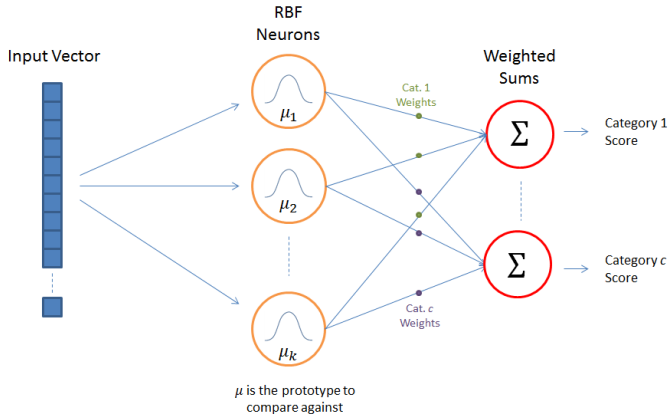


RBF Radial Basis Functions

Sieci neuronowe z radialnymi
funkcjami bazowymi

Radial Basis Function Network



Rys: Chris McCormick, Radial Basis Function Network (RBFN) Tutorial

Sieć RBF

Funkcja realizowana przez sieć RBF z jednym wyjściem:

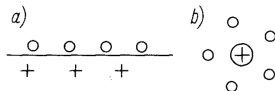
$$f(\mathbf{x}) = \sum_{i=1}^K w_i \varphi(\|\mathbf{x} - \mathbf{c}_i\|) + w_0$$

gdzie K - liczba funkcji bazowych z centrami w $\mathbf{c}_i$

- jedna warstwa ukryta, neurony z radialnymi funkcjami
- wagi połączeń neuronów ukrytych definiują centrum $\mathbf{c}_i$ funkcji radialnej
- funkcja bazowa (radialna) φ określa podobieństwo przypadku $\mathbf{x}$ do centrum (prototypu) $\mathbf{c}_i$
- wyjścia liniowe - ważona suma pobudzeń neuronów radialnych

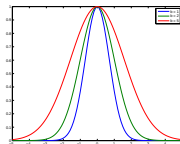
Funkcje radialne

- funkcje radialne mapują relacje lokalne, gdzie perceptrony (np. neurony sigmoidalne) mapują relacje globalne



- argumentem funkcji radialnej jest odległość od pewnego centrum (od prototypu) $\|\mathbf{x} - \mathbf{c}\|$. Zazwyczaj stosuje się metrykę euklidesową.
- najczęściej stosowane funkcje radialne $\varphi(x)$ posiadają maksimum w $x = 0$ i zanikają w obszarach dalekich od $x = 0$, np. funkcja Gaussa

$$\varphi(x) = e^{-\frac{x^2}{\sigma}}$$



- współczynnik dyspersji σ decyduje o zasięgu funkcji bazowej

Przykłady funkcji radialnych

Niech $r = \|\mathbf{x} - \mathbf{c}_i\|$

$$h(r) = r$$

liniowa

$$h(r) = e^{-\left(\frac{r}{\sigma}\right)^2}$$

gaussowska

$$h(r) = (\sigma^2 + r^2)^\beta, \quad 0 < \beta < 1$$

multiquadratic (wielokwadratowa)

$$h(r) = (\sigma^2 + r^2)^{-\alpha}, \quad \alpha > 0$$

inverse multiquadratic

$$h(r) = (\sigma r^2) \ln(\sigma r)$$

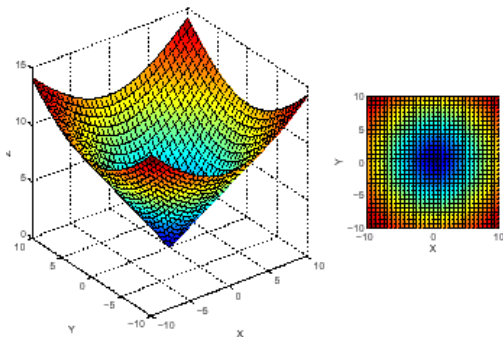
thin-plate spline (cienkiej płytki)

