**Московский авиационный институт**

**(Национальный исследовательский университет)**

Факультет прикладной математики и физики

Кафедра вычислительной математики и программирования

**Лабораторная работа № 8**

по курсу «Нейроинформатика».

Тема: «Динамические сети».

Студент: Якимович А.И.

Группа: 80-408Б

Вариант: 20

Оценка:

Москва, 2018

Постановка задачи.

Целью работы является исследование свойств некоторых динамических нейронных сетей,

алгоритмов обучения, а также применение сетей в задачах аппроксимации функций и распознавания динамических образов.

Ход работы.

1. Использовать сеть прямого распространения с запаздыванием для предсказания значений временного ряда и выполнения многошагового прогноза.

2. Использовать сеть сеть прямого распространения с распределенным запаздыванием для распознавания динамических образов.

3. Использовать нелинейную авторегрессионную сеть с внешними входами для аппроксимации траектории динамической системы и выполнения многошагового прогноза.

Теория.

Они построены из динамических нейронов, чье поведение описывается дифференциальными или разностными уравнениями, как правило, первого порядка. Сеть организована так, что ка­ждый нейрон получает входную информацию от других нейронов (возможно, и от себя самого) и из окружающей среды. Этот тип сетей имеет важное значение, так как с его помощью можно моделировать нелинейные динамические системы, например: ассоциативная па­мять, нелинейная обработка сигналов, моделирование конечных автоматов, идентификация систем, задачи управления.

Такие сети используются, главным образом, в качестве ассоциативной памяти, а также в задачах нелинейной фильтрации данных, для предсказывания и для распознавания закономерностей.

Входные данные и результаты.

Вариант 20.

|  |  |
| --- | --- |
|  | **C:\Users\now20\AppData\Local\Microsoft\Windows\INetCache\Content.Word\8_1perf.pngC:\Users\now20\AppData\Local\Microsoft\Windows\INetCache\Content.Word\8_1_test.pngC:\Users\now20\AppData\Local\Microsoft\Windows\INetCache\Content.Word\8_1_train.png** |
|  | C:\Users\now20\AppData\Local\Microsoft\Windows\INetCache\Content.Word\8_2perf.pngC:\Users\now20\AppData\Local\Microsoft\Windows\INetCache\Content.Word\8_2fig2.png |
|  | C:\Users\now20\AppData\Local\Microsoft\Windows\INetCache\Content.Word\8_3_ans_out_err.pngC:\Users\now20\AppData\Local\Microsoft\Windows\INetCache\Content.Word\8_3start.png |

Выводы:

Выполнив лабораторную работу, я научился применять динамические сети для предсказания значений временного ряда и выполнения многошагового прогноза, распознавания динамических образов, аппроксимации траектории динамической систем и выполнения многошагового прогноза.

Исходный код.

filename = 'C:\Users\now20\Desktop\lab8\sun\_month\_mean\_activity.txt';

delimiterIn = ' ';

sun\_dataset = importdata(filename, delimiterIn);

sun\_dataset = sun\_dataset(1690:3227, 4);

x = smooth(sun\_dataset, 12);

D = 5;

ntrain = 500;

nval = 100;

ntest = 50;

trainInd = 1 : ntrain; % 1..500

valInd = ntrain + 1 : ntrain + nval; % 501..600

testInd = ntrain + nval + 1 : ntrain + nval + ntest; % 601..650

trainSet = x(1: ntrain + nval + ntest)';

X = con2seq(trainSet);

net = timedelaynet(1: D, 8, 'trainlm');

net.layers{1}.transferFcn = 'tansig';

net.layers{2}.transferFcn = 'purelin';

net.divideFcn = 'divideind';

net.divideParam.trainInd = trainInd;

net.divideParam.valInd = valInd;

net.divideParam.testInd = testInd;

net = configure(net, X, X);

net = init(net);

net.trainParam.epochs = 2000;

net.trainParam.max\_fail = 2000;

net.trainParam.goal = 1.0e-5;

[Xs, Xi, Ai, Ts] = preparets(net, X, X);

net = train(net, Xs, Ts);

view(net);

x\_train = x(trainInd)';

[Xs, Xi, Ai, Ts] = preparets(net, con2seq(x\_train), con2seq(x\_train));

Y = net(Xs, Xi, Ai);

figure;

hold on;

plot(x\_train, '-b');

plot([cell2mat(Xi) cell2mat(Y)], '-r');

grid on;

figure;

plot(x\_train - [cell2mat(Xi) cell2mat(Y)]);

grid on;

x\_test = x(testInd)';

[Xs, Xi, Ai, Ts] = preparets(net, con2seq(x\_test), con2seq(x\_test));

x\_val = x(valInd)';

Xi = x\_val(length(x\_val) - D + 1: length(x\_val));

Xi = con2seq(Xi);

Y = net(Xs, Xi, Ai);

figure;

hold on;

plot(x\_test, '-b');

plot([cell2mat(Xi) cell2mat(Y)], '-r');

grid on;

figure;

plot(x\_test - [cell2mat(Xi) cell2mat(Y)]);

grid on;

k1 = 0 : 0.025 : 1;

p1 = sin(4 \* pi \* k1);

t1 = -ones(size(p1));

k2 = 2.9 : 0.025 : 4.55;

%g = @(k)cos(-cos(k) .\* k .\* k + k);

g = @(k) 1.5\*sin(-5 .\* k .\* k + 10 .\* k - 5) + 0.4;

p2 = g(k2);

t2 = ones(size(p2));

