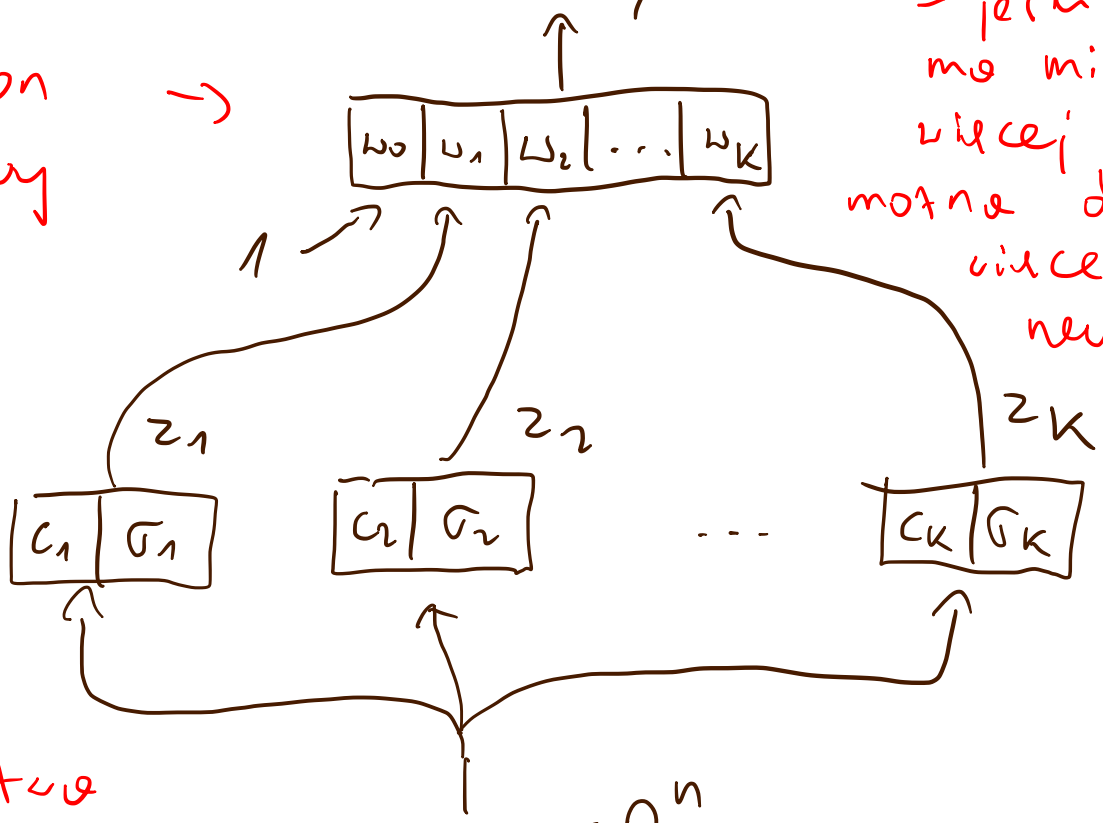
sieć RBF



neuron liniowy →

- jeśli sieć ma mieć więcej wyjść można dodać więcej neuronów liniowych

warstwa neuronów radialnych

$$y = f(x) = w_0 + \sum_{k=1}^{K} w_k \cdot z_k(x)$$ - funkcja, która realizuje sieć

$$z_k(x) = R(\underbrace{\|x - c_k\|}_{d(x, c_k)}, \sigma_k)$$ - odległość x od wzorca c_k

$$R(d, \sigma) = e^{-d^2/2\sigma^2}$$ - przykładowa gaussowska f-cja radialna

$$d(x, c_k) = \sqrt{\sum_{i=1}^{n} (x_i - c_{k,i})^2}$$ - odległość Euklidesowa w przestrzeni R^n

