**智能计算第二次实验报告**

**一、实验目的及要求**

**理解并掌握Hopfield神经网络，并用所学 求解TSP问题，且能自主设计一个Hopfield神经网络，使其能够正确识别阿拉伯数字，并具有联想记忆功能。**

**二、实验内容**

**1.Hopfield网络求解TSP问题**

**北京等六城市的距离表如下。请据此表应用Hopfield网络求解TSP（Travelling Salesman Problem） 问题。即给定N个城市，从某一城市开始走遍所有城市但不许重复，最后回到出发点，要求总路径最短。**

**六城市间距离表**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **上海** | **天津** | **石家庄** | **太原** | **呼和浩特** |
| **北京** | **1078** | **119** | **263** | **398** | **401** |
| **上海** | **0** | **963** | **989** | **1096** | **1381** |
| **天津** |  | **0** | **262** | **426** | **504** |
| **石家庄** |  |  | **0** | **171** | **394** |
| **太原** |  |  |  | **0** | **341** |

**2设计一个Hopfield网络，使其具有联想记忆功能，能正确识别阿拉伯数字，当数字被噪声污染后仍可以正确地识别。**

**三、实验基本原理**

**3.1 TSP问题描述**

**Hopfield神经网络是一种递归神经网络，由约翰·霍普菲尔德在1982年发明。Hopfield网络是一种结合存储系统和二元系统的神经网络。它保证了向局部极小的收敛，但收敛到错误的局部极小值（local minimum），而非全局极小（global minimum）的情况也可能发生。Hopfield网络也提供了模拟人类记忆的模型。**

**离散Hopfield网络是一个单层网络，有n个神经元节点，每个神经元的输出均接到其它神经元的输入。各节点没有自反馈。每个节点都可处于一种可能的状态（1或－1），即当该神经元所受的刺激超过其阀值时，神经元就处于一种状态（比如1），否则神经元就始终处于另一状态（比如 －1）。 整个网络有两种工作方式：即异步方式和同步方式。**

**Hopfield网络的单元是二元的（binary），即这些单元只能接受两个不同的值，并且值取决于输入的大小是否达到阈值。Hopfield网络通常接受值为-1或1，也可以是0或者1。输入是由sigmoid函数处理得到的。用于将输入化简为两个极值。**

**每一对Hopfiled网络的单元*i*和*j*间都有一对以一定权重（weight）的连接。因此，Hopfiled网络可被描述为一个完整的无向图G=<V,f>，其中V是人工神经元集合。**

**Hopfiled网络的连接有以下特征：**

**1没有神经元和自身相连**

**2连接权重是对称的**

**权重对称的要求是一个重要特征，因为它保证了能量方程（称向函数某一点收敛的过程为势能转化为能量）在神经元激活时单调递减，而不对称的权重可能导致周期性的递增或者噪声。然而，Hopfiled网络也证明噪声过程会被局限在很小的范围，并且并不影响网络的最终性能。**

**联想记忆功能是离散Hopfield网络的一个重要应用范围。要想实现联想记忆，反馈网络必须具有两个基本条件：**

**① 网络能收敛到稳定的平衡状态，并以其作为样本的记忆信息；**

**② 具有回忆能力，能够从某一残缺的信息回忆起所属的完整的记忆信息。 离散Hopfield网络实现联想记忆的过程分为两个阶段：学习记忆阶段和联想回忆阶段。在学习记忆阶段中，设计者通过某一设计方法确定一组合适的权值，使网络记忆期望的稳定平衡点。联想回忆阶段则是网络的工作过程。**

