

Poznanie działania reguły Hebb'a na przykładzie rozpoznawania emotikon.

1. Cel projektu

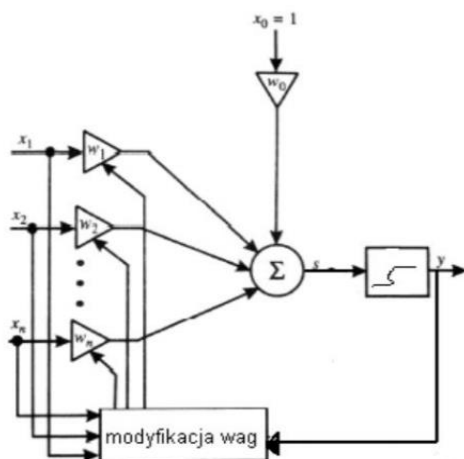
Celem ćwiczenia jest wygenerowanie danych uczących i testujących, zawierających 4 różne emotikony np. czarnobiałe, wymiar 8x8 pikseli dla jednej emotikony. Przygotowanie sieci oraz reguły Hebb'a w wersji z i bez współczynnika zapominania. Uczenie sieci dla różnych współczynników uczenia i zapominania. Testowanie sieci.

2. Podstawowe pojęcia

Sieć neuronowa (sztuczna sieć neuronowa) – ogólna nazwa struktur matematycznych i ich programowych lub sprzętowych modeli, realizujących obliczenia lub przetwarzanie sygnałów poprzez rzędy elementów, zwanych sztucznymi neuronami, wykonujących pewną podstawową operację na swoim wejściu.

Jest to zbiór neuronów realizujących różne cele. W przypadku sztucznych sieci neuronowych jest to sztuczna struktura zaprojektowana i zbudowana w taki sposób, aby modelowała działanie naturalnego układu nerwowego, a w szczególności mózgu.

Reguła Hebb'a - Polega na tym, że sieci dostarcza się kolejne przykłady sygnałów wejściowych, nie podając informacji o tym, co z tymi sygnałami należy zrobić.



$$w_i(t+1) = w_i(t) + \eta y x_i$$

gdzie:

- i -numer wagi neuronu,
- t -numer iteracji w epoce,
- y -sygnał wyjściowy neuronu,
- x -wartość wejściowa neuronu,
- η - współczynnik uczenia (0,1).

Sieć obserwując otoczenie odbiera różne sygnały, nikt nie określa jednak, jakie znaczenie mają pokazujące się obiekty i jakie są pomiędzy nimi zależności, ponieważ sieć sama określa po czasie jakie jest znaczenie poszczególnych sygnałów. Modyfikację wag przeprowadza się następująco:

Po odczytaniu przez sieć neuronową każdego z zestawów sygnałów wejściowych tworzy się w tej sieci rozkład sygnałów wyjściowych, w którym niektóre neurony sieci są pobudzone bardzo silnie, a inne słabiej.

Niestety ten proces uczenia jest wolniejszy w porównaniu do nauki z nauczycielem. Nie wiadomo również, który neuron będzie rozpoznawał którą klasę neuronów (ważniejszą lub mniej ważną).

Ważne jest, aby dobrze wybrać początkowe wartości wag neuronów.

3. Opis działania

Wygenerowanie danych uczących – 4 wybranych emotikon.

Kolor żółty na rysunku pierwszym oznacza jedynki, a białe zera w macierzy reprezentującej emotikony.

:O

0	0	0	0	0
0	1	0	1	0
0	0	0	0	0
0	1	1	1	0
0	1	1	1	0

:)

0	0	0	0	0
0	1	0	1	0
0	0	0	0	0
1	0	0	0	1
0	1	1	1	0

:(

0	0	0	0	0
0	1	0	1	0
0	0	0	0	0
0	1	1	1	0
1	0	0	0	1

:I

0	0	0	0	0
0	1	0	1	0
0	0	0	0	0
0	1	1	1	0
0	0	0	0	0

Wygenerowanie danych wejściowych w postaci tabeli „WEJSCIE”, w której każdy wektor (emotikona) zajmował jedną kolumnę.

Wykorzystanie funkcją ‘newff’ ze zmiennymi parametrami uczenia (epokami, średnim błędem kwadratowym oraz współczynnikiem zapominania jak i współczynnikiem uczenia).

