Sieci neuronowe typu RBF

Polecenia tworzące sieć

net = newrb(we, wy, blad, spread, n_max, disp) – polecenie tworzy sieć typu RBF

we – macierz danych wejściowych
wy – wektor zadanych danych wyjściowych
blad – średni błąd kwadratowy sieci jaki chcemy osiągnąć
spread – szerokość funkcji RBF
n_max – maksymalna ilość neuronów w sieci
disp – krok, co jaki wyświetlany jest postęp w nauce sieci

Przykładowo:

```
net = newrb(we, wy, 0.01, 1, 100, 1);
```

net = newrbe(we, wy, spread) – polecenie tworzy sieć typu RBF, z ilością neuronów równą ilości próbek

Przykład:

```
we = load('dane_sin1a_i.txt'); we = we';
wy = load('dane_sin1a_o.txt'); wy = wy';

er = 0.01;
spread = 1;

net = newrb(we, wy, er, spread, 100, 1);
ilosc_neuronow = length(net.IW{1,1})

we1 = linspace(min(we), max(we), 1000);
wy1 = sim(net, we1);
figure(2); plot(we1,wy1,'-', we,wy,'.');

wy2 = sim(net,we);
blad = abs(wy-wy2);
figure(3); plot(blad);
```

Przykład dla danych 2-wejściowych:

```
we = load('dane3d_i.txt'); we = we';
wy = load('dane3d_o.txt'); wy = wy';

er = 0.01;
spread = 1;

net = newrb(we, wy, er, spread, 200, 1);
ilosc_neuronow = length(net.IW{1,1})


x_lin = linspace(min(we(1,:)),max(we(1,:)),50);
y_lin = linspace(min(we(2,:)),max(we(2,:)),50);
[X,Y] = meshgrid(x_lin,y_lin);
Z = griddata(we(1,:), we(2,:), wy, X, Y, 'cubic');

figure(1); mesh(X,Y,Z); axis tight; hold on;
plot3(we(1,:), we(2,:), wy,'.'); title('Powierzchnia danych');

wy2 = sim(net,we);
blad = abs(wy-wy2);
figure(2); plot(blad); title('Blad sieci');

Z1 = griddata(we(1,:), we(2,:), wy2, X, Y, 'cubic');
```

```
figure(3); mesh(X,Y,Z1); axis tight; hold on;
plot3(we(1,:), we(2,:), wy,'.'); title('Powierzchnia sieci');
```

Zadania do wykonania:

1. Dla wybranych plików z danymi:
   a. zapoznać się z danymi
   b. dobrać najkorzystniejszą szerokość neuronów RBF
   c. dla badanych sieci przeprowadzić uczenie i porównać jakość wyników
2. Opracować samodzielnie skrypt, który:
   a. podzieli dane na część uczącą i testującą w zadanej proporcji,
   b. dobierze najkorzystniejszą wartość parametru `spread` sieci przy uczeniu z wykorzystaniem polecenia `newrbe`. W trakcie uczenia dane testujące powinny być wykorzystane do określenia momentu przerwania procesu zmniejszania parametru `spread` i do określenia rzeczywistej dokładności sieci.

W sprawozdaniu zamieszczamy raport z przeprowadzonych eksperymentów dla każdych badanych danych. Zamieścić należy także skrypty opracowane w czasie eksperymentów.

Opis plików z danymi:

Każdy zestaw danych składa się z dwóch plików. Końcówka '_i' w nazwie oznacza dane wejściowe, końcówka '_o' oznacza wyjścia zadane.

Plik(i)	Ilość wejść	Ilość wyjść	Opis

`dane_sin1`
`dane_sin2`
`dane_sin3`
	1	1	Sinusoida (w kolejnych plikach próbkowana coraz rzadziej)

`dane_sin1a`
`dane_sin2a`
`dane_sin3a`
	1	1	Zaszumiona sinusoida (w kolejnych plikach próbkowana coraz rzadziej)

`dane3d`	2	1	Funkcja: z = sin(x)*cos(y)/(x^2+y^2+1)
`percep` `dane_a`	2	1	Dane separowalne za pomocą 1 neuronu typu perceptron prosty
`parity`	5	1	Wyjście = 1 gdy suma wejść jest parzysta, 0 gdy nie
`transak`	6	1	Wejścia to: kwota, indeks firmy, godzina, typ osoby autoryzującej transakcję, dzień tygodnia, typ dnia (czy wolny czy nie) Wyjście to: wiarygodność transakcji w %
`kapitan`	3	1	Wyjście to ocena stopnia niebezpieczeństwa dla statku dokonana przez eksperta-kapitana (0, 0.5, 1) Wejścia to prędkości statku
`diabet`	8	1	Wejścia: dane o pacjentach Wyjście: ryzyko zachorowania na cukrzycę