ЛАБОРАТОРНА РОБОТА №3

Тривалість роботи – 2 години.

Функції створення нейронних мереж

network – функція створення нейронної мережі користувача.

Запис:

net = network

net = network(numInputs, numLayers, biasConnect, inputConnect layerConnect, outputConnect, targetConnect)

Опис. Функція повертає створену нейронну мережу з ім'ям net і з наступними характеристиками (у дужках представлено значення за замовчанням):

numInputs – кількість входів (0);

numLayers – кількість шарів (0);

biasConnect – булевий вектор з числом елементів, рівним кількості шарів (нулі);

inputConnect – булева матриця з числом рядків, рівним кількості шарів, і числом стовпців, рівним кількості входів (нулі);

layerConnect – булева матриця з числом рядків і стовпців, рівним кількості шарів (нулі);

outputConnect – булевий вектор-рядок з числом елементів, рівним кількості шарів (нулі);

targetConnect – вектор-рядок, такий же, як попередня (нулі).

net = newc(PR,S,KLR,CLR) – функція створення шару Кохонена.

Функція використовує аргументи:

PR – Rx2 матрицю мінімальних і максимальних значень для R вхідних елементів;

S – число нейронів;

KLR – коефіцієнт навчання Кохонена (за умовчанням 0,01);

CLR – коефіцієнт «справедливості» (за умовчанням 0,001).

net = newcf(PR,[S1 S2...SNI],TF1 TF2...TFNI,BTF,BLF,PF) – функція створення різновиду багатошарової НМ із зворотним розповсюдженням помилки (каскадна НМ). Така мережа містить прихованих шарів NI, використовує вхідні функції типу **dotprod** і **netsum**, ініціалізація мережі здійснюється функцією **initnw**.

Аргументи функції:

PR – Rx2 матриця мінімальних і максимальних значень R вхідних елементів;

Si – розмір i-го прихованого шару, для NI шарів;

TFi – функція активації нейронів i-го шару, за умовчанням '**tansig**';

BTF – функція навчання мережі, за умовчанням '**traingd**';

BLF – функція налаштування вагів і зсувів, за умовчанням '**learngdm**';

PF – функція помилки, за умовчанням '**mse**'.

Приклад

```
»P = [0 1 2 3 4 5 6 7 8 9 10];
»T = [0 1 2 3 4 3 2 1 2 3 4];
» net = newcf([0 10],[5 1],{'tansig' 'purelin'}); % Створення нової мережі
» net.trainParam.epochs = 50; % Задання кількості циклів навчання
» net = train(net,P,T); % Навчання НМ
TRAINLM, Epoch 0/50, MSE 7.77493/0, Gradient 138.282/1e-
010
TRAINLM, Epoch 25/50, MSE 4.01014e-010/0, Gradient
0.00028557/1e-010
TRAINLM, Epoch 50/50, MSE 1.13636e-011/0, Gradient 1.76513e-
006/1e-010
TRAINLM, Maximum epoch reached, performance goal was not met.
» Y = sim(net,P); % Використання НМ
» plot(P,T,'d',P,Y)% Графічна ілюстрація роботи мережі
```

На мал. 1.1 крапками відображено елементи навчальної вибірки, лінією вихід мережі.

