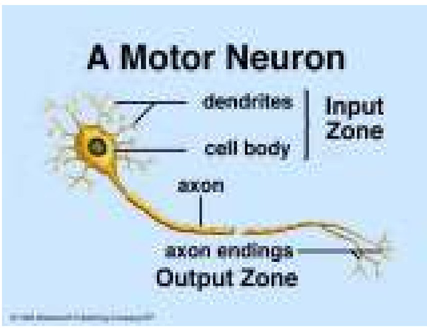
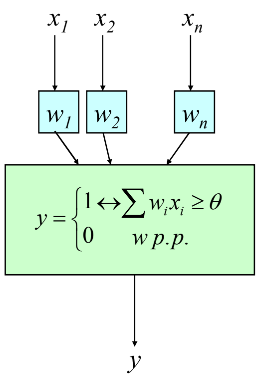
Sieci neuronowe - uczenie

`http://zajecia.jakubw.pl/nai/`

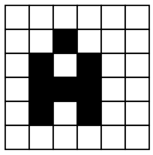
Perceptron - przypomnienie



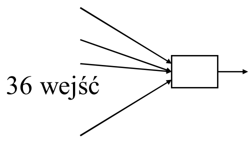


Uczenie perceptronu

Przykład: rozpoznawanie znaków

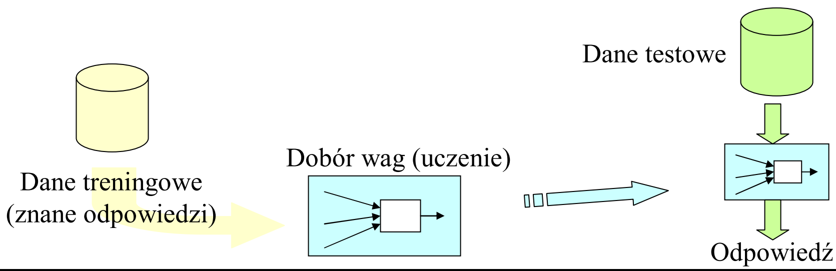


Siatka 6 × 6



<u>Wyjście</u>: 1, jeśli na wejściu pojawia się litera "A", zaś 0 w p.p.

<u>Zadanie</u>: dobrać wagi wejść i wartość progową tak, by uzyskać zaplanowany efekt



Uczenie perceptronu

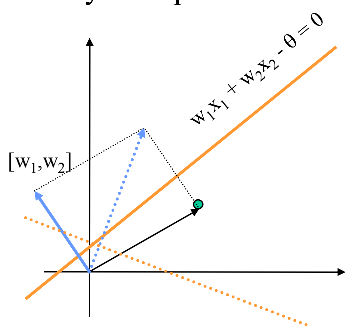
- **Wejście:**
  - Ciąg przykładów uczących ze znanymi odpowiedziami
- **Proces uczenia:**
  - Inicjujemy wagi losowo
  - Dla każdego przykładu, jeśli odpowiedź jest nieprawidłowa, to

    $\mathbf{w_1} += \alpha\ \mathbf{x_1}$

    $\mathbf{w_2} += \alpha\ \mathbf{x_2}$

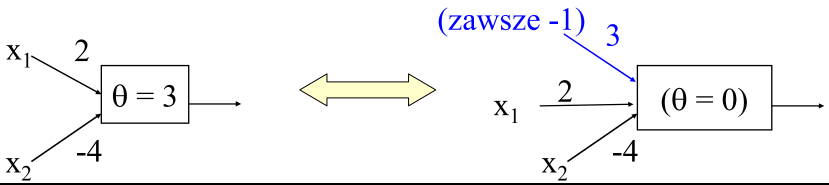
    $\theta -= \alpha$

    gdzie α jest równe różnicy między odpowiedzią prawidłową a otrzymaną.