**离散Hopfield网络用于联想记忆有两个突出的特点：即记忆是分布式的，而联想是动态的。 离散Hopfield网络局限性，主要表现在以下几点：**

**① 记忆容量的有限性；**

**② 伪稳定点的联想与记忆；**

**③ 当记忆样本较接近时，网络不能始终回忆出正确的记忆等。另外网络的平衡稳定点并不可以任意设置的，也没有一个通用的方式来事先知道平衡稳定点。**

**TSP问题是典型的组合优化问题。给定个城市以及各个城市之间的距离，要求找到一条遍历所有城市且每个城市只被访问一次的路线，并使得总路线距离最短。其可用数学式来表达：**

**设有一个城市集合，其中每对城市之间的距离。求一对经过中每个城市一次的路线。使**

**其中是的一个置换。**

**Hopfield神经网络模型是一种循环神经网络，输出和输入都有反馈连接。在输出的激励下，会产生不断的状态变化。与BP神经网络不同的是，它引入类似于函数的能量函数概念，把用连接权矩阵表示的神经网络的拓扑结构与所求TSP问题的函数描述相对应，并将其转换为神经网动力学系统的演化问题。**

**Hopfield神经网络有稳定连续的，也有不稳定离散的，判断两者的关键在于确定它在稳定条件下的权系数。**

**连续的Hopfield神经网络(CHNN)主要用于优化计算，由3.1可知，TSP问题是一种优化问题，故可用其求解。**

**离散的Hopfield神经网络（DHNN）主要用于联想记忆。输入和输出模式的各元素之间，并不存在一对一的映射关系，输入-输出模式的维数也不要求相同：联想记忆时，只需给出输入模式部分信息就能联想出完整的输出模式，即具有容错性。故本实验第二题可用离散的Hopfield神经网络来解决。**

**3.3 算法步骤**

**由于连续型Hopfield神经网络是由离散型Hopfield神经网络发展而来，故这里只说明离散型Hopfield神经网络的计算步骤。连续型Hopfield神经网络的概述放在4.1.1里称述。**

**离散型Hopfield神经网络主要有两种运行规则**

**一是异步工作方式，即在任一时刻，只有某一神经元依照激励函数:**

**变化，而其他神经元的状态不变。**

**对网络进行初始化；**

**从网络中随机抽取一个神经元；**

**求出神经元i的输出；**

**求出该神经元经过激活函数处理后的输出，此时其余在网络中的神经元的输出保持不变；**

**判断网络是否达到稳定状态（这同样也是连续型网络的关键）若达到稳定状态或满足给定条件则结束，否则将转到之前继续运行。**

**四、程序简述**

**Hopfield神经网络应用于求解TSP问题，就是把目标函数转化为网络的能量函数，把问题的变量对应于网络的状态。这样当网络的能量函数收敛于极小值时，问题的最优解也随之求出。**

**将问题映射到一个神经网络。**

**将题目给出的城市距离，为所有城市定出坐标，然后对数据进行归一化处理，即将神经元的输出限制在0和1上，映射时可以用换位矩阵来表示。对n个城市，需要有个神经元构成的的矩阵来表示行进路线。换位矩阵中每列和每行都只有一个元素为1，其余元素都为0.通过这样的矩阵，可以唯一地确定一条路径。**