$$h(r) = \sigma^2 + r^2$$

wielomianowa

Funkcja liniowa współrzędnej radialnej

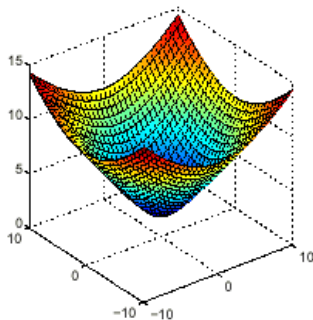
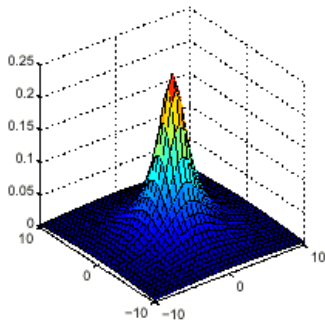
$$h_i(r) = r = \|\mathbf{x} - \mathbf{c}_i\|$$



Funkcje wielokwadratowe

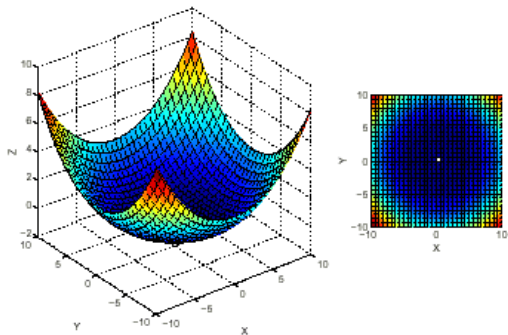
$$h(r) = (\sigma^2 + r^2)^{-\alpha}, \quad \alpha = 1$$

$$h(r) = (\sigma^2 + r^2)^{\beta}, \quad \beta = 0.5$$



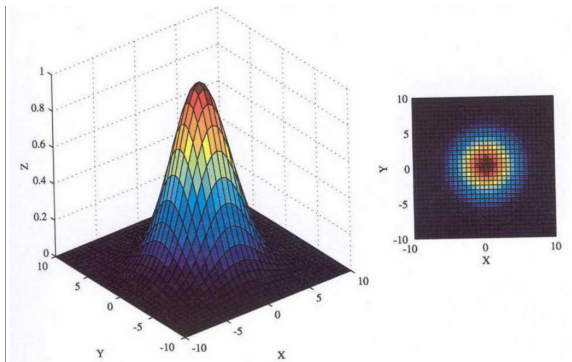
Funkcja cienkiej płytki

$$h(r) = (\sigma r^2) \ln(\sigma r)$$



Funkcja Gaussa

$$h(r) = e^{-\frac{r^2}{2\sigma^2}}$$

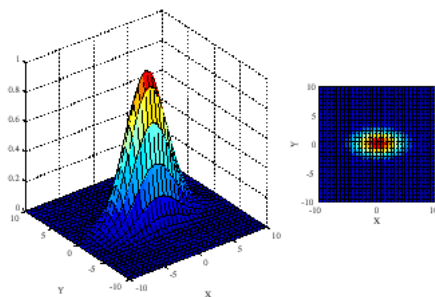


Najczęściej używana w roli funkcji bazowej RBF

Funkcja Gaussa z macierzą dyspersji

$$h(r) = e^{-\frac{r^2}{2\sigma^2}} = e^{-\frac{1}{2}(\mathbf{x}-\mathbf{c})^T \Sigma^{-1}(\mathbf{x}-\mathbf{c})}$$

Funkcja Gaussa wielu zmiennych



Macierz Σ określa rozmycia i rotacje funkcji w wielowymiarowej przestrzeni

Uczenie RBF jako problem aproksymacji

Dla N punktów $\mathbf{x}_i, y_i$ znajdź funkcję spełniającą

$$f(\mathbf{x}_i) = y_i \quad i = 1, \dots, N$$

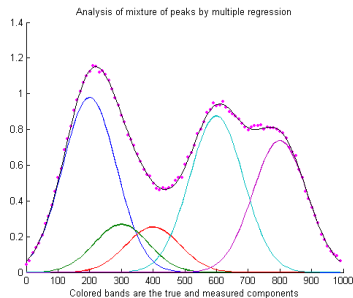
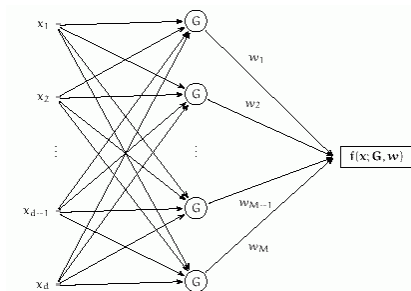
Funkcja sieci RBF

