Podstawy Sztucznej Inteligencji

Przemysław Bartkowski Gr 1 Inżynieria Obliczeniowa

Budowa i działanie sieci wielowarstwowej typu feedforward

Cel ćwiczeń:

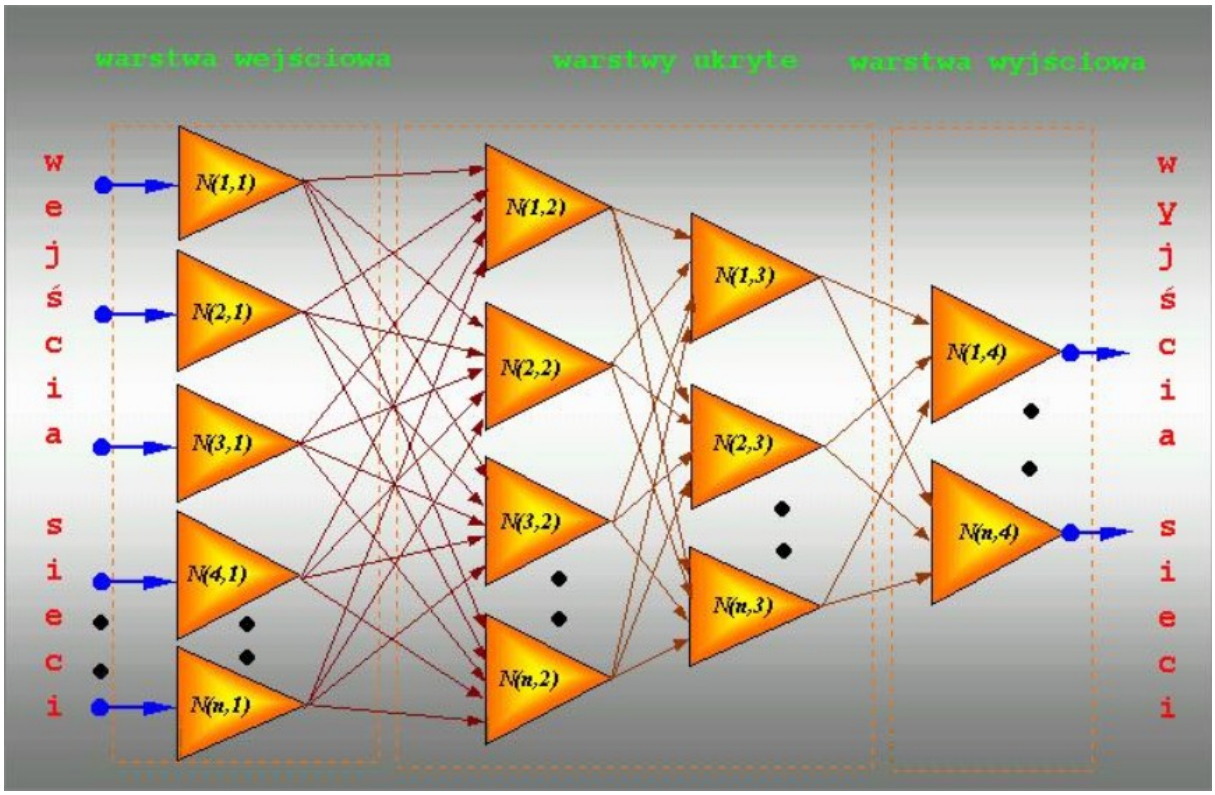
Poznanie budowy i działania wielowarstwowych sieci neuronowych poprzez uczenie kształtu funkcji matematycznej z użyciem algorytmu wstecznej propagacji błędu.

Zadania do wykonania:

- Wygenerowanie danych uczących i testujących dla funkcji Rastrigin 3D dla danych wejściowych z przedziałów od-2 do 2.
- Implementacja wielowarstwowej sieci oraz algorytmy wstecznej propagacji błędu.
- Uczenie sieci dla różnych współczynników uczenia i bezwładności.
- Testowanie sieci.

Zagadnienia teoretyczne:

Sieci jednokierunkowe (feedforward) - Sieci jednokierunkowe wielowarstwowe cieszą się największym zainteresowaniem spośród wszystkich znanych rodzajów sztucznych sieci neuronowych. Spowodowane jest to prostą ich strukturą - łatwą do opisania, jak również prostymi i łatwymi do realizacji metodami uczenia tychże sieci. Należy podkreślić, że taka struktura sieci jest zbliżona do budowy mózgu, który również posiada strukturę warstwową, w dużej części jednokierunkową.



