Anna Szylak
IO rok 3, WIMiIP

Scenariusz nr 3

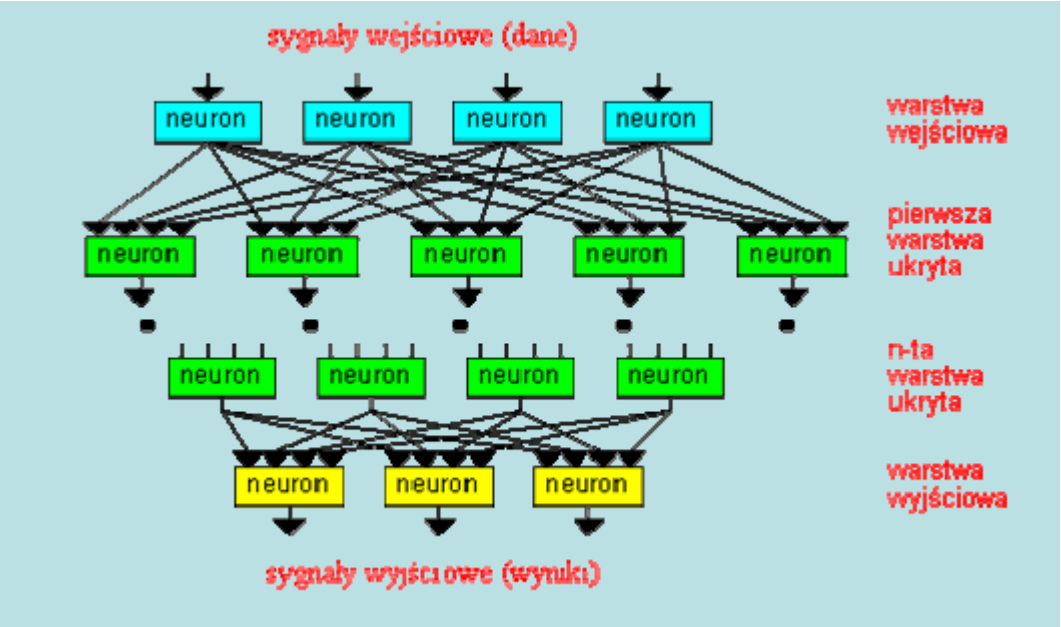
Budowa i działanie sieci wielowarstwowej typu feedforward

1. Cel ćwiczenia

Celem ćwiczenia jest poznanie budowy i działania wielowarstwowych sieci neuronowych poprzez uczenie kształtu funkcji matematycznej z użyciem algorytmu wstecznej propagacji.

2. Teoria

- Sieci wielowarstwowe to odpowiednio połączone warstwy neuronów zwykle o nieliniowych funkcjach aktywacji (np. neurony sigmoidalne, radialne), aczkolwiek czasami w warstwie wyjściowej pojawiają się neurony liniowe. Sieci wielowarstwowe muszą posiadać minimalnie dwie warstwy neuronów: warstwę wejściową i warstwę wyjściową pomiędzy którymi może być jedna lub więcej warstw ukrytych. W każdej warstwie może występować różna ilość neuronów: W warstwie wejściowej odpowiada ona zwykle ilości parametrów opisujących dane wejściowe, w warstwie wyjściowej np. ilości klas, natomiast ilość neuronów w warstwach ukrytych odpowiada za możliwości modelu. Neurony łączą się tylko pomiędzy (zwykle) sąsiednimi warstwami, zaś wewnątrz warstw nie występują połączenia pomiędzy neuronami. Neurony zwykle łączymy pomiędzy sąsiednimi warstwami na zasadzie każdy z każdym.