**构造与TSP问题对应的能量函数**

**为了让网络的能量最低点，其所对应的行进路线最短，需要构造一个能量函数。用表示第行，第列的元素，则其取值为1或0，同时这也是网络神经元的状态。**

**根据网络的约束条件，构造的能量函数包含四项：**

1. **每行元素两两相乘的和为**

**B. 每列元素两两相乘的和为**

**C. 换位矩阵的元素和为**

**D. 所有经过的路径最短为：**

**则能量函数的最终形式为：**

**当E达到极小值时，有网络状态构成的换位矩阵表达最短行进路径。**

**求出网络的连接权系数：**

**根据上式，有公式可以求出Hopfield神经网络的连接权系数：**

**其中在时为1，其余时候为0。**

**求解TSP网络的迭代方程。**

**求出各神经元的输出，即输出方程如下：**

**状态更新公式：**

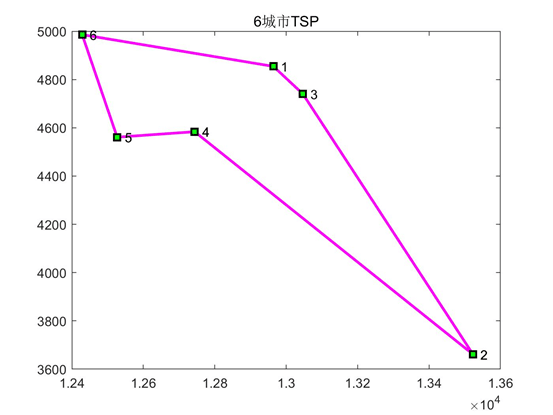
**4.2.1问题的求解**

**由3.2实验原理中，说明了选择离散型Hopfield神经网络的原因。**

**设网络由10个初始稳态值1、2、3、4、5、6、7、8、9、0构成，每一个稳态值由的矩阵组成，用以模拟阿拉伯数字点阵，其中，有点处用1表示，空白处用-1的表示如代码所示，矩阵内共100个元素**

**给定一个加有噪点的数字2，其矩阵在6.2的代码已经给出，在实验过程中可以注意到，这些噪点是有那些原来是1的值变异成-1得出的，或者原来是-1的值变异成1的得出的。然后训练网络，仿真结果表明该网络具有联想记忆功能，在数字2被污染后仍能识别出原本的数字。**

**五、实验结果与分析**

****

**图中数字分别代表的城市：1.北京 2.上海 3.天津 4.石家庄 5.太原 6.呼和浩特。从任何一个数字开始顺时针或者逆时针绕一圈就能使走过的路程最短。**

**（1）具有良好的收敛性，但若想要达成收敛性需要找准问题对应的能量函数**

**（2）具有有限个平衡点。**

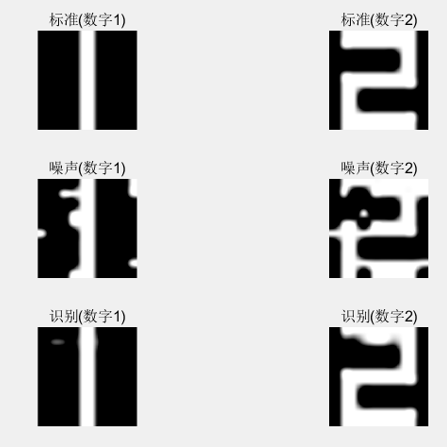
**（3）网络的存储信息表现为神经元之间互连的分布式动态存储，这点与BP算法不同。**

**（4）Hopfield神经网络以大规模、非线性、连续时间并行方式处理问题的基础数据，其计算时间就是网络趋向于平衡点的时间。**

**经由查阅文献发现，有研究表明，基于Hopfield模型的优化计算易于收敛到非法解或局部极小解。以及算法对模型的参数和初始条件具有很强的依赖性，存在不稳健性。可针对随机扰动的缺点，归一化坐标到（0,1）区间，引入一个偏置到初值来改善优化性能，通过TSP网络的动态分析修正TSP的能量函数，从而获得有效解。**

