Katedra Mechaniki Stosowanej i Robotyki
Wydział Budowy Maszyn i Lotnictwa, Politechnika Rzeszowska

METODY SZTUCZNEJ INTELIGENCJI

Laboratorium nr 5

Temat: **Uczenie jednowarstwowych sieci neuronowych z zastosowaniem algorytmu wstecznej propagacji błędów**

Uczenie jednowarstwowych sieci neuronowych z zastosowaniem algorytmu wstecznej propagacji błędów

1. Cel ćwiczenia

Celem ćwiczenia jest zapoznanie się z metodami uczenia dwuwarstwowych sieci neuronowych (SN) zastosowaniem algorytmu wstecznej propagacji błędów oraz algorytmów przybornika Matlab Neural Network Toolbox. Do aproksymacji funkcji nieliniowych jednej zmiennej zastosowane zostaną SN z lokalnymi funkcjami bazowymi (***funkcje Gaussa***) oraz nielokalnymi funkcjami bazowymi w warstwie ukrytej (***funkcje liniowe, funkcje sigmoidalne unipolarne, bipolarne***).

2. Uczenie SN z zastosowaniem algorytmu wstecznej propagacji błędów

Algorytm wstecznej propagacji błędu określa strategię doboru wag w sieci wielowarstwowej przy wykorzystaniu gradientowych metod optymalizacji.

W poprzednich ćwiczeniach stosowano metody gradientowe do adaptacji wag SN, gdzie poszczególne wagi były aktualizowane w każdym kroku procedur prezentacji wzorców uczących. W bieżącym laboratorium zajmiemy się uczeniem SN, gdzie aktualizacja wag SN będzie się odbywać po prezentacji wszystkich wzorców uczących dla danej epoki (termin epoka oznacza prezentację wszystkich wzorców uczących ze zbioru treningowego w procedurze uczenia wag).

Przykład 1. Realizacja SN dwuwarstwowej z zastosowaniem Toolboxa Neural Network

SN dwuwarstwowa z sigmoidlanymi bipolarnymi funkcjami aktywacji neuronów w warstwie ukrytej wraz z algorytmem uczenia wag sieci, została zrealizowana przy pomocy następującej procedury w Matlab-ie:

```
x=[…];                          %Zadanie wektora wejściowego do SN
y=[…];                          %Wartości zadanej funkcji y(x)

ogr=[0 2];                      %Wartości ograniczeń wektora wejściowego x

siec_neur = newff(ogr,[5,1],{'tansig', 'purelin'}, 'traingdm', 'learngd', 'sse');                        %Inicjalizacja SN z sigmoidlanymi bipolarnymi funkcjami aktywacji neuronów (tansig) w warstwie ukrytej

NN_out(1,:)=sim(siec_neur,x); %Obliczanie wartości wyjścia dla zainicjalizowanej sieci

%purelin  - funkcja aktywacji liniowa,
%tansig   - sigmoidlana bipolarna funkcja aktywacji,
%logsig   - sigmoidlana unipolarna funkcja aktywacji,
%radbas   - radialna funkcja aktywacji

figure
subplot(3,1,1)
plot(x,y,'o',x,NN_out(1,:),'-')            %Wizualizacja wyniku
subplot(3,1,2)

siec_neur.trainParam.epochs = 1500;        %Liczba epok procesu uczenia
siec_neur.trainParam.show = 100;           %Wyświetlanie błędu co … epok
siec_neur.trainParam.goal = 0.01;          %Zadany minimalny błąd odwzorowania
siec_neur.trainParam.min_grad = 1e-10;
net.trainParam.lr = 0.01;                  %Zadanie wartości współczynnika wzmocnienia uczenia
```

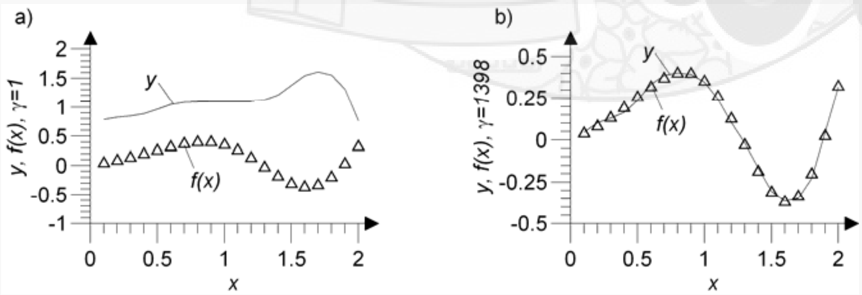
```
[siec_neur] = train(siec_neur,x,y);        %Procedura uczenia wag SN
NN_out(2,:)=sim(siec_neur,x);              %Obliczanie wartości wyjścia z wyuczonej SN

subplot(3,1,3)
plot(x,y,'o',x,NN_out(2,:),'-')            %Wizualizacja wyniku
```

