Akademia Górniczo-Hutnicza
WIMIiP, Inżynieria Obliczeniowa
grupa laboratoryjna 01
Agnieszka Kępka

15.11.2018,Kraków

Podstawy Sztucznej Inteligencji

Sprawozdanie numer 2

Budowa i działanie sieci jednowarstwowej

Wprowadzenie:

Celem projektu było poznanie budowy i działanie jednowarstwowych sieci neuronowych oraz uczenie rozpoznawania wielkości liter.

Zadania do wykonania:

- wygenerowanie danych uczących i testujących zawierających litery wielkie i małe
- przygotowanie dwóch jednowarstwowych sieci
- uczenie sieci dla różnych współczynników uczenia
- testowanie sieci

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. Oryginalną inspiracją takiej struktury była budowa naturalnych neuronów, łączących je synaps, oraz układów nerwowych, w szczególności mózgu.

Sieci jednowarstwowe - neurony w tej sieci ułożone są w jednej warstwie, zasilanej jedynie z węzłów wejściowych. Węzły wejściowe nie tworzą warstwy neuronowej, ponieważ nie zachodzi w nich proces obliczeniowy.

Uczenie sieci neuronowych - wymuszenie na niej określonej reakcji na zadane sygnały wejściowe. Uczenie jest konieczne tam, gdzie brak jest informacji doświadczalnych o powiązaniu wejścia z wyjściem lub jest ona niekompletna, co uniemożliwia szczegółowe zaprojektowanie sieci. Uczenie może być realizowane krok po kroku lub poprzez pojedynczy zapis. Istotnym czynnikiem przy uczeniu jest wybór odpowiedniej strategii (metody) uczenia. Wyróżnić możemy dwa podstawowe podejścia: uczenie z nauczycielem (supervised learning) i uczenie bez nauczyciela (unsupervised learning).

Reguła delta jest regułą uczenia z nauczycielem. Polega ona na tym, że każdy neuron po otrzymaniu na swoich wejściach określone sygnały (z wejść sieci albo od innych neuronów, stanowiących wcześniejsze piętra przetwarzania informacji) wyznacza swój sygnał wyjściowy wykorzystując posiadaną wiedzę w postaci wcześniej ustalonych wartości współczynników wzmocnienia (wag) wszystkich wejść oraz ew progu. Wartość sygnału wyjściowego, wyznaczonego przez neuron na danym kroku procesu uczenia porównywana jest z odpowiedzią wzorcową podaną przez nauczyciela w ciągu uczącym. Jeśli występuje rozbieżność - neuron wyznacza różnicę pomiędzy swoim sygnałem wyjściowym a tą wartością sygnału, która była by - według nauczyciela prawidłowa. Ta różnica oznaczana jest zwykle symbolem greckiej litery d (delta) i stąd nazwa opisywanej metody.

Przeprowadzenie Ćwiczenia:

W programie Matlab, za pomocą biblioteki Nueral Network Toolbox, zaimplementowałam sztuczną sieć neuronową. Wykorzystałam do tego funkcje:

- **newlin** – tworzy jednowarstwową sieć neuronową
- **newff** – tworzy wielowarstwową sieć neuronową
- **sim** – symuluje działanie perceptronu
- **train** – służy do nauki sieci na podstawie wektorów wejściowego i wyjściowego
- **disp** – wyświetla informacje
- **round** – funkcja zaokrąglająca

Net jest to struktura zawierająca opis sieci neuronowej.

