SOM Self Orgnizing Map

Samoorganizacja i uczenie konkurencyjne

Uczenie nadzorowane i nienadzorowane

Uczenie nadzorowane:

- sygnał wejściowy $\mathbf{x}_i$ posiada oczekiwaną odpowiedź $\mathbf{y}_i$
- trening dąży do minimalizacji błędu pomiędzy odpowiedzią sieci $f(\mathbf{x}_i)$ a wartością oczekiwaną $\mathbf{y}_i$
- przykłady: klasyfikacja, aproksymacja

Uczenie nienadzorowane:

- modelowanie rozkładu danych $\mathbf{x}_i$, brak sygnału zwrotnego $\mathbf{y}_i$
- samoorganizacja - prowadzi do uporządkowania sieci poprzez lokalne, niezależne oddziaływania

Mechanizmy samoorganizacji

- oparte o mechanizm konkurencji - reguła Kohonena
- oparte o regułę asocjacji Hebba (metody korelacyjne)

Aspekty samoorganizacji

- trening dostosowuje wagi neuronów (a przez to ich aktywacje) do rozkładu wartości sygnałów uczących (korelacje, podobieństwo)
- pewne grupy neuronów reagują na podobne wzorce (grupowanie)
- większe sygnały -> większe wagi -> większe aktywacje -> większe różnice pomiędzy grupami neuronów reagujących na odmienne bodźce
- nadmiarowość danych (redundancja) - wielokrotne powtarzanie podobnych wzorców jest niezbędne w samoorganizacji

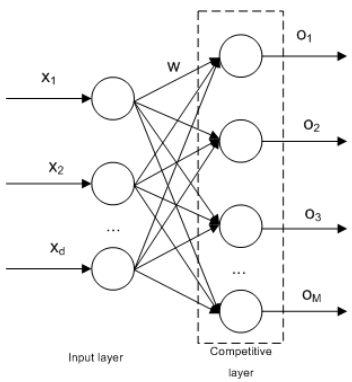
Zastosowanie samoorganizujących się sieci

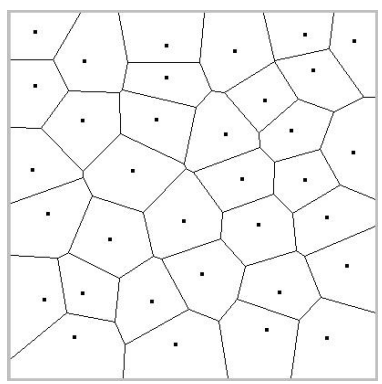
- wykrywanie istotnych cech w sygnale, podobieństw i różnic
- modelowanie rozkładu danych
- **analiza skupień** (klasteryzacja) - grupowanie wektorów podobnych
- analiza składowych głównych, **redukcja wymiarowości** do najistotniejszych cech
- **prototypy** - wykrycie typowego reprezentanta pewnej grupy wektorów treningowych
- **wektorowa kwantyzacja, kodowanie, kompresja** - opisanie zbioru danych mniejsza liczbą wektorów kodujących (prototypów)
- **mapy cech** - wizualizacja organizacji danych w niskowymiarowych przestrzeniach (1D, 3D, 3D) przy zachowaniu porządku topologicznego
- klasyfikacja

Uczenie konkurencyjne

Uczenie konkurencyjne (*competitive learning*) - w trakcie uczenia neurony konkurują ze sobą o prawo do reprezentacji danych wejściowych

- **Zwycięzca bierze wszystko WTA** (*winner takes all*) nazywane również Hard Competitive Learning, tylko sygnał neuronu najbardziej pobudzonego (zwycięzcy) jest brany pod uwagę
- **Zwycięzca bierze większość WTM** (*winner take most*) - łagodniejsza forma (*soft competitive*) biorąca pod uwagę obok zwycięzcy również neurony o słabszym pobudzeniu, w pewnym otoczeniu $h(\mathbf{r}_i, \mathbf{r})$





Neuron zwycięzca k o największym pobudzeniu

$$\mathbf{w}_k^T \mathbf{x} > \mathbf{w_j}^T \mathbf{x}, \qquad j = 1, \ldots, N$$

Dla znormalizowanych wektorów $\mathbf{w}_i$ oraz $\mathbf{x}$

$$\hat{\mathbf{w}} = \frac{\mathbf{w}}{\|\mathbf{w}\|}$$

neuron zwycięzca $\mathbf{w}_k$ ma najmniejszą odległość do $\mathbf{x}$

$$\|\mathbf{x} - \mathbf{w}\|^2 = \|\mathbf{x}\|^2 + \|\mathbf{w}\|^2 - 2\mathbf{w}^T\mathbf{x}$$

$$\min \|\mathbf{x} - \mathbf{w}\| = \max \mathbf{w}^T\mathbf{x}$$

stąd neuron zwycięzca

$$k = \arg\min_j \|\mathbf{w}_j - \mathbf{x}\| = \sqrt{\sum_{i=1}^{d} (x_i - w_{ij})^2}, \quad j = 1, \cdots, N$$

Miary podobieństwa

- odległość Euklidesowa L_2

$$d\left(\mathbf{x}, \mathbf{w}_j\right) = \sqrt{\sum_{k=1}^{d} \left(x_k - w_{jk}\right)^2}$$

- odległość Manhattan L_1

$$d\left(\mathbf{x}, \mathbf{w}_j\right) = \sum_{k=1}^{N} \left|x_k - w_{jk}\right|$$

- iloczyn skalarny (odległość kątowa)

$$d\left(\mathbf{x}, \mathbf{w}_j\right) = 1 - \mathbf{w}_j^T \mathbf{x} = 1 - \|\mathbf{x}\| \|\mathbf{w}_j\| \cos(\mathbf{x}, \mathbf{w}_j)$$

- norma L_∞

$$d\left(\mathbf{x}, \mathbf{w}_j\right) = \max_k \left|x_k - w_{jk}\right|$$

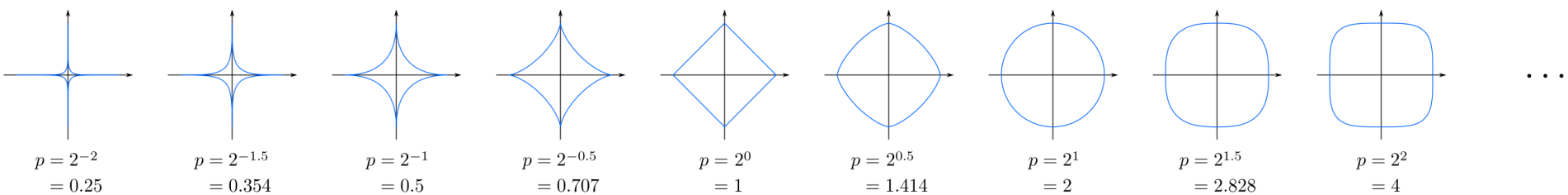
Odległość Minkowskiego

Uogólniona funkcja odległości

$$d(\mathbf{x}, \mathbf{w}) = \left(\sum_{k=1}^{K} (x_k - w_k)^p \right)^{\frac{1}{p}}$$

- gdy $p = 1$ odległość Manhattan
- gdy $p = 2$ odległość euklidesowa

Jednostkowy okrąg w przestrzeni R^2

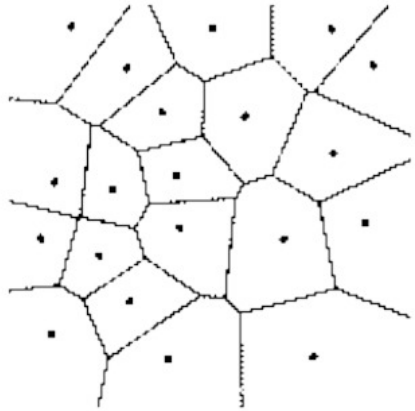


Rys: wikipedia.org

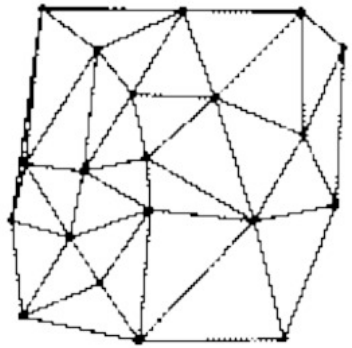
Voronoi i Delaunay

Zbiór punktów

diagram Voronoi



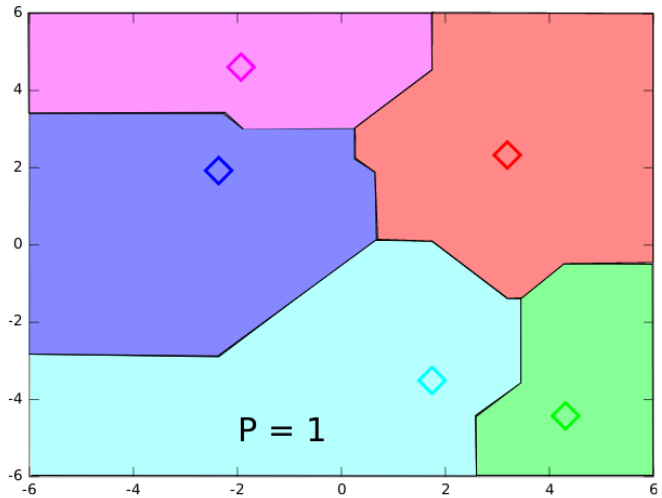
Triangulacja Delauny

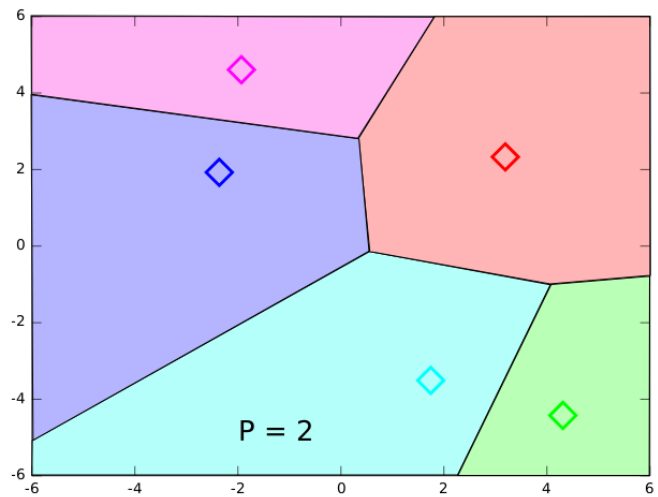


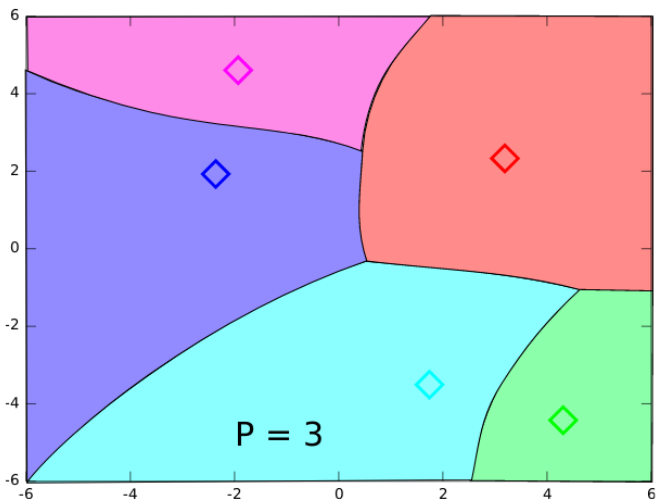
Neurony dzielą przestrzeń cech na rozdzielne obszary (obszary Voronoi), w danym obszarze tylko jeden neuron zwycięża konkurencję