- Algorytm wczesnej propagacji błędów:

  BP (ang. BackPropagation) określa strategię doboru wag w sieci wielowarstwowej przy wykorzystaniu gradientowych metod optymalizacji. Podczas procesu uczenia sieci dokonuje się prezentacji pewnej ilości zestawów uczących (tzn. wektorów wejściowych oraz odpowiadających im wektorów sygnałów wzorcowych (wyjściowych)). Uczenie polega na takim doborze wag neuronów by w efekcie końcowym błąd popełniany przez sieć był mniejszy od zadanego. Nazwa "wsteczna propagacja" pochodzi od sposobu obliczania błędów w poszczególnych warstwach sieci. Najpierw obliczane są błędy w warstwie ostatniej (na podstawie sygnałów wyjściowych i wzorcowych. Błąd dla neuronów w dowolnej warstwie wcześniejszej obliczany jest jako pewna funkcja błędów neuronów warstwy poprzedzającej. Sygnał błędu rozprzestrzenia się od warstwy ostatniej, aż do warstwy wejściowej, a więc wstecz.

3. Kod programu

Główny plik:

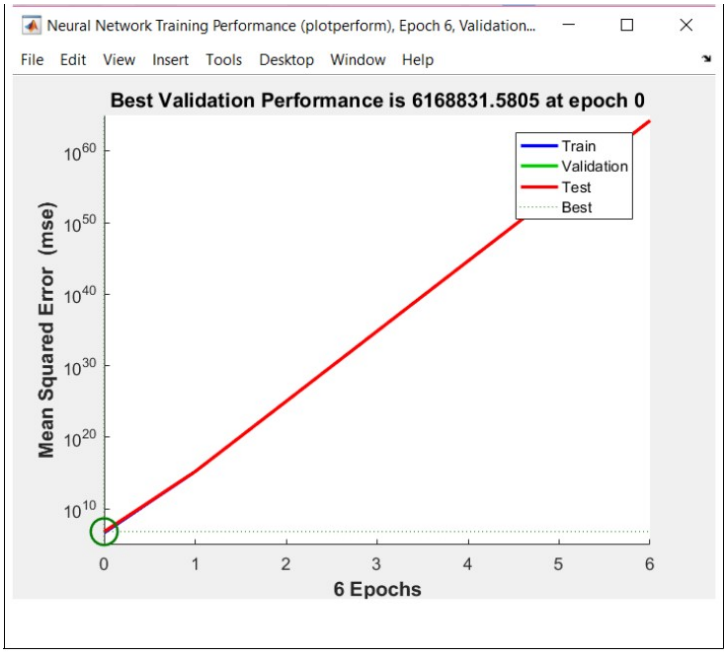
```
close all; clear all; clc;
%Poprawne dane wejsciowe i wyjsciowe
data_in = [0.4 0.8 1.2 1.6 -2 -1.6 -1.2 -0.8 -0.4 0];
data_out = [1949.5 1982.5 2147.7 2179.1 1663.3 1688.0 1686.7 1776.4
1780.0 0.0];
%tworzenie sieci i jej ustawianie
N = feedforwardnet(5);
%uzycie algorytmu wstecznej propagacji
N.trainFcn = 'traingd';
%wspolczynnik uczenia:
N.trainParam.lr = 0.3;
%bezwladnosc
N.trainParam.mc = 0.5
%trenowanie;
N = train(N,data_in, data_out);
%miejsce na wyniki
result = zeros(size(N));

%Testowanie sieci
while n < 10
      n=n+1;
      result(n) = sim(N, data_in(i));
end
```

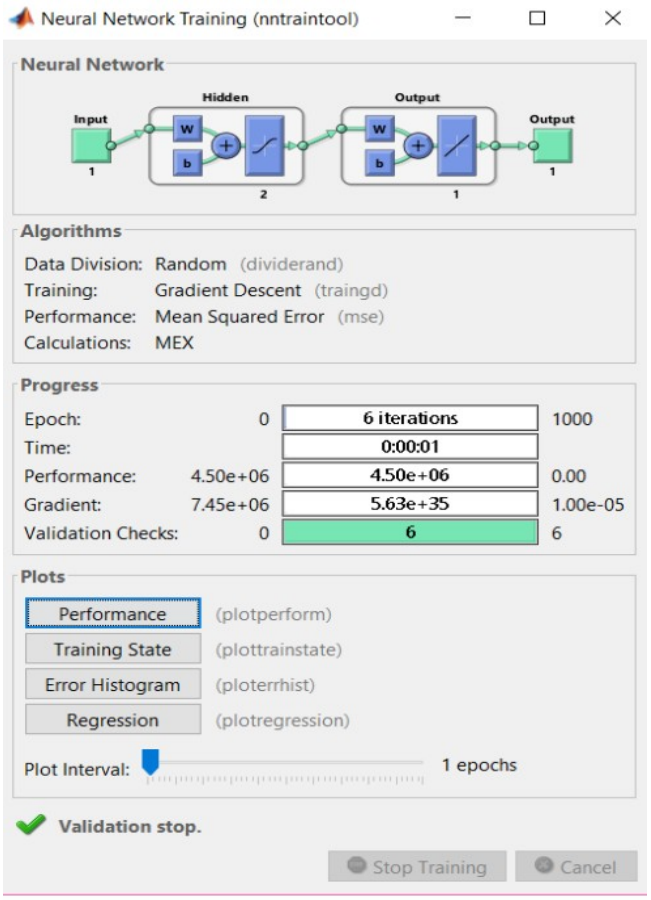
Funkcja rastrigin:

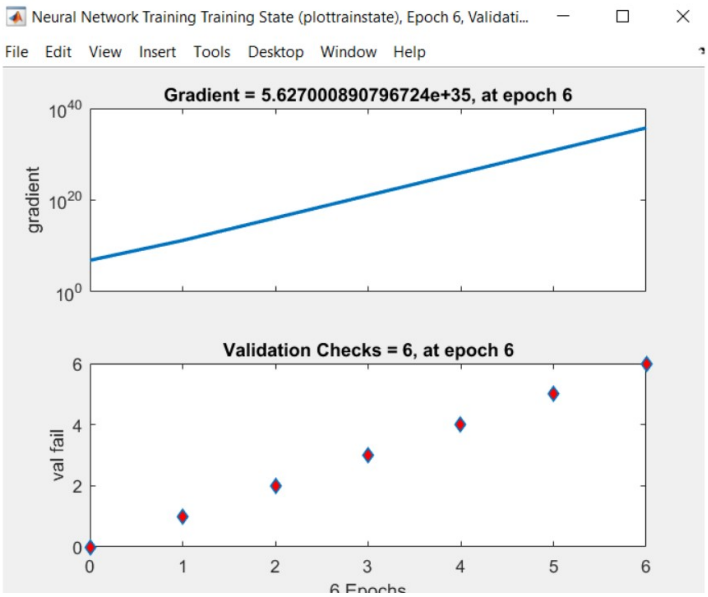
```
function fx = RastrignTest3D(x)
    if x == 0
        fx = 0;
    else
        A = 10;
        n = 100;
        x1 = x;
        dx = (5.12-x)/n;
        fx = A * n;
        for i = 1:n
            x = x1 + (i * dx);
            fx = fx + (x^2) - (A * cos(2 * pi * x));
        end
    end
end
```