$$f(\mathbf{x}_i) = \sum_{j=1}^K w_j \varphi(\|\mathbf{x}_i - \mathbf{c}_j\|)$$

Szukamy minimum funkcji błędu MSE

$$E = \sum_{i=1}^N (y_i - f(\mathbf{x}_i))^2$$

RBF z jedną warstwą ukrytą i linowym wyjściem jest uniwersalnym aproksymatorem



Rys: <https://terpconnect.umd.edu/~toh/spectrum/CurveFittingB.html>

Rozwiązanie RBF

Dla $K = N$ gdzie centra funkcji bazowych są ustalone w punktach treningowych $\mathbf{x}_i = \mathbf{c}_i$

$$\begin{pmatrix} h_{1,1} & h_{1,2} & \cdots & h_{1,N} \\ h_{2,1} & h_{2,2} & \cdots & h_{2,N} \\ \vdots & \vdots & \ddots & \vdots \\ h_{N,1} & h_{N,2} & \cdots & h_{N,N} \end{pmatrix} \begin{pmatrix} w_1 \\ w_2 \\ \vdots \\ w_N \end{pmatrix} = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{pmatrix}$$

$$h_{ij} = \varphi(\|\mathbf{x}_i - \mathbf{x}_j\|)$$

$$\mathbf{H}\mathbf{w} = \mathbf{y}$$

dla $\mathbf{x}_1 \neq \mathbf{x}_2 \neq \dots \neq \mathbf{x}_N$ macierz $\mathbf{H}$ jest nieosobliwa i dodatnio określona, więc istnieje

$$\mathbf{w} = \mathbf{H}^{-1}\mathbf{y}$$

RBF jako problem aproksymacji

- dla radialnych funkcji bazowych macierz interpolacji $\mathbf{H}$ dodatnio określona (Light 1992)
- dla wąskich funkcji Gaussa (gdy $\mathbf{H} \approx \mathbf{I}$) istnieje rozwiązanie trywialne $\mathbf{w} = \mathbf{y}$ o niskiej generalizacji
- problem źle określony, przewymiarowanie
- hiperpłaszczyzna interpolacji nie jest gładka
- większe dyspersje σ i mniejsza liczba funkcji ($K < N$)
→ lepsza generalizacja

Aproksymacja RBF

Dla ustalonych $\mathbf{c}_i$ oraz σ_i gdy $K < N$

$$\begin{pmatrix} h_{1,1} & h_{1,2} & \cdots & h_{1,K} \\ h_{2,1} & h_{2,2} & \cdots & h_{2,K} \\ \vdots & \vdots & \ddots & \vdots \\ h_{N,1} & h_{N,2} & \cdots & h_{N,K} \end{pmatrix} \begin{pmatrix} w_1 \\ w_2 \\ \vdots \\ w_K \end{pmatrix} = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_K \end{pmatrix}$$

$$h_{ij} = \varphi(\|\mathbf{x}_i - \mathbf{c}_j\|)$$

$$\mathbf{H}\mathbf{w} = \mathbf{y}$$

rozwiązanie układu równań przez pseudoinwersję

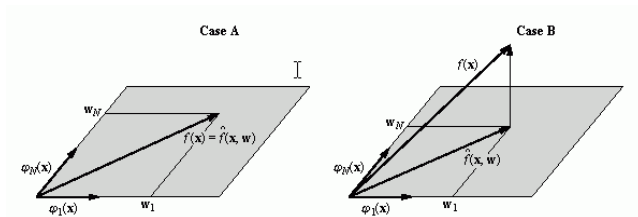
$$\mathbf{w} = \mathbf{H}^+ \mathbf{y}, \quad \text{gdzie} \quad \mathbf{H}^+ = (\mathbf{H}^T \mathbf{H})^{-1} \mathbf{H}^T$$