Metoda wstecznej propagacji błędów – Pierwszą czynnością w procesie uczenia jest przygotowanie dwóch ciągów danych: uczącego i weryfikującego. Ciąg uczący jest to zbiór takich danych, które w miarę dokładnie charakteryzują dany problem. Jednorazowa porcja danych nazywana jest wektorem uczącym. W jego skład wchodzi wektor wejściowy czyli te dane wejściowe, które podawane są na wejścia sieci i wektor wyjściowy czyli takie dane oczekiwane, jakie sieć powinna wygenerować na swoich wyjściach. Po przetworzeniu wektora wejściowego, nauczyciel porównuje wartości otrzymane z wartościami oczekiwanymi i informuje sieć czy odpowiedź jest poprawna, a jeżeli nie, to jaki powstał błąd odpowiedzi. Błąd ten jest następnie propagowany do sieci ale w odwrotnej niż wektor wejściowy kolejności (od warstwy wyjściowej do wejściowej) i na jego podstawie następuje taka korekcja wag w każdym neuronie, aby ponowne przetworzenie tego samego wektora wejściowego spowodowało zmniejszenie błędu odpowiedzi. Procedurę taką powtarza się do momentu wygenerowania przez sieć błędu mniejszego niż założony. Wtedy na wejście sieci podaje się kolejny wektor wejściowy i powtarza te czynności. Po przetworzeniu całego ciągu uczącego (proces ten nazywany jest epoką) oblicza się błąd dla epoki i cały cykl powtarzany jest do momentu, aż błąd ten spadnie poniżej dopuszczalnego. Jak to już było zasygnalizowane wcześniej, SSN wykazują tolerancję na nieciągłości, przypadkowe zaburzenia lub wręcz niewielkie braki w zbiorze uczącym. Jest to wynikiem właśnie zdolności do uogólniania wiedzy.

Funkcja obliczająca błąd:

$$d(w_1,\ldots,w_K)=\frac{1}{2}\left(f(w_1z_1+\ldots+w_Kz_K)-t\right)^2$$

Rozwiązanie problemu:

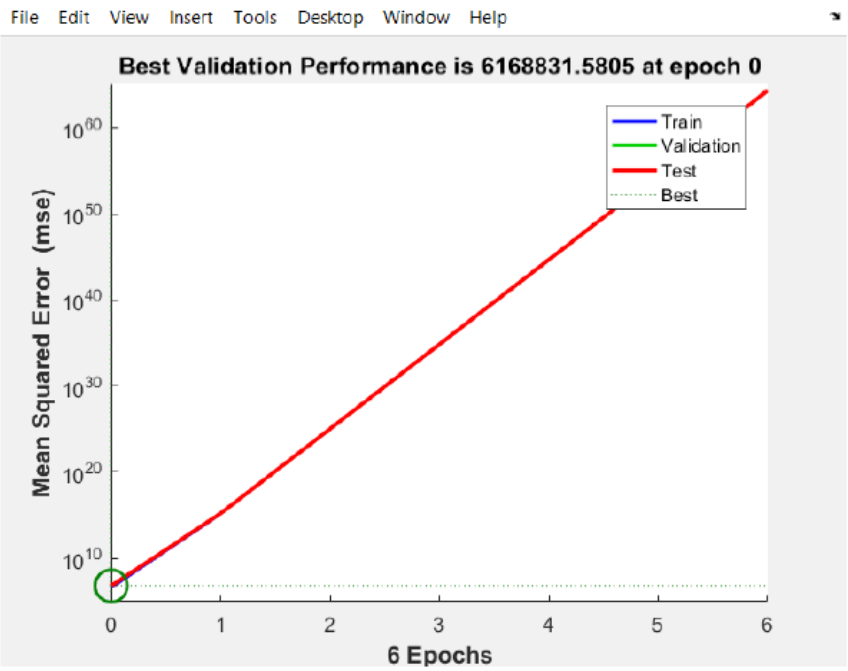
Do wykonania zadania został wykorzystany pakiet Matlab oraz narzędzie Neural Networking Training Tool. Narzędzie to pozwala na utworzenie bardziej złożonych sieci korzystając z modelów wielowarstwowych. Funkcja Rastrigin 3D przyjmowała wartości:

