深圳大学研究生课程论文

题目 模式识别与人工神经网络 成绩

专业 生物医学工程(工) 课程名称、代码 152022020015

年级 2014 级 姓名 李婉君

学 号 2140220212 时间 2015 年 6 月

任课教师 汪天富

模式识别

第零章　纹理特征提取

编程实现至少两种纹理特征提取方法。

第四章　线性判别函数

随机产生两类二维样本，编写程序用 Fisher 线性判别函数和感知器准则函数对其进行分类。

第五章　非线性判别函数

随机产生多类二维样本，编写程序用最小距离分类器和 K 近邻分类器对其进行分类。

第七章　聚类分析

随机产生多类二维样本，编写程序用 C--均值聚类和 ISODATA 聚类算法对其进行分类。

第八章　模糊模式识别

随机产生多类二维样本，编写程序用模糊 C--均值聚类算法对其进行分类。

神经网络

第二章　感知器网络

用三层感知器网络解决异或分类问题。

第三章　反向传播网络（BPN)

设计一个三层 BP 网络，实现非线性函数的拟合。

第四章　自组织神经网络（SONN)

用自组织神经网络实现模式样本的自动聚类。

第五章　霍普菲尔德（Hopfield）神经网络

用 Hopfield 神经网络实现联想记忆。

模式识别

第零章　纹理特征提取

实验目的：编程实现纹理特征提取方法。

实验原理：一幅图像的纹理是在图像计算中经过量化的图像特征。图像纹理描述图像或其中小块区域的空间颜色分布和光强分布，反映物体的质地，如粗糙度、光滑性、颗粒度、随机性和规范性等。纹理特征的提取分为基于结构的方法和基于统计数据的方法。描述一块图象区域的纹理有三种主要的方法，统计分析方法、结构分析方法和频谱分析方法。统计方法有自相关函数、纹理边缘、结构元素、灰度的空间共生概率、灰度行程和自回归模型。结构方法研究基元及其空间关系。基元一般定义为具有某种属性而彼此相连的单元的集合，属性包括灰度、连通区域的形状、局部一致性等。空间关系包括基元的相邻性、在一定角度范围内的最近距离等等。根据基元间的空间联系，纹理可以分为弱纹理和强纹理。进一步细分，可以根据基元的空间共生频率来划分，也可以根据单位面积内的边缘数来区别。基元也可以定义为灰度行程。频谱方法是根据傅立叶频谱的峰值所占的能量比例将图象分类。包括计算峰值处的面积、峰值处的相位、峰值与原点的距离平方、两个峰值间的相位差等方法。

灰度共生矩阵是另一种纹理特征提取方法,描述的是某方向上间隔一定距离的一对图象点灰度出现的统计规律。

首先对于一幅图像定义一个方向（orientati- on）和一个以 pixel 为单位的步长(step)，灰度共生矩阵 T（N×N)，则定义 M(i,j)为灰度级为 i 和 j 的像素同时出现在一个点和沿所定义的方向跨度步长的点上的频率。其中 N 是灰度级划分数目。由于共生矩阵有方向和步长的组合定义，而决定频率的一个因素是对矩阵有贡献的像素数目，而这个数目要比总共数目少，且随着步长的增加而减少。因此所得到的共生矩阵是一个稀疏矩阵，所以灰度级划分 N 常常减少到 8 级。如在水平方向上计算左右方向上像素的共生矩阵，则为对称共生矩阵。类似的，如果仅考虑当前像素单方向（左或右）上的像素，则称为非对称共生矩阵。

从灰度共生矩阵上可以简单的看出，如果对角附近的元素有较大的值，说明图像的像素具有相似的像素值，如果偏离对角线的元素会有比较大的值，说明像素灰度在局部有较大变化。为了得到更多的纹理特征，我们还需要在进行计算：