W praktyce pseudoinwersję realizuje się za pomocą dekompozycji SVD

Interpretacja geometryczna

Jeśli prawdziwa aproksymowana funkcja $f(\mathbf{x})$ leży w przestrzeni rozpiętej przez wektory bazowe $\Phi(\mathbf{x})$ to możliwe jest rozwiązanie bez błędu, w przeciwnym razie aproksymowana jest projekcja ortogonalna (błąd jest ortogonalny do p-ni bazowej).

$$\hat{f}(\mathbf{x}; \mathbf{w}) = \sum_i w_i \varphi_i(\mathbf{x})$$



Separacja w wielowymiarowej przestrzeni

Twierdzenie (Cover 1965):

Jeśli przekształcić wzorce $\mathbb{X} = \{\mathbf{x}_i\}, i = 1, \dots, N$, nieliniową funkcją na wektory $\Phi(\mathbf{x}_i) = [\varphi_1(\mathbf{x}_i), \varphi_2(\mathbf{x}_i), \dots, \varphi_K(\mathbf{x}_i)]^T$ gdzie $K > N$ to rośnie prawdopodobieństwo liniowej separacji, tj. istnieje płaszczyzna

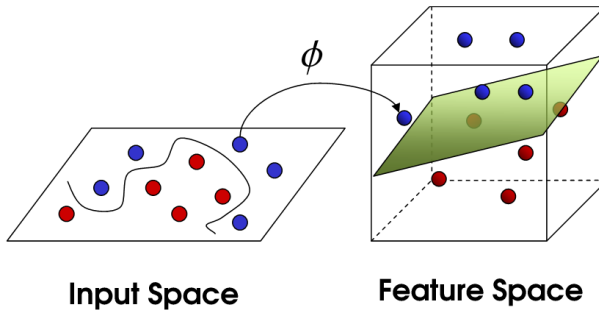
$$\begin{aligned}\mathbf{w}^T \Phi(\mathbf{x}_i) &\geq 0 & \text{dla } \mathbf{x}_i \in \mathbb{C}_1 \\ \mathbf{w}^T \Phi(\mathbf{x}_i) &< 0 & \text{dla } \mathbf{x}_i \in \mathbb{C}_2\end{aligned}$$

granica decyzyjna zdefiniowana jest równaniem

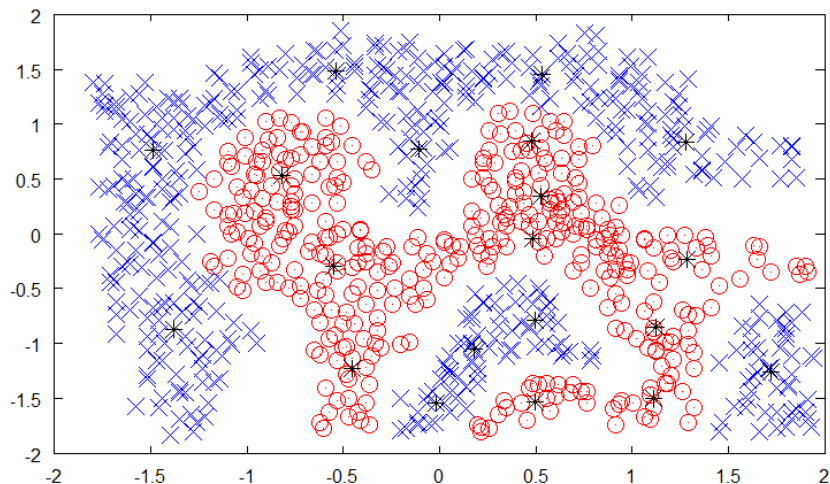
$$\mathbf{w}^T \Phi(\mathbf{x}) = 0$$

Przy dostatecznie dużej wymiarowości przestrzeni do której rzutujemy wzorce prawdopodobieństwo separowalności liniowej rośnie do 1.

