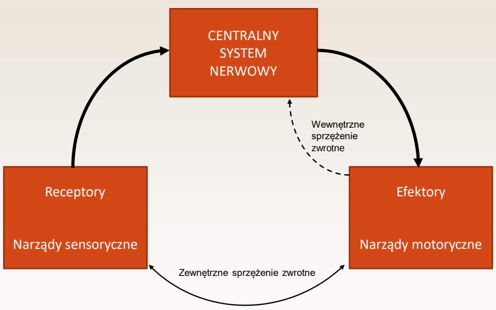
Data Mining
Wykład 7

Sztuczne sieci neuronowe (SNN)

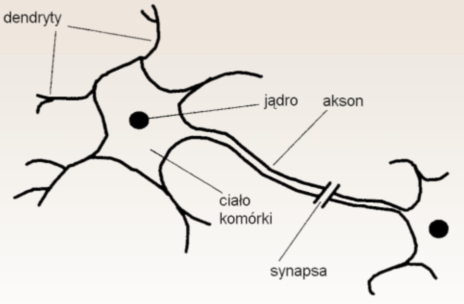
Przepływ informacji w systemie nerwowym



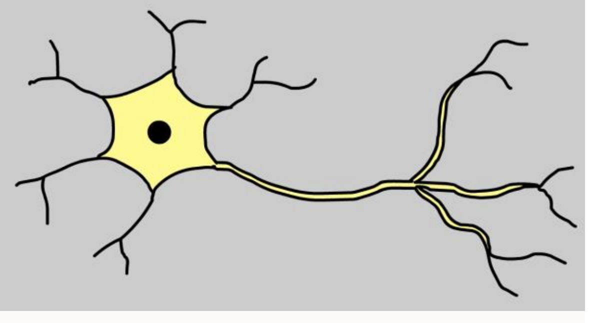
Budowa i działanie mózgu

- Mózg składa się ze 100 miliardów (10^{11}) neuronów
- Każdy neuron jest wstanie stworzyć nawet do 10 tysięcy połączeń z innymi komórkami nerwowymi
- Neuron ma ok. 0,1 mm średnicy, a jego długość może osiągnąć kilka metrów
- Łączna długość aksonów w mózgu to 160 000 km
- Na każdy neuron przypada od 1000 do 10000 synaps
- Komórki nerwowe wysyłają i przyjmują impulsy
  - Częstotliwości 1-100Hz
  - Czasie trwania 1-2 ms
  - Szybkość propagacji 1-100 m/s
- Informacje docierają do mózgu z prędkością 100 megabajtów na sekundę
- Szybkość pracy mózgu to 10^{18} operacji/s

Budowa neuronu biologicznego



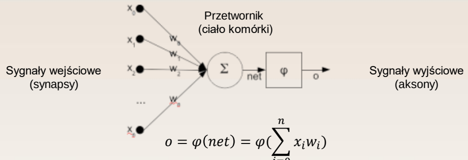
Zasada działania neuronu biologicznego



Sztuczna sieć neuronowa (SSN)

- łączy w sobie małe procesory (jednostki przetwarzania), z których każdy naśladuje funkcje pojedynczego neuronu biologicznego
- każdy procesor, podobnie jak neuron, ma poziom progowy
- aby nastąpiła transmisja sygnału poziom progowy, musi zostać przekroczony
- binarny charakter procesora można porównać do włącznika światła
- dzięki połączeniu wielu takich sztucznych neuronów, z których każdy posiada jedno wyjście, SSN może wykonywać złożone funkcje
- tworzy sztuczny „mózg” do przetwarzania informacji

Sztuczny neuron (perceptron)



$$o = \varphi(net) = \varphi(\sum_{i=0}^{n} x_i w_i)$$

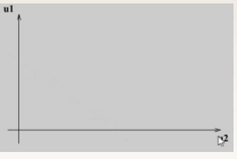
gdzie x_i - wejścia i x_0= 1, w_i - wagi, φ - funkcja aktywacji, net - pobudzenie neuronu, o - wyjście z neuronu