$k = 1, \dots, K$

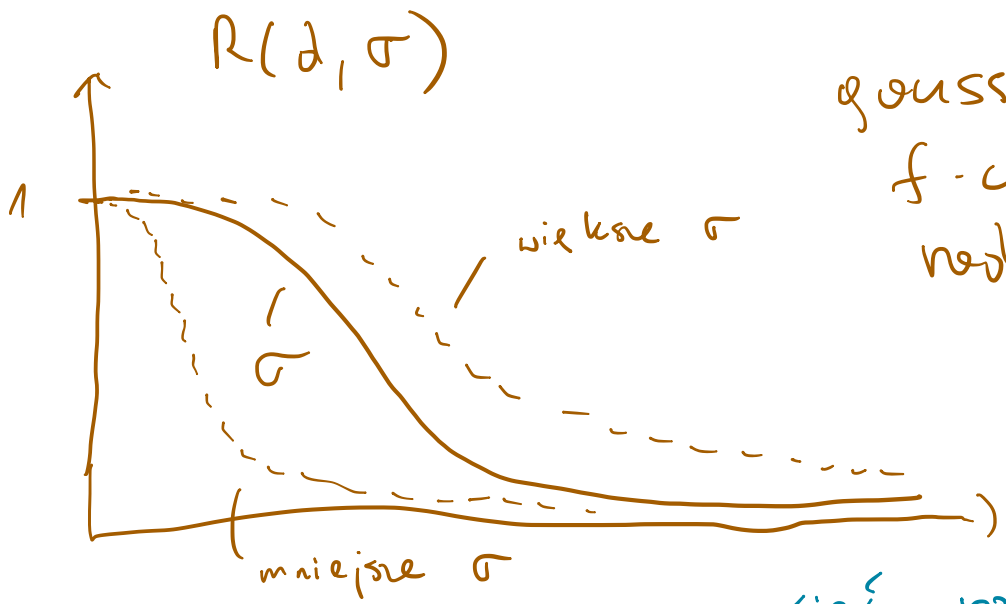
$x \in R^n$ - wejście sieci $\quad k = 1, \ldots, K$

$y \in R$ - wyjście sieci

c_k - wzorzec / centrum ($\in R^n$) neuronu radialnego

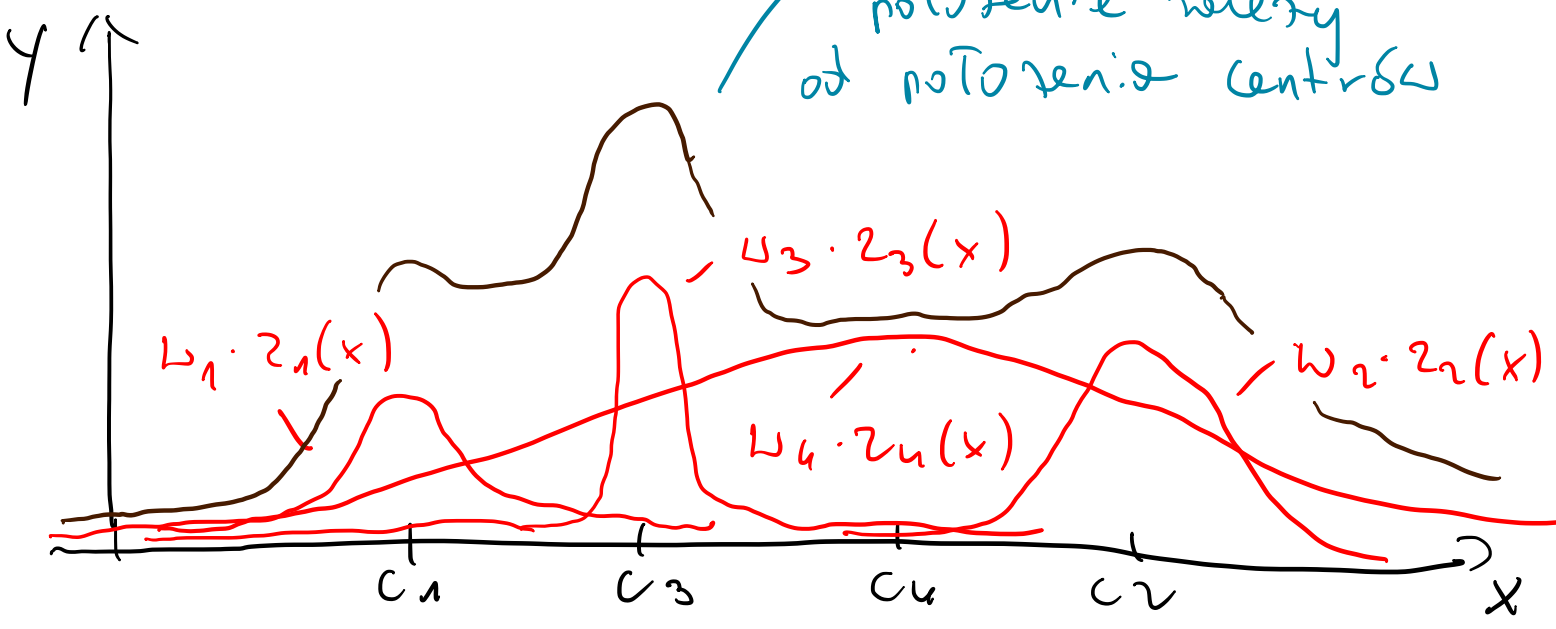
σ_k - parametr wpływający na kształt funkcji radialnej ($\in R$)