Projekcja do przestrzeni cech



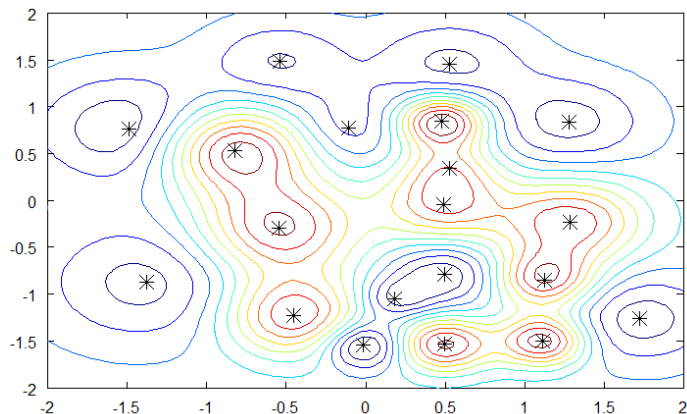
Przykład: 2 klasy



Rys: Chris McCormick, Radial Basis Function Network (RBFN) Tutorial

Przykład: 2 klasy

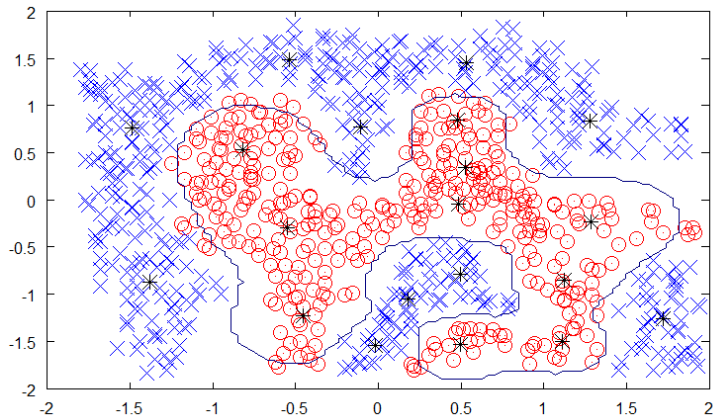
Wartości dla klasy 1, najczęściej wagi powiązane z funkcjami radialnymi dla klasy 1 mają wartości dodatnie a dla klasy 2 - ujemne



Rys: Chris McCormick, Radial Basis Function Network (RBFN) Tutorial

Przykład: 2 klasy

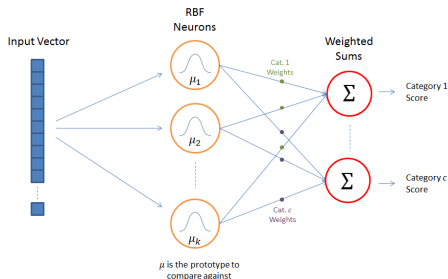
Granica decyzyjna



Rys: Chris McCormick, Radial Basis Function Network (RBFN) Tutorial

Sieć RBF

- jedna warstwa ukryta z radialnymi funkcjami + wyjścia liniowe
- parametry: wagi $\mathbf{w}$, centra $\mathbf{c}_i$, dyspersja σ_i (lub pełna macierz Σ), liczba funkcji radialnych K



Rys: Chris McCormick, Radial Basis Function Network (RBFN) Tutorial

Trening RBF

Różne strategie uczenia:

- Trening 2 etapowy:
 - najpierw ustawienie funkcji bazowych (centra i dyspersje)
 - potem rozwiązanie układu równań $\mathbf{HW} = \mathbf{Y}$
- Możliwe jednoczesne uczenie centrów, dyspersji i wag, np. za pomocą metod gradientowych (wsteczna propagacja)
- Inicjalizacja początkowych położeń centrów: losowa, klasteryzacja, samoorganizacja, probabilistyczna
- Inicjalizacja dyspersji: wartość stała, dobór zależnie od gęstości danych w różnych rejonach, średnie odległości od wektorów z innych klas, ...