R = {6; 7; 1};

P = [repmat(p1, 1, R{1}), p2, repmat(p1, 1, R{2}), p2, repmat(p1, 1, R{3}), p2];

T = [repmat(t1, 1, R{1}), t2, repmat(t1, 1, R{2}), t2, repmat(t1, 1, R{3}), t2];

Pseq = con2seq(P);

Tseq = con2seq(T);

net = distdelaynet({0: 4, 0: 4}, 10, 'trainoss');

net.layers{1}.transferFcn = 'tansig';

net.layers{2}.transferFcn = 'tansig';

net.divideFcn = '';

net = configure(net, Pseq, Tseq);

[Xs, Xi, Ai, Ts] = preparets(net, Pseq, Tseq);

net.trainParam.epochs = 1000;

net.trainParam.goal = 1.0e-5;

net = train(net, Xs, Ts, Xi, Ai);

Y = net(Xs, Xi, Ai);

figure;

hold on;

grid on;

plot(cell2mat(Tseq), '-b');

plot([cell2mat(Xi) cell2mat(Y)], '-r');

Yc = zeros(1, numel(Xi) + numel(Y));

for i = 1 : numel(Xi)

if Xi{i} >= 0

Yc(i) = 1;

else

Yc(i) = -1;

end

end

for i = numel(Xi) + 1 : numel(Y)

if Y{i} >= 0

Yc(i) = 1;

else

Yc(i) = -1;

end

end

display(nnz(Yc == cell2mat(Tseq)) / length(Tseq) \* 100)

R = {6; 3; 1};

P = [repmat(p1, 1, R{1}), p2, repmat(p1, 1, R{2}), p2, repmat(p1, 1, R{3}), p2];

T = [repmat(t1, 1, R{1}), t2, repmat(t1, 1, R{2}), t2, repmat(t1, 1, R{3}), t2];

Pseq = con2seq(P);

Tseq = con2seq(T);

[Xs, Xi, Ai, Ts] = preparets(net, Pseq, Tseq);

Y = net(Xs, Xi, Ai);

figure;

hold on;

grid on;

plot(cell2mat(Tseq), '-b');

plot([cell2mat(Xi) cell2mat(Y)], '-r');

Yc = zeros(1, numel(Xi) + numel(Y));

for i = 1 : numel(Xi)

if Xi{i} >= 0

Yc(i) = 1;

else

Yc(i) = -1;

end

end

for i = numel(Xi) + 1 : numel(Y)

if Y{i} >= 0

Yc(i) = 1;

else

Yc(i) = -1;

end

end

display(nnz(Yc == cell2mat(Tseq)) / length(Tseq) \* 100)

t0 = 0;

tn = 10;

dt = 0.01;

n = (tn - t0) / dt + 1;

fun = @(k)cos(k.^2-2.\*k + 3);

fun2 = @(y, u)y ./ (1 + y.^2) + u.^3;

u = zeros(1, n);

y = zeros(1, n);

u(1) = fun(0);

for i = 2 : n

t = t0 + (i - 1) \* dt;

y(i) = fun2(y(i - 1), u(i - 1));

u(i) = fun(t);

end

figure

subplot(2, 1, 1)

plot(t0 : dt : tn, u, '-b'), grid

ylabel('control')

subplot(2, 1, 2)

plot(t0 : dt : tn, y, '-r'), grid

ylabel('state')

xlabel('t')

x = u;

D = 3;

ntrain = 700;

nval = 200;

ntest = 97;

trainInd = 1 : ntrain;

valInd = ntrain + 1 : ntrain + nval;

testInd = ntrain + nval + 1 : ntrain + nval + ntest;

net = narxnet(1 : 3, 1 : 3, 10);

net.trainFcn = 'trainlm';

net.layers{1}.transferFcn = 'tansig';

net.layers{2}.transferFcn = 'purelin';

net.divideFcn = 'divideind';

net.divideParam.trainInd = trainInd;

net.divideParam.valInd = valInd;

net.divideParam.testInd = testInd;

net = init(net);

net.trainParam.epochs = 600;

net.trainParam.max\_fail = 600;

net.trainParam.goal = 1.0e-8;

[Xs, Xi, Ai, Ts] = preparets(net, con2seq(x), {}, con2seq(y));

net = train(net, Xs, Ts, Xi, Ai);

Y = sim(net, Xs, Xi, Ai);

figure

subplot(3, 1, 1)

plot(t0 : dt : tn, u, '-b'),grid

ylabel('Answer')

subplot(3, 1, 2)

plot(t0 : dt : tn, x, '-b', t0 : dt : tn, [x(1:D) cell2mat(Y)], '-r'), grid

ylabel('Out')

subplot(3, 1, 3)

plot(t0+D\*dt : dt : tn, x(D+1:end) - cell2mat(Y)), grid

ylabel('Error(train)')

last\_elem\_x = x(valInd(length(valInd)-2 : length(valInd)));

last\_elem\_u = u(valInd(length(valInd)-2 : length(valInd)));

inp = [con2seq(u(testInd)); con2seq(x(testInd))];

delay = [con2seq(last\_elem\_u); con2seq(last\_elem\_x)];

out\_for\_test = sim(net, inp, delay, Ai);

figure;

hold on;

plot(x(testInd), '.-b')

plot(cell2mat(out\_for\_test), '-or');

grid minor;

grid on;

figure;

hold on;

errortst = cell2mat(out\_for\_test) - x(testInd);

plot(errortst, '-r');

legend('Error (test)');

grid minor;

grid on;