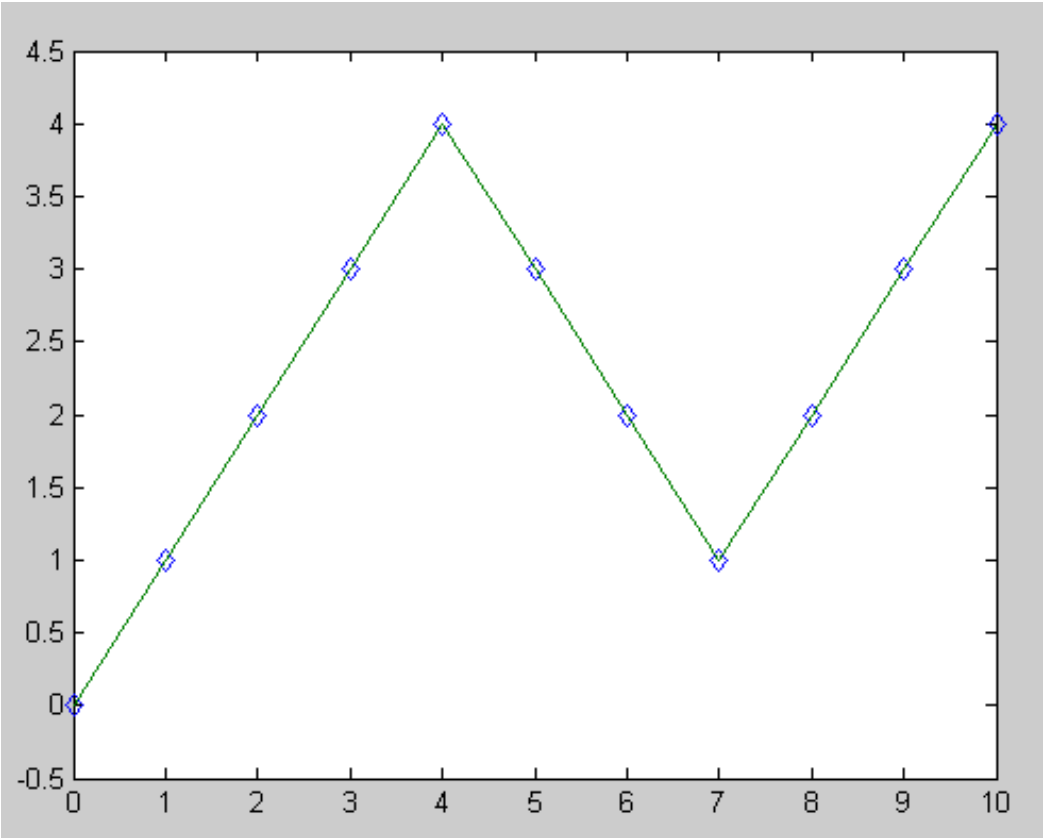

Рис. 1.1. Ілюстрація роботи мережі

net = newelm(PR,[S1 S2...SNI],TF1 TF2...TFNI,BTF,BLF,PF) – функція створення мережі Елмана. Аргументи такі ж, як і у попередньої функції.

net = newff(PR,[S1 S2...SNI],TF1 TF2...TFNI,BTF,BLF,PF) – функція створення «класичною» багатошарової НМ з навчанням за методом зворотного розповсюдження помилки.

net = newfftd(PR,ID,[S1 S2...SNI],TF1 TF2...TFNI,BTF,BLF, PF) – те ж, що і попередня функція, але з наявністю затримок за входами. Додатковий аргумент ID вектор вхідних затримок.

net = newgrnn(P,T,spread)- функція створення загальнооегресійної моделі мережі.

Аргументи:

P – R×Q матриця Q вхідних векторів;

T – S×Q матриця Q цільових векторів;

spread – відхилення (за замовчанням 1,0).

net = newhop(T) – функція створення мережі Хопфілда. Використовується тільки один аргумент.

T – R×Q матриця Q цільових векторів (значення елементів мають бути +1 або -1).

net = newlin(PR,S,ID,LR) – функція створення шару лінійних нейронів.

Аргументи:

PR – R×2 матриця мінімальних і максимальних значень для R вхідних елементів;

S – число елементів у вихідному векторі;

ID – вектор вхідної затримки (за замовчанням [0]);

LR – коефіцієнт навчання (за замовчанням 0,01).

Повертається новий лінійний шар. При записі у формі **net = newlin(PR,S,0,P)** використовується аргумент: **P –** матриця вхідних векторів; повертається лінійний шар з максимально можливим коефіцієнтом навчання при заданій P.

net = newlind(P,T) – функція проектування нового лінійного шару. Функція матриць вхідних і вихідних векторів, отриманих методом найменших квадратів, визначає ваги і зсуви лінійної НМ.

net = newlvq(PR,S1,PC,LR,LF) – функція створення мережі зустрічного розповсюдження.

Аргументи:

PR – Rx2 матриця мінімальних і максимальних значень R вхідних елементів;

S1 – число прихованих нейронів;

PC – S2 елементів вектора, які задають ступінь належності до різних класів;

LR – коефіцієнт навчання, за умовчанням 0,01;

LF – функція навчання, за умовчанням 'learnlv2'.

net = newp(PR,S,TF,LF) – функція створення персептрона.

Аргументи:

PR – Rx2 матриця мінімальних і максимальних значень R вхідних елементів;

S – число нейронів;

TF – функція активації, за умовчанням **'hardlim'**;

LF – функція навчання, за умовчанням **'learnp'**.

net = newpnn(P,T,spread) – створення ймовірнісної НМ. Аргументи – як у функції **newgrnn**.

net = newrb(P,T,goal,spread) – функція створення мережі з радиальними базисними елементами. Аргументи: **P, T, spread –** такі ж, як у функції **newgrnn**; аргумент **goal –** задана средньоквадратична похибка.

net = newrbe(P,T,spread) – функція створення мережі з радиальними базисними елементами з нульовою помилкою на навчальній вибірці.

net = newsom(PR,[D1,D2...],TFCN,DFCN,OLR,OSTEPS,TLR,TND) – функція створення самонавчальної карти з аргументами:

PR – Rx2 матриця мінімальних і максимальних значень R вхідних елементів;
I – розмір i-го шару, за умовчанням [5 8];
TFCN – топологічна функція, за замовчанням '**hextop**';
DFCN – функція відстані, за замовчанням '**linkdist**';
OLR – коефіцієнт навчання фази впорядковування, за замовчанням 0,9;
OSTEPS – число кроків фази впорядковування, за замовчанням 1000;
TLR – коефіцієнт навчання фази налаштування, за замовчанням 0,02;
TND – відстань для фази налаштування, за замовчанням 1.

