Gabriela Ciołek

Nr indeksu: 293083

Inżynieria Obliczeniowa

Wydział Inżynierii Metali i Informatyki Przemysłowej

Akademia Górniczo-Hutnicza w Krakowie

**Sprawozdanie 3**

Przedmiot: Podstawy sztucznej inteligencji

Temat: Budowa i działanie sieci wielowarstwowej typu feedforward​

**CEL:**

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

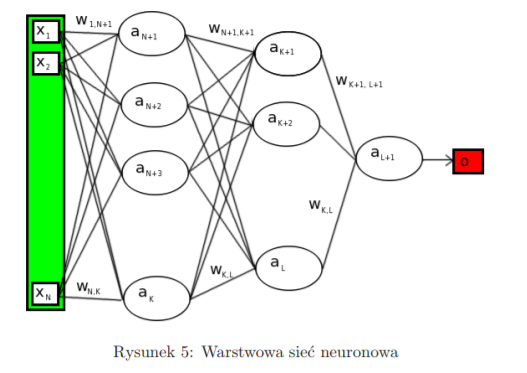
**TEORIA:**

**- Sieć neuronowa:** 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, w szczególności mózgu.

**- Sieć feedforward:** • Sieci skierowane (ang. feed-forward) z jednoznacznie zdefiniowanym kierunkiem przepływu impulsów, nie dopuszczają skierowanych cykli.

**Sieci warstwowe**

Sieci warstwowe są przykładem sieci feed-forward. Perceptrony pogrupowane są w warstwy, dane wejściowe wchodzą do wszystkich jednostek w najniższej warstwie Wyniki policzone w warstwie niższej są traktowane jako wejścia w warstwie bezpośrednio wyżej. Wyniki z warstwy najwyższej są zwracane jako wyniki całej sieci na danych wejściowych (rys. 5). Warstwy, które nie są wyjściowymi nazywane są ukrytymi. Sieci neuronowe z jedną warstwą ukrytą mogą przybliżać każdą funkcję ciągłą. Sieci z dwiema i więcej warstwami nieciągłymi mogą przybliżać funkcje nieciągłe, o ile ilość jednostek w sieci jest wystarczająca.

****

**Algorytm wstecznej propagacji błędu**

Algorytm wstecznej propagacji błędu zdecydowanie dominuje wśród metod uczenia jednokierunkowych sieci wielowarstwowych. Nazwa metody oddaje zasadę jej działania, która polega na „przenoszeniu" błędu, jaki popełniła sieć, w kierunku od warstwy wyjściowej do warstwy wejściowej (a więc wstecz w stosunku do kierunku przepływu informacji).

Cykl uczenia metodą wstecznej propagacji błędu (backpropagation) składa się z następujących etapów:

1. Wyznaczenie odpowiedzi neuronów warstwy wyjściowej oraz warstw ukrytych na zadany sygnał wejściowy.

2. Wyznaczenie błędu popełnianego przez neurony znajdujące się w warstwie wyjściowej i przesłanie go w kierunku warstwy wejściowej.

3. Adaptacja wag.

Algorytm wstecznej propagacji błędu (backpropagation) określa procedurę korekty wag w sieci wielowarstwowej przy wykorzystaniu gradientowych metod optymalizacji. Korekta wektora wag sieci oparta jest na minimalizacji funkcji miary błędu (funkcji celu), którą określono jako sumę kwadratów błędów na wyjściach sieci.

**LISTING KODU WRAZ Z KOMENTARZAMI:**

Główny kod

close all; clear all; clc;

%Dane wejściowe i wyjściowe

input = [-2 -1.5 -1 -0.5 0 0.5 1 1.5 2];

otput = [1.6633e+03 1.6800e+03 1.7329e+03 1.7820e+03 0 1.9484e+03 2.0655e+03 2.1782e+03 2.3262e+03];

test = zeros(1);

%Tworzenie sieci z warstwami ukrytymi

net = feedforwardnet(2);

%Użycie algorytmu wstecznej propagacji

net.trainFcn = 'traingd';

%Dobranie współczynników dla funkcji

net.trainParam.lr = 0.5; %wsp. uczenia

net.trainParam.mc = 0.5; %bezwladnosc

%Trenowanie sieci

net = train(net, input, otput);

%Zapisanie wyniku do tablicy

wynik = zeros(size(net));