**设计一个具有联想记忆，并能正确识别阿拉伯数字的Hopfield神经网络**

**运行6.2的代码可绘出以下图像：**

****

**结果和数字2的正常点阵是一致的，说明网络从受到污染的数字2点阵中识别出了数字2，此网络是有效的。**

**六、实验代码**

**Clc;clear;**

**desktop\_text\_CN=10;**

**[desktop\_text\_dislist,Clist]=tsp(desktop\_text\_CN);**

**desktop\_text\_A=500;desktop\_text\_B=500;**

**desktop\_text\_C=200;desktop\_text\_D=500;**

**desktop\_text\_F=1;desktop\_text\_m=0.02;**

**desktop\_text\_la=0.00001;desktop\_text\_EN=1000;**

**desktop\_text\_y=zeros(desktop\_text\_CN,desktop\_text\_CN);**

**for desktop\_text\_i=1:desktop\_text\_CN**

**desktop\_text\_y(desktop\_text\_i,desktop\_text\_i)=1;**

**end**

**desktop\_text\_z=-desktop\_text\_m/2\*log(9)\*ones(desktop\_text\_CN,desktop\_text\_CN);**

**delu=0.1\*desktop\_text\_m\*rand(desktop\_text\_CN,desktop\_text\_CN);**

**figure(1);**

**for k=1:desktop\_text\_EN**

**desktop\_text\_z=desktop\_text\_z+desktop\_text\_la\*delu;**

**for desktop\_text\_u=1:desktop\_text\_CN**

**for desktop\_text\_i=1:desktop\_text\_CN**

**desktop\_text\_y(desktop\_text\_u,desktop\_text\_i)=1/(1+exp(-2\*desktop\_text\_z(desktop\_text\_u,desktop\_text\_i)/desktop\_text\_m));**

**end**

**end**

**for desktop\_text\_u=1:desktop\_text\_CN**

**for desktop\_text\_i=1:desktop\_text\_CN**

**desktop\_text\_A1=0;**

**desktop\_text\_B1=0;**

**for desktop\_text\_aa=1:desktop\_text\_CN**

**desktop\_text\_A1=desktop\_text\_A1+desktop\_text\_y(desktop\_text\_u,desktop\_text\_aa);**

**desktop\_text\_B1=desktop\_text\_B1+desktop\_text\_y(desktop\_text\_aa,desktop\_text\_i);**

**end**

**desktop\_text\_A1=desktop\_text\_A1-desktop\_text\_y(desktop\_text\_u,desktop\_text\_i);**

**desktop\_text\_B1=desktop\_text\_B1-desktop\_text\_y(desktop\_text\_u,desktop\_text\_i);**

**desktop\_text\_C1=0;**

**for desktop\_text\_aa=1:desktop\_text\_CN**

**for bb=1:desktop\_text\_CN**

**desktop\_text\_C1=desktop\_text\_C1+desktop\_text\_y(desktop\_text\_aa,bb);**

**end**

**end**

**desktop\_text\_C1=desktop\_text\_C1-desktop\_text\_CN;**

**desktop\_text\_D1=0;**

**for desktop\_text\_x=1:desktop\_text\_CN**

**if desktop\_text\_x~=desktop\_text\_u**

**if desktop\_text\_i==1**

**desktop\_text\_D1=desktop\_text\_D1+desktop\_text\_dislist(desktop\_text\_u,desktop\_text\_x)\*(desktop\_text\_y(desktop\_text\_x,2)+desktop\_text\_y(desktop\_text\_x,desktop\_text\_CN));**

**elseif desktop\_text\_i==desktop\_text\_CN**

**desktop\_text\_D1=desktop\_text\_D1+desktop\_text\_dislist(desktop\_text\_u,desktop\_text\_x)\*(desktop\_text\_y(desktop\_text\_x,1)+desktop\_text\_y(desktop\_text\_x,desktop\_text\_CN-1));**

**else**

**desktop\_text\_D1=desktop\_text\_D1+desktop\_text\_dislist(desktop\_text\_u,desktop\_text\_x)\*(desktop\_text\_y(desktop\_text\_x,desktop\_text\_i+1)+desktop\_text\_y(desktop\_text\_x,desktop\_text\_i-1));**

**end**

**end**

**end**

**delu(desktop\_text\_u,desktop\_text\_i)=-desktop\_text\_z(desktop\_text\_u,desktop\_text\_i)\*desktop\_text\_F-desktop\_text\_A\*desktop\_text\_A1-desktop\_text\_B\*desktop\_text\_B1-desktop\_text\_C\*desktop\_text\_C1-desktop\_text\_D\*desktop\_text\_D1;**