- Każdy neuron ma bardzo prostą funkcje i oblicza sumę ważoną sumę danych wejściowych pochodzących od innych neuronów,
- Wejścia i wagi są liczbami rzeczywistymi dodatnimi i ujemnymi,
- Jeżeli jakaś cecha (wejście) powoduje odpalenie neuronu, waga będzie dodatnia, a jeżeli cecha ma działanie hamujące to waga jest ujemna,
- Funkcje aktywacji dobiera się do rozwiązywanego zadania,
- Wagi są dopasowywane przez pewna regułę uczenia tak, by zależność wejście/wyjście w neuronie spełniało pewien określony cel

Interpretacja geometryczna

Jeśliby przedstawić sygnały wejściowe w postaci przestrzeni n-wymiarowej, gdzie n - liczba sygnałów wejściowych, to w przypadku dwóch wejść mamy 2 - wymiarową przestrzeń.

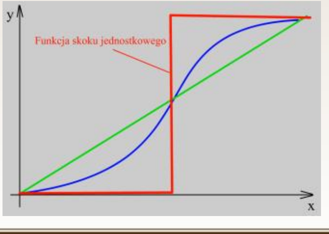
Na obu osiach mamy sygnały wejściowe, zaś sygnał wyjściowy z perceptronu y to punkt znajdujący się w obrębie przestrzeni układu. Odpowiada on sygnałom wejściowym w tym przypadku u_1 i u_2.



Perceptron dzieli powstałą w ten sposób przestrzeń na dwie półprzestrzenie oddzielone m-1 wymiarową hiperpłaszczyzną. Dla wejść z jednej półprzestrzeni perceptron jest aktywy (y=1) a dla drugiej nie (y=0).

Funkcja aktywacji

- skoku jednostkowego - tzw. funkcja progowa
- funkcja liniowa
- funkcja nieliniowa



Stosowane w SSN funkcje aktywacji (1)

Name	Input/Output Relation	Icon
Hard Limit	$a = 0 \quad n < 0$ $a = 1 \quad n \ge 0$	
Symmetrical Hard Limit	$a = -1 \quad n < 0$ $a = +1 \quad n \ge 0$	
Linear	$a = n$	
Saturating Linear	$a = 0 \quad n < 0$ $a = n \quad 0 \le n \le 1$ $a = 1 \quad n > 1$	

Stosowane w SSN funkcje aktywacji (2)

Symmetric Saturating Linear	$a = -1 \quad n < -1$ $a = n \quad -1 \le n \le 1$ $a = 1 \quad n > 1$	
Log-Sigmoid	$a = \frac{1}{1 + e^{-n}}$	
Hyperbolic Tangent Sigmoid	$a = \frac{e^{n} - e^{-n}}{e^{n} + e^{-n}}$	
Positive Linear	$a = 0 \quad n < 0$ $a = n \quad 0 \le n$	
Competitive	$a = 1$ neuron with max n $a = 0$ all other neurons	C

Uczenie SSN

Uczenie polega na automatycznym doborze wag w SSN, na podstawie zbioru przykładów nazwanych zbiorem uczącym. Zaczyna się z losowymi małymi wagami i iteracyjnie zmienia się wagi, dopóki wszystkie przykłady uczące nie zostaną poprawnie zaklasyfikowane (lub z niewielkim błędem).

Wyróżnia się dwa typy uczenia, co zależy od wykorzystanej sieci przykładów (zbioru uczącego):

- **nadzorowane** (z nauczycielem), gdy w zbiorze danych do każdej próbki podana jest poprawna klasa
- **nienadzorowane** (bez nauczyciela), zbiór danych nie ma wektora odpowiedzi.

Algorytm uczenia perceptronu

1. Niech $\boldsymbol{w}(0) = (0, \dots, 0)$ lub wartości losowe z przedziału $[-1,1], k = 0$
2. Dopóki zbiór punktów uczących pozostaje błędnie klasyfikowany tj. zbiór $A = \{x_i : y_i \neq f(\langle \boldsymbol{w}, \boldsymbol{x_i} \rangle)\}$ pozostaje niepusty, powtarzaj:
   1. Wylosuj ze zbioru A dowolny punkt
   2. Aktualizuj wagi według następującej reguły:
      $\boldsymbol{w}(k+1) = \boldsymbol{w}(k) + \eta e \boldsymbol{x_i}$
   3. $k = k + 1$

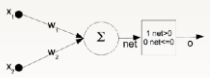
Gdzie η współczynnik uczenia $\eta \in (0,1]$, a e jest wartością błędu popełnianego na prezentowanej próbce.

