Zagadnienia optymalizacji i aproksymacji.
Sieci neuronowe.

zajecia.jakubw.pl/nai

Literatura:
S. Osowski, *Sieci neuronowe w ujęciu algorytmicznym*. WNT, Warszawa 1997.

PODSTAWOWE ZAGADNIENIA TECHNICZNE AI

- Zadania optymalizacyjne
  – szukanie najlepszego rozwiązania (ocenianego liczbowo)
  – przykłady: minimalizacja kosztu, minimalizacja funkcji błędu, maksymalizacja wygranej (gry logiczne)
- Zadania aproksymacji (ekstrapolacji)
  – znajdowanie zależności między danymi
  – klasyfikacja nieznanych obiektów na podstawie znanych przykładów (rozpoznawanie mowy, OCR, KDD, sterowanie, prognozowanie trendów...)

ZADANIA OPTYMALIZACYJNE

Wiele problemów rozwiązywanych przez komputery ma postać **zadań optymalizacyjnych**:
„znaleźć wśród różnych możliwych rozwiązań takie, które najbardziej nam odpowiada"

Niech X - dowolny zbiór skończony (*przestrzeń stanów*)

Niech $f : X \rightarrow R$ - pewna rzeczywista funkcja na X (*funkcja celu*)

Zadanie optymalizacyjne polega na znalezieniu punktu x_0 ze zbioru X takiego, że:

$$f(x_0) = \max(f(x)), \qquad x \in X$$

lub

$$f(x_0) = \min(f(x)), \qquad x \in X$$

PRZYKŁAD

Posortować *n* nazwisk alfabetycznie (rosnąco).

Przestrzeń stanów: wszystkie możliwe ustawienia *n* nazwisk.
Wielkość przestrzeni stanów: *n!* (**nie *n***).
Przestrzeń stanów to zbiór wszystkich możliwych (również nieoptymalnych) rozwiązań problemu.

Funkcja celu: np. wartość 1 (sukces), jeśli porządek jest właściwy, lub 0 (porażka).
W bardziej złożonych problemach funkcja celu ma zwykle więcej wartości, niż 0 i 1.

Każde dyskretne zadanie optymalizacyjne można rozwiązać przez przejrzenie wszystkich możliwości (wszystkich elementów przestrzeni stanów). Często jednak istnieją skuteczniejsze algorytmy (np. w przypadku sortowania).

ZADANIA APROKSYMACJI

Problemy kojarzone z terminem „sztuczna inteligencja" to często **zadania aproksymacji**:
„mamy daną pewną dziedzinę i niektóre wartości nieznanej funkcji *f,* chcemy zgadnąć wartości *f* w innych punktach"

Niech X - dowolny zbiór (*przestrzeń stanów, uniwersum*)

Niech f : X⟶Y - pewna (nieznana) funkcja na X

Załóżmy, że mamy dany ciąg $(x_1, f(x_1)), \ldots (x_n, f(x_n))$. Zadanie aproksymacji polega tym, by dla dowolnego punktu x_0 ze zbioru X znaleźć wartość y taką, że:

$f(x_0) = y$ lub $|f(x_0) - y|$ było minimalne

lub

$P(f(x_0) = y)$ było maksymalne

PRZYKŁAD

Mamy zbiór obrazków binarnych 32x32 przedstawiających litery (pisane ręcznie). Chcemy rozpoznawać nowe, nieznane wcześniej przypadki (odczytywać nowe napisy).

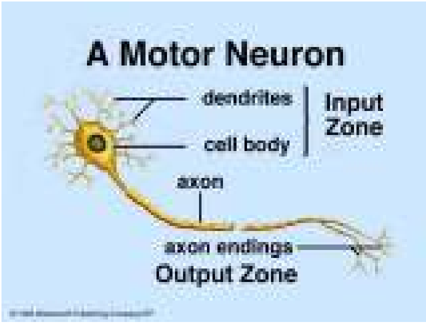
Przestrzeń stanów: wszystkie możliwe obrazki 32x32.
Wielkość przestrzeni stanów: 2^{32*32}.
Przestrzeń stanów to zbiór wszystkich potencjalnych obiektów, jakie mogą pojawić się w zbiorze treningowym i później jako nowe obiekty.