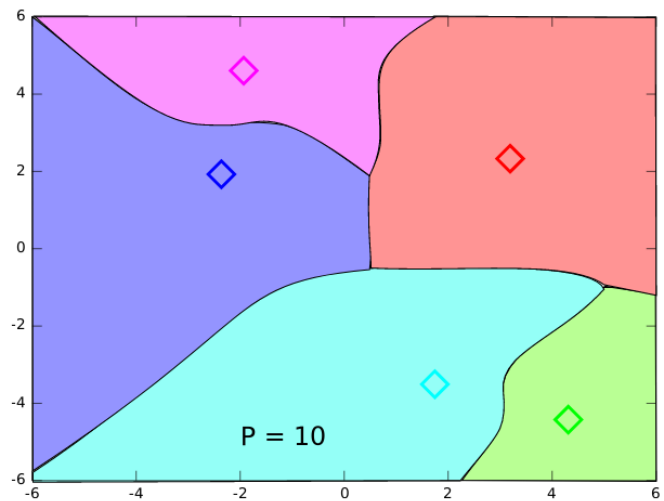
- **zbiór Voronoi** - zbiór wektorów wewnątrz obszaru Voronoi.
- łącząc węzły, których obszary Voronoia mają wspólną krawędź otrzymujemy **traingulację Delaunaya**

Strefy wpływów dla różnych metryk





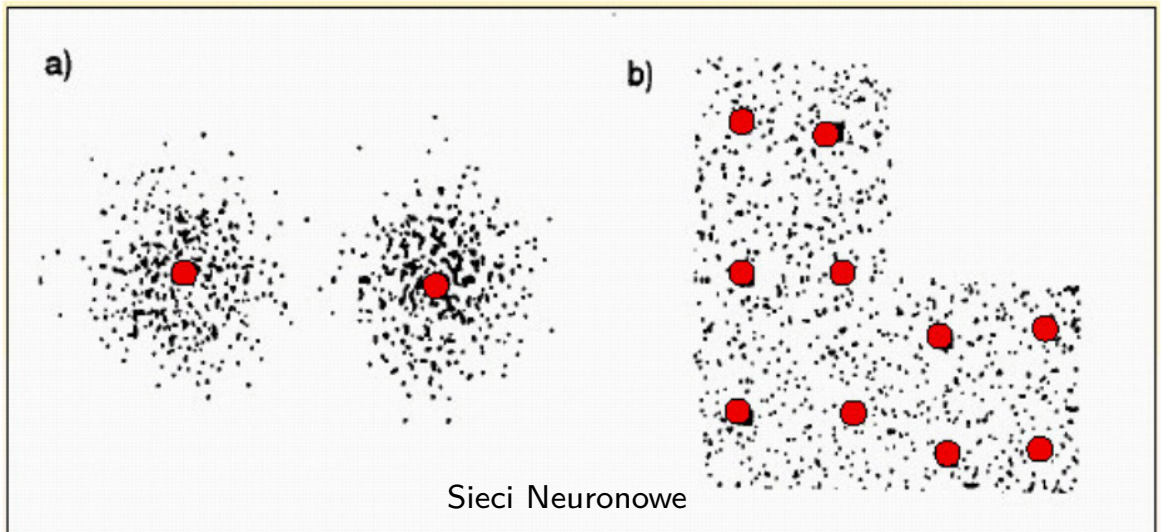




Rys: M. Czoków, J. Piersa, Wstęp do sieci neuronowych

Kwantowanie wektorowe

- **Kwantowanie wektorowe** - zakodowanie danych za pomocą wektorów prototypowych. Pojedynczy prototyp reprezentuje grupę wektorów wejściowych
- **Księga kodów** - zbiór prototypów kodujących (ich numery)
- Stratna kompresja danych - sygnał kodowany sekwencją numerów prototypów z księgi kodów, liczba prototypów k jest mniejsza od liczby wzorców n
- Do zakodowania wektora $\mathbf{x} \in \mathbb{R}^d$ wystarczy $\log_2 k$ bitów, gdzie k to rozmiar księgi kodów. Cały zbiór danych o rozmiarze $d \cdot n$ wymaga $n \cdot \log_2 k + d \cdot k$ bitów



Algorytm kwantowania wektorowego WTA

1. Zainicjuj wagi k neuronów
2. Dla każdego wzorca uczącego $\mathbf{x}$ znajdź neuron zwycięzcę o największej aktywacji
3. Przesuń wagi zwycięskiego neuronu w kierunku wektora $\mathbf{x}$ (**reguła Kohonena**)

$$\mathbf{w}(t+1) = \mathbf{w}(t) + \eta(t)\|\mathbf{x} - \mathbf{w}(t)\|$$

4. Powtarzaj 2 i 3 aż do uzyskania zbieżności

Błąd kwantowania wektorowego dla k neuronów

$$E(\mathbf{x}; \mathbf{w}, t) = \frac{1}{k} \sum_{\mathbf{i}} \left\| \mathbf{x}_i - \mathbf{w}_{m(i)} \right\|^2$$

gdzie $\mathbf{w}_{m(i)}$ to wagi neuronu zwycięskiego dla sygnału $\mathbf{x}_i$

Przykład: kompresja obrazów

- obraz o wymiarach $N_x \times N_y$ dzielimy na równe ramki o wymiarach $n_x \times n_y$
- wektor $\mathbf{x}_i$ to wektor $n_x \times n_y$ pikseli (np. odcienie szarości) Zbiór wszystkich wektorów $\mathbf{x}_i$ tworzy zbiór treningowy
- uczenie samoorganizujące dowolną metodą w celu minimalizacji błędu kwantyzacji
- podobne ramki pobudzają ten sam neuron zwycięski
- kompresja: prezentacja wszystkich ramek $\mathbf{x}_i$ pozwala ustalić kolejność neuronów zwycięskich $1, 5, 1, 30, \ldots$, ich numery tworzą księgę kodów
- współczynnik kompresji

$$k_r = \frac{N n_x n_y T}{N \log_2 k + k n_x n_y t}$$

gdzie N liczba ramek, k - liczba neuronów, T liczba bitów potrzebna do reprezentacji wartości $\mathbf{x}_i$, t liczba bitów potrzebna do reprezentacji wagi $\mathbf{w}_i$

oryginał

VQ 1024 codebook

kompresja: 2.6 bits/pixel

☞ Image Compression with Vector Quantization by Ivan-Assen Ivanov

Własności kwantyzacji wektorowej

- VQ nie musi odzwierciedlać gęstości rozkładu danych, przecenia rejony mało prawdopodobnych danych i nie docenia rejonów o dużym prawdopodobieństwie danych
- wynik zależy od punktu początkowego i kolejności prezentacji wzorców - każde uruchomienie może prowadzić do innej kwantyzacji
- zasada jednakowego zniekształcenia: każdy neuron powinien mieć podobny wkład do końcowego błędu kwantyzacji
- problem **martwych neuronów**: przy losowej inicjalizacji część neuronów może znaleźć się w obszarach gdzie nie będzie pobudzeń
- rozwiązanie: mechanizmy zmęczenia (np. neurony z sumieniem), odpowiadają chwilowemu zmęczeniu neuronu po wzbudzeniu w sieciach neuronowych biologicznych

Mechanizm sumienia (DeSieno)

Neurony z sumieniem (*conscience learning*) - mechanizm zmęczenia zapobiegający powstawaniu martwych neuronów

Miara częstości zwycięstw $p_i(t)$

$$p_i(t+1) = \begin{cases} p_i(t) + B \cdot (1 - p_i(t)) & \text{dla neuronu zwycięzcy} \\ p_i(t) - B \cdot p_i(t) & \text{dla pozostałych} \end{cases}$$

gdzie $0 < B \ll 1$ (np. $B = 0.0001$)

Zwycięzca spośród N neuronów

$$k = \arg\min_j \left(\|\mathbf{w}_j - \mathbf{x}\| - C \left(\frac{1}{N} - p_j(t) \right) \right) \quad j = 1, \cdots, N$$

- u często wygrywających neuronów rośnie „poczucie winy”
- dla $C = 0$ nie występuje zmęczenie (standardowy algorytm Kohonena)
- potencjał p_i ograniczony do 1

Mechanizm zmęczenia ze współczynnikiem czułości

Każdy prototyp $\mathbf{w}_i$ posiada współczynnik czułości $s_i \geq 0$ początkowo równy 0.
W kolejnych krokach treningu kwantyzacji wektorowej

$$s_i(t+1) = \begin{cases} 0 & \text{dla neuronu zwycięzcy} \\ s_i(t) + \gamma & \text{dla pozostałych} \end{cases}$$

gdzie $\gamma > 0$

Zwycięzca wybierany jest spośród N neuronów

$$k = \arg\min_i \left(\|\mathbf{w}_i - \mathbf{x}\| - s_i \right) \quad i = 1, \cdots, N$$

- lepsze odwzorowanie gęstości danych, nawet przy dużej zmienności gęstości w różnych obszarach

Mechanizm zmęczenia z modyfikacją odległości

Efektywna odległość zależna od częstości zwycięstw N_i

$$\hat{d}(\mathbf{x}, \mathbf{w}_i) = N_i d(\mathbf{x}, \mathbf{w}_i)$$

- kara dodawana przy określaniu zwycięzcy, częściej wygrywające neurony mają większą odległość efektywną
- w trakcie aktualizacji wag używa się oryginalnej miary odległości $d(\mathbf{x}, \mathbf{w})$
- po kilku początkowych cyklach, wszystkie neurony mają szansę wygrać przynajmniej raz

Gaz neuronowy (Schulten i Martinez 1991)

Wektorowa kwantyzacja WTM - krok aktualizacji zależny od pozycji w rankingu zwycięzców

1. Losowa inicjalizacja N neuronów
2. Powtarzaj następne kroki T razy
3. Wybierz losowy wektor $\mathbf{x}$
4. Zrób ranking neuronów posortowanych względem odległości od $\mathbf{x}$

   $$\|\mathbf{w}_i - \mathbf{x}\| \leq \|\mathbf{w}_j - \mathbf{x}\|, \qquad i < j$$

