**Московский авиационный институт**

**(Национальный исследовательский университет)**

Факультет прикладной математики и физики

Кафедра вычислительной математики и программирования

**Лабораторная работа № 5**

по курсу «Нейроинформатика».

Тема: «Сети с обратными связями».

Студент: Якимович А.И.

Группа: 80-408Б

Вариант: 20

Оценка:

Москва, 2018

Постановка задачи.

Целью работы является исследование свойств сетей Хопфилда, Хэмминга и Элмана, алгоритмов обучения, а также применение сетей в задачах распознавания статических и динамических образов.

Ход работы.

1. Построить и обучить сеть Элмана, которая будет выполнять распознавание динамического образа. Проверить качество распознавания. Подать значения из варианта на вход алгоритма.
2. Построить сеть Хопфилда, которая будет хранить образы из заданного набора. Эталонными образами являются двоичные изображения цифр 0, 1, 2 размером 12х10. Проверить работу сети с зашумленными образами.
3. Построить сеть Хэммиинга, которая будет хранить образы из заданного набора. Эталонными образами являются двоичные изображения цифр 0, 1, 2 размером 12х10. Проверить работу сети с зашумленными образами.

Теория.

Сеть Хопфилда - полносвязная нейронная сеть с симметричной матрицей связей. В процессе работы динамика таких сетей сходится (конвергирует) к одному из положений равновесия. Эти положения равновесия определяются заранее в процессе обучения, они являются локальными минимумами функционала, называемого энергией сети (в простейшем случае — локальными минимумами отрицательно определённой квадратичной формы на n-мерном кубе). Такая сеть может быть использована как автоассоциативная память, как фильтр, а также для решения некоторых задач оптимизации. В отличие от многих нейронных сетей, работающих до получения ответа через определённое количество тактов, сети Хопфилда работают до достижения равновесия, когда следующее состояние сети в точности равно предыдущему: начальное состояние является входным образом, а при равновесии получают выходной образ.

Нейронная сеть Хэмминга (Хемминга) — вид нейронной сети, использующийся для классификации бинарных векторов, основным критерием в которой является расстояние Хэмминга. Является развитием нейронной сети Хопфилда.

Нейронная сеть Элмана — один из видов рекуррентной сети, которая получается из многослойного перцептрона введением обратных связей, только связи идут не от выхода сети, а от выходов внутренних нейронов. Это позволяет учесть предысторию наблюдаемых процессов и накопить информацию для выработки правильной стратегии управления. Эти сети могут применяться в системах управления движущимися объектами, так как их главной особенностью является запоминание последовательностей.

Входные данные и результаты.

Вариант 20.

|  |  |
| --- | --- |
| **Вход** | **Выход** |
| C:\Users\now20\AppData\Local\Microsoft\Windows\INetCache\Content.Word\5_1nnt.png | C:\Users\now20\AppData\Local\Microsoft\Windows\INetCache\Content.Word\5_1.jpg |
|  | C:\Users\now20\AppData\Local\Microsoft\Windows\INetCache\Content.Word\5_2image0.jpgC:\Users\now20\AppData\Local\Microsoft\Windows\INetCache\Content.Word\5_2image2.jpgC:\Users\now20\AppData\Local\Microsoft\Windows\INetCache\Content.Word\5_2image3.jpg |
| {noised2,noised3} | {2,3} |

Выводы:

Выполнив лабораторную работу, я научился использовать сети Хопфилда, Хэмминга для распознавания статических образов и сеть Элмана для распознавания динамических образов, применять метод секущих для обучения сетей.

У нейронной сети Хопфилда есть ряд недостатков. Во первых, это относительно небольшой объём памяти, величину которого можно оценить выражением: N/ 2lnN (попытка записи большего числа образов приводит к тому, что нейронная сеть перестаёт их распознавать.Во вторых, достижение устойчивого состояния не гарантирует правильный ответ сети. Это происходит из-за того, что сеть может сойтись к так называемым ложным аттракторам, иногда называемым «химерами».