对比度）（或反差）（contrast)：纹理沟纹越深，其对比度越大，视觉效果越清晰；反之，对比度小，则沟纹浅，效果模糊。灰度差即对比度大的象素对越多，这个值越大。灰度公生矩阵中远离对角线的元素值越大，con 越大。所以 con 越大图像越清晰。

$$Con = \sum_{i}\sum_{j}(i-j)^2 P(i,j)$$

相关度（inverse different moment)：度量空间灰度共生矩阵元素在行或列方向上的相似程度，因此，相关值大小反映了图像中局部灰度相关性。当矩阵元素值均匀相等时，相关值就大;相反，如果矩阵像元值相差很大则相关值小。

$$IDE = \sum_{i}\sum_{j}\frac{P(i,j)}{1+(i-j)^2}$$

能量：是灰度共生矩阵元素值的平方和，所以也称之为能量，反映了图像灰度分布均匀程度和纹理粗细度。ASM 值大表明一种较均一和规则变化的纹理模式。

$$Asm = \sum_i \sum_j P(i,j)^2$$

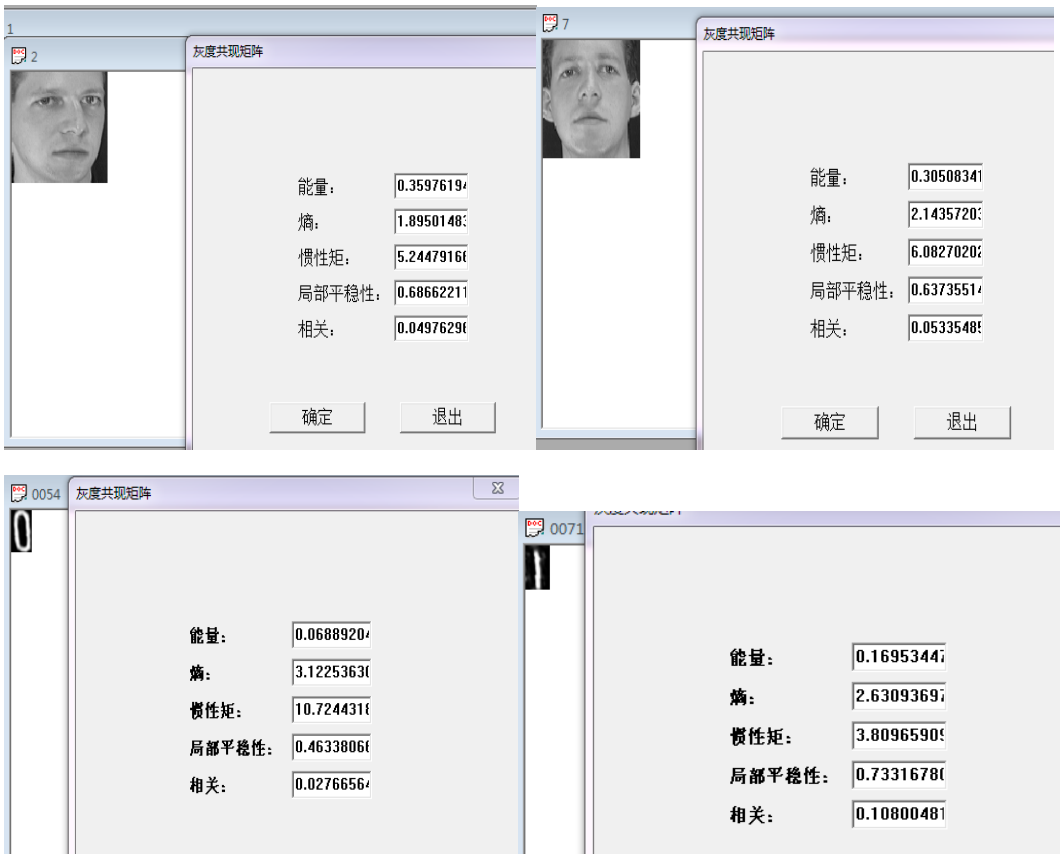
熵（entropy）：熵在物理中的含义就是物体的规则度，越有序熵越小，越无序熵越大。此处熵同样表示图像的信息量，当共生矩阵中所有元素有最大的随机性、空间共生矩阵中所有值几乎相等时，共生矩阵中元素分散分布时，熵较大。它表示了图像中纹理的非均匀程度或复杂程度。

$$Ent = \sum_i \sum_j P(i,j) \log P(i,j)$$

自相关（correlation）：反映了图像纹理的一致性。如果图像中有水平方向纹理，则水平方向矩阵的 COR 大于其余矩阵的 COR 值。

算法解释：灰度共现矩阵部分实现了能狼、熵、惯性矩、局部平稳性，以及相关五维纹理特征的计算。LBP 部分将检测窗口划分为 16×16 的小区域，对每个小区域进行计算，直方图统计，进行连接，得到整幅图的 LBP 特征。