w_0, w_k - wagi neuronu liniowego ($\in R$)



gaussowska f-cja radialna (są też inne)

sieć radialna odpowiada sumie ważonej f-cji radialnych położenie wzlezy od położenia centrów



I dwuetapowe trenowanie sieci RBF w oparciu o zbiór przykładów:

(x^j, y^j) dla $j = 1, \dots, N$

gdzie $x^j \in R^n$ i $y^j \in R$

wymaga:

a) doboru wzorców $c_k \in R^n$

b) doboru $\sigma_k \in R$

c) doboru wag w_0 oraz w_k dla $k = 1, \dots, K$

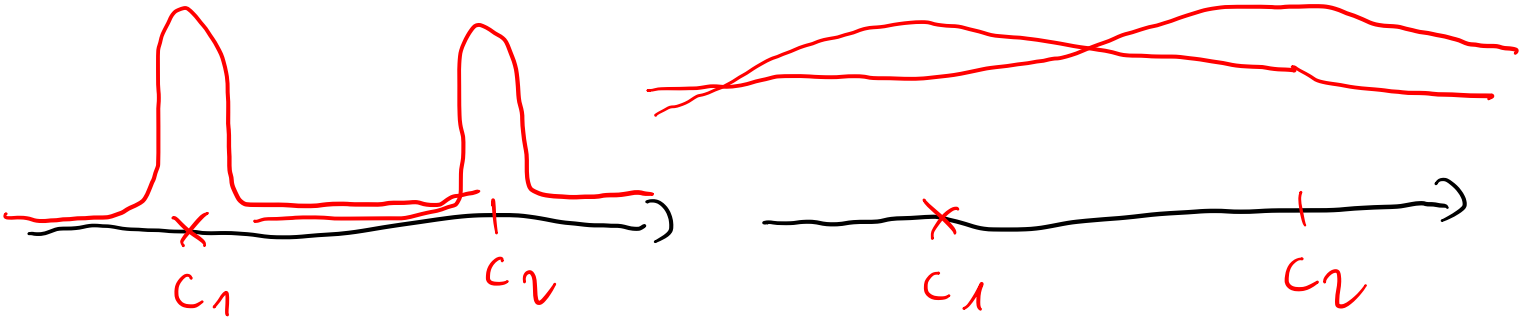
w przypadku gdy sieć ma więcej wyjść (więcej neuronów liniowych) procedura jest analogiczna dla każdego z nich

a) dobór centrów powinien zmierzać ku równomiernie (minimalny błąd kwantyzacji) na danym pokrywanym przez dane treningowe x^j