   Niech $m(i)$ określa miejsce neuronu w rankingu, gdzie dla zwycięzcy $m(i) = 0$
5. Adaptacja

   $$\Delta\mathbf{w}_i = \eta(t) h_i(t) (\mathbf{x} - \mathbf{w}_i)$$

   $$h_i(t) = e^{-\frac{m(i)}{\lambda(t)}}$$

6. Zmniejsz eksponencjalnie obszar $\lambda(t)$ i stałą uczenia $\eta(t)$

Promień sąsiedztwa

Promień sąsiedztwa maleje w trakcie uczenia od λ_0 do pewnej wartości λ_{min}

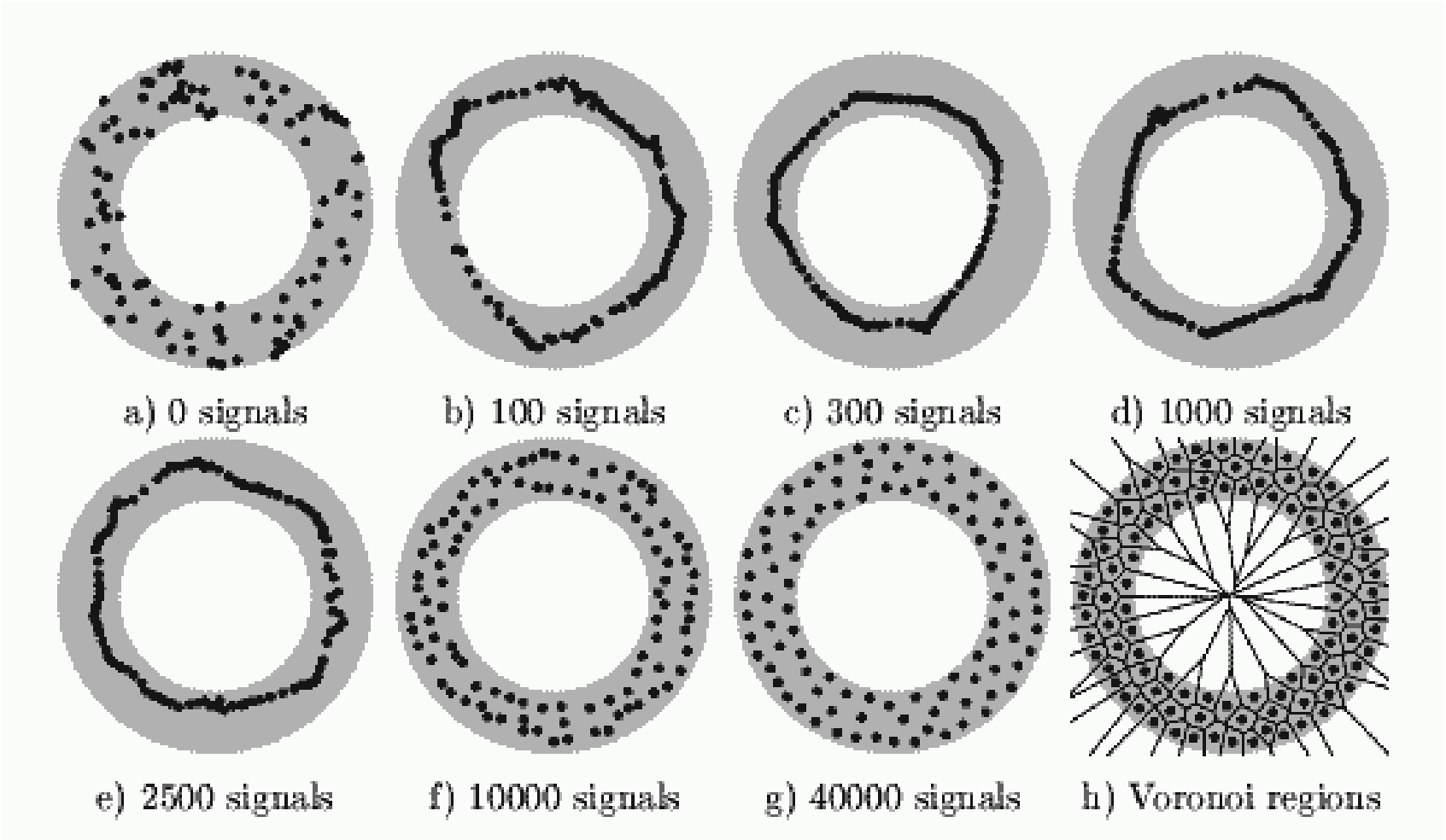
$$\lambda(t) = \lambda_0 \left(\frac{\lambda_{min}}{\lambda_0} \right)^{\frac{t}{T}}$$

- dla $\lambda = 0$ uczenie WTA
- dla $\lambda > 0$ aktualizowane są wagi wszystkich neuronów z siłą zależna od pozycji w rankingu

Współczynnik uczenia $\eta(t)$ zazwyczaj maleje wykładniczo

$$\eta(t) = \eta_0 \left(\frac{\eta_{min}}{\eta_0} \right)^{\frac{t}{T}}$$

Przykład kwantyzacji gazu neuronowego

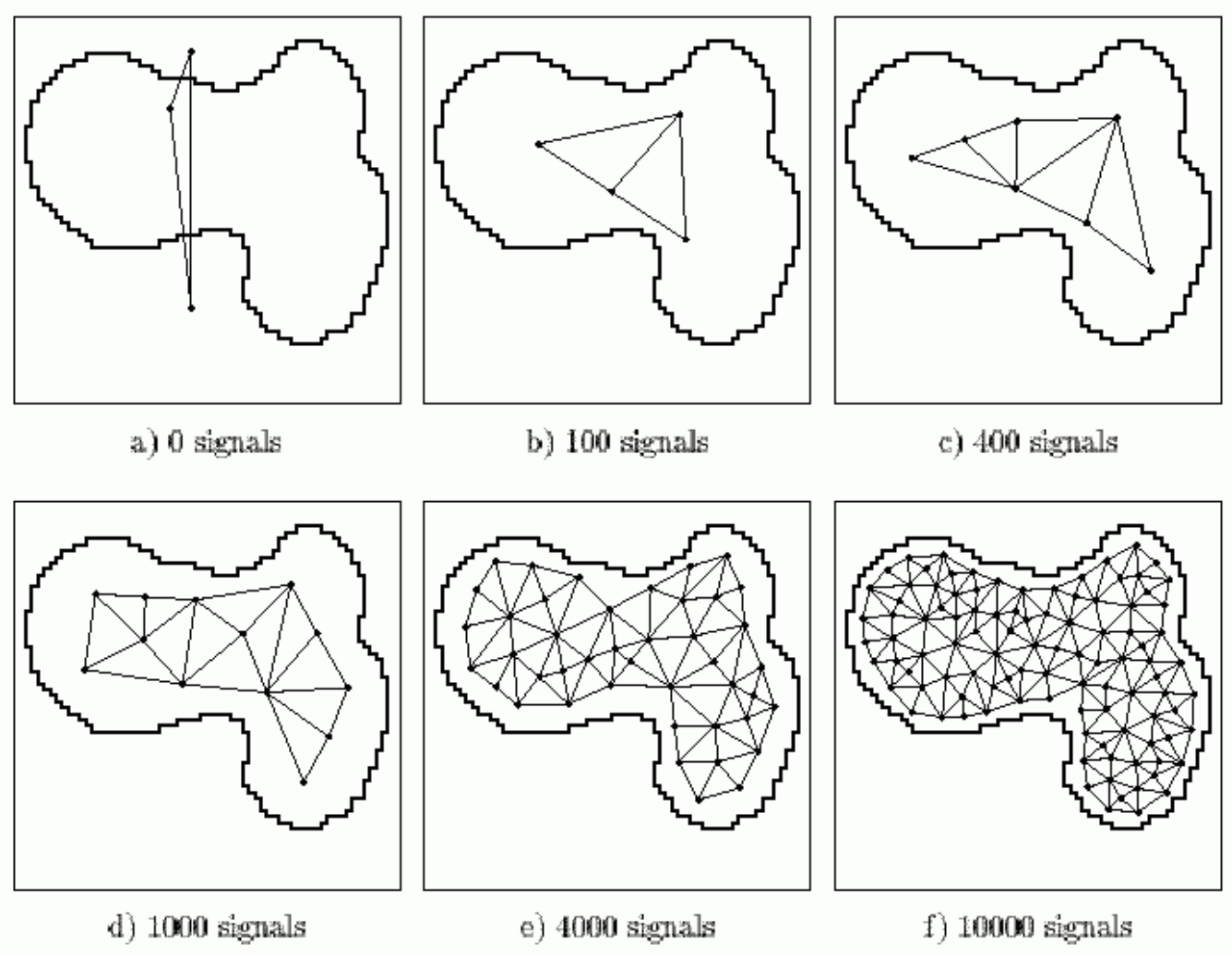


Growing Cell Structures

Rozrastająca się wektorowa kwantyzacja. Neurony połączone są krawędziami definiującymi ich sąsiedztwo.
Początkowa topologia: k-wymiarowy sympleks (np. k=1, 2, 3).

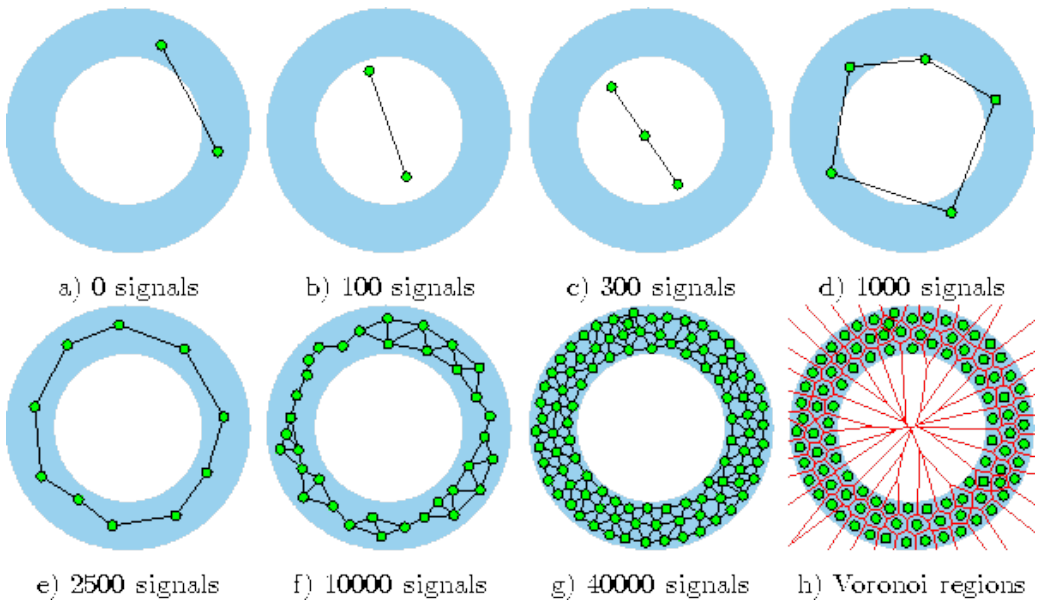
1. Dla losowego $\mathbf{x}$ znajdź zwycięzcę c
2. Aktualizuj wagi zwycięzcy oraz połączonych sąsiadów: $\Delta \mathbf{w}_i = \eta_s(\mathbf{x} - \mathbf{w}_i)$
3. Dla zwycięzcy c zwiększ sumaryczny błąd $E_c = E_c + ||\mathbf{x} - \mathbf{w}||^2$
4. Po ustalonej liczbie epok T znajdź neuron o największym E_c i wstaw nowy neuron r w środku krawędzi łączącej go z najdalszym sąsiadem tworząc lokalny k-wymiarowy sympleks. Błąd sąsiadów r zmniejsz o wartość proporcjonalną do liczby tych sąsiadów. Błąd nowego neuronu ustaw na średni błąd jego sąsiadów.
5. Powtarzaj powyższe kroki aż do osiągnięcia kryterium stopu (np. określony rozmiar sieci)