Ustalenie centrów - różne strategie

- losowe równomierne pokrycie przestrzeni wejściowej - może nie oddawać specyfiki problemu
- w problemie aproksymacji ustawiamy centra w miejscach minimum i maksimum a następnie usuwamy wzorce z otoczenia tych centrów a resztę rozmieszczamy równomiernie wśród pozostałych wzorców
- dla klasyfikacji ustalamy centra funkcji bazowych w pobliżu granic decyzyjnych lub wewnątrz skupisk wektorów należących do wspólnej klasy
- wybieramy losowy podzbiór K wzorców treningowych - prosta ale skuteczna metoda
- ustalenie centrów w procesie uczenia nadzorowanego lub nienadzorowanego: klasteryzacja, samoorganizacja, LVQ, ...

Ustalenie dyspersji - różne strategie

- wymagane gładkie odwzorowanie, rozmycie pełni rolę regularyzacji
- „pola recepcyjne” wszystkich funkcji bazowych powinny pokrywać cały obszar danych wejściowych
- pola recepcyjne sąsiadujących funkcji bazowych powinny nakrywać się w niewielkim stopniu
- rozmycie σ jednakowe dla wszystkich funkcji np. (Haykin, 1994):

$$\sigma = \frac{d}{\sqrt{2K}}$$

gdzie K ilość funkcji bazowych, d maksymalna odległość pomiędzy centrami. Funkcja Gaussa przybiera wówczas postać:

$$\varphi(\|\mathbf{x} - \mathbf{c}_i\|) = e^{-\frac{K\|\mathbf{x} - \mathbf{c}_i\|^2}{d^2}}$$

- rozmycie równe średniej odległości od sąsiadujących centrów

Ustawienie dyspersji - różne strategie

- średnia odległość euklidesowa wektorów treningowych od najbliższego centrum

$$\sigma_k = \frac{1}{m} \sum_{i=1}^m \|\mathbf{x}_i - \mathbf{c}_k\|$$

gdzie m to liczba wektorów $\mathbf{x}_i$ przypisanych do centrum $\mathbf{c}_k$.

- σ_i równe odległości euklidesowej od najbliższego sąsiedniego centrum klastra (Tarasenko, 1994)
- uwzględniając odległość P najbliższych położonych centrów sąsiadujących funkcji radialnych (gdzie zwykle $P \leq 3$) (Moody, 1989)

$$\sigma_i = \sqrt{\frac{1}{P} \sum_{i=1}^P \|\mathbf{c}_i - \mathbf{c}_k\|^2}$$

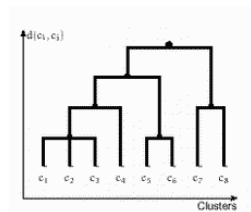
Incjalizacja RBF: klasteryzacja

Inicjalizacja centrów funkcji radialnych w centrach skupisk (klastrow) uzyskanych za pomocą algorytmów klasteryzacji (analizy skupień).

Przykład: metoda dendrogramów, algorytm k -średnich

Metoda dendrogramów:

1. przypisz każdy x_i do odrębnego klastra
2. połącz najbliższą parę klastrów x_i i x_j w jeden klaster
3. powtarzaj punkt 2 aż do uzyskania zadowalającej liczby klastrów lub gdy najmniejsza odległość między klastrami przekroczy ustalony poziom



Odległość między klastrami może być definiowana na wiele sposobów, np. minimalna odległość między obserwacjami z różnych klastrów

Incjalizacja RBF: klasteryzacja

Algorytm k -średnich

1. Rozmieść k centrów $\mathbf{c}_i$ równomiernie w przestrzeni wejściowej lub losowo wybierając k obserwacji ze zbioru treningowego
2. Dla każdego centrum $\mathbf{c}_i$ znajdź zbiór wszystkich punktów $\mathbf{x}_j$ leżących najbliżej tego centrum
3. Aktualizuj położenie centrum $\mathbf{c}_i$ ustawiając je w punkcie wyznaczonym przez średnią położenia punktów przypisanych do danego centrum

$$\mathbf{c}_i(k+1) = \frac{1}{N_i} \sum_{j=1}^{N_i} \bar{\mathbf{x}}_j(k)$$

4. Powtarzaj dwa ostatnie kroki aż do uzyskania zbieżności

👉 Clustering Demo

Inicjalizacja RBF: samoorganizacja

Metoda on-line k -średnich: inicjalizacja centrów funkcji radialnych przez samoorganizację metodami uczenia konkurencyjnego