- można to zrobić wybierając losowo punkty spośród x^j
- można do tego celu wykorzystać algorytmy: k-średnich, Kohonena

ny algorytm fazy uczenia

b) parametry σ_k powinny być tak dobrane aby umiarkowanej sytuacji przedstawionych na poniższych rysunkach (przypadek jednowymiarowy):



Uwaga: powinien być tak dobrany aby wpływ funkcji radialnej w miejscu wystąpienia sąsiednich wzorców nie był ani za duży, ani za mały

c) mając ustalone wartości parametrów c_k i σ_k dobór wag neuronu liniowego można przeprowadzić metodą najszybszego spadku minimalizując błąd średniokwadratowy:

przypadek off-line:

$$E(w) = \frac{1}{2N} \sum_{j=1}^{N} \left(f(x^j) - y^j \right)^2$$

w przypadku online nie uśredniamy po wszystkich danych treningowych tylko nie ma z pozostałym cząstkowe sumy (aktualizujemy wagi po kolejnej danej treningowej)

gdzie

$$f(x^j) = w_0 + \sum_{k=1}^{K} w_k \cdot z_k(x^j)$$

wówczas metoda najszybszego spadku przyjmuje postać:

$$w_k(t+1) = w_k(t) - \alpha \cdot \frac{\partial E}{\partial w_k}(w(t))$$

dla $k = 0, 1, \ldots, K$

Łatwo wyliczyć, że:

$$\frac{\partial E}{\partial w_0} = \frac{1}{N} \sum_{j=1}^{N} \left(f(x^j) - y^j \right)$$

$$\frac{\partial E}{\partial w_k} = \frac{1}{N} \cdot \sum_{j=0}^{N} \left(f(x^j) - y^j \right) \cdot z_k(x^j)$$

dla $k = 1, \ldots, K$

II jednoetapowe trenowanie sieci RBF w oparciu o zbiór przykładów:

$$(x^j, y^j) \quad \text{dla} \quad j = 1, \ldots, N$$

gdzie $x^j \in R^n$ i $y^j \in R$

w przypadku trenowania off-line błąd średniokwadratowy zależy od wag w, centrów c oraz parametru σ:

$$E(w, c, \sigma) = \frac{1}{N} \sum_{j=1}^{N} (f(x^j) - y^j)^2$$

gdzie:

$$f(x^j) = w_0 + \sum_{k=1}^{K} w_k z_k(x^j)$$

$$z_k(x^j) = e^{-d^2(x^j, c_k)/2\sigma_k^2}$$

$$d(x^j, c_k) = \sqrt{\sum_{i=1}^{n} (x_i^j - c_{k,i})^2}$$

funkcja E jest prawie wszędzie wielokrotnie ciągle różniczkowalna jest zastosowanie metody najszybszego spadku do wszystkich parametrów;

$$w_k(t+1) = w_k(t) - \alpha \cdot \frac{\partial E}{\partial w_k}$$

dla $k = 0, 1, \ldots, K$

$$\sigma_k(t+1) = \sigma_k(t) - \alpha \frac{\partial E}{\partial \sigma_k}$$

dla $k = 1, \ldots, K$

$$c_{k,i}(t+1) = c_{k,i}(t) - \alpha \frac{\partial E}{\partial c_{k,i}}$$

dla $k = 1, \ldots, K$ oraz $i = 1, \ldots, n$

pochodne $\frac{\partial E}{\partial w_k}$ należało obliczone

wcześniej przy okazji dwuetapowej,

pochodne $\frac{\partial E}{\partial \sigma_k}$ oraz $\frac{\partial E}{\partial c_{k,i}}$

należy obliczyć samodzielnie :)