Algorytmy Ewolucyjne – projekt NN 2017
Stanisław Jankowski

Zaprojektować sieć neuronową o architekturze perceptronu wielowarstwowego do aproksymacji funkcji $y=f(x)$ na podstawie zbioru uczącego.

Struktura sieci neuronowej: 2 neurony wejściowe: x i 1 (składowa stała, ang. bias), 1 warstwa n neuronów ukrytych, 1 neuron wyjściowy.
- funkcja aktywacji neuronów ukrytych:

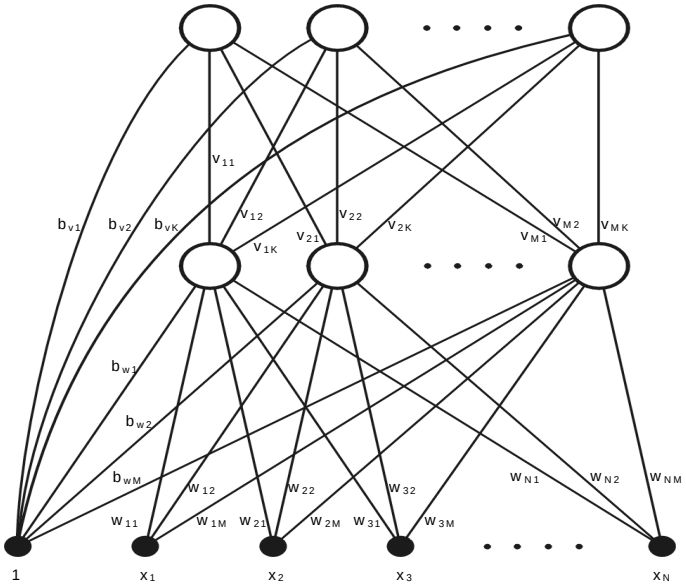
$$g(h) = \tanh(h) = \frac{\exp(h) - \exp(-h)}{\exp(h) + \exp(-h)}$$

- funkcja aktywacji neuronu wyjściowego: liniowa.

Zbiory danych pary $\{x,y\}$:
- zbiór uczący approx_train.asc;
- zbiór testowy approx_test.asc.

Metoda uczenia:

Cel: minimalizacja błędu średniokwadratowego aproksymacji funkcji (regresja).
Struktura sieci: Dobrać optymalną liczbę neuronów ukrytych na podstawie otrzymanych błędów średniokwadratowych dla zbioru uczącego i testowego.
Inicjalizacja wag: liczby losowe z przedziału [–0.15, 0.15]. Dla sieci o optymalnej liczbie neuronów ukrytych wykonać 20 symulacji sieci dla różnych warunków początkowych i wybrać najlepszą strukturę.
Algorytm uczenia: algorytm Levenberga-Marquardta 200 iteracji (`trainlm`).



Rys.. 1. Schemat ogólny perceptronu wielowarstwowego.

Sprawozdanie plik pdf :max. 4 strony A4.

1. Cel: perceptron wielowarstwowy do aproksymacji funkcji.
2. Opis i wizualizacja zbiorów danych.
3. Schemat sieci neuronowej – najlepsza struktura.
4. Wyniki obliczeń: błąd średniokwadratowy: tabela błędów średniokwadratowych dla zbioru uczącego i testowego w zależności od liczby neuronów ukrytych od 1 do *m*:

Liczba neuronów ukrytych	Błąd średniokwadratowy – zbiór uczący	Błąd średniokwadratowy – zbiór testowy
1 2 … *m*		

5. Sporządzić wykres aproksymowanej funkcji.
6. Komentarze:
   a) informacja o narzędziach programistycznych;
   b) uzasadnienie struktury sieci (liczba neuronów ukrytych) na podstawie wyników symulacji, związek struktury sieci i postaci badanej funkcji (kształt).
7. Wnioski.
8. Wykorzystane źródła.

Uczenie sieci neuronowej

```
function [net]= train_net(train_set,labels,hidden_neurons_count)
    %    %Parameters:
    %   train_set:
    %   labels - y
    %   hidden_neurons_count:
    %Return value:
    %   net – object representing a neural network

    %initialization
    %hidden neuron activation function- tanh,
    % output neuron activation - linear
    %
    net=newff(train_set',labels',hidden_neurons_count,...
              {'tansig', 'purelin'},'trainlm');

    rand('state',sum(100*clock));   %random numbers generator
initialization
    net=init(net);                          %weights initialization
    net.trainParam.goal = 0.01;             %stop- mse criterion
    net.trainParam.epochs = 400;            %number of epochs iterations
    net=train(net,train_set',labels');      %network training
```