Uczenie perceptronu

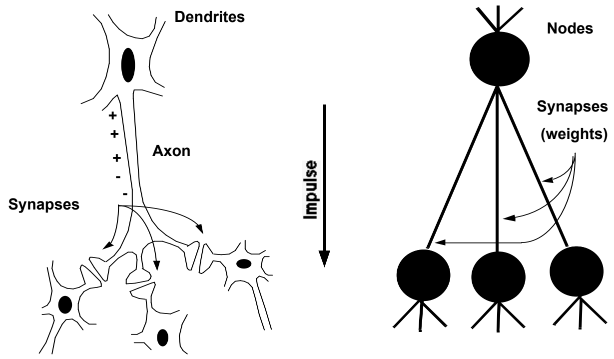
- Często α mnoży się dodatkowo przez niewielki współczynnik uczenia
- Po wyczerpaniu przykładów, zaczynamy proces uczenia od początku, dopóki następują jakiekolwiek zmiany wag połączeń
- Próg θ można traktować jako wagę dodatkowego wejścia o wartości -1:



Uczenie perceptronu

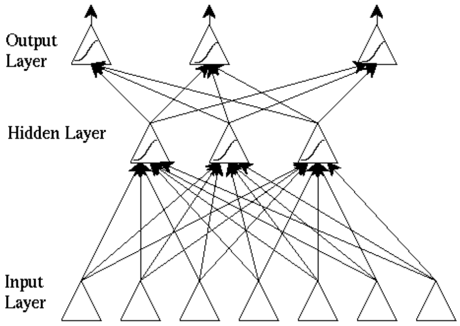
- Opisany schemat jest w miarę przejrzysty tylko dla pojedynczych perceptronów, lub niewielkich sieci
- Ciężko jest stosować reguły tego typu dla skomplikowanych modeli
  - Tymczasem np. do rozpoznawania wszystkich liter potrzeba by sieci złożonej z 26 takich perceptronów

Sieci perceptronów



Ograniczenia pojedynczych perceptronów spowodowały w latach 80-tych wzrost zainteresowania sieciami wielowarstwowymi i opracowanie algorytmu ich uczenia (propagacja wsteczna)

SIECI WIELOWARSTWOWE

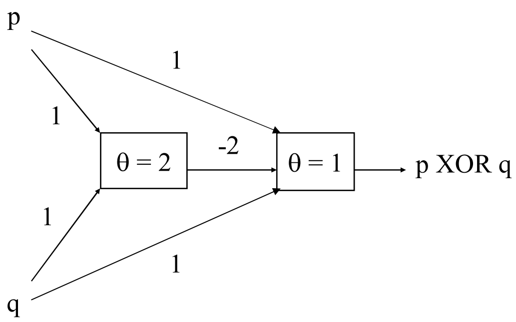


- Wyjścia neuronów należących do warstwy niższej połączone są z wejściami neuronów należących do warstwy wyższej
  - np. metodą „każdy z każdym”

- Działanie sieci polega na liczeniu odpowiedzi neuronów w kolejnych warstwach
- Nie jest znana ogólna metoda projektowania optymalnej architektury sieci neuronowej

SIECI PERCEPTRONÓW

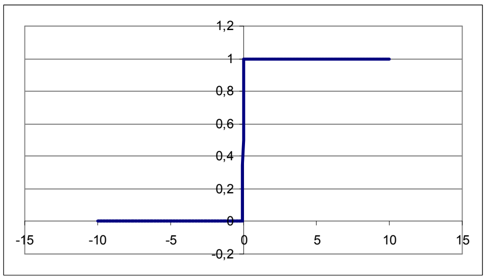
Potrafią reprezentować dowolną funkcję boolowską (opartą na rachunku zdań)

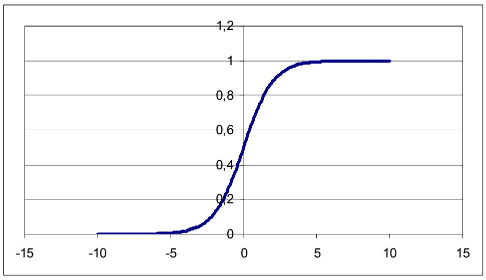


Funkcje aktywacji

- Progowe

$$f(s)=\begin{cases}1 \Leftrightarrow s \geq 0\\ 0 \Leftrightarrow s < 0\end{cases}$$