%Wywołanie funkcji Rastrigin dla liczb z przedziału [-2,2]

for i = 1:9

test(i) = Rast3D(input(i)); %Główne wywołanie funkcji

wynik(i) = sim(net, input(i)); %Testowanie sieci

end

Funkcja dodatkowa Rastrign

function fx = Rast3D(x)

if x == 0

fx = 0;

else

A = 10;

n=100;

x1 = x;

dx = (5.12-x)/n;

for i = 1:n

x = x1 + (i \* dx);

fx = (A\*n) + ((x.^2) - (A \* cos(2 \* pi \* x)));

end

% disp(fx)

end

end

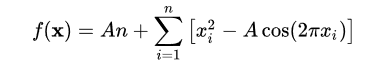
**OPIS WYKONANEGO ZADANIA:**

*Do moich zadań należało:*

1. Wygenerowanie​ ​danych​ ​uczących​ ​i​ ​testujących​ ​dla​ ​funkcji​ ​Rastrigin​ ​3D​ ​dla​ ​danych wejściowych​ ​z​ ​przedziałów​ ​od​ ​-2​ ​do​ ​2.

Stworzyłam tablicę wartości [-2,2], która zmieniała się z krokiem równym 0.5. Dane wyjściowe otrzymałam na podstawie napisanej funkcji Rast3D, dla której za wartość x podstawiałam kolejne wartości z tablicy.

Funkcja Rast3D ma postać :



gdzie A=10, xi[-5.12;5.12] a dla f(0)=0.

1. Przygotowanie​ ​(implementacja​ ​lub​ ​wykorzystanie​ ​gotowych​ ​narzędzi) wielowarstwowej​ ​sieci​ ​oraz​ ​algorytmu​ ​wstecznej​ ​propagacji​ ​błędu.

Użyłam już gotowych narzędzi z pakietu Matlab, które służą do implementacji wyżej wymienionych rzeczy. Do stworzenia wielowarstwowej sieci użyłam polecenia feedforwardnet(), natomiast algorytm wstecznej propagacji błędu jest zapisany przy użyciu gotowej funkcji 'traingd'.

c) Uczenie​ ​sieci​ ​dla​ ​różnych​ ​współczynników​ ​uczenia​ ​(np.​ ​0.5,​ ​0.1,​ ​0.01)​ ​i​ ​bezwładności (np.​ ​0,​ ​0.5,​ ​1).

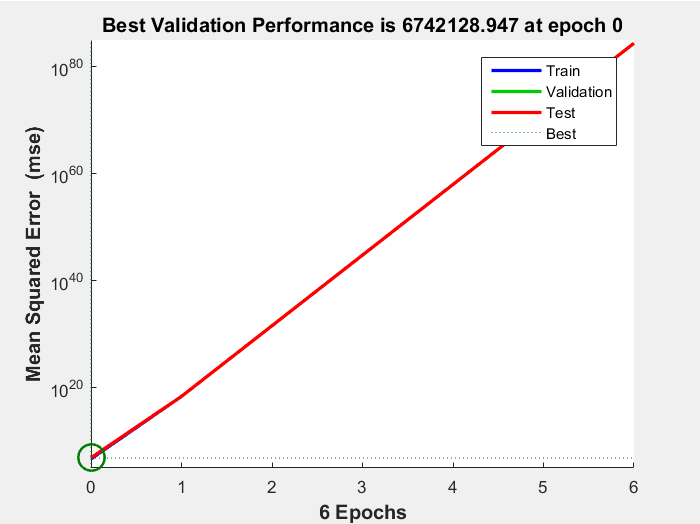
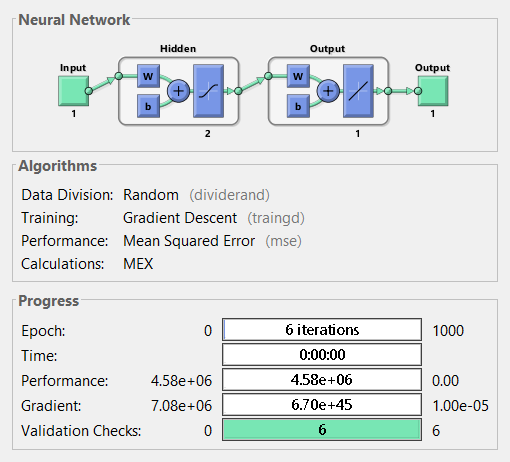
d) Testowanie​ ​sieci.