Trening GCS

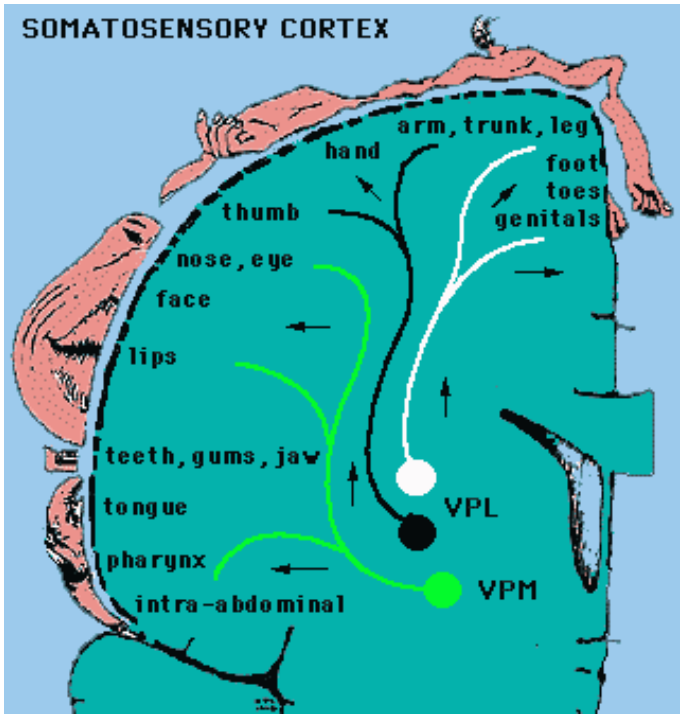

a) 0 signals
b) 100 signals
c) 400 signals
d) 1000 signals
e) 4000 signals
f) 10000 signals

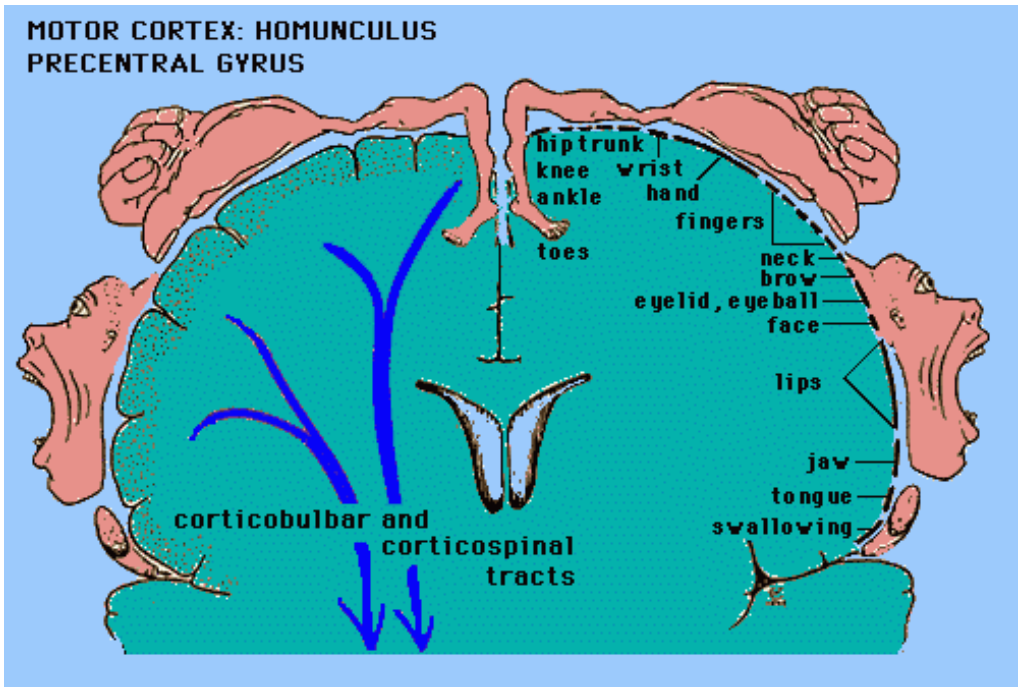
Growing Neural Gas (Fritzke, 1994)

- GNG, rozrastający się gaz neuronowy, algorytm zbliżony do GCS, startuje od pary neuronów i wprowadza mechanizmem starzenia się połączeń
- nowe połączenie powstaje między zwycięzcą i drugim w rankingu neuronem
- połączenia, które żyją za długo są usuwane



Mapy czuciowe i motoryczne





Inspiracje biologiczne SOM

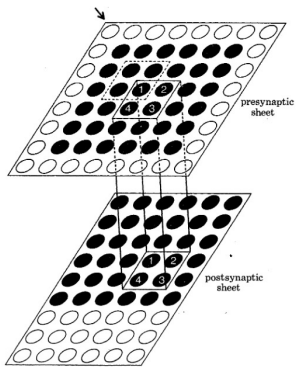
- przetwarzanie różnych danych zmysłowych i motorycznych (wzrok, słuch, dotyk, ...) odbywa się w różnych obszarach kory mózgowej
- neurony o podobnych funkcjach są obok siebie $=>$ mapy topograficzne
- istnieje topologiczna zależność w obliczeniach realizowanych przez mózg
- Przykłady
  - mapy somatosensoryczne układu czuciowego,
  - mapy motoryczne kory i móżdżku,
  - mapy tonotopiczne układu słuchowego,
  - mapy orientacji dwuocznej układu wzrokowego,
  - mapy wielomodalne układu orientacji (wzgórki czworacze górne)

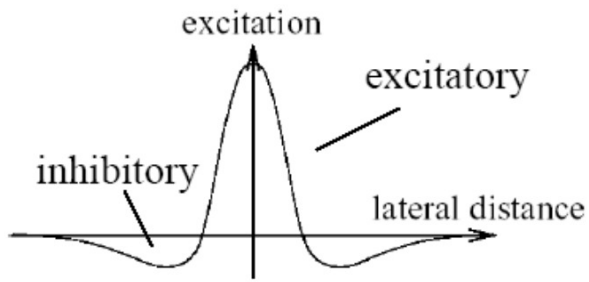
Modele samoorganiacji

- Samoorganizująca się mapa cech - SOM lub SOFM (Self-Organized Feature Mapping)
- Połączenia lokalne (sąsiednie neurony): neuron silnie pobudzany przez pobliskie, słabo przez odległe, hamowany przez neurony pośrednie
- von der Malsburg i Willshaw (1976) - model układu wzrokowego
- Amari (1980) - model ciągłej tkanki neuronowej uwzględniający pobudzenie i hamowanie zależne od odległości
- Kohonen (1981) uproszczony model, bez hamowania, dwie fazy uczenia: konkurencja i kooperacja.

Model Willshaw, van der Malsburg (1976)

Samoorganizacja w mapowaniu sygnału z siatkówki na korę wzrokową, transformacja 2D -> 2D , brak redukcji wymiaru, połączenia wewnątrzwarstwowe,

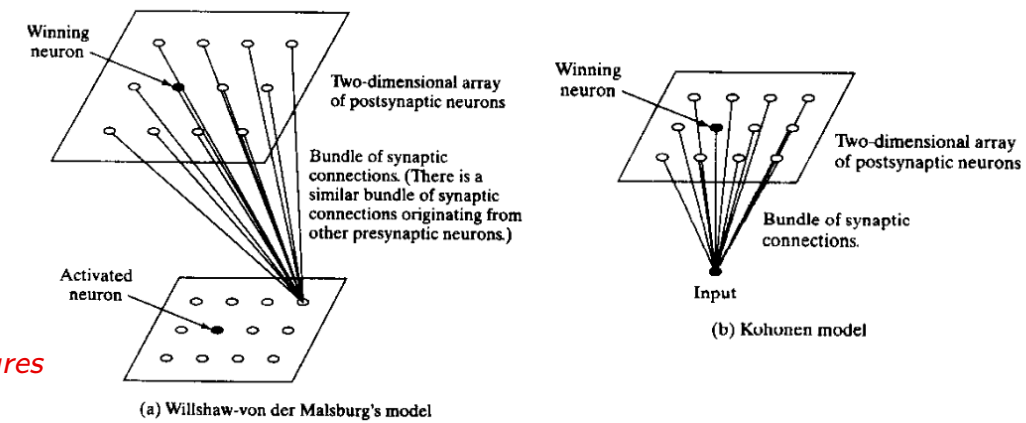




- aktywacja typu „Meksykańskiego kapelusza”
  - kooperacja - pobudzenia bliskiego zasięgu
  - konkurencja - hamowanie w dalszych regionach
- siatkówka (wejście) posiada narzuconą organizację topologiczną
- wybór zwycięskiego neuronu wynikiem dynamiki neuronalnej

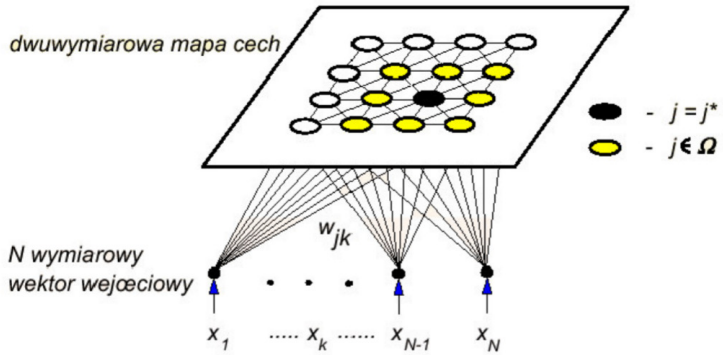
Sieć Kohonena (1981)

- Sieć Kohonena - najbardziej popularny model tworzenia map, stąd pojęcie SOM często utożsamiane z „siecią Kohonena”
- dwie fazy uczenia
  - konkurencja - najpierw wybierz globalnego zwycięzcę
  - kooperacja - uczenie neuronu zwycięzcy oraz neuronów w sąsiedztwie zwycięzcy
- sieci bez hamowania, brak połączeń wewnątrz warstwy
- wejście jest sygnałem ciągłym, nie posiada narzucoenej struktury topologicznej, dowolna wymiarowość sygnału wejściowego, mapowanie nD do 1D, 2D, 3D



Rys: Marcus Kaiser, Neuroinformatics, lectures

Mapa cech Kohonena



- neurony zachowują porządek topologiczny (np. siatka 2D) tworząc mapę cech
- sąsiadujące na siatce neurony powinny reagować na podobne wzorce
- podobieństwo wzorców określa odległość $\|\mathbf{x} - \mathbf{w}_i\|$ w przestrzeni wejściowej $\mathbf{x}, \mathbf{w}_i \in \mathbb{R}^N$
- sąsiadujące neurony na siatce, odległość neuronów $\|\mathbf{r}_i - \mathbf{r}_j\|$

Rys: Małgorzata Krętowska, Sztuczne sieci neuronowe

Algorytm SOM

1. Zainicjuj wagi połączeń neuronów ukrytych
2. Dla danego sygnału $\mathbf{x}(\mathbf{t})$ w kroku $t = 1, 2, \ldots, T$ znajdź najsilniej reagujący neuron c (np. najbliższy)

$$\|\mathbf{x} - \mathbf{w}_j\| = \sqrt{\sum_i (x_i - w_{ij})^2}, \quad c = \arg\min_j \|\mathbf{x} - \mathbf{w}_j\|$$