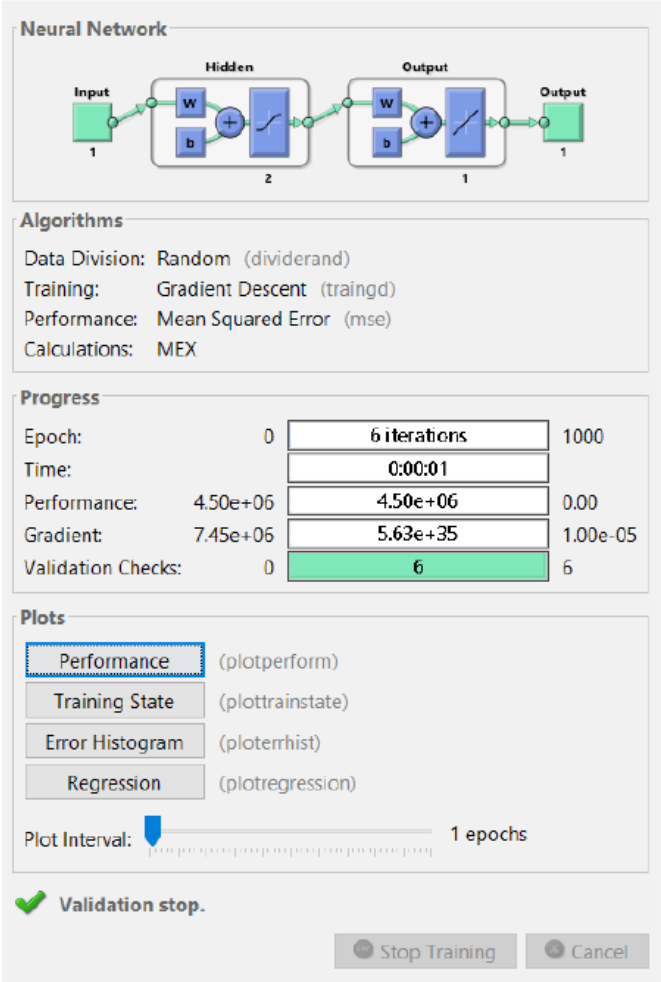
$$f(\mathbf{x}) = An + \sum_{i=1}^{n} \left[x_i^2 - A\cos(2\pi x_i)\right]$$

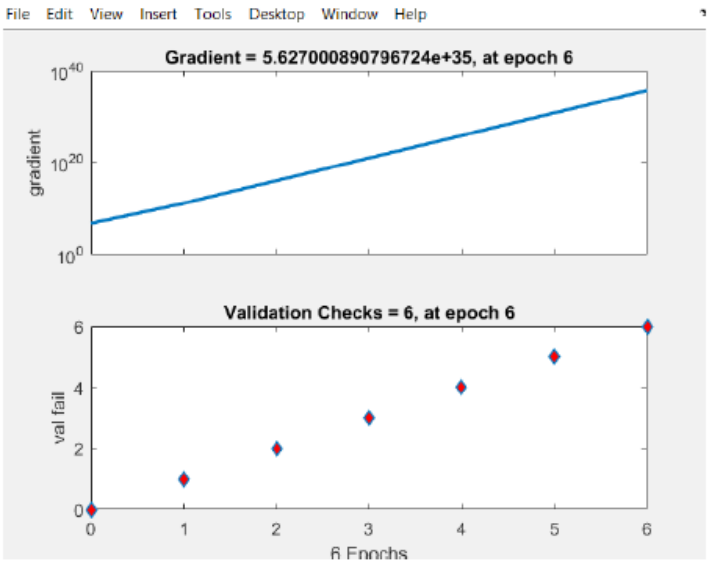
Wykorzystane funkcje:

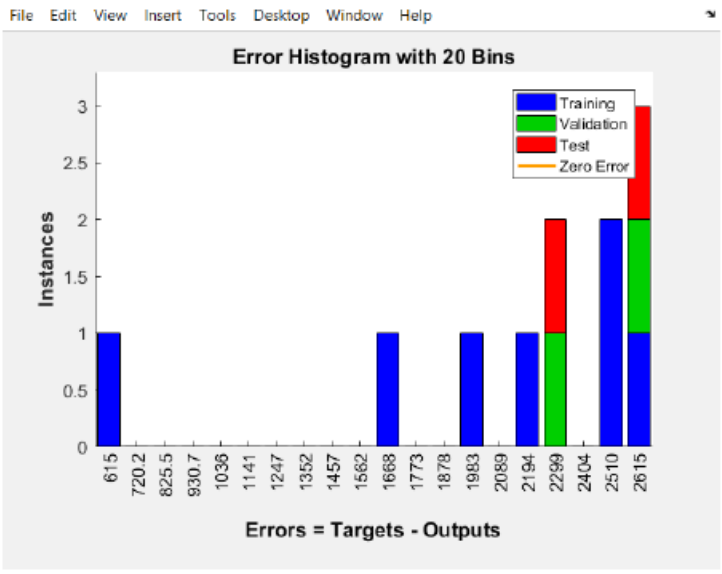
- feedforwardnet(2) – implementacja sieci wielowarstwowej o 2 warstwach ukrytych.
- net.trainFcn = ‘traingd’ – algorytm wstecznej propagacji.
- net.trainParam.lr = 0.1 – współczynnik uczenia.
- net.trainParam.mc = 0.5 – współczynnik bezwładności.
- train() – uczenie sieci dla zadanych danych.