Kod programu:

```
PR = [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;                    % PR - liczba wektorów wejścia
      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];
S = 1;                                                                                          % N = Liczba neuronow (wyjścia)
net = newlin(PR,S,0,0.01);
net.name = 'Sieć rozpoznaje wielkie i małe litery';

net1 = newff(PR,S,{'tansig'},'trainbr');

                                                % tablica znaków wejściowych
                                                % A a B b C c D d E e F f H h I i K k L l
Wejscie = [0 0 1 1 0 0 1 0 1 0 1 0 1 1 0 0 1 1 1 0;          % 1.
          1 0 1 0 1 0 1 0 1 0 0 0 1 0 0 0 0 0;
          1 0 1 0 1 0 1 0 1 0 1 1 0 0 1 0 0 0 0 0;
          1 0 1 0 1 0 1 1 1 0 1 1 0 0 1 0 0 0 0 0;
          0 0 0 0 1 0 0 0 1 0 1 0 1 0 0 0 1 0 0 0;

          1 0 1 1 1 0 1 0 1 0 1 0 1 1 0 0 1 1 1 0;           % 2.
          0 1 0 0 0 0 0 0 0 1 0 1 0 0 0 0 0 0 0 1;
          0 1 0 0 0 0 0 0 0 1 0 0 0 0 1 1 0 0 0 1;
          0 1 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0;
          1 0 1 0 0 0 1 0 0 0 0 1 1 0 0 0 0 0 0 0;

          1 0 1 1 1 0 1 0 1 1 1 0 1 1 0 0 1 1 1 0;           % 3.
          0 0 1 0 0 1 0 0 1 0 1 1 1 1 0 0 0 0 0 0;
          0 0 1 0 0 1 0 0 1 0 1 0 1 1 1 0 1 0 0 1;
          0 0 1 0 0 1 0 1 0 1 1 0 1 0 0 0 0 1 0 0;
          1 1 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 0 0;

          1 0 1 1 1 1 1 0 1 1 1 1 1 1 0 0 1 1 1 0;           % 4.
          1 1 0 1 0 0 0 1 0 1 0 1 0 0 0 1 1 1 0 0;
          1 1 0 1 0 0 0 1 0 1 0 1 0 0 1 1 0 1 0 1;
          1 1 0 0 0 0 0 1 0 1 0 0 0 1 0 0 1 0 0 0;
          1 1 1 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 0 0;

          1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 0 0 1 1 1 0;         % 5.
          0 0 0 0 0 0 0 0 0 0 0 0 1 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 0 0 0 1;
          0 0 0 1 0 0 0 1 0 0 0 0 0 1 0 0 0 1 0 0;
          1 1 1 0 0 0 1 0 0 0 0 0 1 0 0 0 1 0 0 0;

          1 0 1 1 0 0 1 0 1 0 1 0 1 1 0 0 1 1 1 0;           % 6.
          0 1 1 1 1 1 1 1 1 1 0 1 0 0 1 1 0 0 0 1;
          0 1 1 1 1 1 1 1 1 1 0 0 0 0 1 1 0 0 0 1;
          0 1 1 0 1 1 1 1 1 1 0 0 0 1 1 1 0 0 0 1;
          1 1 0 0 1 0 0 0 1 0 0 0 1 0 0 0 1 1 1 0; ];

                                                % tablica znaków wyjściowych
                                                % A a B b C c D d E e F f H h I i K k L l
Wyjscie = [1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0];
                                                % wartość 1 - wielka litera
                                                % wartość 0 - mala litera

                                    % Parametry sieci do testowania
net.trainParam.epochs = 1000;     % liczba epok
net.trainParam.goal = 0.001;      % błąd średniokwadratowy
net.trainParam.mu = 0.001;        % współczynnik uczenia
net = train(net, Wejscie, Wyjscie);  % uczenie sieci
% symulacja = sim(net,Wejscie);  % symulacja

net1.trainParam.epochs = 100;
net1.trainParam.goal = 0.001;
net1.trainParam.mu = 0.001;
net1 = train(net1, Wejscie, Wyjscie);
% symulacja1 = sim(net1,Wejscie);  % symulacja

   test_A = [ 0; 1; 1; 1; 0;
              1; 0; 0; 0; 1;
              1; 0; 0; 0; 1;
              1; 1; 1; 1; 1;
              1; 0; 0; 0; 1;
              1; 0; 0; 0; 1 ];

    test_i = [ 0; 0; 0; 0; 0;
               0; 0; 1; 0; 0;
               0; 0; 0; 0; 0;
               0; 1; 1; 0; 0;
```