Przykładowy algorytm sieci zbudowano dla następujących danych:
- estymowana funkcja: $f(x) = e^{-(x-2)(x-2)} sin(3x) + 0.3x$,
- przedział zmienności wartości wejściowej: $x \in \langle 0,2 \rangle$,
- przyjęto liczbę neuronów warstwy ukrytej *SN* $l_{neur} = 5$,
- zastosowanej funkcje aktywacji neuronów: sigmoidalne bipolarne, $\mathbf{S}(\mathbf{x}) = \frac{2}{1+\exp(-\beta\mathbf{x})} - 1$,
- przyjęto wartość współczynnika wzmocnienia uczenia α=0.01.

Algorytm uczenia SN z zastosowaniem gradientowej metody uczenia z momentum, zrealizowany przy użyciu funkcji *train*, podaje wartość błędu SSE co zadaną ilość epok:

TRAINGDM, Epoch 0/1500, SSE 25.3514/0.005, Gradient 38.5982/1e-010
TRAINGDM, Epoch 100/1500, SSE 0.292753/0.005, Gradient 0.53014/1e-010
TRAINGDM, Epoch 200/1500, SSE 0.0584726/0.005, Gradient 0.187566/1e-010
TRAINGDM, Epoch 300/1500, SSE 0.0276189/0.005, Gradient 0.0684556/1e-010
TRAINGDM, Epoch 400/1500, SSE 0.0210571/0.005, Gradient 0.0485388/1e-010
TRAINGDM, Epoch 500/1500, SSE 0.0169979/0.005, Gradient 0.0412495/1e-010
TRAINGDM, Epoch 600/1500, SSE 0.0140021/0.005, Gradient 0.0357522/1e-010
TRAINGDM, Epoch 700/1500, SSE 0.0117347/0.005, Gradient 0.0312464/1e-010
TRAINGDM, Epoch 800/1500, SSE 0.00999171/0.005, Gradient 0.0275105/1e-010
TRAINGDM, Epoch 900/1500, SSE 0.00863258/0.005, Gradient 0.024384/1e-010
TRAINGDM, Epoch 1000/1500, SSE 0.00755922/0.005, Gradient 0.021741/1e-010
TRAINGDM, Epoch 1100/1500, SSE 0.00670204/0.005, Gradient 0.0194837/1e-010
TRAINGDM, Epoch 1200/1500, SSE 0.00601093/0.005, Gradient 0.0175369/1e-010
TRAINGDM, Epoch 1300/1500, SSE 0.00544918/0.005, Gradient 0.0158427/1e-010
TRAINGDM, Epoch 1398/1500, SSE 0.0049978/0.005, Gradient 0.0143844/1e-010
TRAINGDM, Performance goal met.

W wyniku symulacji SN z sigmoidlanymi bipolarnymi funkcjami aktywacji neuronów, uczonej metodą gradientową, w zadaniu estymacji zadanej nieliniowej funkcji *f(x)* dla podanych wcześniej parametrów pracy, otrzymano następujące przebiegi wartości wyjścia z SN *y* (rys. 1.a)) dla pierwszej epoki procesu uczenia (γ=1), oraz ostatniej epoki procesu uczenia γ=1398 (rys. 1.b)).



Rys. 1.a) Zadana nieliniowa funkcja *f(x)* oraz wartości wyjścia z SN *y* dla γ=1, b) zadana nieliniowa funkcja *f(x)* oraz wartości wyjścia z SN *y* dla γ=1398

Analizując przebieg wartości błędu SSE oraz przebieg wyjścia *y* SN na rys. 6, zauważamy, iż wartości wyjścia zainicjalizowanej SN, z dobranymi losowo wartościami wag, w znacznym stopniu odbiegają od zadanego odwzorowania. W początkowych epokach procesu uczenia dynamika zmian wartości błędu jest znaczna, błąd jest szybko redukowany. Po γ=1398 epokach procesu uczenia SN osiąga zadany poziom dokładności odwzorowania.