1. Wybierz losowo k wektorów $\mathbf{c}_i$
2. Dla przypadku $\mathbf{x}$ ze zbioru uczącego znajdź najbliższe centrum $\mathbf{c}_j$ (zwycięzcę)
3. Zmień położenie centrum $\mathbf{c}_j$ zwycięzcy przesuwając je w kierunku punktu $\mathbf{x}$

$$\mathbf{c}_i(t+1) = \mathbf{c}_i(t) + \eta_t(\mathbf{x} - \mathbf{c}_i(t))$$

4. Powtarzaj kroki 2 i 3 określoną liczbę razy lub do uzyskania zbieżności

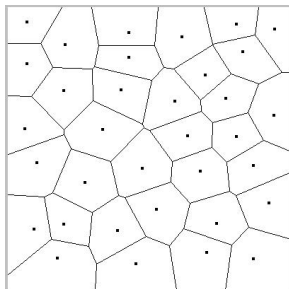
Stała uczenia η_t powinna zanikać ze wzrostem liczby iteracji t

$$\eta_t = \frac{\eta_0}{1 + \frac{t}{T}}$$

gdzie T to maksymalna liczba iteracji a η_0 - początkowa wartość

Diagram Voronoi

W drodze samoorganizacji przestrzeń dzielona jest na obszary Voronoi z centrum funkcji radialnej osadzonym w każdej komórce definiującymi teselację



Trening RBF: metoda probabilistyczna

Zał: rozkład równomierny danych w zbiorze treningowym,
diagonalna macierz skalująca Σ

Funkcja radialna $\varphi_i(\mathbf{x})$ reprezentuje prawdopodobieństwo warunkowe, że $\mathbf{x}$ należy do klastra w centrum $\mathbf{c}_i$

$$\varphi_i(\mathbf{x}) = e^{-\frac{1}{2}(\mathbf{x}-\mathbf{c}_i)^T \Sigma^{-1}(\mathbf{x}-\mathbf{c}_i)}$$

Wszystkie centra $\mathbf{c}_i$ oraz rozmycia Σ_i optymalizowane są równocześnie

$$\mathbf{c}_i(k+1) = \frac{\mathbf{c}_i(k) + \eta_k(\mathbf{x}\varphi_i(\mathbf{x}) - \mathbf{c}_i(k))}{(1 - \eta_k) + \eta_k\varphi_i(\mathbf{x})}$$

$$\Sigma_i(k+1) = \frac{\Sigma_i(k) + \eta_k(\varphi_i(\mathbf{x})[\mathbf{x} - \mathbf{c}_i(k)][\mathbf{x} - \mathbf{c}_i(k)]^T - \Sigma_i(k))}{(1 - \eta_k) + \eta_k\varphi_i(\mathbf{x})}$$

Współczynnik uczenia maleje w czasie $\eta_k = \frac{\eta_0}{k}$, gdzie $\eta_0 \in [0, 1]$

Trening wag $\mathbf{w}_i$ warstwy wyjściowej odbywa się po ustaleniu parametrów funkcji radialnych (np. jak wcześniej z rozwiązywania układu równań)

Uczenie metodami gradientowymi

Minimalizacja funkcji błędu

$$E = \sum_{i=1}^N (y_i - f(\mathbf{x}_i))^2 = \sum_{i=1}^N \left(y_i - \sum_{k=0}^K w_k \varphi_k(\|\mathbf{x}_i - \mathbf{c}_k\|) \right)^2$$

dla funkcji gusowskiej i metryki euklidesowej metodą największego spadku gradientu

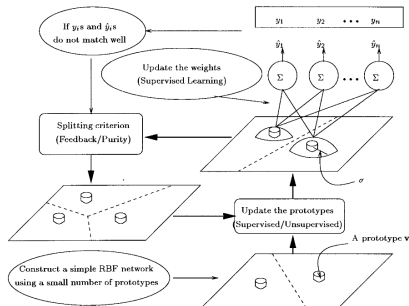
$$\Delta w_i = -\eta \frac{\partial E}{\partial w_i} = -\eta (y - f(\mathbf{x})) \varphi_i(\mathbf{x})$$

$$\Delta c_{ji} = -\eta \frac{\partial E}{\partial c_{ji}} = -\eta (y - f(\mathbf{x})) w_j \varphi_j(\mathbf{x}) \frac{(x_i - c_{ji})}{\sigma_j^2}$$

$$\Delta \sigma_j = -\eta \frac{\partial E}{\partial \sigma_j} = \eta (y - f(\mathbf{x})) w_j \varphi_j(\mathbf{x}) \frac{\|\mathbf{x} - \mathbf{c}_j\|^2}{\sigma_j^3}$$