Algorytm uczenia perceptronu - przykład

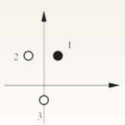
- Dany jest zbiór uczący:

$\{x_1 = (1,2), y_1{=}1\}$
$\{x_2 = (-1,2), y_2{=}0\}$
$\{x_3 = (0,-1), y_3{=}0\}$

Architektura sieci:



Przestrzeń wejść:



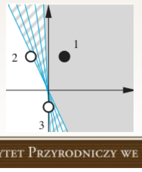
Konsekwencje braku wejścia x_0

Usuniecie dodatkowego wejścia powoduje, ze płaszczyzna decyzyjna musi przejść przez początek układu współrzędnych, bo:

$$w_1 x_1 + w_2 x_2 = 0$$

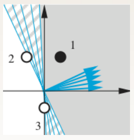
to równanie prostej przechodzącej przez punkt (0, 0).
Trzeba być pewnym, ze taka siec rozwiąże problem separacji punktu x_1 od punktów x_2 i x_3.

Jak widać rozwiązań jest nieskończenie wiele.



Cel uczenia

Rysunek pokazuje wektory wag, które odpowiadają granicom decyzyjnym. Wektor jest prostopadły do granicy decyzyjnej. Reguła ucząca ma znaleźć wektor wag, który wskazuje w jednym ze wskazanych kierunków. Długość wektora wag nie ma znaczenia, ważny jest tylko kierunek.

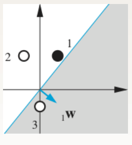


Krok 1: Ustalenie wartości początkowych wag

Zakłada się losowe ustalenie wartości wag początkowych i tak:

$$w(0) = (1, -0.8)$$

Na wykresie zaprezentowano początkowy wektor wag i granice decyzyjna dla takich wag.



Krok 2.1: Prezentacja punktu x_1

Pierwszym prezentowanym punktem jest $x_1 = (1,2)$

$$f(\langle \boldsymbol{w}, \boldsymbol{x}_1 \rangle) = f(1 \cdot 1 + (-0.8) \cdot 2) = f(-0.6) = 0$$

Wiemy, ze $y_1 = 1$, zatem sieć popełniła błąd i wskazała zła klasę dla punktu x_1.

Jak widać na rysunku z poprzedniego slajdu granica decyzyjna jest nieprawidłowo położona w stosunku do punktu uczącego. Powinno się zatem przesunąć wektor wag nieznacznie w kierunku punktu x_1. Można tego dokonać dodając do wektora wag wektor x_1 .

Krok 2.2: Wnioski co do poprawki wag dla x_1

Dla uproszczenia obliczeń przyjmijmy że funkcja błędu ma postać:

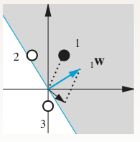
$$e = y_i - f(\langle \boldsymbol{w}, \boldsymbol{x}_i \rangle)$$

a współczynnik uczenia $\eta = 1$.

Jeżeli $y_1 = 1$ i $f(\langle \boldsymbol{w}, \boldsymbol{x}_1 \rangle) = 0$ to :

$$\boldsymbol{w}(k+1) = \boldsymbol{w}(k) + \boldsymbol{x}_i$$

Stąd, wagi po korekcie: $\boldsymbol{w}(1) = (1+1, -0.8+2) = (2,1.2)$



Krok 2.1: Prezentacja punktu x_2

Drugim prezentowanym punktem jest $x_2 = (-1,2)$

$$f(\langle \boldsymbol{w}, \boldsymbol{x}_2 \rangle) = f(2 \cdot (-1) + 1.2 \cdot 2) = f(0.4) = 1$$

Wiemy, ze $y_2 = 0$, zatem sieć popełniła błąd i wskazała zła klasę dla punktu x_2.

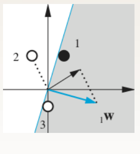
Jak widać na rysunku z poprzedniego slajdu granica decyzyjna jest nieprawidłowo położona w stosunku do punktu uczącego. Powinno się zatem odsunąć wektor wag nieznacznie od punktu x_2.

