Kamil Udziela
Nr. ind.: 286133
Kierunek: Inżynieria Obliczeniowa

AGH

Sprawozdanie:

Scenariusz 1 - Budowa i działanie perceptronu

Spis Treści:

Kamil Udziela
Nr. ind.: 286133
Kierunek: Inżynieria Obliczeniowa

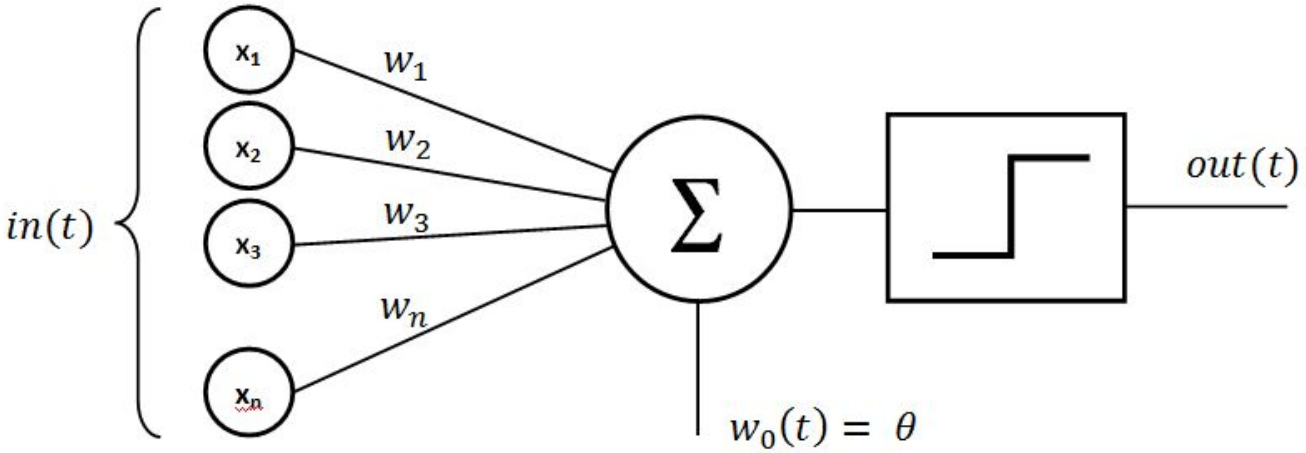
1.Cel ćwiczenia:

Poznanie budowy i działania perceptronu. Implementacja perceptronu, wygenerowanie danych uczących i testujących wybranej funkcji logicznej, uczenie i testowanie perceptronu.

2. Podstawowe zagadnienia:

Perceptron (rys.1)- najprostsza sieć neuronowa, składająca się z jednego bądź wielu niezależnych neuronów.
Działanie perceptronu polega na klasyfikowaniu danych pojawiających się na wejściu i ustawieniu ich stosownie do tego wartości wyjścia.

Sztuczna sieć neuronowa - zbiór neuronów realizujących różne cele.



rys.1 - Budowa perceptronu.

Kamil Udziela
Nr. ind.: 286133
Kierunek: Inżynieria Obliczeniowa

AGH

3. Wykonane zadania:

a) Stworzenie pojedynczego neuronu o dwóch wejściach:
- pierwsze wejście ma zakres 0, 1;
- drugie wejście ma zakres -2, 2;

kod Matlab:

```
net = newp([0 1; -2 2], 1);
```

b) Stworzenie wektorów W1(0 0 1 1),(0 1 0 1) i T1(0 0 0 1). Wektory te opisują działanie bramki logicznej OR.

kod Matlab:

```
W1 = [0 0 1 1; 0 1 0 1];
T1 = [0 0 0 1];
```

c) Inicjalizacja sieci perceptronowej, w której wartości wag i progów są losowe. Użycie funkcji ‘init’

kod Matlab:

```
net = init(net);
```

d) Symulacja sieci przed treningiem. Użycie funkcji ‘sim’

kod Matlab:

```
Symulacja_przed = sim(net, W1)
```

Kamil Udziela
Nr. ind.: 286133
Kierunek: Inżynieria Obliczeniowa

e) Określanie liczby epok, w treningu sieci.

kod Matlab:

net.trainParam.epochs = 20;

f) Trening sieci, wywołanie treningu funkcją train.

kod Matlab:

net = train(net, W1, T1);

g) Symulacja sieci dla tych wartości, które były wyznaczone podczas treningu.

