambda \to \infty$, wtedy przy wolnej zmianie λ dostajemy funkcję wielomodalną, o wielu minimach lokalnych. W tym momencie funkcja ta zapewnia dojście procesu uczenia do globalnego minimum. Wskutek tego algorytm ten jest określany jako najbardziej skuteczny wśród takich algorytmów jak mapa Kohonena czy k-means [3,6].

Dendrogram podobieństwa geometrycznego [1]



[1] Zastosowania sztucznej inteligencji w inżynierii produkcji. Pod Red. R. Knosali, WNT 2002