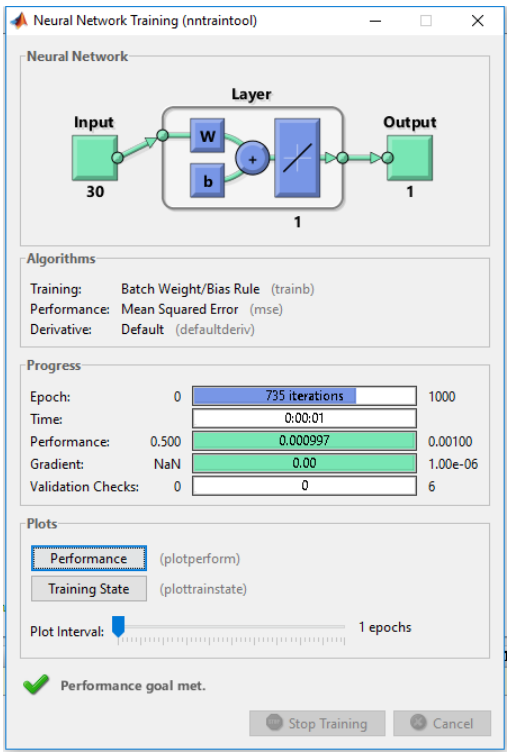
```
            0; 0; 1; 0; 0;
            0; 1; 1; 1; 0 ];

    test_F = [ 1; 1; 1; 1; 1;
            1; 0; 0; 0; 0;
            1; 1; 1; 1; 0;
            1; 0; 0; 0; 0;
            1; 0; 0; 0; 0;
            1; 0; 0; 0; 0 ];

              test_d = [ 0; 0; 0; 1; 0;
           0; 0; 0; 1; 0;
           0; 0; 0; 1; 0;
           0; 1; 1; 1; 0;
           1; 0; 0; 1; 0;
           0; 1; 1; 1; 0 ];

              test_H = [ 1; 0; 0; 0; 1;
           1; 0; 0; 0; 1;
           1; 1; 1; 1; 1;
           1; 0; 0; 0; 1;
           1; 0; 0; 0; 1 ];


              test_K = [ 1; 0; 0; 1; 0;
           1; 0; 1; 0; 0;
           1; 1; 0; 0; 0;
           1; 0; 1; 0; 0;
           1; 0; 0; 1; 0 ];

              test_l = [ 1; 0; 0; 0; 0;
           1; 0; 0; 0; 0;
           1; 0; 0; 0; 0;
           1; 0; 0; 0; 0;
           1; 1; 1; 0; 0 ];


symulacja = sim(net, test_A);       % symulacja z danymi testowymi
if round(symulacja) == 1
   disp('Podana litera jest wielka'); disp(round(symulacja));
else
   disp('Podana litera jest mala'); disp(round(symulacja));
end

symulacja1 = sim(net1, test_A);
if round(symulacja1) == 1
   disp('Podana litera jest wielka'); disp(round(symulacja1));
else
   disp('Podana litera jest mala'); disp(round(symulacja1));
end
```

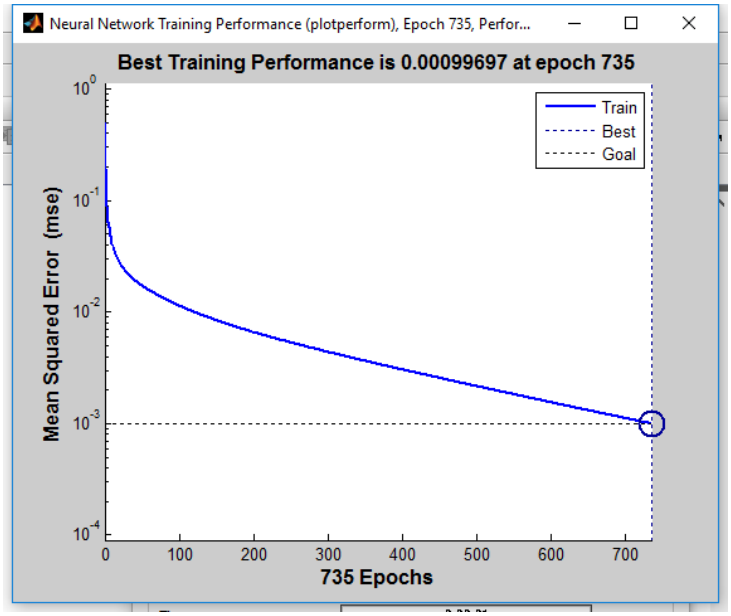
Dane wejściowe: litery; Aa, Bb, Cc, Dd, Ee, Ff, Hh, Ii, Kk, Ll, reprezentowane przez wartości binarne (0 i 1) o rozmiarze 6x4
Dane wyjściowe: macierz odpowiadająca ilości liter oraz ich wartości. Wielka litera równa 1, mała 0.