Исходный код.

p1\_args = 0:0.025:1;

p1\_values = sin(4 \* pi \* p1\_args);

t1 = ones(1,length(p1\_args));

p2\_args = 0.78:0.025:2.35;

p2\_values = 1.5 \* sin(-5 \* power(p2\_args,2) + 10 \* p2\_args - 5) + 0.4;

t2 = -1 \* ones(1, length(p2\_args));

R = [2 2 5];

P = [repmat(p1\_values,1,5),p2\_values,repmat(p1\_values,1,3),p2\_values,repmat(p1\_values,1,3),p2\_values];

T = [repmat(t1,1,5),t2,repmat(t1,1,3),t2,repmat(t1,1,3),t2];

P = con2seq(P);

T = con2seq(T);

net = layrecnet(1:2,8,'trainoss');

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

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

net = configure(net, P, T);

[Ps,Pi,Ai,Ts] = preparets(net,P,T);

net.trainParam.epochs = 100;

net.trainParam.goal = 1e-5;

% view(net)

net = train(net, Ps, Ts, Pi, Ai);

Y = sim(net, Ps);

Y = cell2mat(Y);

Y(Y >= 0) = 1;

Y(Y < 0) = -1;

Ps = cell2mat(Ps);

Ts = cell2mat(Ts);

hold on

plot(1:1:(length(Ts)), Ts, 'color','r')

plot(1:1:(length(Ts)), Y, 'color','g')

hold off

target0 = [

-1 -1 -1 -1 -1 -1 -1 -1 -1 -1;

-1 -1 -1 1 1 1 1 -1 -1 -1;

-1 -1 1 1 1 1 1 1 -1 -1;

-1 1 1 -1 -1 -1 -1 1 1 -1;

-1 1 1 -1 -1 -1 -1 1 1 -1;

-1 1 1 -1 -1 -1 -1 1 1 -1;

-1 1 1 -1 -1 -1 -1 1 1 -1;

-1 1 1 -1 -1 -1 -1 1 1 -1;

-1 1 1 -1 -1 -1 -1 1 1 -1;

-1 -1 1 1 1 1 1 1 -1 -1;

-1 -1 -1 1 1 1 1 -1 -1 -1;

-1 -1 -1 -1 -1 -1 -1 -1 -1 -1;

];

target2 = [

1 1 1 1 1 1 1 1 -1 -1;

1 1 1 1 1 1 1 1 -1 -1;

-1 -1 -1 -1 -1 -1 1 1 -1 -1;

-1 -1 -1 -1 -1 -1 1 1 -1 -1;

-1 -1 -1 -1 -1 -1 1 1 -1 -1;

1 1 1 1 1 1 1 1 -1 -1;

1 1 1 1 1 1 1 1 -1 -1;

1 1 -1 -1 -1 -1 -1 -1 -1 -1;

1 1 -1 -1 -1 -1 -1 -1 -1 -1;

1 1 -1 -1 -1 -1 -1 -1 -1 -1;

1 1 1 1 1 1 1 1 -1 -1;

1 1 1 1 1 1 1 1 -1 -1;

];

target3 = [

-1 -1 1 1 1 1 1 1 -1 -1;

-1 -1 1 1 1 1 1 1 1 -1;

-1 -1 -1 -1 -1 -1 -1 1 1 -1;

-1 -1 -1 -1 -1 -1 -1 1 1 -1;

-1 -1 -1 -1 -1 -1 -1 1 1 -1;

-1 -1 -1 -1 1 1 1 1 -1 -1;

-1 -1 -1 -1 1 1 1 1 -1 -1;

-1 -1 -1 -1 -1 -1 -1 1 1 -1;

-1 -1 -1 -1 -1 -1 -1 1 1 -1;

-1 -1 -1 -1 -1 -1 -1 1 1 -1;

-1 -1 1 1 1 1 1 1 1 -1;

-1 -1 1 1 1 1 1 1 -1 -1;

];

target6 = [

1 1 1 1 1 1 -1 -1 -1 -1;

1 1 1 1 1 1 -1 -1 -1 -1;

1 1 -1 -1 -1 -1 -1 -1 -1 -1;

1 1 -1 -1 -1 -1 -1 -1 -1 -1;

1 1 -1 -1 -1 -1 -1 -1 -1 -1;

1 1 1 1 1 1 -1 -1 -1 -1;

1 1 1 1 1 1 -1 -1 -1 -1;

1 1 -1 -1 1 1 -1 -1 -1 -1;

1 1 -1 -1 1 1 -1 -1 -1 -1;

1 1 -1 -1 1 1 -1 -1 -1 -1;

1 1 1 1 1 1 -1 -1 -1 -1;

1 1 1 1 1 1 -1 -1 -1 -1;

];

T = [target6(:), target2(:), target3(:)];

net = newhop(T);

noised2 = noise(target0(:), 24);

noised3 = noise(target3(:), 36);

noised6 = noise(target6(:), 24);

res\_1 = sim(net,1,[],noised6);

res\_1(res\_1 >= 0) = 2;

res\_1(res\_1 < 0) = 1;

res\_1 = reshape(res\_1,12,10);

noised3(noised3 >= 0) = 2;

noised3(noised3 < 0) = 1;

noised3 = reshape(noised3,12,10);

noised6(noised6 >= 0) = 2;

noised6(noised6 < 0) = 1;

noised6 = reshape(noised6,12,10);

image(noised6);