Rozrastające się architektury RBF

- GAL (*Growing and Learning*)
- GrRBF (*Growing Radial Basis Function*)
- FEN (*Function Estimation Networks*)
- GCS (*Growing Cell Structures*)
- RAN (*Resource Allocation Networks*)



Nicolaos, *Growing Radial Basis Neural Networks: Merging Supervised and Unsupervised Learning with Network Growth Techniques*, 1997

Supervised Growing Cell Structures (Fritzke)

1. Zainicjuj małą sieć RBF
2. Wytrenuj sieć dowolną metodą
3. Oblicz błąd związany z każdym neuronem radialnym e_i akumulując błąd MSE sieci dla najlepiej dopasowanych centrów (zwycięzców położonych najbliżej przypadku uczącego $\mathbf{x}$)
4. Wstaw nową funkcję radialną w obszarze o największym błędzie

$$\mathbf{c}_{new} = \frac{1}{2}(\mathbf{c}_m - \mathbf{c}_n)$$

gdzie $\mathbf{c}_m$ to centrum z największym błędem,
 $\mathbf{c}_n$ to sąsiadujące centrum $\mathbf{c}_m$ (tj. położone w sąsiedniej komórce Voronoi) o największym błędzie

5. Powtarzaj dopóki błąd nie zmaleje do pożądanej wartości

Resource-Allocating Network (RAN)

- RAN (Pratt, 1991) algorytm rozrostu sieci RBF
- Startuje od pustej warstwy ukrytej i dla danego wektora $\mathbf{x}_t$ dodawany jest nowy neuron jeżeli spełnione są kryteria „odkrywczości”

$$\|\mathbf{x}_t - \mathbf{c}_i\| \geq \epsilon(t)$$

$$\|\mathbf{e}(t)\| = \|\mathbf{y}_t - f(\mathbf{x}_t)\| > e_{min}$$

gdzie $\mathbf{c}_i$ to centrum najbliższe $\mathbf{x}_t$

- Wartość progu $\epsilon(t)$ maleje z każdym krokiem

$$\epsilon(t) = \max\{\epsilon_{max}e^{-\frac{t}{T}}, \epsilon_{min}\}$$

- zakładając, że w kroku t istnieje k neuronów, wówczas nowy neuron dodawany jest z parametrami

$$\mathbf{c}_{k+1} = \mathbf{x}_t, \quad w_{k+1,j} = e_j(t), \quad \sigma_{k+1} = \alpha \|\mathbf{x}_t - \mathbf{c}_i\|$$

- gdy kryteria „odkrywczości” nie są spełnione to błąd jest korygowany aktualizacją parametrów (spadkiem gradientu)

Porównanie MLP vs RBF

MLP	RBF
nielocalne relacje, wymagają douczania	lokalne efekty, stabilność, nie-wrażliwość na odstające przypadki
jeden rodzaj parametrów (wagi)	kilka rodzajów parametrów: wagi, centra, rozmycia
trudna inicjalizacja	łatwa inicjalizacja
trudna interpretacja	łatwa interpretacja (prototypy)
uczenie tylko pod nadzorem	możliwe uczenie bez nadzoru
zawsze wie	czasami nie wie (przypadki w obszarach nie objętych polami recepcyjnymi)
BP dość skomplikowane dla wielu warstw	uczenie łatwe bo 1 warstwa
granica decyzji perceptronu: hiperpłaszczyzna	granica decyzyjna funkcji radialnej: hypersfera
aktywacja: ważona suma sygnałów $\mathbf{w}^T \mathbf{x}$	odległość od centrum funkcji bazowej $\ \mathbf{x} - \mathbf{c}\ $