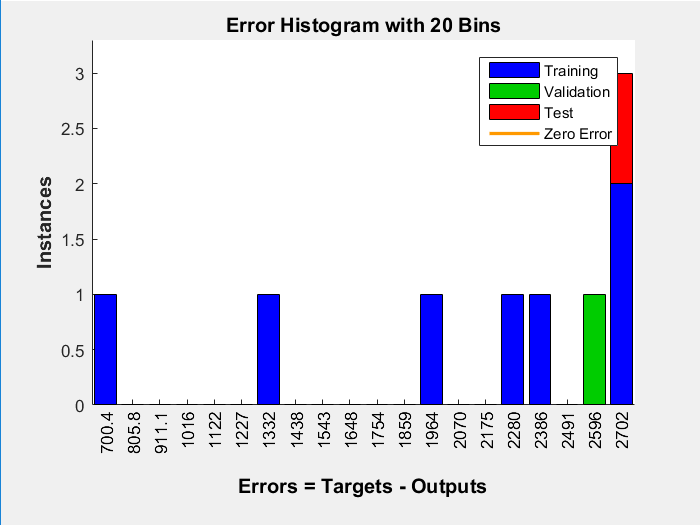
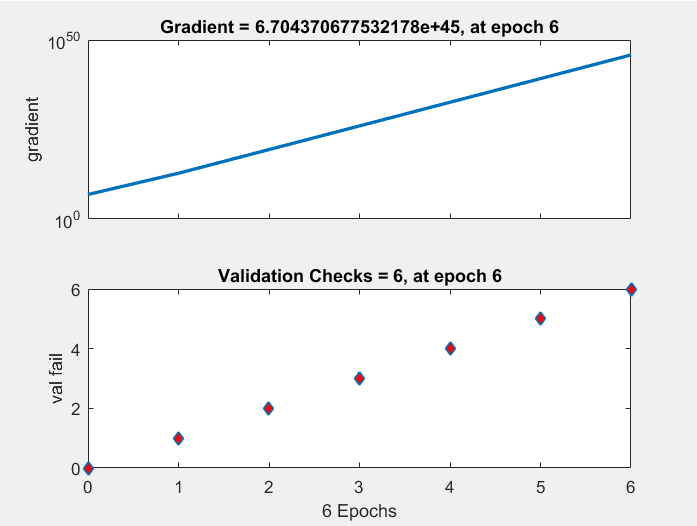
Wyniki zebrane w tabeli dla podanych współczynników:

|  |  |  |  |
| --- | --- | --- | --- |
| Wsp. Uczenia | 0.5 | 0.1 | 0.01 |
| Momentum | 0 | 0.5 | 1 |
| -2 | 1,52355e+03 | 2,71041e+03 | 2,63471e+03 |
| -1.5 | 1,31786e+03 | 9,68540e+03 | 8,40806e+03 |
| -1 | 1,22224e+03 | 1,29279e+03 | 1,10926e+03 |
| -0.5 | 1,19348e+03 | 1,39088e+03 | 1,19078e+03 |
| 0 | 1,18663e+03 | 1,41599e+03 | 1,21285e+03 |
| 0.5 | 1,18755e+03 | 1,42041e+03 | 1,22168e+03 |
| 1 | 1,19785e+03 | 1,41440e+03 | 1,23657e+03 |
| 1.5 | 1,23330e+03 | 1,38973e+03 | 1,28173e+03 |
| 2 | 1,30978e+03 | 1,33592e+03 | 1,37823e+03 |

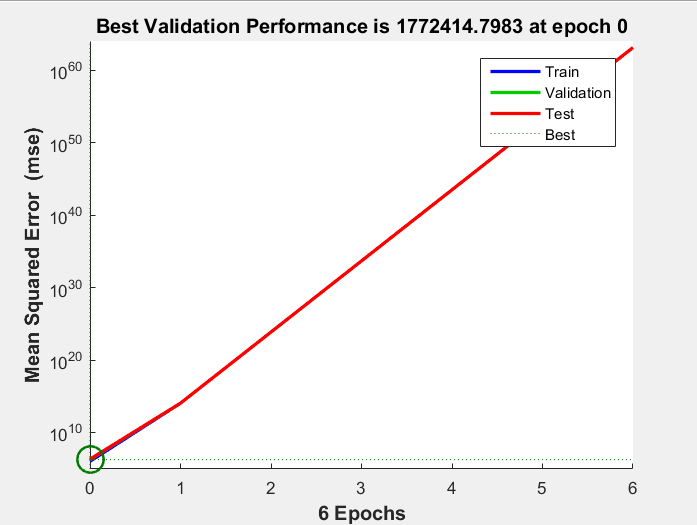
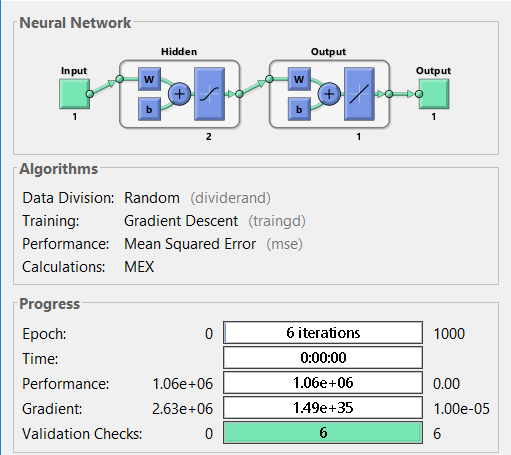
**Przykładowe wykresy dla różnych wartości współczynników:**