Kod Matlab:

Symulacja_po = sim(net, W1)

Cały kod programu:

```
close all; clear all; clc;

%   tworzenie pojedynczego neuronu o dwóch wejściach.
%   pierwsze wejście ma zakres 0, 1
%   drugie wejscie ma zakres -2, 2
net = newp([0 1; -2 2], 1);

%   %   wektor W2 i T2 opisują działanie bramki OR
W1 = [0 0 1 1; 0 1 0 1];
T1 = [0 0 0 1];

%   Inicjalizacja sieci perceptronowej
%   (wartości wag i progów są losowe)
net = init(net);

%   symulacja sieci przed treningiem
Sym_przed = sim(net, W1)

%   trening sieci ( określamy liczbę epok )
net.trainParam.epochs = 20;


%   wywołanie funkcji 'train'
net = train(net, W1, T1);

%   sumulacja sieci dla tych wartości,
%   które były wyznaczone podczas treningu
Sym_po = sim(net, W1)
```

Kamil Udziela
Nr. ind.: 286133
Kierunek: Inżynieria Obliczeniowa

4.Wyniki i wnioski:

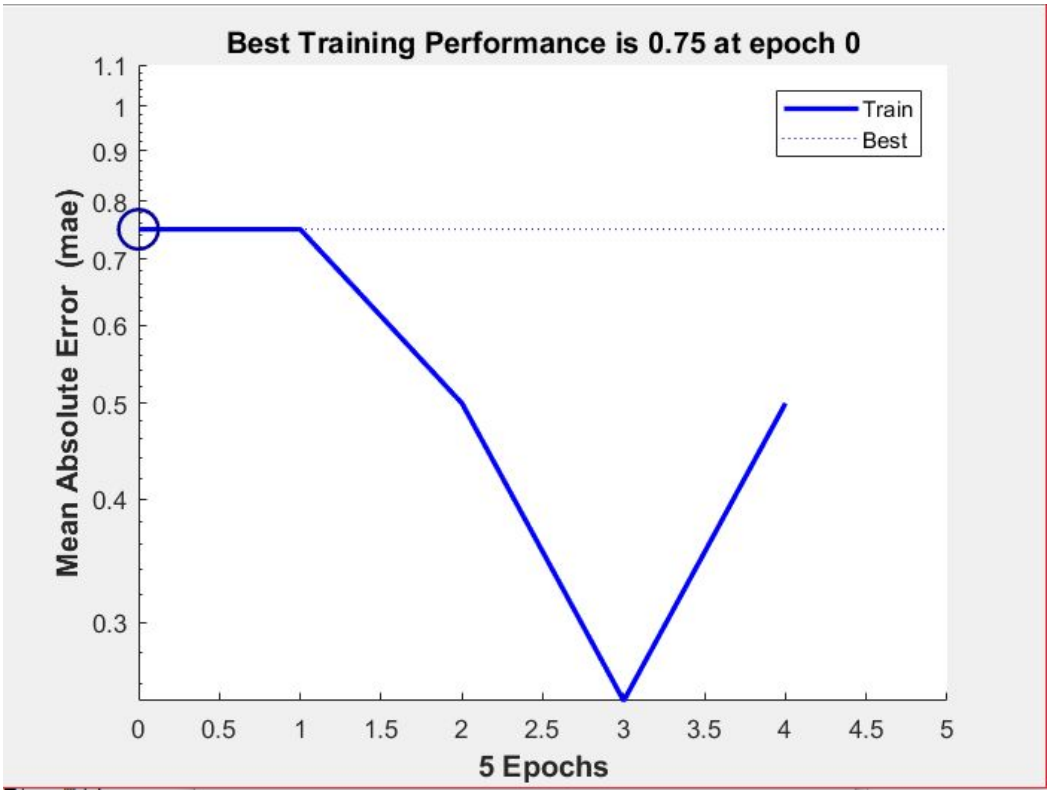
a) Symulacja sieci przed:

Sym_przed = 1 1 1 1

b) Symulacja sieci po:

Sym_po = 0 0 0 1

c) Wykres (najlepsza iteracja) :



Aby zmienić naszą bramkę logiczną należy zmienić wartości W1 i T1. W moim projekcie użyłem bramki OR. Powyższy wykres pokazuje, że proces uczenia w naszym przypadku jest efektywny. Został przeprowadzony już po paru iteracjach. Jeżeli użyjemy innych bramek szybkość i efektywność nauki zmienią się (głównie zmiania się liczba iteracji).