Krótki opis zmiennych:
PR – zmienna wejściowa dla funkcji tworzących sieć neuronową
S – zmienna przechowująca ilość wyjść (w tym wypadku 1, bo litera albo jest wielka albo mała)
net – struktura zawierająca sieć neuronową
Wejście – tablica znaków wejściowych
Wyjście – tablica znaków wyjściowych
net.trainParam.* - określenie parametrów treningu
***epochs** – maksymalna liczba epok
***goal** – błąd średniokwadratowy
***mu** – współczynnik uczenia
test_(litera) – dane testujące reprezentowane przez 0 i 1; w nawiasie: odpowiednia litera, która odpowiada danej zmiennej

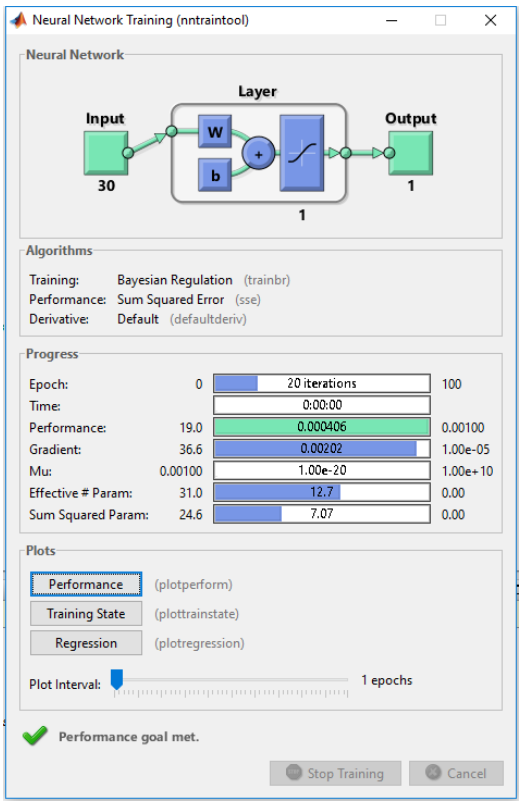
Otrzymane wyniki:

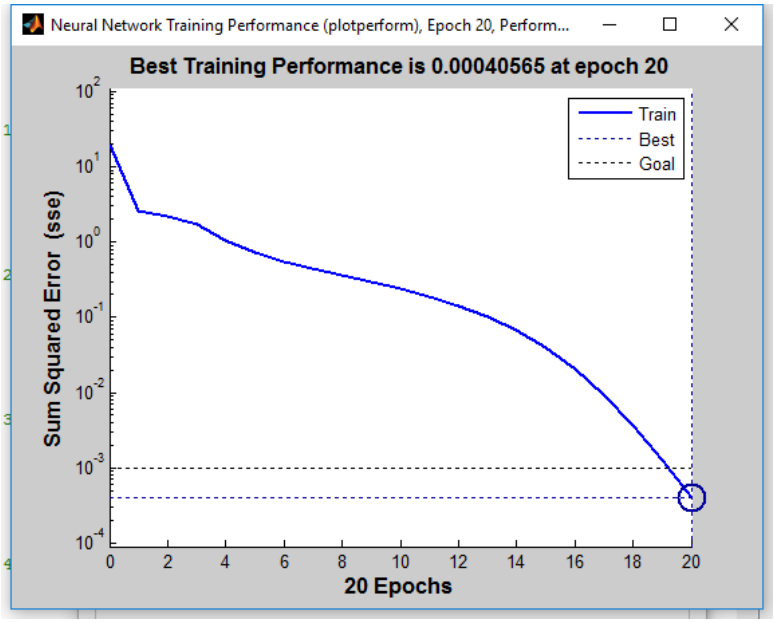
1. Wynik nauczania dla funkcji newlin oraz wykres jej wydajności