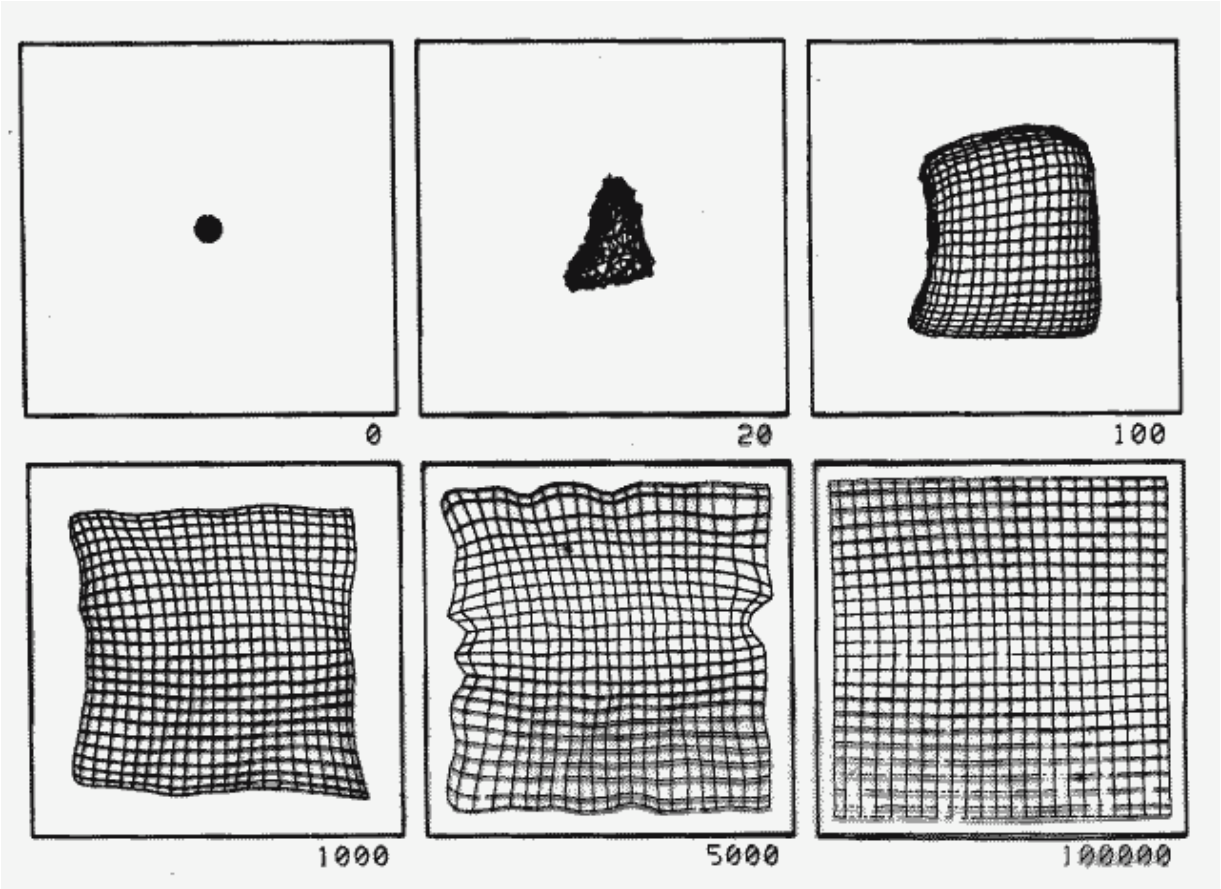
3. Zmień wagi neuronu c oraz neuronów w sąsiedztwie $O(c)$ przybliżając je w stronę wektora $\mathbf{x}$

$$\mathbf{w}_i(t+1) = \mathbf{w}_i(t) + \eta(t) h\left(\mathbf{r}_i, \mathbf{r}_c, t\right) \left[\mathbf{x}(t) - \mathbf{w}_i(t)\right] \text{ dla } i \in O(c)$$

Funkcja $h(\mathbf{r}, \mathbf{r}_c, t)$ określa zasięg sąsiedztwa zwycięzcy

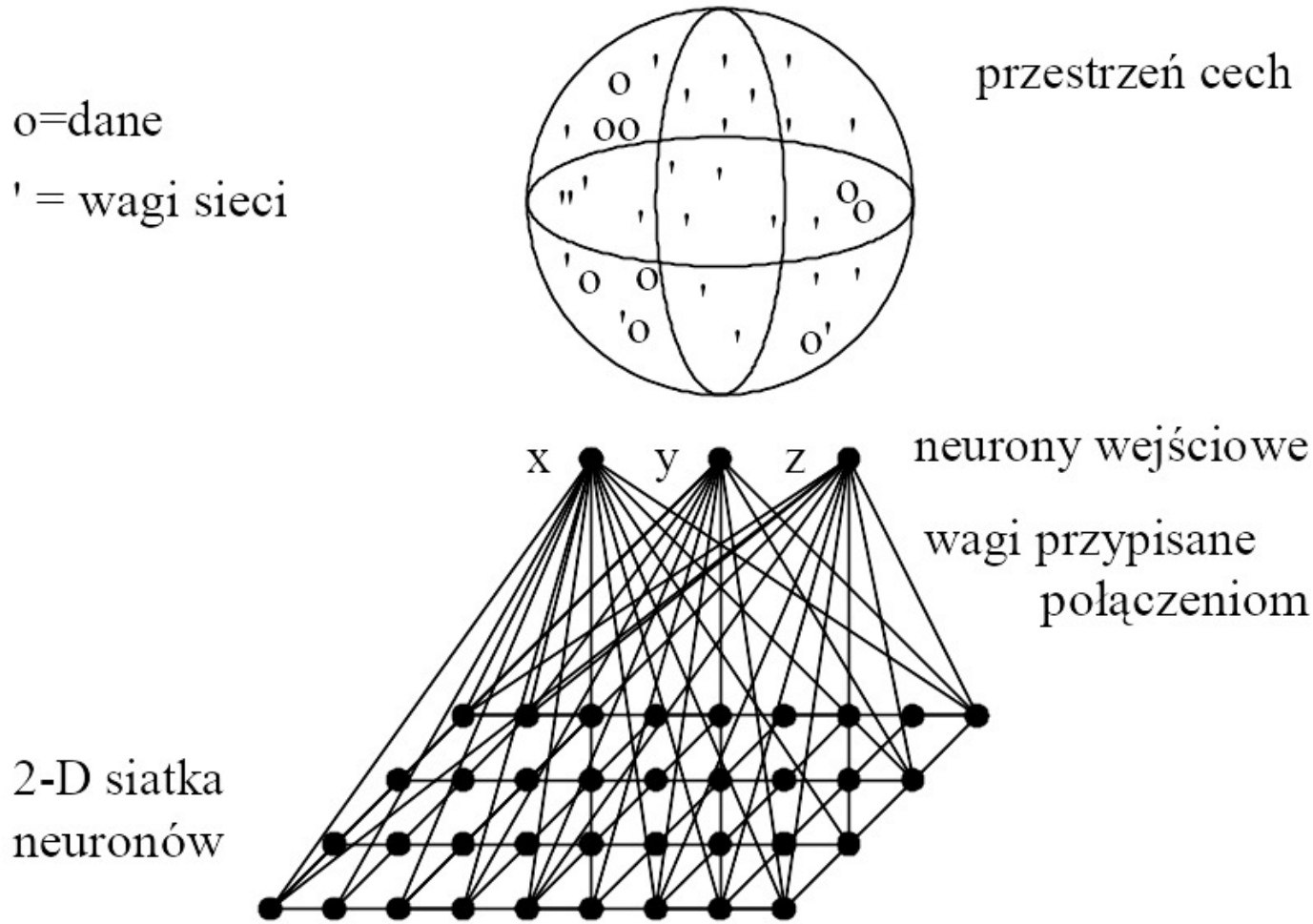
$$h\left(\mathbf{r}, \mathbf{r}_c, t\right) = h_0(t) \exp\left(-\frac{\|\mathbf{r} - \mathbf{r}_c\|^2}{2\sigma^2(t)}\right)$$

Przykład: mapowanie 2D -> 2D

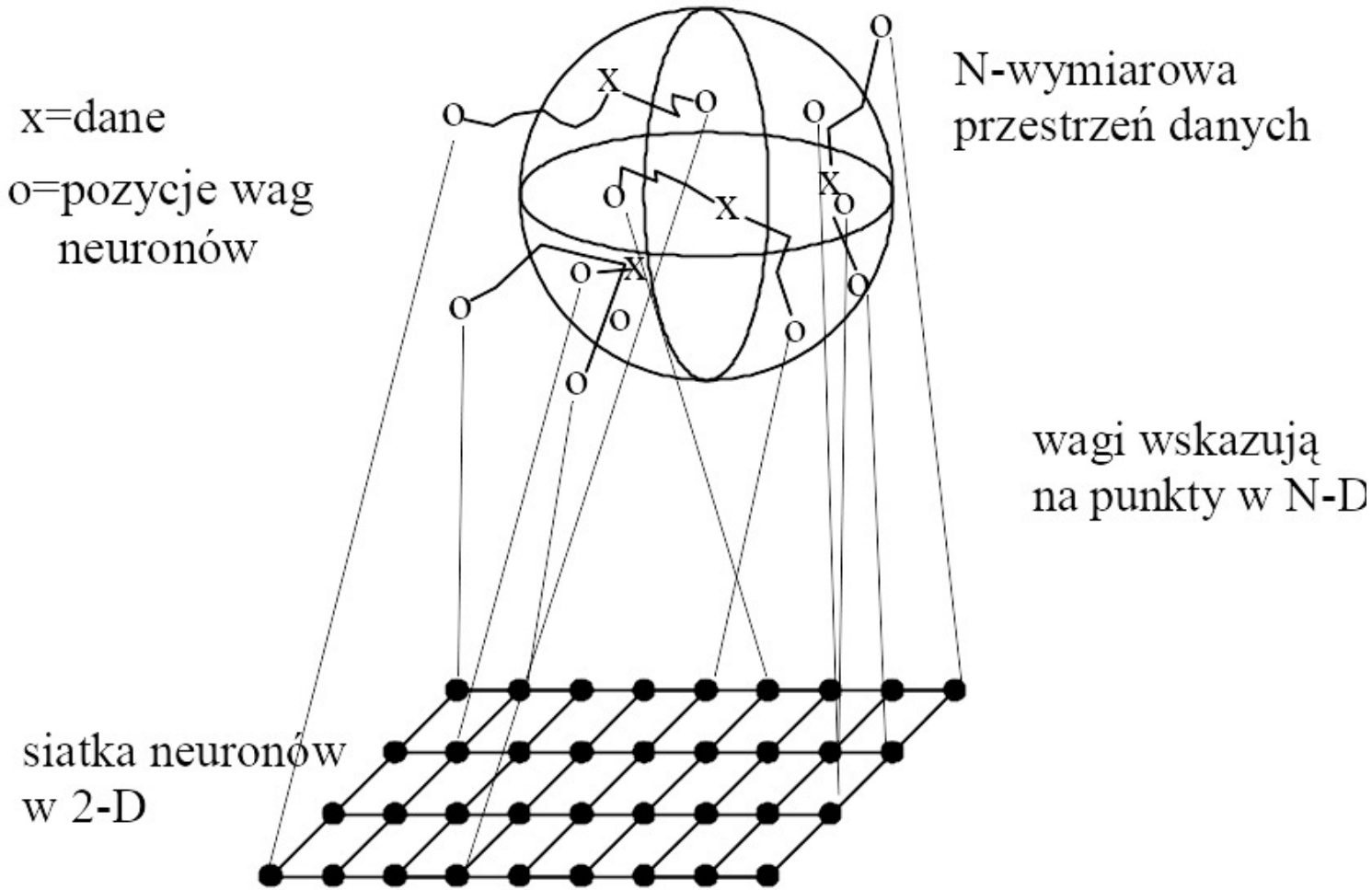


Rozkład jednostajny w kwadracie. SOM uczy się jednorodnego rozkładu. Początkowo wszystkie $\mathbf{w} \approx 0$