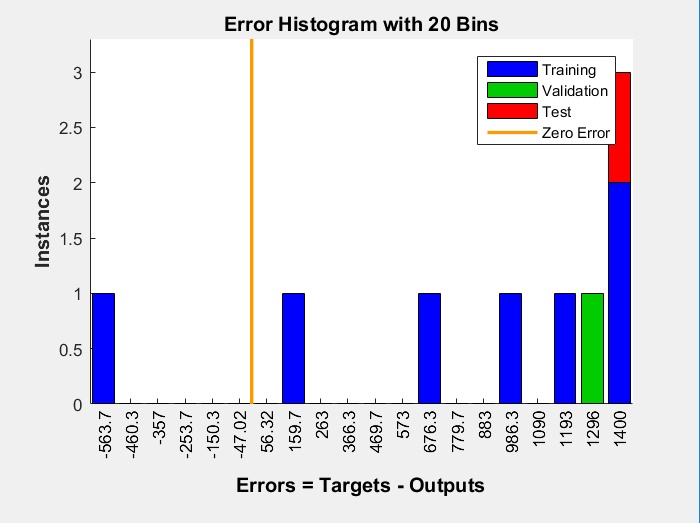
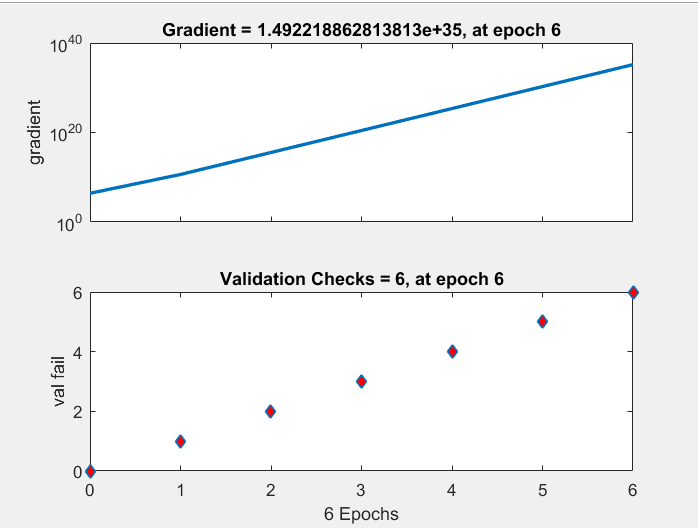
**WSPÓŁCZYNNIK UCZENIA – 0.5 ; MOMENTUM - 0**





**WSPÓŁCZYNNIK UCZENIA – 0.01; MOMENTUM – 1**

****

****

**WNIOSKI:**

- Parametry uczenia pozwalają nam sterować wynikami końcowymi i mają główny wpływ na wygląd i wartości otrzymanych funkcji.

- Dla początkowych wartości tablicy wejściowej i dla danych uczących (0.1,0.5) oraz (0.01,1) wyniki bardzo odbiegają od reszty, co może być spowodowane faktem, że na początku sieć uczy się i nie przyjmuje dokładnych wartości, a z czasem liczy co raz lepiej.

- Im bardziej współczynniki odbiegają od siebie wartością tym większe błędy zachodzą podczas liczenia. Aby otrzymać w miarę poprawne wyniki powinny te wartości nie być zbyt mocno od siebie odległe.

- Wyniki w większości przypadków wychodzą w miarę podobne więc można założyć, że program został zrealizowany poprawnie.