Przeprowadzenie uczenia sieci według zasad Hebba przy zmiennych parametrach oraz w wersji z i bez współczynnika zapominania.

4. Wykonanie zadania i kod

```
close all; clear all; clc;

%wejścia do sieci oraz minimalne oraz maksymalne wartości
%wejść
%(25 par 0&1 - osobno dla każdej z danych uczących)

start=[0 1; 0 1; 0 1; 0 1; 0 1; 0 1; 0 1; 0 1; 0 1; 0 1;
       0 1; 0 1; 0 1; 0 1; 0 1; 0 1; 0 1; 0 1; 0 1; 0 1;
       0 1; 0 1; 0 1; 0 1; 0 1;];

%ilość wyjść z sieci (jedna warstwa - 25 neuronów na
%wyjściu)
wyjścia_s = 25;

%użycie funkcji newff
net = newff(start, wyjścia_s, {'tansig'}, 'trainlm',
'learnh');

%kolumnowa reprezentacja binarna 4 emotikonów dla tablicy
8x4
WEJSCIE = %.):O:(:|
[ 0 0 0 0
  0 0 0 0
  0 0 0 0
  0 0 0 0
  0 0 0 0
  0 0 0 0
  1 1 1 1
  0 0 0 0
  1 1 1 1
  0 0 0 0
  0 0 0 0
  0 0 0 0
  0 0 0 0
  0 0 0 0
  0 0 0 0
  1 0 0 0
  0 1 1 1
  0 1 1 1
  0 1 1 1
  1 0 0 0
  0 0 1 0
  1 1 0 0
  1 1 0 0]
```

```

        1 1 0 0
        0 0 1 0
    ];
    %zmienna, ktora reprezentuje, czy uzytkownik "trafil" w
    wybrana przez
    %siebie litere - 1 oznacza trafienie, 0 - chybiecie
    WYJSCIE = [1 0 0 0 ; %:)
               0 1 0 0 ; %:O
               0 0 1 0 ; %:(
               0 0 0 1 ]; %:|

    %PARAMETRY ALGORYTMU HEBBA
    % * wspolczynnik zapominania
    lp.dr = 0.5;
    % * wspolczynnik uczenia
    lp.lr = 0.99;

    %dostosowanie parametrów sieci do metody Hebba
    wagiHebba = learnh([], WEJSCIE, [], [], WYJSCIE, [], [], [],
    [], [], lp, []);

    %PARAMETRY TRENINGU SIECI:
    % * maksymalna ilosc epok
    net.trainParam.epochs = 1000;
    % * cel wydajnosci sieci
    net.trainParam.goal = 0.001;
    % * wskaznik uczenia sieci
    net.trainParam.lr=0.5;
    whebb=wagiHebba';
    net = train(net, WEJSCIE, whebb);

    %dane testowe
    a_testowe= [0;0;0;0;0;
                0;1;0;1;0;
                0;0;0;0;0;
                1;0;0;0;1;
                0;1;1;1;0]; %:)

    b_testowe=[0;0;0;0;0;
                0;1;0;1;0;
                0;0;0;0;0;
                0;1;1;1;0;
                0;1;1;1;0]; %:O

    c_testowe=[0;0;0;0;0;
                0;1;0;1;0;
                0;0;0;0;0;
                0;1;1;1;0;
                0;1;1;1;0;

```

```

1;0;0;0;1;]; %:(

d_testowe=[0;0;0;0;0;
            0;1;0;1;0;
            0;0;0;0;0;
            0;1;1;1;0;
            0;0;0;0;0;]; %:|

efekt = wagiHebba;
%symulacja sieci net
efekt_1 = sim(net, a_testowe);

%wypisywanie wartosci reguly Hebba, wypisywanie kolejnych
wierszy
disp('Jednokrotne wykorzystanie reguly Hebba: ')
disp(':) = '), disp(sum(efekt(1, ':')));
disp(':O = '), disp(sum(efekt(2, ':')));
disp(':( = '), disp(sum(efekt(3, ':')));
disp(':| = '), disp(sum(efekt(4, ':')));

%wypisywanie wartosci
disp('Dzialanie algorytmu z wykorzystaniem r. Hebba dla
emotikon: ')
disp(':) = '), disp(efekt_1(1));
disp(':O = '), disp(efekt_1(2));
disp(':( = '), disp(efekt_1(3));
disp(':| = '), disp(efekt_1(4));