Aproksymowana funkcja: pewna (teoretycznie istniejąca) funkcja przyporządkowująca każdemu możliwemu obrazkowi kod ASCII znaku, który jest na nim przedstawiony.

Nie ma prostych, „siłowych" rozwiązań problemu aproksymacji: jeżeli obiekt testowy x_0 nie występował wśród znanych przykładów, musimy zastosować jakąś metodę uogólnienia tych przykładów.

Sztuczne sieci neuronowe

Geneza: Naśladowanie działania naturalnych neuronów
Cel historyczny: osiągnięcie zdolności uogólniania (aproksymacji) i uczenia się, właściwej mózgowi ludzkiemu.



Perceptron (Rosenblatt 1958)

- Wejście
  - n stanów wejściowych $x_1,\ldots,x_n$
  - stany mogą być cyfrowe lub analogowe
- Wyjście
  - 0 lub 1
- Parametry perceptronu
  - n wag połączeń $w_1,\ldots,w_n \in \Re$
  - wartość progowa $\theta \in \Re$

Uwaga: pod pojęciem "perceptronu" rozumie się też czasem sieć połączonych jednostek (neuronów).

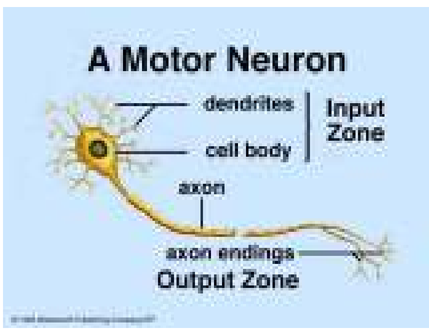
Perceptron

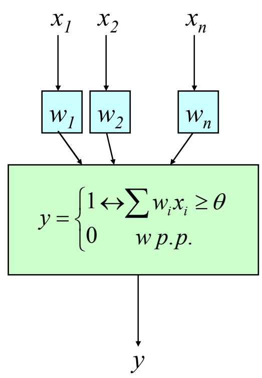
- Zasada działania
  - Do każdego i-tego wejścia przypisana jest waga w_i
  - Dla danych stanów wejściowych $x_1,\dots,x_n$ liczymy sumę ważoną:

$$s = \sum_{i=1}^{n} w_i x_i$$

  - Jeżeli $s \geq \theta$, to ustawiamy wyjście $y = 1$, zaś w przeciwnym przypadku ustawiamy $y = 0$

Analogia z neuronem naturalnym



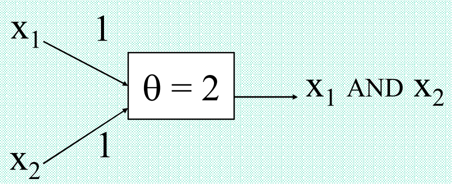


Jak opisać perceptron

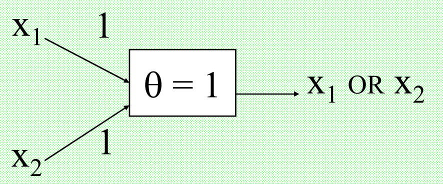
- Perceptron opisuje jednoznacznie zbiór wag $w_1,\ldots,w_n \in \Re$ oraz wartość progowa $\theta \in \Re$
- Wartości $x_1,\ldots,x_n \in \Re$ to zmienne pojawiające się na wejściu do modelu perceptronu

