Klaudia Pakos

IO, rok 3, 267561

PODSTAWY SZTUCZNEJ INTELIGENCJI

SPRAWOZDANIE NR 1

BUDOWA I DZIAŁANIE PERCEPTRONU

1. Cel ćwiczenia

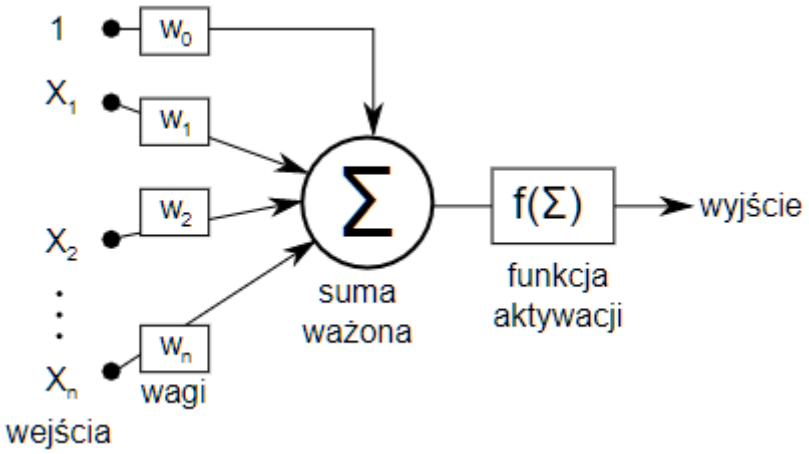
Celem ćwiczenia jest poznanie budowy i działania perceptronu poprzez implementację oraz uczenie perceptronu realizującego wybraną funkcję logiczną dwóch zmiennych.

2. Zadania do wykonania

a) Implementacja sztucznego neuronu wg algorytmu podanego na wykładzie lub dowolnego innego (z podaniem źródła).
b) Wygenerowanie danych uczących i testujących wybranej funkcji logicznej dwóch zmiennych.
c) Uczenie perceptronu dla różnej liczby danych uczących, różnych współczynników uczenia.
d) Testowanie perceptronu.

3. Opis budowy perceptronu

Neuron McCullocha-Pittsa – jest to jeden z matematycznych modeli neuronu oraz podstawowy budulec sieci neuronowej (perceptron). Posiada wiele wejść gdzie każde ma swoją wagę (tzw. wagę wejścia), oraz jedno wyjście którego waga jest obliczona dodatkowo za pomocą wzoru (funkcja aktywacji).



4. Przebieg zadania

W przedstawionym poniżej sprawozdaniu przedstawiono sieć neuronową uczącą się funkcji logicznej OR.

Bramka OR (suma logiczna)

a	b	x
0	0	0
0	1	1
1	0	1
1	1	1

Na początku stworzono pojedynczy neuron o zakresach [0,1] [0,2]. Następnie stworzono dwa wektory o wartościach odpowiadających bramce logicznej OR przedstawionej powyżej, a także odpowiadający im wektor wynikowy. Następnie zainicjalizowano sieci perceptronowe z losową wartością wag i wywołano symulacje przed treningiem. Przetestowano również działanie sieci dla innych par wektorów. Kolejno ustawiono 15 epok do określenia treningu sieci i uruchomiono owy trening. Po nim przeprowadzono symulację sieci dla wag określonych po treningu. Wykonano także kolejny test tych samych wektorów co poprzednio.

Kod programu wraz komentarzami przedstawia się następująco:

```
%%
%OR
close all; clear all; clc;

%stworzenie pojedynczego neuronu o zakresach [0,1] [0,2]
net=newp([0 1;0 2],1);
%stworzenie dwóch wektorów, odpowiadających wszystkim
możliwym wejściom
%bramki logiczniej OR
A=[0 0 1 1];
B=[0 1 0 1];
%połączenie ich w jedną macierz
P=[A;B];
%stworzenie wektora  opisującego wartości wynikowe
odpowiadające bramce
%logicznej NOR dla powyższych wektorów A i B
T=[0 1 1 1];
%inicjalizacja sieci perceptronowe z losową wartością wag
net=init(net);
%wywołanie symulacji przed treningiem
Przed_treningiem=sim(net,P)
%przetestowanie działania sieci dla innych wektorów
test1=sim(net,[1 0 1 1;0 0 1 1])
test2=sim(net,[0 1 0 0;0 0 0 0])
%ustawienie 15 epok do określenia treningu sieci
net.trainParam.epochs=15;
```