```

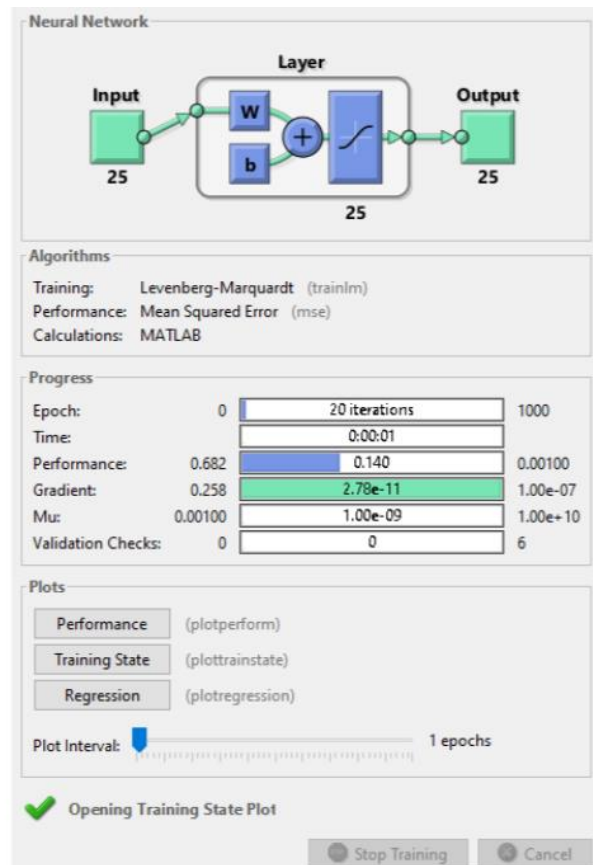
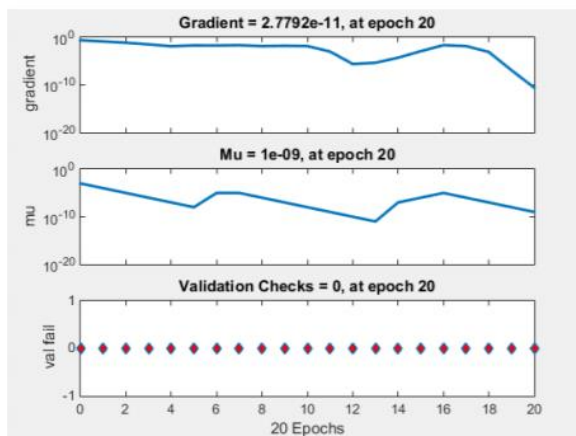
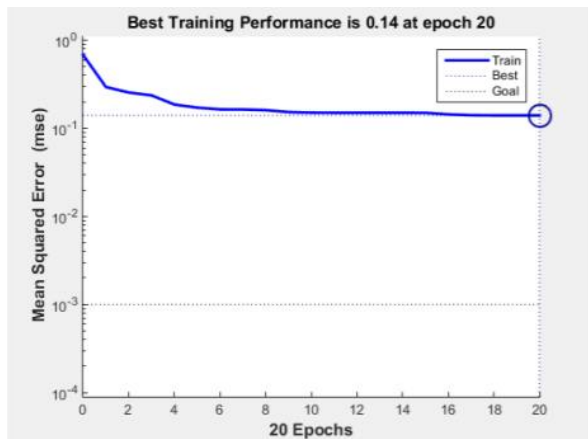
5. Wnioski

Reguła Hebba pozwala na uczenie sieci bez nauczyciela. Ze względu na użyty algorytm można zauważyć, że sieć w porównaniu do wcześniejszych metod użytych na zajęciach zdecydowanie potrzebuje więcej czasu na naukę. Przebieg uczenia zależy od wartości początkowych wag.

Reguła Hebba zawiera wiele ograniczeń.

- Nie ma gwarancji, że jednej klasie wzorców będzie odpowiadał jeden neuron
- Nie ma gwarancji, że wszystkie klasy wzorców będą miały oddzielne reprezentacje w postaci oddzielnych zbiorów aktywnych neuronów.

Zmiany wag w dużym stopniu zależą od współczynnika zapominania. Gdy go nie mamy nie ma stabilizacji i wagi mogą rosnąć w nieskończoność



współczynnik zapominania = 0.5

współczynnik uczenia = 0.99

maksymalna ilość epok= 1000

cel wydajności sieci = 0.001

wskaźnik uczenia sieci=0.5