Завдання

1. Реалізувати у *Neural Networks Toolbox* наведений приклад створення НМ.
2. Згідно завдання викладача реалізувати 3 фунції сторення НМ.

Функції перетворення входів мережі, вагів та відстаней

Функції даної групи перетворять значення входів з використанням операцій множення або додавання.

netprod(Z1,Z2...) – повертає матрицю, елементи якої визначаються як елементи вхідних векторів і зсувів. Аргументи **Z1,Z2... –** матриці, чиї стовпці асоційоровані з входами або зсувами.

Приклади

```
»z1 = [1 2 4;3 4 1];
»z2 = [-1 2 2;-5-6 1];
» n = netprod(z1,z2)
n =
  -14    8
 -15 -24   1
»b = [0;-1];
» n = netprod(z1,z2,concur(b,3))% Функція concur(b,3) створює 3 копії вектора зсуву
n=
     0     0     0
    15    24    -1
```

netsum(Z1,Z2...) – те ж, що у попередньому випадку, але замість множення використовується додавання.

dnetprod(Z,N) – повертає матрицю значень першої похідної входів, перетворених функцією **N = netprod(Z1,Z2...).**

Приклад

```
»Z1 = [0; 1; -1];
»Z2 = [1; 0.5; 1.2];
»N = netprod(Z1,Z2)
 N=
0
0.5000
-1.2000
» dN_dZ2 = dnetprod(Z2,N)
»  dN_dZ2 =
 0
 1
-1
```

dnetsum(Z,N) – те ж, що і у попередньому випадку, але по відношенню до функції **netsum(Z1,Z2...)**.

boxdist(pos) – функція визначення box-відстані між нейронами в шарі. Має один аргумент pos-матрицу розміру N×S, елементи якої визначають координати нейронів, повертає матрицю розміру S×S відстаней.

Відстані (елементи матриці, що повертається) обчислюються за виразом:

$$D_{ij} = max(abs(P_i — P_j)),$$

де: P_i і P_j – вектори, що містять координати нейронів i і j.

Приклад

```
» pos = rand(3,4)% Випадкове розміщення 4-х нейронів в 3-мірному просторі (генерація випадкової матриці 3x4)
pos =
    0.8600    0.4966    0.6449    0.3420
    0.8537    0.8998    0.8180    0.2897
    0.5936    0.8216    0.6602    0.3412

» d=boxdist(pos)
d=
    0         0.3635    0.2151    0.5639
    0.3635    0         0.1614    0.6100
    0.2151    0.1614    0         0.5282
    0.5639    0.6100    0.5282    0
```

dist – функція обчислення евклідової відстані.

Z = dist(W,P) – повертає матрицю, елементи якої являються евклідовими відстанями між рядками (векторами) матриці **W** і стовпцями матриці **P** (матриці повинні мати відповідні розміри).

D = dist(pos) – у такій формі функція аналогічна функції **boxdist(pos)** за тим виключенням, що повертається матриця евклідових відстаней.

negdist(W,P) – функція ідентична попередній, але елементи матриці, що повертається, є евклідовими відстанями, взятими зі знаком мінус.

mandist(W,P) – функція аналогічна попередній, але елементи матриці, що повертається, є відстанями за Манхеттеном, які для векторів *x* і *y* визначаються співвідношенням: *D = sum(abs(x — y)).*

linkdist(pos) – функція визначення лінійної відстані між нейронами в шарі. Аналогічна функції *boxdist;* відрізняється від останньої алгоритмом визначення відстані:

D ij = 0, якщо i = j;

D ij = 1, якщо евклідова відстань між i і P j менше або рівна 1;

Dij = 2, якщо існує D таке, що D ik = D kj = 1;

D ij = 3, якщо існують k1 і k2 такі, що -Oik1 = D k1k2=... = D k2j =1;

D ij = N, якщо існують k1, k2 :::, kN такі, що -Oik1 = D k1k2 =...= D kNj =1;

D ij = S, якщо не виконана жодна з попередніх умов.

dotprod(W,P) – функція додавання входам **P** деяких вагів **W.** Повертає матрицю **Z=W*P**.

normprod(W,P) – функція аналогічна попередній, але кожний елемент матриці, що повертається, додатково ділиться на суму елементів відповідного стовпця-співмножника матриці **P**.

Завдання

1. Реалізувати у *Neural Networks Toolbox* наведені приклади щодо перетворення входів мережі та її вагів.
2. Згідно завдання викладача реалізувати 5 фунцій перетворення входів, вагів та відстаней НМ.