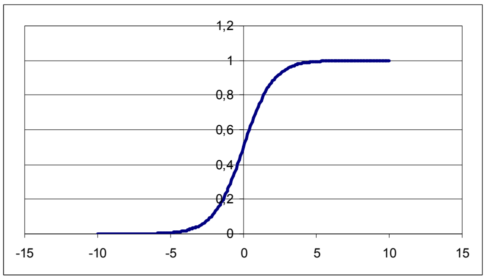
- Sigmoidalne

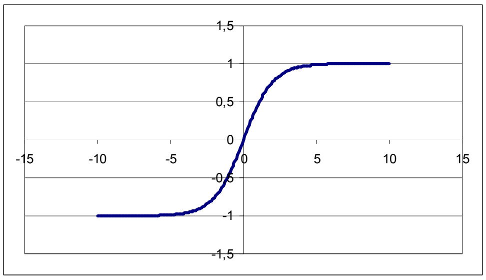
$$f(s)=\frac{1}{1+e^{-s}}$$

FUNKCJE AKTYWACJI (2)

- Unipolarne

$$f(s)=\frac{1}{1+e^{-s}}$$

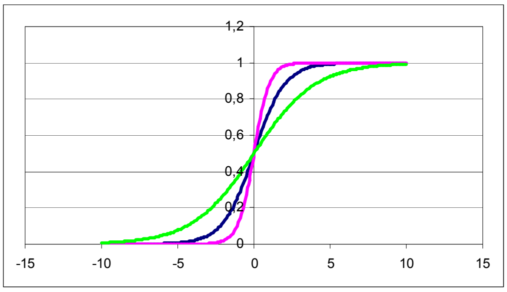




- Bipolarne

$$f(s)=\frac{2}{1+e^{-s}}-1$$

FUNKCJE AKTYWACJI (3)



$$f_{\alpha}(s)=\frac{1}{1+e^{-\alpha s}}$$

α = 2.0

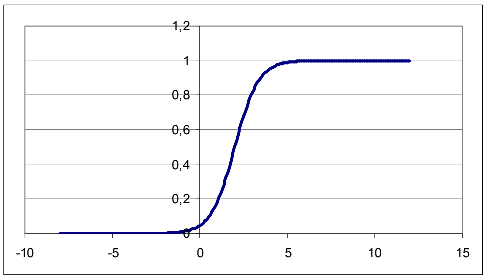
α = 1.0

α = 0.5

$$\lim_{\alpha\to 0} f_{\alpha}(s)=0.5 \qquad \lim_{\alpha\to +\infty} f_{\alpha}(s)=\begin{cases} 1 & \Leftrightarrow \quad s>0 \\ 0.5 & \Leftrightarrow \quad s=0 \\ 0 & \Leftrightarrow \quad s<0 \end{cases}$$

FUNKCJE AKTYWACJI (4)

$$f_{\theta,\alpha}(s) = \frac{1}{1+e^{-\alpha(s-\theta)}}$$



$$\theta = 2$$

$$\alpha = 1.5$$

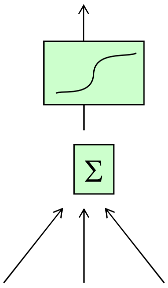
FUNKCJE AKTYWACJI (5)

- Zasady ogólne:
  - Ciągłość (zachowanie stabilności sieci jako modelu rzeczywistego)
  - Różniczkowalność (zastosowanie propagacji wstecznej)
  - Monotoniczność (intuicje związane z aktywacją komórek neuronowych)
  - Nieliniowość (możliwości ekspresji)

SIECI NEURONOWE

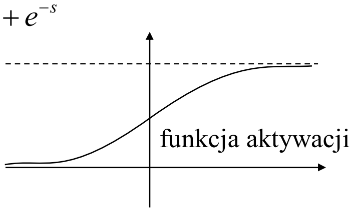
Potrafią modelować (dowolnie dokładnie przybliżać) funkcje rzeczywiste

(z tw. Kołmogorowa)