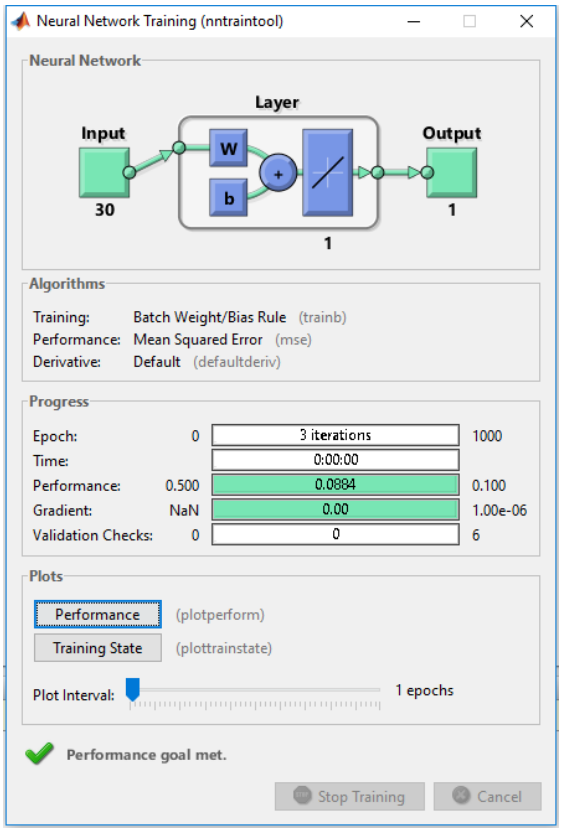
2.Wynik nauczania dla funkcji newff oraz wykres jej wydajności