Krok 2.2: Wnioski co do poprawki wag dla x_1

Jeżeli $y_1 = 0$ i $f(\langle \boldsymbol{w}, \boldsymbol{x}_2 \rangle) = 1$ to $e = y_i - f(\langle \boldsymbol{w}, \boldsymbol{x}_i \rangle) = -1$, stąd:

$$\boldsymbol{w}(k+1) = \boldsymbol{w}(k) - \boldsymbol{x}_i$$

Wagi po korekcie: $\boldsymbol{w}(2) = (2-(-1), 1.2-2) = (3, -0.8)$



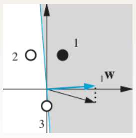
Krok 2.1&2: Prezentacja punktu x_3

Trzecim prezentowanym punktem jest $x_3 = (0, -1)$

$$f(\langle \mathbf{w}, \mathbf{x}_3 \rangle) = f(3 \cdot 0 + (-0.8) \cdot (-1)) = f(0.8) = 1$$

Wiemy, ze $y_3 = 0$, zatem sieć popełniła błąd i wskazała zła klasę dla punktu x_3. Stosując regułę z poprzedniej iteracji poprawia się wagi:

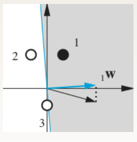
$$w(3) = (3 - 0, (-0.8) - (-1) = (3{,}0.2)$$



Kolejna seria prezentacji wzorców uczących

Prezentując ponownie wszystkie punkty uczące uzyskujemy bezbłędne sklasyfikowanie wszystkich 3 punktów.

Oznacza to, że wagi $\mathbf{w} = (3{,}0.2)$ są rozwiązaniem dla uczonego klasyfikatora co kończy proces uczenia.

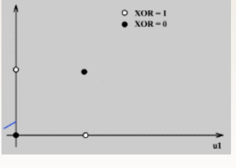


Problem klasyfikacji pojedynczym perceptronem

Z powyższego przykładu wynika, że aby funkcja mogła być realizowana przez perceptron prosty sygnały wejściowe dla których funkcja aktywacji przyjmuje wartość 0 muszą leżeć w innej półpłaszczyźnie niż te sygnały wejściowe dla których funkcja ta przyjmuje wartość 1. Mówi się że **muszą one być liniowo separowalne**.

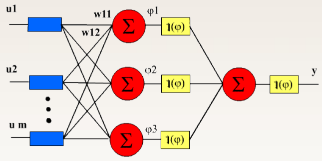
To ograniczenie powoduje, że wiele ciekawych funkcji nie może być realizowanych przez perceptron prosty. Przykładem takiej funkcji może być XOR. Nie istnieje bowiem taka prosta, która rozdzielałaby punkty o wartościach równych 0 od punktów o wartościach 1.

u1	u2	XOR (u1,u2)
0	0	0
0	1	1
1	0	1
1	1	0



Perceptron wielowarstwowy

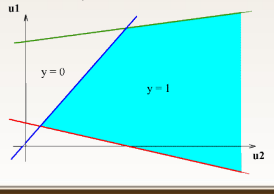
- Ograniczenia takie eliminuje się poprzez wprowadzanie warstw ukrytych uzyskując w ten sposób strukturę nazwaną ***perceptronem wielowarstwowym***.



Perceptron dwuwarstwowy

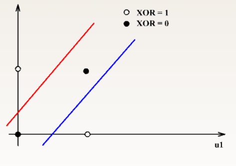
Perceptron wielowarstwowy

- Każdy element warstwy ukrytej dzieli przestrzeń sygnałów wejściowych na dwie półprzestrzenie rozdzielone hiperpłaszczyzną (tak jak perceptron jednowarstwowy).
- Element wyjściowy, dzieli przestrzeń sygnałów wejściowych również na dwie podprzestrzenie.
- W efekcie otrzymujemy podprzestrzeń, która jest zborem wielościennym wypukłym *.
- Jeśli więc jeden z dwóch zbiorów sygnałów wejściowych o wartościach odpowiednio równych 0 i 1 zawiera się w tym zbiorze wypukłym to funkcja może być zrealizowana przez perceptron dwuwarstwowy.