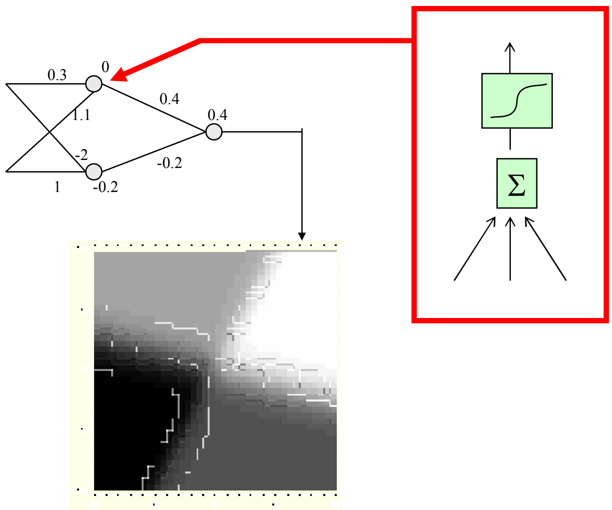


$$y = f\left(w_0 + \sum_{i=1}^{n} w_i x_i \right)$$

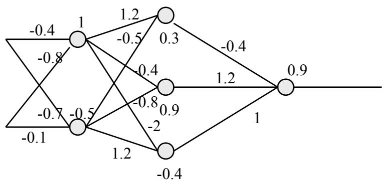
$$f(s) = \frac{1}{1+e^{-s}}$$

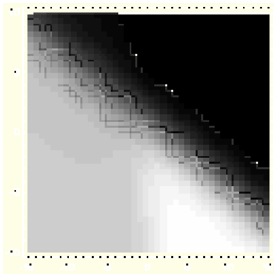


SIECI NEURONOWE

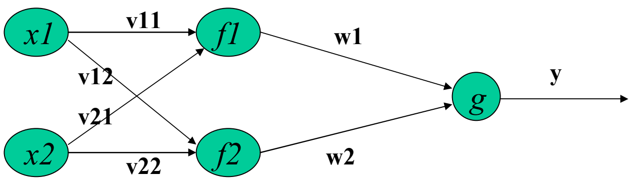


SIECI NEURONOWE





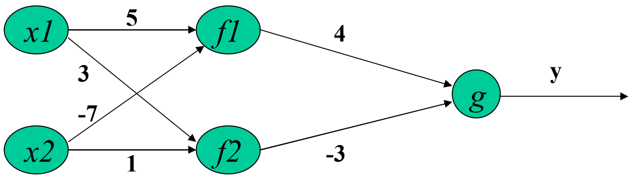
SIECI JAKO FUNKCJE ZŁOŻONE (1)



$$y = g\left(w_1 f_1\left(v_{11}x_1 + v_{21}x_2\right) + w_2 f_2\left(v_{12}x_1 + v_{22}x_2\right)\right)$$

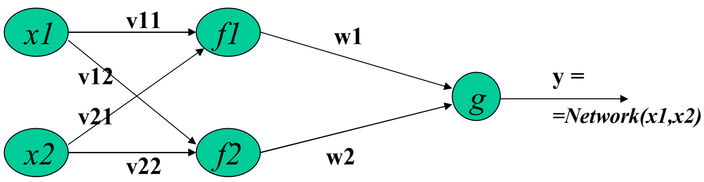
$$y = Network\left(x_1, x_2\right)$$

SIECI JAKO FUNKCJE ZŁOŻONE (2)



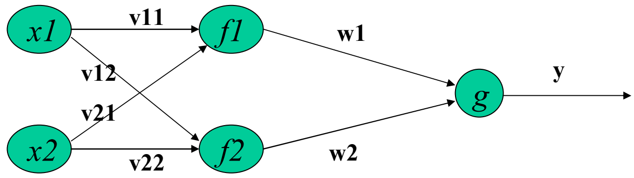
$$y=\begin{cases}1 \Leftrightarrow \dfrac{4}{1+e^{-3(5x_1-7x_2)}}-3\left(\dfrac{2}{1+e^{-2(3x_1+x_2)}}-1\right)\geq 8\\ 0 \Leftrightarrow \dfrac{4}{1+e^{-3(5x_1-7x_2)}}-3\left(\dfrac{2}{1+e^{-2(3x_1+x_2)}}-1\right)<8\end{cases}$$

SIECI JAKO FUNKCJE ZŁOŻONE (3)



- Jeśli wszystkie poszczególne funkcje aktywacji są liniowe, to funkcja *Network* jest również liniowa
- Architektura wielowarstwowa daje zatem nowe możliwości tylko w przypadku stosowania funkcji nieliniowych

SIECI JAKO FUNKCJE ZŁOŻONE – przypadek liniowy



- Niech

  $f_i(x1,x2) = a_i*(x1*v1_i + x2*v2_i) + b_i$

  $g(z1,z2) = a*(z1*w1 + z2*w2) + b$
- Wtedy

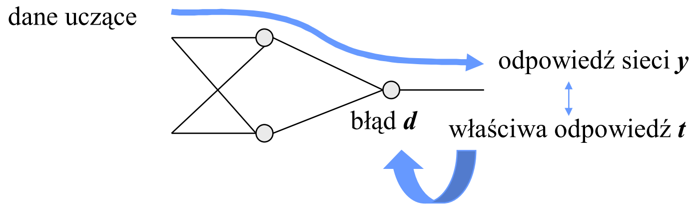
  Network(x1,x2) = A1*x1 + A2*x2 + B
- Np.:

  A1 = a*(a1*v1*w1 + a2*v2*w2)

PROPAGACJA WSTECZNA (1)

- <u>Chcemy “wytrenować” wagi połączeń między kolejnymi warstwami neuronów</u>
- Inicjujemy wagi losowo (na małe wartości)
- Dla danego wektora uczącego obliczamy odpowiedź sieci (warstwa po warstwie)
- Każdy neuron wyjściowy oblicza swój błąd, odnoszący się do różnicy pomiędzy obliczoną odpowiedzią **y** oraz poprawną odpowiedzią **t**

PROPAGACJA WSTECZNA (2)



Błąd sieci definiowany jest zazwyczaj jako

$$d = \frac{1}{2}(y-t)^2$$

PROPAGACJA WSTECZNA (3)

- Oznaczmy przez:
  - ***f: R → R*** – funkcję aktywacji w neuronie
  - $\boldsymbol{w_1, \ldots, w_K}$ – wagi połączeń wchodzących
  - $\boldsymbol{z_1, \ldots, z_K}$ – sygnały napływające do neuronu z poprzedniej warstwy
- Błąd neuronu traktujemy jako funkcję wag połączeń do niego prowadzących:

$$d(w_1, \ldots, w_K) = \frac{1}{2}\left(f(w_1 z_1 + \ldots + w_K z_K) - t\right)^2$$

PRZYKŁAD (1)

- Rozpatrzmy model, w którym:
  - Funkcja aktywacji przyjmuje postać

$$f(s) = \frac{1}{1 + e^{-3(s+2)}}$$

  - Wektor wag połączeń = [1;-3;2]
- Załóżmy, że dla danego przykładu:
  - Odpowiedź powinna wynosić ***t*** = 0.5
  - Z poprzedniej warstwy dochodzą sygnały [0;1;0.3]

PRZYKŁAD (2)

- Liczymy wejściową sumę ważoną:

$$s = w_1 x_1 + w_2 x_2 + w_3 x_3 = 1 \cdot 0 + (-3) \cdot 1 + 2 \cdot 0.3 = -2.4$$

- Liczymy odpowiedź neuronu:

$$y = f(s) = \frac{1}{1 + e^{-3(-2.4+2)}} = \frac{1}{1 + e^{1.2}} \approx 0.23$$

- Błąd wynosi:

$$d = \frac{1}{2}(0.23 - 0.5)^2 \approx 0.036$$

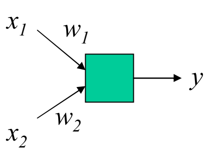
IDEA ROZKŁADU BŁĘDU

- Musimy „rozłożyć” otrzymany błąd na połączenia wprowadzające sygnały do danego neuronu
- Składową błędu dla każdego ***j***-tego połączenia określamy jako pochodną cząstkową błędu względem ***j***-tej wagi
- Składowych tych będziemy mogli użyć do zmodyfikowania ustawień poszczególnych wag połączeń

IDEA ROZKŁADU BŁĘDU

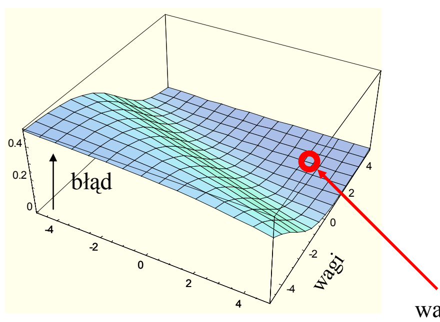
Załóżmy, że mamy neuron z wagami w_0=0, w_1=2, w_2=3. Mamy dane wektor wejściowy: [0.3 , 0.7], przy czym oczekiwana odpowiedź to t=1. Jak należy zmienić wagi, aby błąd był jak najmniejszy?