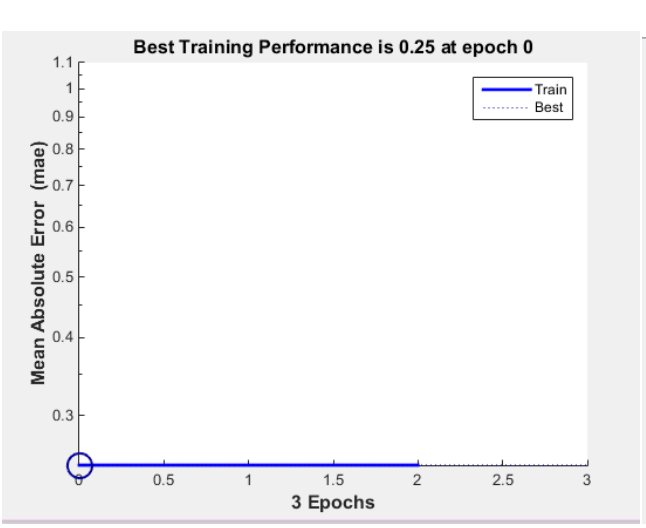
```
%uruchomienie treningu
net=train(net,P,T);
%symulacja sieci neuronowej dla wartości określonych po
treningu
Po_treningu=sim(net,P)
%ponowne przetestowanie dla tych samych wektorów, co
poprzednio
test1po=sim(net,[1 0 1 1;0 0 1 1])
test2po=sim(net,[0 1 0 0;0 0 0 0])
```

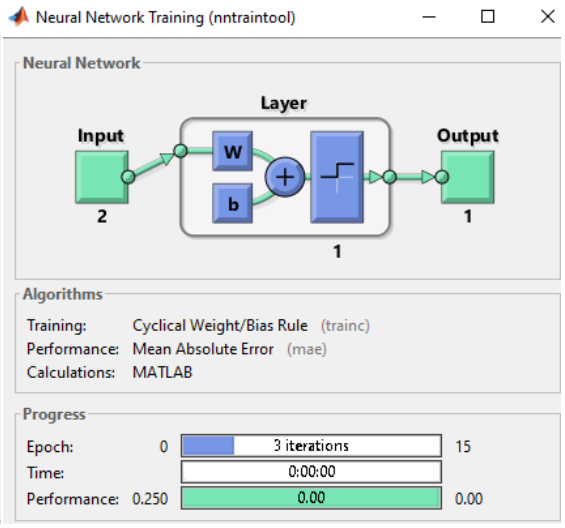
5. Przedstawienie wyników

Wykonano kilka testów różnej ilości danych, których wyniki przedstawiono w poniższej tabeli, wartości błędne zaznaczono na czerwono:

Wektory	**Wektory początkowe**			
	Wektory1	**Wektory2**	**Wektory3**	**Wektory4**
	[0 0 1 1;0 1 0 1]	**[0 0 1 1 1 1; 0 1 0 1 1 0]**	**[0 0; 0 1]**	**[0 0 1; 0 1 1]**
Wynik przed treningiem	[1 1 1 1]	[1 1 1 1]	[1 1]	[1 1 1]
Wynik po treningu	[0 1 1 1]	[0 1 1 1 1 1]	[0,1]	[0 1 1]
błąd	0.25	0.167	0.5	0.33
iteracje	3	3	2	2
Test 1: [1 0 1 1;0 0 1 1] po treningu	[1 0 1 1]	[1 0 1 1]	[0 0 1 1]	[0 0 1 1]
Test2: [0 1 0 0;0 0 0 0] po treningu	[0 1 0 0]	[0 1 0 0]	[0 0 0 0]	[0 0 0 0]

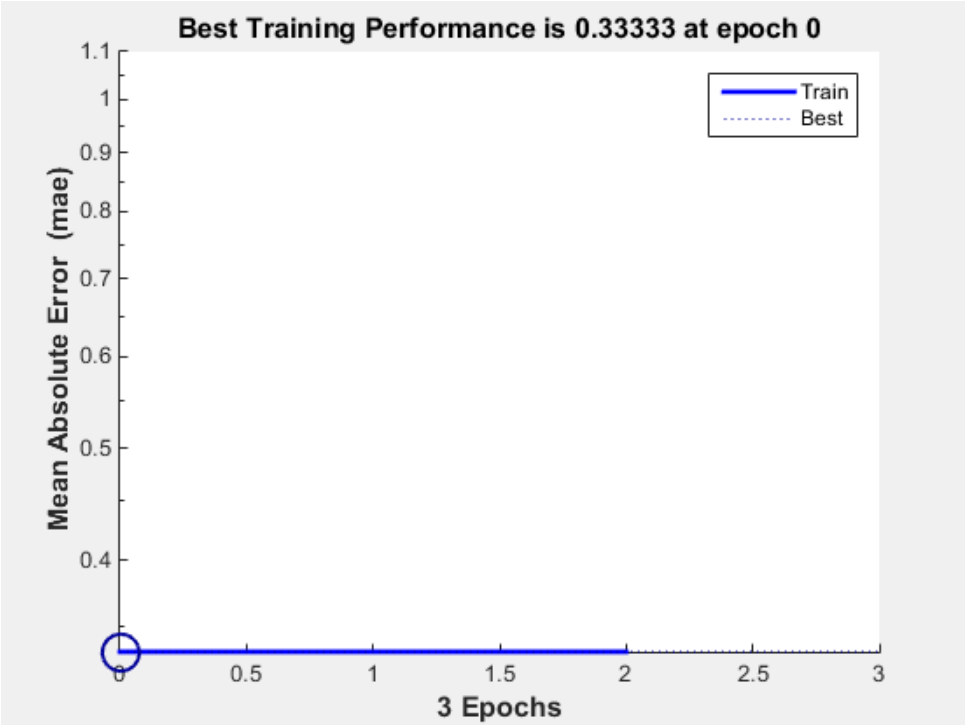
Dla zbioru Wektory1 otrzymano następujący wykres postępu, gdzie widać, że błąd faktycznie jest niewielki i 3 iteracje zupełnie wystarczają do otrzymania poprawnych wyników. Przedstawiony czas był identyczny dla wszystkich zbiorów wektorów:



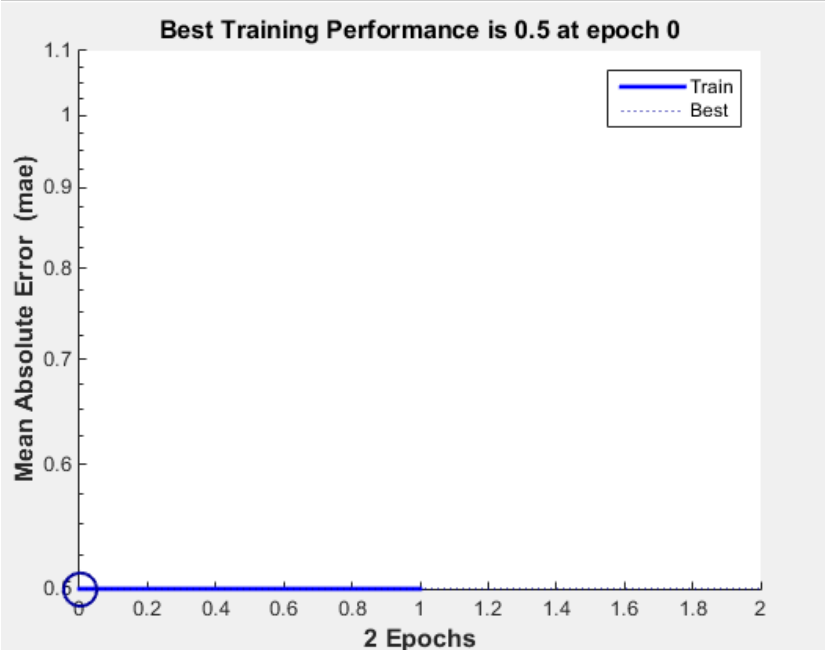


Dla zbioru Wektory2 otrzymano następujący wykres postępu, gdzie widać, że błąd faktycznie jest jeszcze mniejszy niż poprzednio i 3 iteracje zupełnie wystarczają do otrzymania poprawnych wyników :

Dla zbioru Wektory4 otrzymano następujący wykres postępu, gdzie widać, że błąd jest nieduży, a 3 iteracje wystarczają, jednakże przeprowadzone testy wykazały, że sieć popełnia błędy ze względu na brak rozpatrzonych wszystkich możliwych przypadków funkcji logicznej OR:



Dla zbioru Wektory3 otrzymano następujący wykres postępu, gdzie widać, że błąd jest największy, i 2 iteracje wystarczają, jednakże przeprowadzone testy wykazały, że sieć popełnia błędy ze względu na brak znajomości wszystkich możliwych przypadków funkcji logicznej OR:



6. Wnioski

Podczas wykonywania zadania zauważono uczenie się perceptronu rozwiązywania bramki logicznej OR. Perceptron nie wykorzystał w żadnym z testów założonej wartości epok treningowych. Zauważono również, że przy podaniu zbyt małej ilości danych wejściowych i nie rozpatrzeniu wszystkich możliwych przypadków stworzona sieć myli się symulując dane końcowe. Minimalnym wymiarem wektorów wejściowych jest 4, mniejsze dają błędne wyniki. Jeśli natomiast zwiększymy wymiar wektorów wejściowych udaje nam się zmniejszyć błąd otrzymywany w trakcie symulacji, wyniki ostateczne są wówczas również prawidłowe- czyli zabieg taki ulepsza działanie sieci neuronowej Nie zauważono przeuczenia się siec, na wykresach nie widać w żadnym punkcie wzrostu błędu.