%image(res\_1);

colormap([1, 1, 1; 0, 0, 0]);

axis off

axis image

target0 = [

-1 -1 -1 -1 -1 -1 -1 -1 -1 -1;

-1 -1 -1 1 1 1 1 -1 -1 -1;

-1 -1 1 1 1 1 1 1 -1 -1;

-1 1 1 -1 -1 -1 -1 1 1 -1;

-1 1 1 -1 -1 -1 -1 1 1 -1;

-1 1 1 -1 -1 -1 -1 1 1 -1;

-1 1 1 -1 -1 -1 -1 1 1 -1;

-1 1 1 -1 -1 -1 -1 1 1 -1;

-1 1 1 -1 -1 -1 -1 1 1 -1;

-1 -1 1 1 1 1 1 1 -1 -1;

-1 -1 -1 1 1 1 1 -1 -1 -1;

-1 -1 -1 -1 -1 -1 -1 -1 -1 -1;

];

target2 = [

1 1 1 1 1 1 1 1 -1 -1;

1 1 1 1 1 1 1 1 -1 -1;

-1 -1 -1 -1 -1 -1 1 1 -1 -1;

-1 -1 -1 -1 -1 -1 1 1 -1 -1;

-1 -1 -1 -1 -1 -1 1 1 -1 -1;

1 1 1 1 1 1 1 1 -1 -1;

1 1 1 1 1 1 1 1 -1 -1;

1 1 -1 -1 -1 -1 -1 -1 -1 -1;

1 1 -1 -1 -1 -1 -1 -1 -1 -1;

1 1 -1 -1 -1 -1 -1 -1 -1 -1;

1 1 1 1 1 1 1 1 -1 -1;

1 1 1 1 1 1 1 1 -1 -1;

];

target3 = [

-1 -1 1 1 1 1 1 1 -1 -1;

-1 -1 1 1 1 1 1 1 1 -1;

-1 -1 -1 -1 -1 -1 -1 1 1 -1;

-1 -1 -1 -1 -1 -1 -1 1 1 -1;

-1 -1 -1 -1 -1 -1 -1 1 1 -1;

-1 -1 -1 -1 1 1 1 1 -1 -1;

-1 -1 -1 -1 1 1 1 1 -1 -1;

-1 -1 -1 -1 -1 -1 -1 1 1 -1;

-1 -1 -1 -1 -1 -1 -1 1 1 -1;

-1 -1 -1 -1 -1 -1 -1 1 1 -1;

-1 -1 1 1 1 1 1 1 1 -1;

-1 -1 1 1 1 1 1 1 -1 -1;

];

target6 = [

1 1 1 1 1 1 -1 -1 -1 -1;

1 1 1 1 1 1 -1 -1 -1 -1;

1 1 -1 -1 -1 -1 -1 -1 -1 -1;

1 1 -1 -1 -1 -1 -1 -1 -1 -1;

1 1 -1 -1 -1 -1 -1 -1 -1 -1;

1 1 1 1 1 1 -1 -1 -1 -1;

1 1 1 1 1 1 -1 -1 -1 -1;

1 1 -1 -1 1 1 -1 -1 -1 -1;

1 1 -1 -1 1 1 -1 -1 -1 -1;

1 1 -1 -1 1 1 -1 -1 -1 -1;

1 1 1 1 1 1 -1 -1 -1 -1;

1 1 1 1 1 1 -1 -1 -1 -1;

];

T = [target6(:), target2(:), target3(:)];

Q = 3;

R = 120;

eps = 1/(Q);

% fisrt layer init thing

fnet = newlin(T,3);

fnet.IW = {[target6(:)'; target2(:)'; target3(:)']};

fnet.b = {repmat(R,3,1)};

sm = sim(fnet, T);

% second layer init

net = newhop(sm);

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

LW = eye(3);

LW = LW(:);

LW(LW == 0) = -eps;

LW = reshape(LW,3,3);

net.LW = {LW};

net.B = {repmat(R,3,1)};

noised6 = noise(target6(:), 24);

noised3 = noise(target3(:), 36);

temp1 = sim(fnet,noised6);

temp2 = sim(fnet,noised3);

% res = sim(net, 3, [], sm);

res1 = sim(net,1,[],temp1);

res2 = sim(net,1,[],temp2);

[~,j1] = max(res1);

[~,j2] = max(res2);