3. Zadania do wykonania

Zbudować model SN z zastosowaniem procedury pokazanej w przykładzie 1, dla funkcji aktywacji neuronów w postaci:
a) funkcji sigmoidalnych unipolarnych w warstwie ukrytej,
b) funkcji sigmoidalnych bipolarnych w warstwie ukrytej,
a) funkcji liniowych w warstwie ukrytej,
b) funkcji radialnych typu krzywa Gaussa w warstwie ukrytej,
w postaci *m*-pliku w pakiecie Matlab. Zadaniem sieci jest aproksymacja nieliniowej funkcji jednej zmiennej *f(x)*:

a) $f(x) = 1 - e^{-x} - sin(x)$,

b) $f(x) = \frac{1}{2} sin(x)$,

c) $f(x) = sin(2x) + 0.5cos(0.5x)$,

d) $f(x) = 5 - e^{-x} - x^2$,

e) $f(x) = 0.1 \cdot (2x^2 - 5x) - 3e^{-(x-2)(x-2)}$,

f) $f(x) = sin(x) + \frac{1}{1 + e^{-3(x-3)}}$,

g) $f(x) = 3 \cdot e^{-2x} + 0.7 \cdot x$,

h) $f(x) = e^{-(x-2)(x-2)} sin(3x) + 0.3x$,

dla $x \in \langle 0,5 \rangle$. Przykład funkcji zadanej należy dobrać zgodnie z nr zespołu.

Parametry SN dla poszczególnych zespołów przedstawiono w tab.1.

Tab.1. Parametry SN dla poszczególnych zespołów

nr zespołu	1.	2.	3.	4.	5.	6.	7.	8.	9.	10.	11.	12.
$m=l_{neur}$	6	7	5	8	9	10	12	6	8	7	9	8
α	0.02	0.035	0.05	0.023	0.011	0.042	0.021	0.05	0.04	0.03	0.015	0.044

Należy przeprowadzić następujące badania:
Należy przeprowadzić badania symulacyjne dla algorytmu SN z sigmoidlanymi unipolarnymi funkcjami aktywacji neuronów zrealizowanego w postaci *m*-pliku *Matlab*-a, dla danych podanych w tabeli (pozostałe parametry przyjąć jak w przykładzie 1),

- zbadać wpływ liczby neuronów na proces uczenia SN,
- zbadać wpływ wartości współczynnika wzmocnienia uczenia α na proces uczenia SN,
- zbadać wpływ typu zastosowanych funkcji aktywacji neuronów w warstwie ukrytej na jakość aproksymacji i szybkość procesu uczenia (funkcje sigmoidalne unipolarne, sigmoidalne bipolarne, liniowe, Gaussa).

Sprawozdanie powinno zawierać:
1. Wstęp teoretyczny
   - podstawowe wiadomości na temat uczenia SN.
2. Przebieg ćwiczenia
   - przykładowy listing kodu Matlab-a służący do wygenerowania SN z sigmoidlanymi unipolarnymi funkcjami aktywacji neuronów oraz procedurę uczenia SN,
   - przykładowy listing kodu Matlab-a służący do wygenerowania SN z sigmoidlanymi bipolarnymi funkcjami aktywacji neuronów oraz procedurę uczenia SN,
   - przykładowy listing kodu Matlab-a służący do wygenerowania SN z liniowymi funkcjami aktywacji neuronów oraz procedurę uczenia SN,

- przykładowy listing kodu Matlab-a służący do wygenerowania SN z funkcjami aktywacji neuronów typu krzywa Gaussa oraz procedurę uczenia SN,

3. Wyniki symulacji

3.1. Wyniki dla SN z sigmoidlanymi unipolarnymi funkcjami aktywacji neuronów:

- wykres (analogicznie jak w przykładzie 1):
    a) funkcji zadanej *f(x)* oraz wyjścia z SN dla γ=1,
    b) funkcji zadanej *f(x)* oraz wyjścia z SN dla $\gamma=n$, gdzie *n* to zadana liczba epok procesu uczenia wag SN lub liczba przeprowadzonych epok procesu uczenia (w przypadku, gdy uczenie zakończyło się sukcesem dla mniejszej liczby epok niż zadana ilość maksymalna),
- wartości błędu SSE dla poszczególnych epok procesu uczenia (analogicznie jak w przykładzie 1).

3.2. Wyniki dla SN z sigmoidlanymi bipolarnymi funkcjami aktywacji neuronów (analogicznie jak w pkt. 3.1).

3.3. Wyniki dla SN z liniowymi funkcjami aktywacji neuronów (analogicznie jak w pkt. 3.1).

3.4. Wyniki dla SN z radialnymi funkcjami aktywacji neuronów typu krzywe Gaussa (analogicznie jak w pkt. 3.1).

4. Wnioski

Uwaga. Każdy realizowany podpunkt sprawozdania powinien być odpowiednio skomentowany.