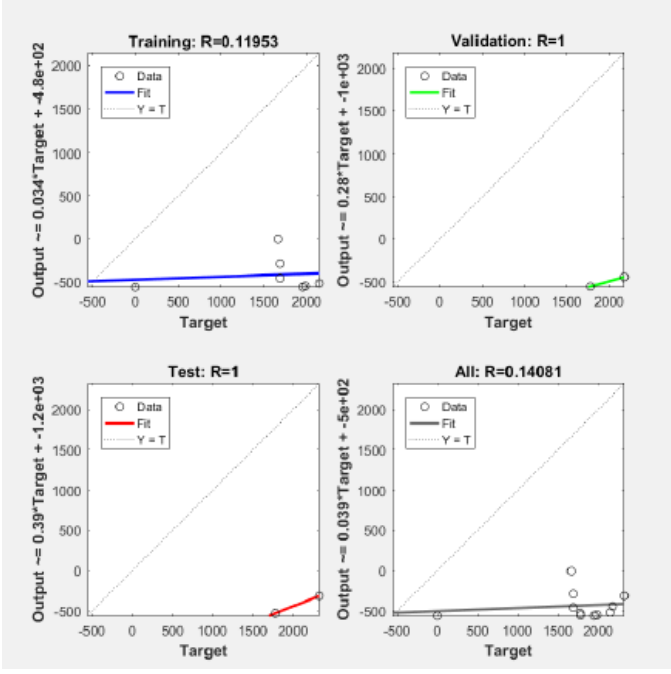
Wyniki dla wariantu sieci wielowarstwowej z algorytmem propagacji błędu. Współczynnik uczenia 0.1 oraz bezwładność 0.5






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


File Edit View Insert Tools Desktop Window Help
Error Histogram with 20 Bins
Training
Validation
Test
Zero Error
Instances
Errors = Targets - Outputs


Training: R=0.11953
Validation: R=1
Test: R=1
All: R=0.14081
Data
Fit
Y = T
Target

Listing kodu programu:

```
function f = Rastrign(x)
    if x == 0
        f = 0;
    else
        N = 100;
        tmp = 10;
        x1 = x;
        dx = (5.12-x)/N;
        f = tmp * N;

        for i = 1:N
            x = x1 + (i * dx);
            f = f + (x^2) - (tmp * cos(2 * pi * x));
        end

        %disp(f)

    end
end
```

```
close all; clear all; clc;

input = [-2 -1.6 -1.2 -0.8 -0.4 0 0.4 0.8 1.2 1.6 2];
output = [1.6633e+03 1.6880e+03 1.6867e+03 1.7764e+03 1.7800e+03 0.0 1.9495e+03 1.9825e+03 2.1477e+03 2.1791e+03 2.3262e+03];
test = zeros(1);


net = feedforwardnet(2);
net.trainFcn = 'traingd';
net.trainParam.lr = 0.1;
net.trainParam.mc = 0.5;
Net = train(Net, input, output);


result = zeros(size(Net));

for i = 1:11

    test(i) = Rastrign(input(i));
    result(i) = sim(Net, input(i));

end
```

Wnioski:

- Algorytm wstecznej propagacji błędów wpływa w sposób niekorzystny na działanie sieci wielowarstwowej. W końcowym efekcie błędy te mogą spowodować dużą różnice między wartościami które sieć powinna zwrócić, a wartościami testowymi.
- Wystąpienie większego współczynnika uczenia od bezwładności powoduje częstsze wystąpienie błędów.
- Najmniej dokładne wyniki osiągają sieci których współczynnik uczenia jest równy bezwładności.
- Algorytm propagacji danych przyspiesza proces uczenia sieci.
- Tak samo jak sieci jednowarstwowe, sieci wielowarstwowe są obciążone błędami.