Możemy błąd przedstawić jako funkcję w_1, w_2:



$$y = f\left(w_0 + \sum_{i=1}^{n} w_i x_i\right)$$

$$f(s) = \frac{1}{1+e^{-s}}$$



wartość błędu dla wag [2, 3]

Wagi powinniśmy zmienić w kierunku spadku wartości błędu.

KIERUNEK ZMIANY WAG

Jeśli rozważymy większą liczbę przykładów, funkcja średniego błędu będzie miała bardziej skomplikowany kształt.

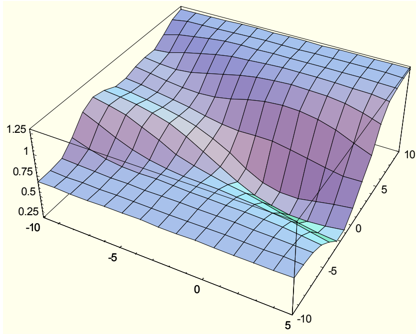
[0.3, 0.7], t=1

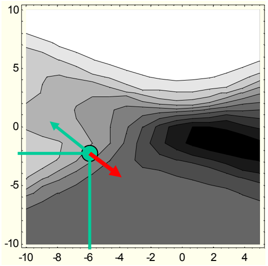
[0.2, 0.9], t=0.1

[-0.6, 1], t=0

[0, -0.8], t=0.5

[0.6, 1], t=0.3





Nachylenie wykresu w danym punkcie (odpowiadającym aktualnym wartościom wag) dane jest przez **gradient**, czyli wektor pochodnych cząstkowych.

Zmiana wag powinna nastąpić w kierunku przeciwnym.

OBLICZANIE POCHODNEJ

$$\boxed{\frac{\partial d(w_1,\ldots,w_K)}{\partial w_j} = (y-t)\cdot f'(s)\cdot z_j}$$

$$= \frac{\partial \frac{1}{2}\left(f(w_1 z_1 + \ldots + w_K z_K) - t\right)^2}{\partial w_j}$$

$$= \frac{\partial \frac{1}{2}(y-t)^2}{\partial y}\cdot\frac{\partial f(s)}{\partial s}\cdot\frac{\partial(w_1 z_1 + \ldots + w_K z_K)}{\partial w_j}$$

PROPAGACJA WSTECZNA

- Idea:
  - Wektor wag połączeń powinniśmy przesunąć w kierunku **przeciwnym** do wektora gradientu błędu (z pewnym współczynnikiem uczenia η)
  - Możemy to zrobić po każdym przykładzie uczącym, albo sumując zmiany po kilku przykładach.
- Realizacja: $\Delta w_j = \eta \cdot (t-y) \cdot f'(s) \cdot z_j$

Prosty przykład: wagi w_1=1, w_2=1, dane wejściowe: [0.5, 0.5], t = 1.

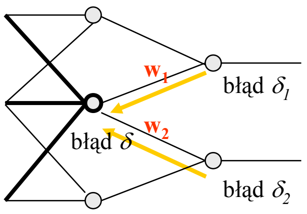
Funkcja sigmoidalna: $f(s) = \frac{1}{1+e^{-s}}$ więc: $f'(s) = \frac{e^{-s}}{(1+e^{-s})^2}$

Stąd: s = 0.5 + 0.5 = 1, y = 0.731, zmiana w= (1- 0.731) * 0.19 * 0.5 = 0.026.
A więc nowe wagi to 1.026. Ten sam przykład da tym razem odpowiedź y=0.736.

PROPAGACJA WSTECZNA

Błędy są następnie propagowane w kierunku poprzednich warstw. Wprowadźmy pomocniczo współczynnik błędu δ zdefiniowany dla ostatniej warstwy jako:

$$\delta = f'(s) \cdot (t-y)$$



a dla pozostałych warstw:

$$\delta = f'(s) \cdot \sum_{i=1}^{n} w_i \delta_i$$

czyli neuron w warstwie ukrytej "zbiera" błąd z neuronów, z którymi jest połączony.

Zmiana wag połączeń następuje po fazie propagacji błędu i odbywa się według wzoru: $\Delta w = \eta \cdot \delta \cdot z$

Oznaczenia: *w - waga wejścia neuronu, z - sygnał wchodzący do neuronu danym wejściem, δ - współczynnik błędu obliczony dla danego neuronu, s - wartość wzbudzenia (suma wartości wejściowych pomnożonych przez wagi) dla danego neuronu.*