**end**

**end**

**end**

**function [desktop\_text\_DLn,desktop\_text\_cityn]=tsp(desktop\_text\_n)**

**if desktop\_text\_n==10**

**desktop\_text\_city10=[0.4 0.4439;0.2439 0.1463;0.1707 0.2293;0.2293 0.761;0.5171 0.9414;**

**0.8732 0.6536;0.6878 0.5219;0.8488 0.3609;0.6683 0.2536;0.6195 0.2634];**

**for desktop\_text\_i=1:10**

**for desktop\_text\_j=1:10**

**DL10(desktop\_text\_i,desktop\_text\_j)=((desktop\_text\_city10(desktop\_text\_i,1)-desktop\_text\_city10(desktop\_text\_j,1))^2+(desktop\_text\_city10(desktop\_text\_i,2)-desktop\_text\_city10(desktop\_text\_j,2))^2)^0.5;**

**end**

**end**

**desktop\_text\_DLn=DL10;**

**desktop\_text\_cityn=desktop\_text\_city10;**

**end**

**if desktop\_text\_n==30**

**desktop\_text\_city30=[41 94;37 84;54 67;25 62;7 64;2 99;68 58;71 44;54 62;83 69;64 60;18 54;22 60;**

**83 46;91 38;25 38;24 42;58 69;71 71;74 78;87 76;18 40;13 40;82 7;62 32;58 35;45 21;41 26;44 35;4 50];**

**for desktop\_text\_i=1:30**

**for desktop\_text\_j=1:30**

**desktop\_text\_DL30(desktop\_text\_i,desktop\_text\_j)=((desktop\_text\_city30(desktop\_text\_i,1)-desktop\_text\_city30(desktop\_text\_j,1))^2+(desktop\_text\_city30(desktop\_text\_i,2)-desktop\_text\_city30(desktop\_text\_j,2))^2)^0.5;**

**end**

**end**

**desktop\_text\_DLn=desktop\_text\_DL30;**

**desktop\_text\_cityn=desktop\_text\_city30;**

**end**

**if desktop\_text\_n==50**

**desktop\_text\_city50=[31 32;32 39;40 30;37 69;27 68;37 52;38 46;31 62;30 48;21 47;25 55;16 57;**

**17 63;42 41;17 33;25 32;5 64;8 52;12 42;7 38;5 25; 10 77;45 35;42 57;32 22;**

**27 23;56 37;52 41;49 49;58 48;57 58;39 10;46 10;59 15;51 21;48 28;52 33;**

**58 27;61 33;62 63;20 26;5 6;13 13;21 10;30 15;36 16;62 42;63 69;52 64;43 67];**

**for desktop\_text\_i=1:50**

**for desktop\_text\_j=1:50**

**desktop\_text\_DL50(desktop\_text\_i,desktop\_text\_j)=((desktop\_text\_city50(desktop\_text\_i,1)-desktop\_text\_city50(desktop\_text\_j,1))^2+(desktop\_text\_city50(desktop\_text\_i,2)-desktop\_text\_city50(desktop\_text\_j,2))^2)^0.5;**

**end**

**end**

**desktop\_text\_DLn=desktop\_text\_DL50;**

**desktop\_text\_cityn=desktop\_text\_city50;**

**end**

**if desktop\_text\_n==75**

**city75=[48 21;52 26;55 50;50 50;41 46;51 42;55 45;38 33;33 34;45 35;40 37;50 30;**

**55 34;54 38;26 13;15 5;21 48;29 39;33 44;15 19;16 19;12 17;50 40;22 53;21 36;**

**20 30;26 29;40 20;36 26;62 48;67 41;62 35;65 27;62 24;55 20;35 51;30 50;**

**45 42;21 45;36 6;6 25;11 28;26 59;30 60;22 22;27 24;30 20;35 16;54 10;50 15;**

**44 13;35 60;40 60;40 66;31 76;47 66;50 70;57 72;55 65;2 38;7 43;9 56;15 56;**

**10 70;17 64;55 57;62 57;70 64;64 4;59 5;50 4;60 15;66 14;66 8;43 26];**

**for desktop\_text\_i=1:75**

**for desktop\_text\_j=1:75**

**DL75(desktop\_text\_i,desktop\_text\_j)=((city75(desktop\_text\_i,1)-city75(desktop\_text\_j,1))^2+(city75(desktop\_text\_i,2)-city75(desktop\_text\_j,2))^2)^0.5;**