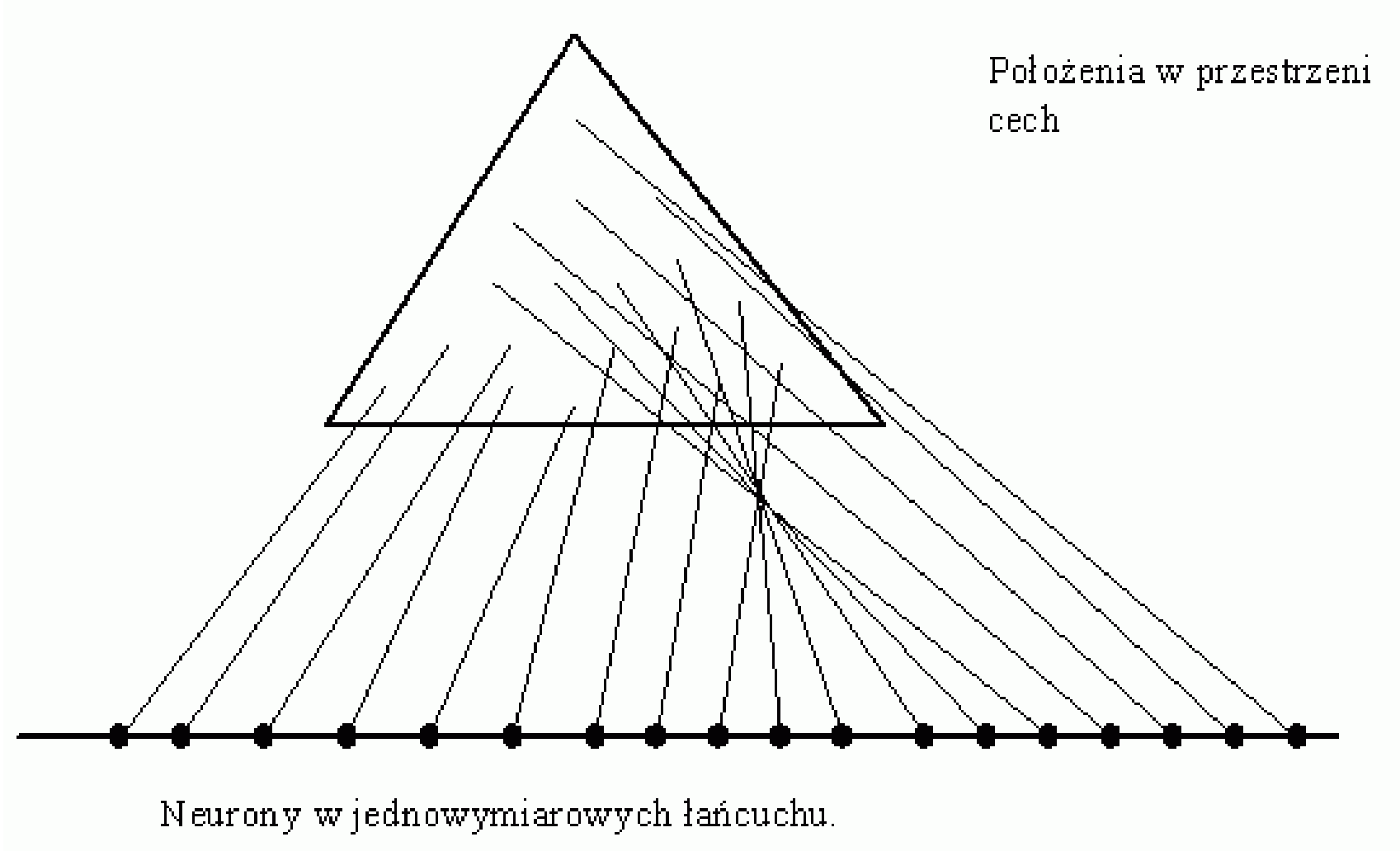
Sieć 2D, cechy 3D (mapowanie 3D->2D)



Uczenie sieci 2D

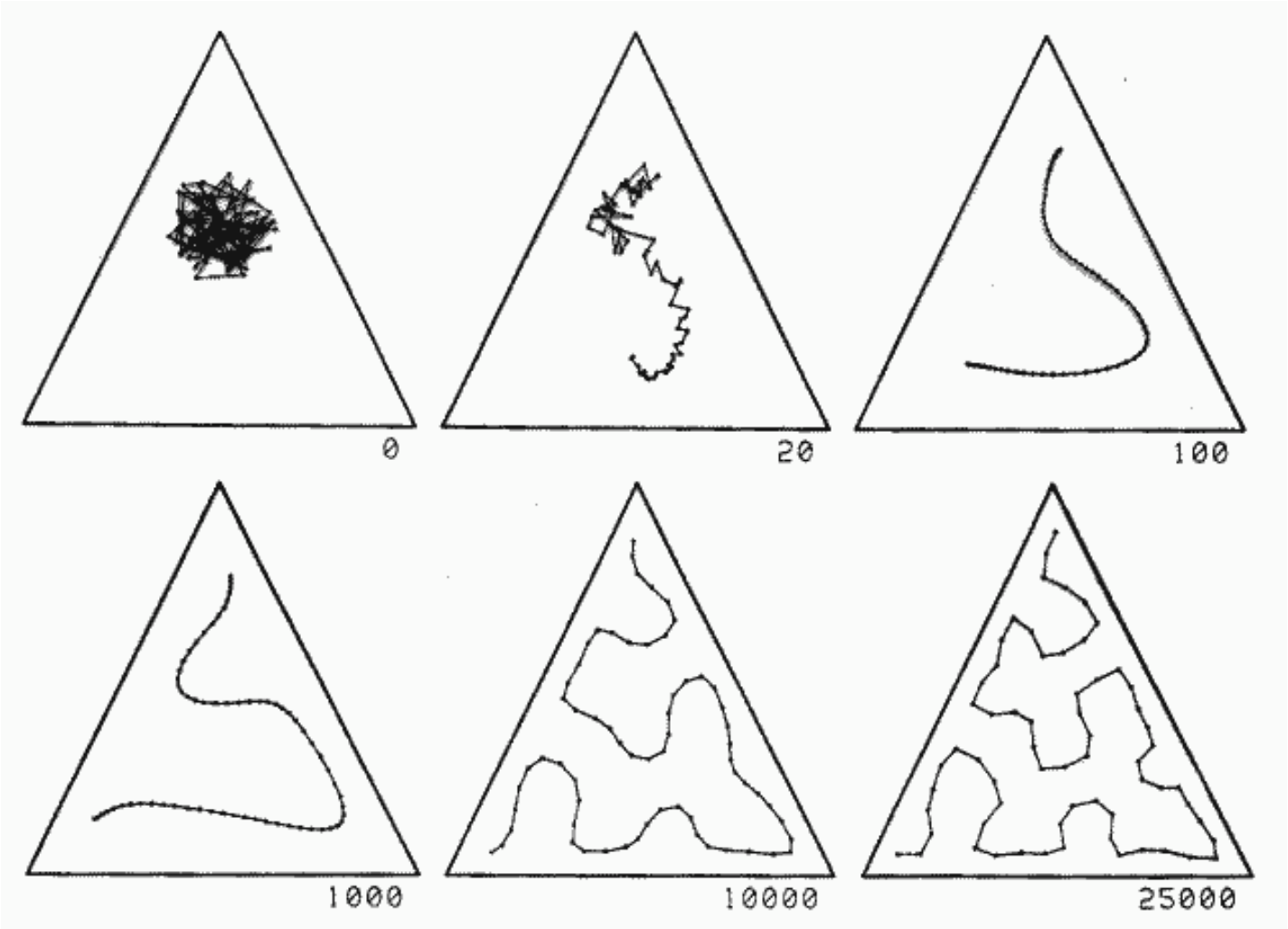


Sieć mapująca dane 2D do 1D (neurony ustawione w linii)