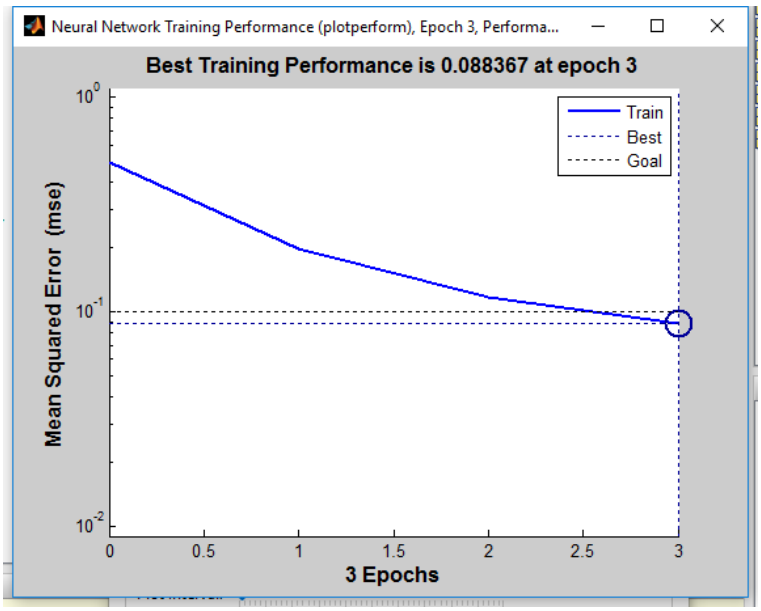




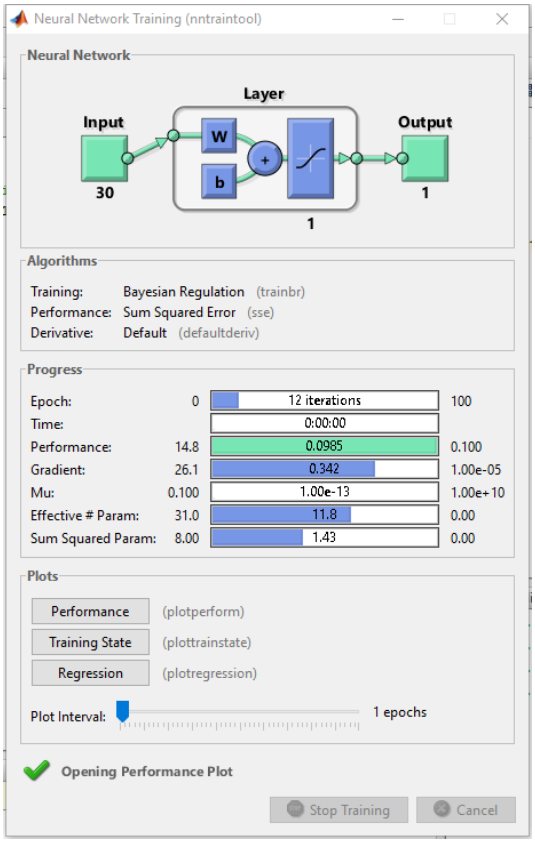
Kolejnym krokiem było przeprowadzenie testu dla różnych współczynników uczenia sieci.
Dla współczynnika równego 0.1 ilość epok, w których sieć się uczyła jest niewiele mniejsza.
Po zwiększeniu błędu średniokwadratowego ilość epok znacznie się skróciła.

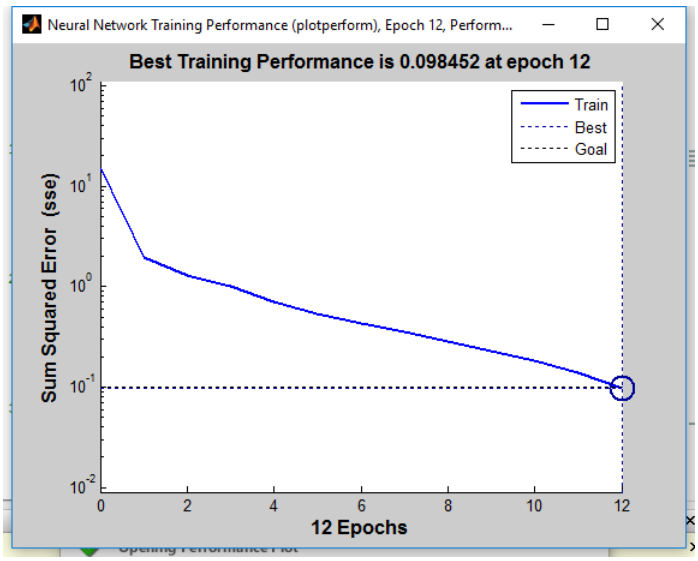
1.Wyniki nauczania dla funkcji newlin dla współczynnika i błędu 0.1 oraz wykres wydajności.





2.Wyniki nauczania dla funkcji newff dla współczynnika i błędu 0.1 oraz wykres wydajności.





Wnioski:

- Dane wejściowe były kompletne co powoduje łatwość uczenia oraz poprawne wyniki.
- Gdyby w danych wejściowych były „dziury” proces nauki wydłużyłby się oraz wyniki mogłyby być błędne.
- Funkcja newff jest szybsza niż newlin.
- Dla obu funkcji testy zostały wykonane poprawnie, a wyniki były różnie dokładne.
- Zwiększenie samego współczynnika uczenia sieci nie miało znacznego wpływu na wyniki.
- Zwiększenie błędu średnokwadratowego przyśpieszyło czas nauki sieci.
- Przy zmianie błędu oraz współczynnika z 0.001 na 0.1 widać było szybkość nauki, tzn z 735 epok nastąpiła zmiana na 3 epoki dla funkcji *newlin,* dla funkcji *newff* liczba epok zmalała z 20 na 12.
- Przy zwiększeniu błędu średniokwadratowego czy współczynnika uczenia na 0.01 wyniki nie różniły się w stopniu znacznym, ilość epok zmniejszyła się o kilka iteracji, nie nastąpiła tak znaczna zmiana jak w błędzie równym 0.1.