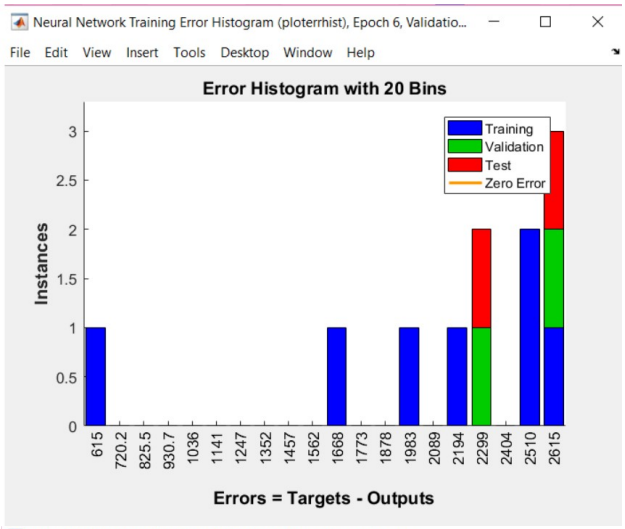
4. Wyniki

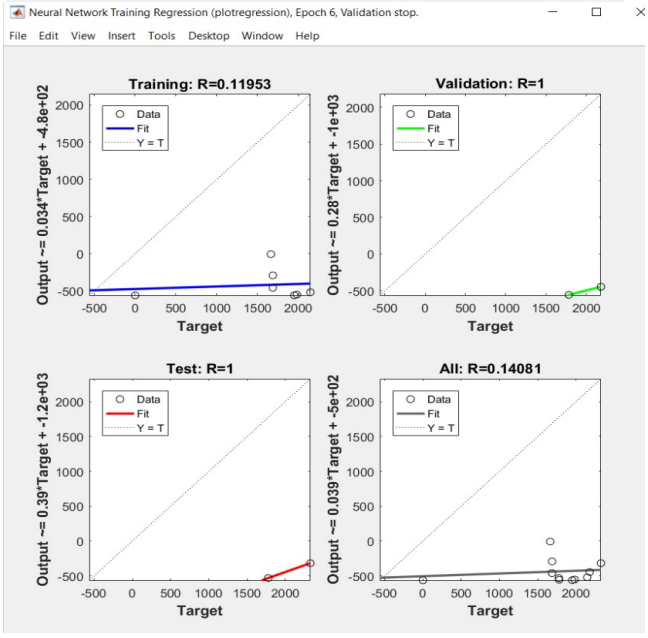
Współczynnik uczenia: 0.1
Bezwładność: 0.5






Neural Network Training Training State (plottrainstate), Epoch 6, Validati...
File Edit View Insert Tools Desktop Window Help
Gradient = 5.627000890796724e+35, at epoch 6
gradient
Validation Checks = 6, at epoch 6
val fail
6 Epochs


Neural Network Training Error Histogram (ploterrhist), Epoch 6, Validatio...
File Edit View Insert Tools Desktop Window Help
Error Histogram with 20 Bins
Training
Validation
Test
Zero Error
Instances
Errors = Targets - Outputs


Neural Network Training Regression (plotregression), Epoch 6, Validation stop.
File Edit View Insert Tools Desktop Window Help
Training: R=0.11953
Validation: R=1
Test: R=1
All: R=0.14081
Data
Fit
Y = T
Target

Reszta wyników:

Wsp. uczenia	0,5			0,1			0,01			prawidłowe
Bezwladnosc	0	0,5	1	0	0,5	1	0	0,5	1	1663
2	2396	2461	1931	1923	2287	1650	1886	1898	2190	1688
-1,6	2532	2374	1824	1821	2341	1706	1844	1845	2200	1687
-1,2	2613	2322	1761	1761	2373	1739	1820	1814	2206	1776
-0,8	2646	2300	1735	1735	2387	1753	1809	1801	2208	1780
-0,4	2658	2292	1726	1726	2391	1758	1805	1797	2209	1867
0	2974	2189	332,2	285,1	2423	2857	1804	3590	2173	1950
0,4	2660	2284	1726	1718	2394	1763	1802	1798	2208	1983
0,8	2654	2275	1735	1707	2394	1769	1797	1805	2205	2148
1,2	2637	2248	1761	1680	2395	1786	1785	1824	2197	2179
1,6	2594	2185	1825	1614	2397	1825	1757	1869	2177	2326
2	2,522	2079	1932	1502	2401	1892	1708	1947	2144	

5. Wnioski

Sieci wielowarstwowe nie są pozbawione błędów, podobnie w stosunku do swoich prostszych, jednowarstwowych odpowiedników. Podobnie jak sieciami jednowarstwowymi, możemy sterować dopasowując poszczególne parametry uczenia, takie jak współczynnik uczenia, czy bezwładność.

Najgorsze wyniki osiągają te sieci, których wsp.uczenia jest równy bezwładności.

Sieci wielowarstwowe posiadające algorytm propagacji danych wykazują praktycznie duże odsetki błędów. W tym przypadku im większy jest wsp. uczenia od bezwładności, tym częściej mogą zdarzać się błędy.

Gradienty uczenia przy sieciach z algorytmem propagacji błędów posiadają charakterystykę niemalże liniową, dzięki czemu można łatwiej przewidzieć, jakich współczynników do nauki sieci najlepiej użyć, by otrzymane rezultaty były zadowalające.