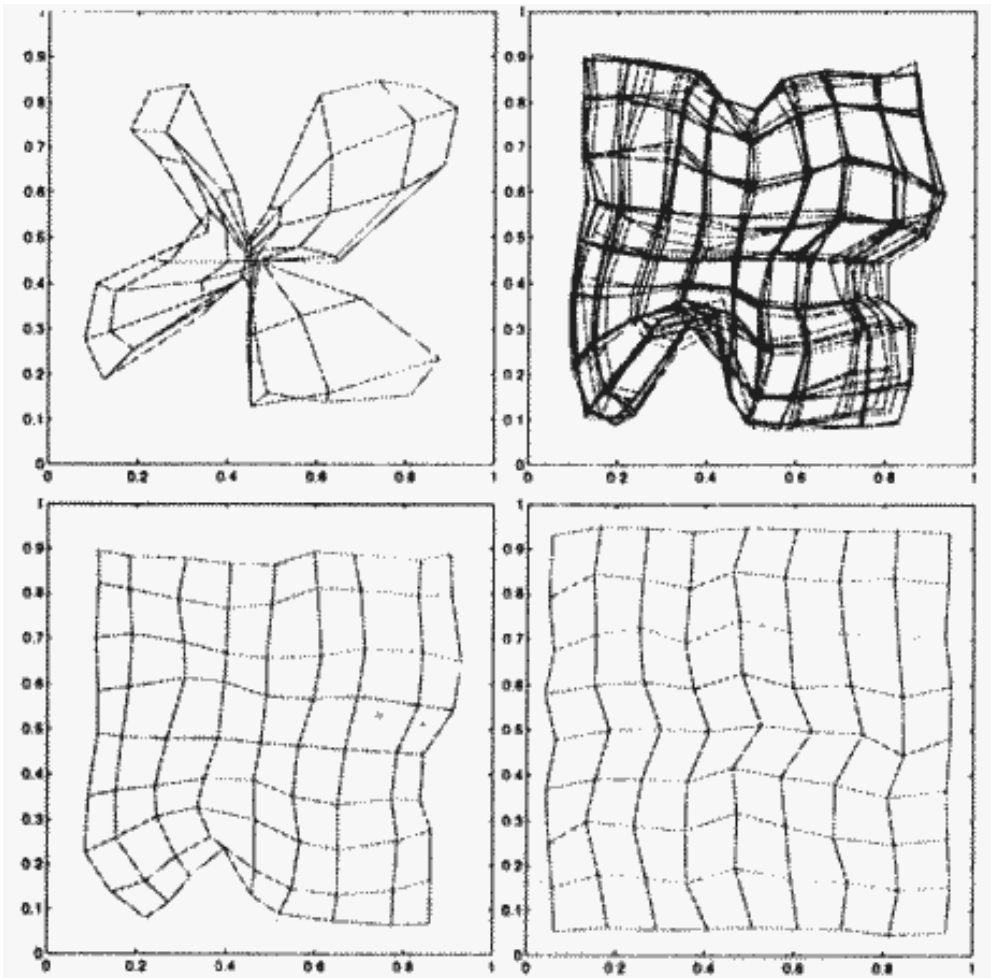


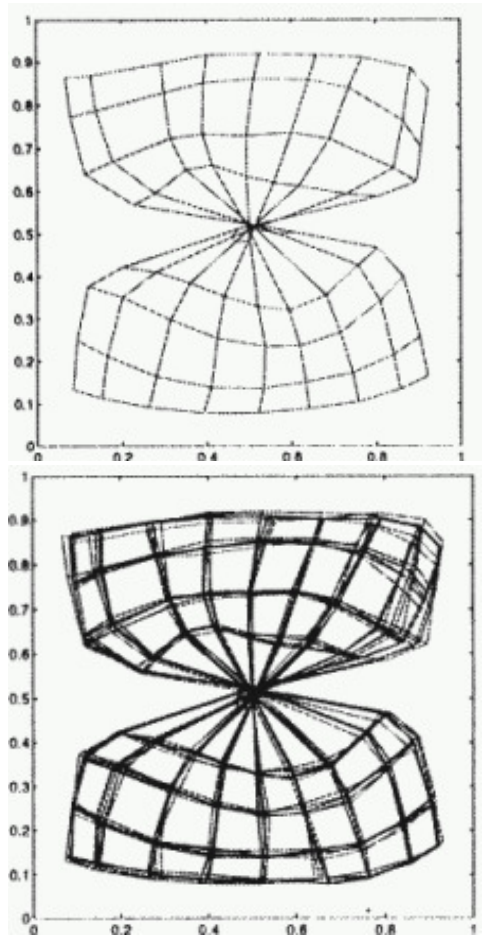
Przykład: fraktalne krzywe Peano

Krzywa Peano (lub Hilberta) - krzywa wypełniająca płaszczyznę



Zniekształcenia





Niektóre zniekształcenia nie mają szans na zniwelowanie

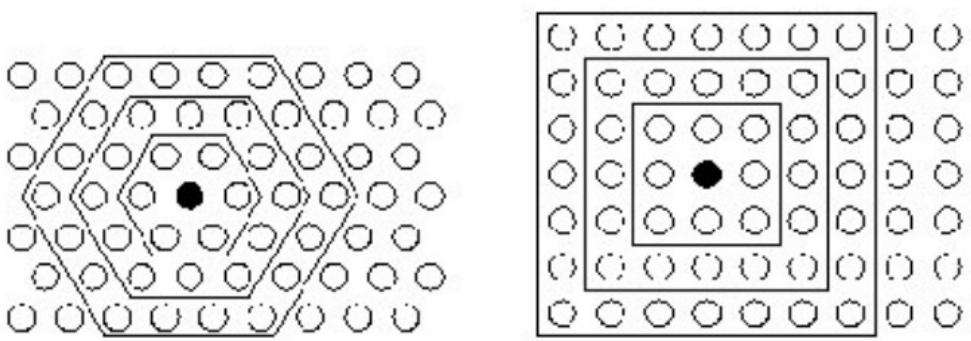
Rozmiar sąsiedztwa

Klasyczny algorytm Kohonena - sąsiedztwo prostokątne

$$h(\mathbf{r}_i, \mathbf{r}, t) = \begin{cases} 1 & \text{dla} \quad d(\mathbf{r}_i, \mathbf{r}) \leq \sigma(t) \\ 0 & \text{dla pozostałych} \end{cases}$$

Promień sąsiedztwa $\sigma(t)$ jest zmniejszany w czasie treningu, może być określony miarą odległości lub liczbą neuronów sąsiadujących

- w początkowej fazie uczenia uwzględnia duże sąsiedztwo
- w końcowej fazie tylko neuron zwycięski jest aktualizowany



(a) Hexagonal grid (b) Rectangular grid

Funkcja sąsiedztwa SOM

Dla gausowskiej funkcji sąsiedztwa

$$h\left(\mathbf{r}_i, \mathbf{r}_j, t\right) = \exp\left(-\frac{\left|\mathbf{r}_i - \mathbf{r}_j\right|^2}{2\sigma(t)^2}\right)$$

zasięg funkcji maleje w czasie treningu

$$\sigma(t) = \sigma_0 \mathrm{e}^{-2\sigma_0 t / t_{\max}}$$

Dla $\sigma = 0$ funkcja sąsiedztwa obejmuje tylko wektor zwycięzcę i SOM sprowadza się do metody kwantyzacji wektorowej (np. on-line k-średnich)

Funkcja błędu SOM

Próba wprowadzenia funkcji błędu (Luttrell; Heskes i Kappen)

Błąd lokalny neuronu i jest sumą po wszystkich neuronach:

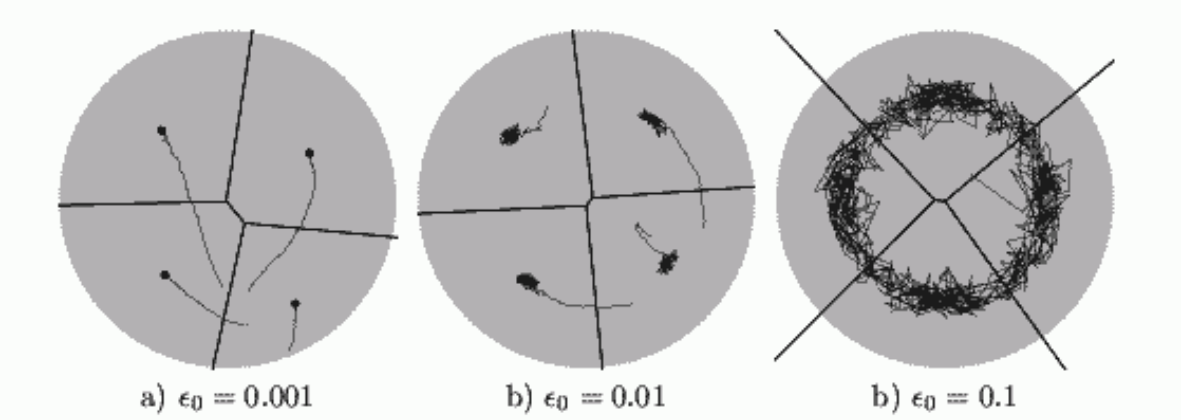
$$E_i(\mathbf{x}; \mathbf{w}, t) = \frac{1}{2} \sum_j h\left(\mathbf{r}_i, \mathbf{r}_j, t\right) \left\|\mathbf{x}(t) - \mathbf{w}_j(t)\right\|^2$$

Neuron-zwycięzca ma najmniejszy błąd lokalny:

$$c = \arg\min_i \sum_j h\left(\left|\mathbf{r}_i - \mathbf{r}_j\right|, t\right) \left\|\mathbf{x}(t) - \mathbf{w}_j(t)\right\|^2$$

Stała uczenia

Duża stała uczenia prowadzi do eksploracji znacznej części przestrzeni



Stała uczenia maleje w trakcie uczenia

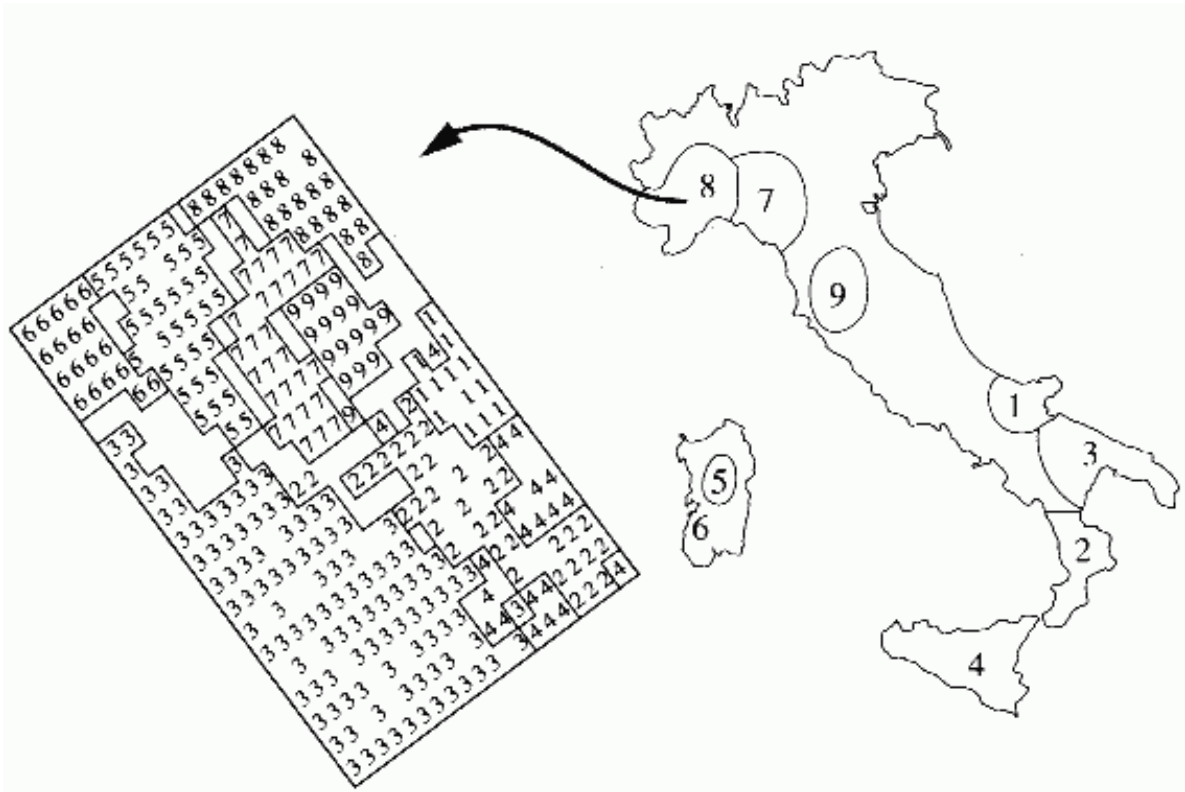
$$\eta(t) = \eta_0 e^{-\frac{t}{t_{max}}}$$

Własności SOM

- sąsiednie neurony kodują sąsiednie obszary, ale sąsiednie obszary mogą być kodowane przez odległe neurony, relacje odległości nie zawsze są zachowane
- powstają „skręcone" konfiguracje przy zbyt szybkiej redukcji sąsiedztwa
- powolna zbieżność algorytmu SOM, wymaga wielu iteracji
- problemy związane z ustaleniem zbieżności i punktów stacjonarnych, wyniki analityczne znane tylko w 1D dla ciągłego czasu: wtedy wartości wag wzdłuż prostej porządkują się
- w oryginalnym SOM nie ma funkcji błędu, brakuje oceny pozwalającej na redukcję zniekształceń, brak szukanego minimum i nie można wyznaczyć gradientu
- zwykle niska jakość klasyfikacji, SOM służy głównie do wizualizacji ... ale nie ma pewności, że wizualizacja oddaje pełną naturę danych