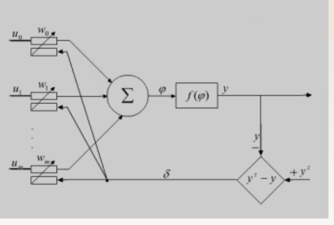


Perceptron wielowarstwowy

- Za pomocą perceptronu dwuwarstwowego można zrealizować już więcej funkcji w tym między innymi funkcję XOR.
- Jak widać na poniższym rysunku jeden z dwóch zbiorów sygnałów wejściowych o wartości funkcji aktywacji równej 0 zawiera się w wydzielonym zbiorze wypukłym. Tak więc warunek jest spełniony.



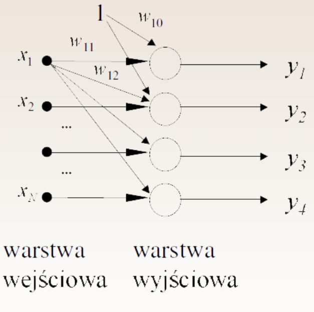
Algorytm uczenia elementu perceptronowego



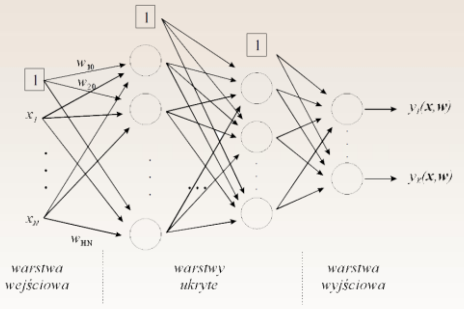
Rodzaje sieci

- Sieci jednokierunkowe – sygnał w sieci rozprzestrzenia się w jednym kierunku
  - Sieci jednowarstwowe
  - Sieci wielowarstwowe (perceptron wielowarstwowy)
- Sieci rekurencyjne

Sieć jednokierunkowa jednowarstwowa

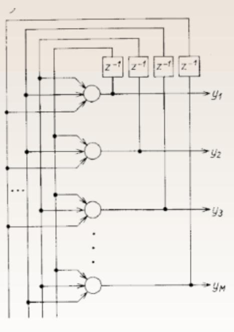


Sieć jednokierunkowa wielowarstwowa



Uniwersytet Przyrodniczy we Wrocławiu

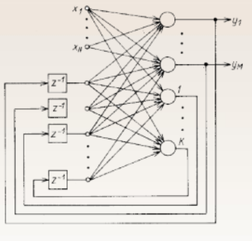
Sieć rekurencyjna jednowarstwowa



- Sieć z jedną warstwą neuronów (wyjściową)
- Sygnały wyjściowe neuronów tworzą jednocześnie wektor wejściowy dla następnego cyklu
- Z reguły nie występuje sprzężenie zwrotne od własnego sygnału wyjściowego

Uniwersytet Przyrodniczy we Wrocławiu

Sieć rekurencyjna dwuwarstwowa



- Neurony 1…M stanowią warstwę wyjściową sieci
- Neurony 1…K warstwę ukrytą
- Wektor wejściowy:
  - sygnały wejściowe x
  - sygnały wyjściowe warstwy ukrytej i wyjściowej

Uniwersytet Przyrodniczy we Wrocławiu

Przykłady zastosowań sieci neuronowych

Funkcje pełnione przez sieć można ująć w kilka podstawowych grup:

- aproksymacji i interpolacji
- rozpoznawania i klasyfikacji wzorców
- kompresji
- predykcji i sterowania
- asocjacji

Sieć neuronowa pełni w każdym z tych zastosowań rolę **uniwersalnego aproksymatora** funkcji wielu zmiennych, realizując **funkcję nieliniową** o postaci **y = f(x),** gdzie x jest wektorem wejściowym, a y realizowaną funkcją wektorową wielu zmiennych.

Duża liczba zadań modelowania, identyfikacji, przetwarzania sygnałów da się sprowadzić do zagadnienia aproksymacyjnego.

Zalety i wady sztucznych sieci neuronowych

Zalety:

- Przetwarzanie równoległe
- Przy dużej liczbie elementów sieć jest odporna na uszkodzenia niewielkiej liczby elementów
- Zdolność uogólniania
- Brak założeń dotyczących rozkładów analizowanych zmiennych

Wady:

- Sieć jako „Czarna skrzynka”