Co potrafi perceptron

x_1	x_2	$x_1 \wedge x_2$
0	0	0
0	1	0
1	0	0
1	1	1

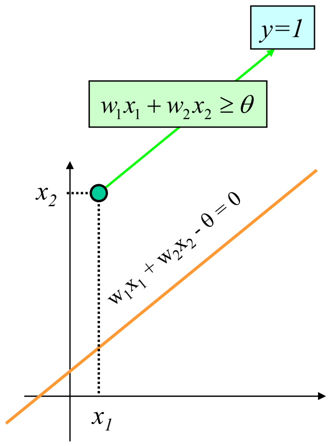


x_1	x_2	$x_1 \vee x_2$
0	0	0
0	1	1
1	0	1
1	1	1



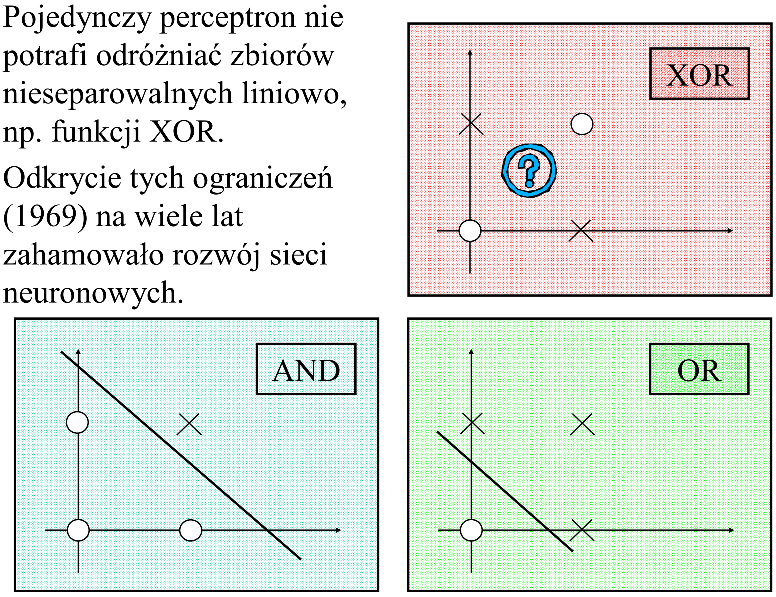
Co potrafi perceptron

- Równanie perceptronu można potraktować jako równanie prostej (ogólnie: hiperpłaszczyzny w przestrzeni n-wymiarowej).
- Punkty leżące nad ową prostą klasyfikujemy jako 1, zaś pozostałe jako 0.



Czego perceptron nie potrafi

- Pojedynczy perceptron nie potrafi odróżniać zbiorów nieseparowalnych liniowo, np. funkcji XOR.
- Odkrycie tych ograniczeń (1969) na wiele lat zahamowało rozwój sieci neuronowych.



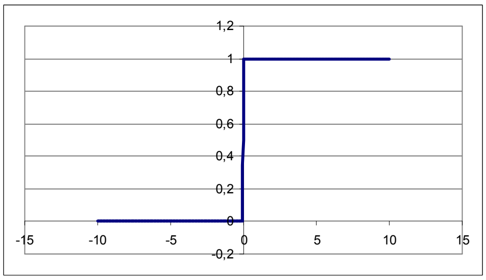
Czego perceptron nie potrafi

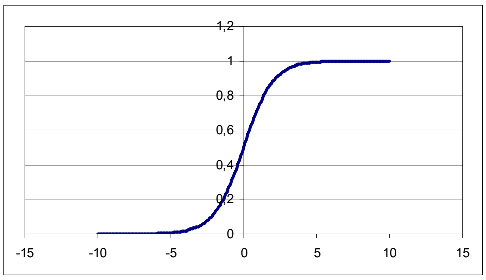
- Zadaniem pojedynczego perceptronu jest jedynie:
  - przetwarzanie jednostkowych informacji
  - podejmowanie prostych decyzji
  - przekazywanie wyników sąsiadom
- Dopiero w połączeniu z innymi węzłami uzyskuje się zdolność podejmowania złożonych decyzji

Funkcje aktywacji

- Progowe

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





- Sigmoidalne

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

Uczenie perceptronu

- Sposób działania perceptronu (wartości wag) w praktycznych problemach nie jest ustawiany ręcznie, tylko **wyuczany** na podstawie przykładów.
- Potrzebujemy zarówno metody uczenia jednego neuronu, jak i procedury obejmującej całą sieć.