**end**

**clear**

**clc**

**desktop\_text\_array\_one=[-1 -1 -1 -1 1 1 -1 -1 -1 -1;-1 -1 -1 -1 1 1 -1 -1 -1 -1;**

**-1 -1 -1 -1 1 1 -1 -1 -1 -1;-1 -1 -1 -1 1 1 -1 -1 -1 -1;**

**-1 -1 -1 -1 1 1 -1 -1 -1 -1;-1 -1 -1 -1 1 1 -1 -1 -1 -1;**

**-1 -1 -1 -1 1 1 -1 -1 -1 -1;-1 -1 -1 -1 1 1 -1 -1 -1 -1;**

**-1 -1 -1 -1 1 1 -1 -1 -1 -1;-1 -1 -1 -1 1 1 -1 -1 -1 -1];**

**desktop\_text\_array\_two=[-1 1 1 1 1 1 1 1 1 -1;-1 1 1 1 1 1 1 1 1 -1;**

**-1 -1 -1 -1 -1 -1 -1 1 1 -1;-1 -1 -1 -1 -1 -1 -1 1 1 -1;**

**-1 1 1 1 1 1 1 1 1 -1;-1 1 1 1 1 1 1 1 1 -1;**

**-1 1 1 -1 -1 -1 -1 -1 -1 -1;-1 1 1 -1 -1 -1 -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=[desktop\_text\_array\_one;desktop\_text\_array\_two]';**

**net=newhop(T);**

**noisy\_array\_one = desktop\_text\_array\_one;**

**noisy\_array\_two = desktop\_text\_array\_two;**

**for desktop\_text\_i=1:100**

**desktop\_text\_a=rand;**

**if (desktop\_text\_a<0.1)**

**noisy\_array\_one(desktop\_text\_i)=-desktop\_text\_array\_one(desktop\_text\_i);**

**noisy\_array\_two(desktop\_text\_i)=-desktop\_text\_array\_two(desktop\_text\_i);**

**end**

**end**

**desktop\_text\_noisy\_one={(noisy\_array\_one)'};**

**desktop\_text\_identify\_one=sim(net,{10,10},{},desktop\_text\_noisy\_one);**

**desktop\_text\_identify\_one{10}';**

**desktop\_text\_noisy\_two={(noisy\_array\_two)'};**

**desktop\_text\_identify\_two=sim(net,{10,10},{},desktop\_text\_noisy\_two);**

**desktop\_text\_identify\_two{10}';**

**%% 结果显示**

**desktop\_text\_Array\_one=imresize(array\_one,20);**

**subplot(3,2,1)**

**imshow(desktop\_text\_Array\_one)**

**title('标准(数字1)')**

**desktop\_text\_Array\_two=imresize(array\_two,20);**

**subplot(3,2,2)**

**imshow(desktop\_text\_Array\_two)**

**title('标准(数字2)')**

**subplot(3,2,3)**

**desktop\_text\_Noisy\_array\_one=imresize(noisy\_array\_one,20);**

**imshow(desktop\_text\_Noisy\_array\_one)**

**title('噪声(数字1)')**

**subplot(3,2,4)**

**desktop\_text\_Noisy\_array\_two=imresize(noisy\_array\_two,20);**

**imshow(desktop\_text\_Noisy\_array\_two)**

**title('噪声(数字2)')**

**subplot(3,2,5)**

**imshow(imresize(desktop\_text\_identify\_one{10}',20))**

**title('识别(数字1)')**

**subplot(3,2,6)**

**imshow(imresize(desktop\_text\_identify\_two{10}',20))**

**title('识别(数字2)')**