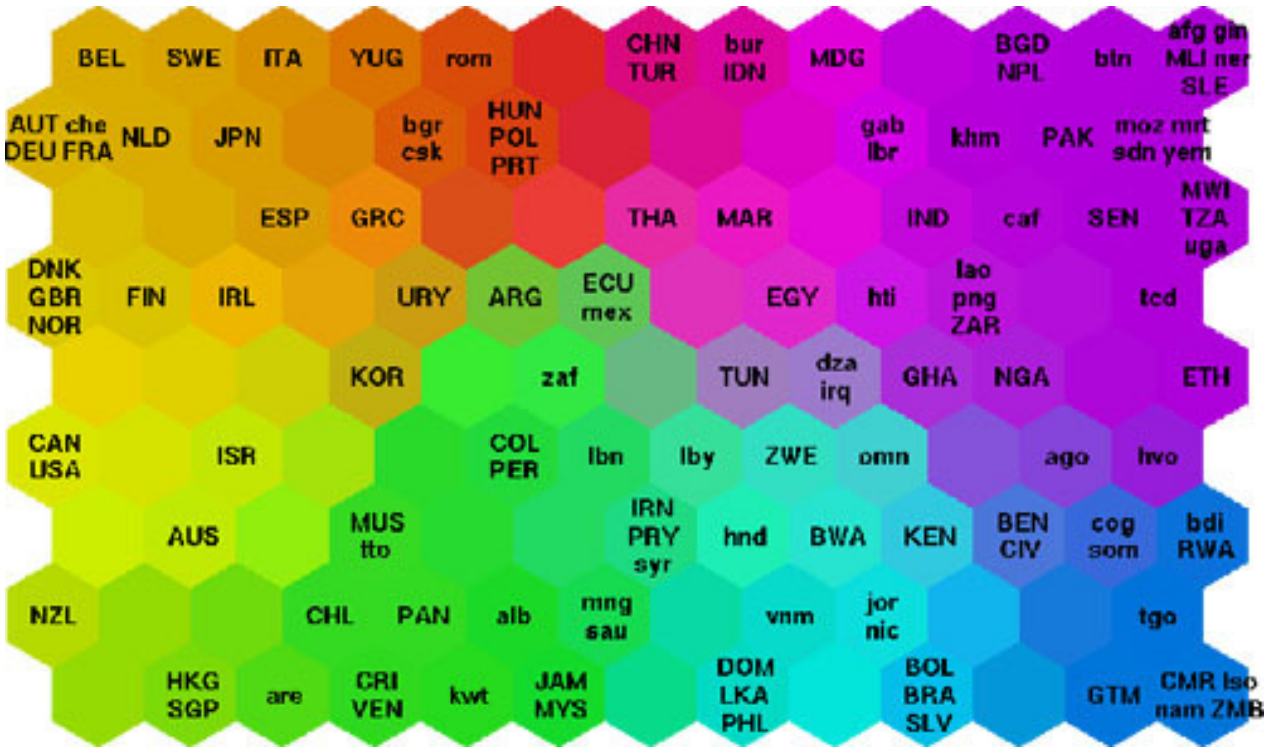
Przykład: Włoska oliwa

Dane: 572 próbki oliwy z 9 prowincji Włoch
Cechy: poziom 8 tłuszczy w każdej próbce.
Mapa SOM: 20 x 20, Redukcja 8D => 2D.
Dokładność klasyfikacji około 95-97%.
Topograficzne relacje zostały zachowane.



Przykład: World powerty map

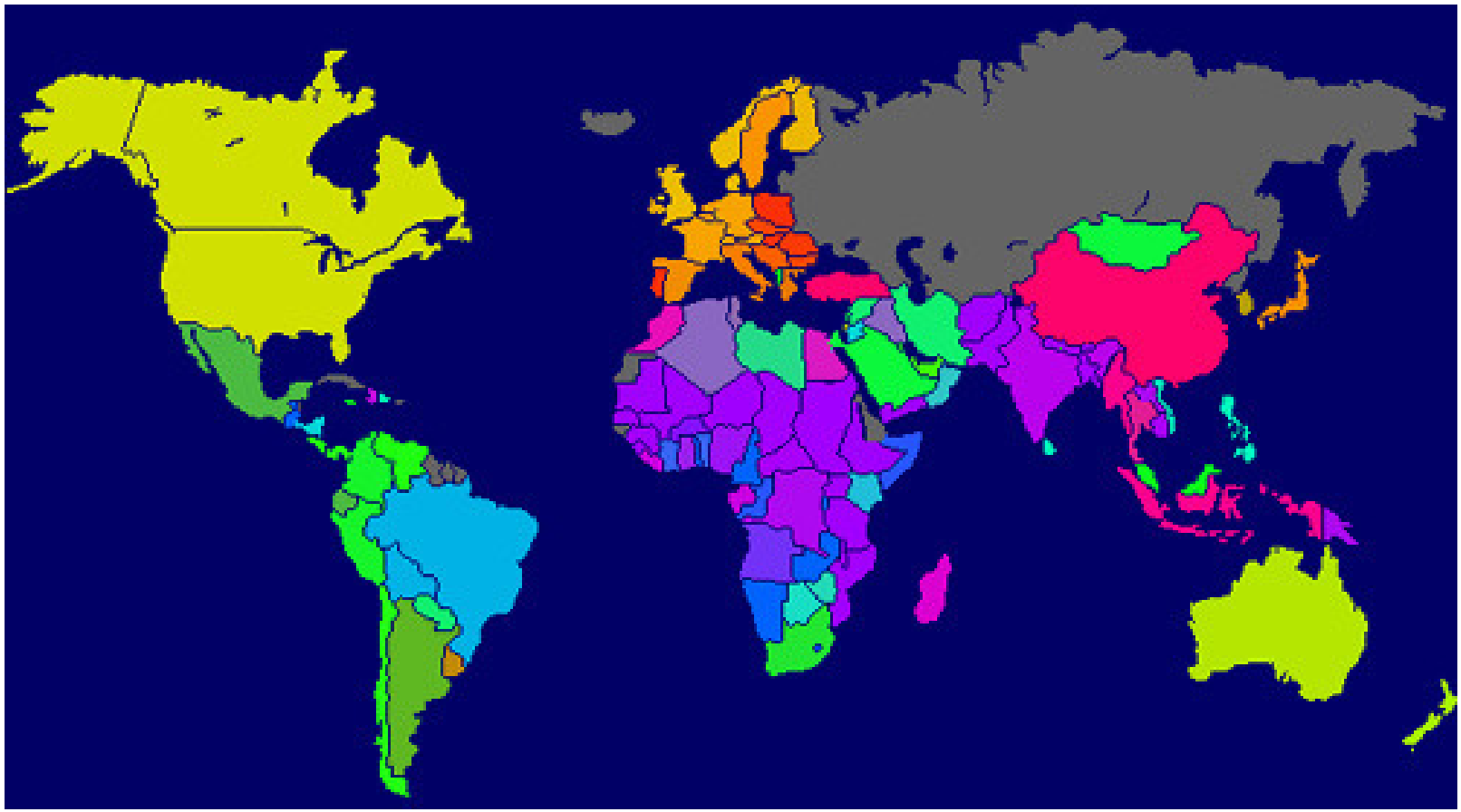
Dane: WorldBank data 1992 rok, 39 współczynników opisujących jakość życia w kraju (poziom służby zdrowia, edukacji, etc.)



Kraje o podobnej wartości poziomu życia pobudzają bliskie grupy neuronów, od krajów o najwyżym wsp. jakości życia (żółty) do najniższego (niebieski)

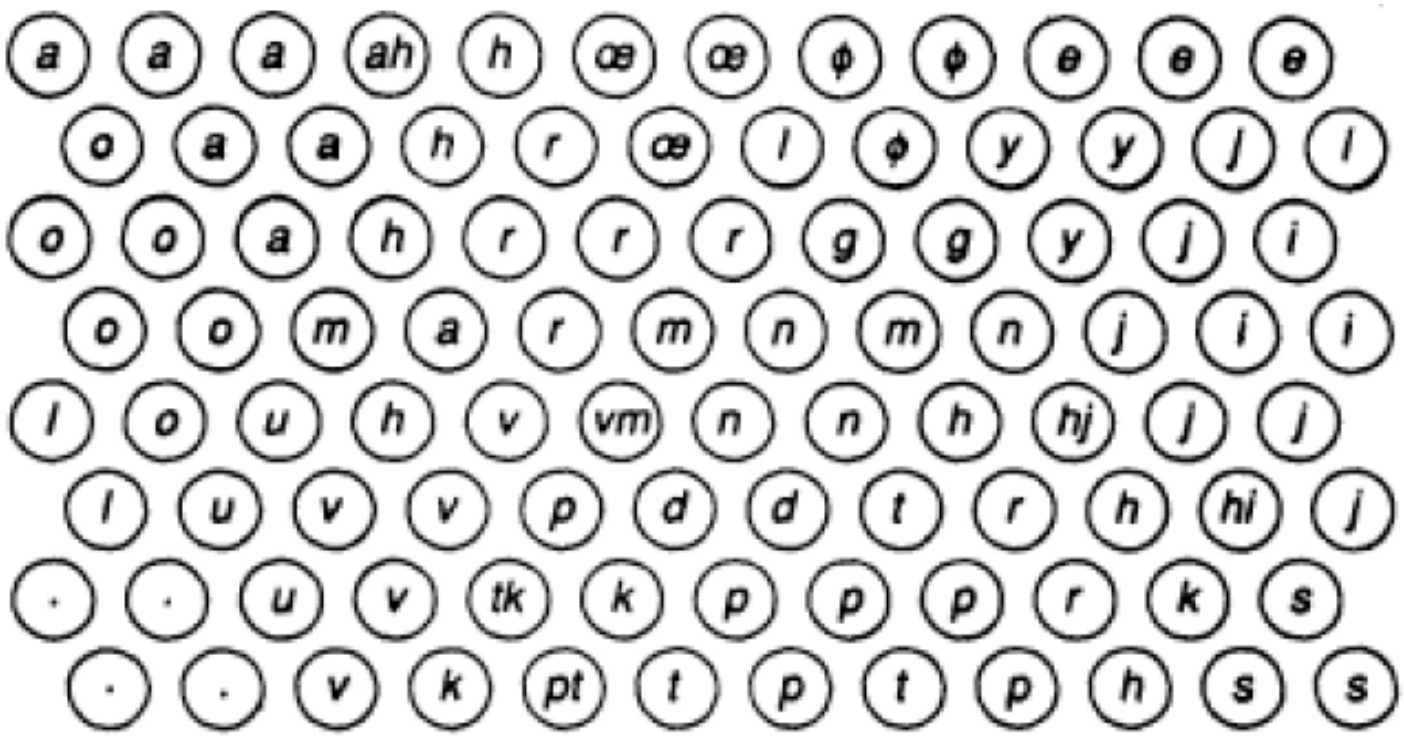
Przykład: Wodr powerty map

Po naniesieniu kolorów na mapę świata



Klawiatura fonetyczna

Dane: 10 ms próbki FFT mowy



Mapa fonetyczna dla języka Fińskiego, Kohonen 1980

Demos

- Wizualizacje do wykładu M. Czoków, J. Piersa: ☞Sześcian, ☞Sześcian z funkcją gaussowską, ☞Kula, ☞Kula z funkcją gaussowską
- ☞SOM 2D dla pogrupowanych danych
- Self-organizing Maps: PyMVPA kolory na mapie 2D
- ☞Demo GNG
- ☞Klasyfikacja kolorów
- ☞GeoSOM Environmental modelling

Przykłady zastosowania SOM

Helsinki University of Technology web site `http://www.cis.hut.fi/research/refs/` has a list (2010) of $>$ 7700 papers on SOM and its applications !

- Badania mózgu: modelowanie topograficznych map w różnych obszarach kory (motoryczna, sensoryczna, wizyjna)
- Robotyka i AI: analiza danych z czujników, sterowanie ruchami robota, mapy orientacji przestrzennej
- Kategoryzacja dokumentów
- Analiza skupień genów, białek, cząsteczek chemicznych, fonemów, dźwięków zwierząt, obiektów astronomicznych, dane ekonomiczne i finansowe
- Kompresja danych (obraz, dźwięk), filtrowanie informacji
- Medical and technical diagnostics.

- Analiza języka naturalnego: linguistic analysis, parsing, learning languages, hyphenation patterns
- Problemy optymaliacyjne: configuration of telephone connections, VLSI design, time series prediction, scheduling algorithms.
- Przetwarzanie sygnałów: adaptive filters, real-time signal analysis, radar, sonar seismic, USG, EKG, EEG and other medical signals ...
- Rozpoznawanie i analiza obrazów: segmentation, object recognition, texture recognition ...
- Content-based retrieval: examples of WebSOM, PicSom – similarity based image retrieval, RGB and textures.
- Viscovery SOMine, komercyjny program do wizualizacji, eksploracji, klasyfikacji, prognozowania oparty na SOM.