结果分析：

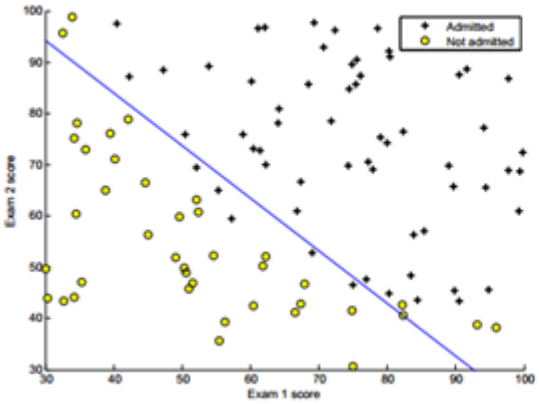


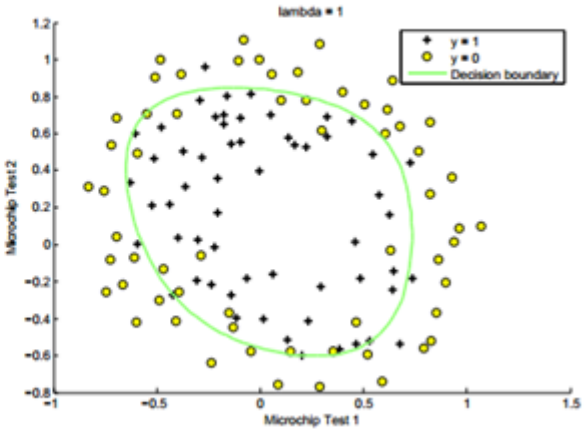
灰度共生矩阵各个变量的物理意义在原理中已经有所表述，熵表示图像的信息量和有序度。

第四章　线性判别函数

实验目的：随机产生两类二维样本，编写程序用 Fisher 线性判别函数和感知器准则函数对其进行分类。

实验原理：设计线性分类器首先要确定准则函数，然后再利用训练样本集确定该分类器的参数，以求使所确定的准则达到最佳。在使用线性分类器时，样本的分类由其判别函数值决定，而每个样本的判别函数值是其各分量的线性加权和再加上一阈值 w_0。如果我们只考虑各分量的线性加权和，则它是各样本向量与向量 W 的向量点积。如果向量 W 的幅度为单位长度，则线性加权和又可看作各样本向量在向量 W 上的投影。显然样本集中向量投影的分布情况与所选择的 W 向量有关。Fisher 准则的基本原理，就是要找到一个最合适的投影轴，使两类样本在该轴上投影的交迭部分最少，从而使分类效果为最佳。左图是线性分类器，右图为非线性分类器。





线性判别函数的一般形式为：$g(\mathbf{x}) = \mathbf{w}^T\mathbf{x} + w_0$ $\mathbf{x}$是样本向量，$\mathbf{w}$是权值向量。w_0是一个常数（阈值）。即

$$\begin{aligned}\mathbf{x} &= [x_1, \quad x_2, \ldots, \quad x_d]^T \\ \mathbf{w} &= [w_1, \quad w_2, \ldots, \quad w_d]^T\end{aligned}$$

对于两类问题而言，有

$$\begin{cases} g(\mathbf{x}^*) > 0, & x \in \omega_1 \\ g(\mathbf{x}^*) < 0, & x \in \omega_2 \end{cases}$$

其中ω_i表示第i类。若出现$g(\mathbf{x}^*) = 0$，则认为判别函数失效，拒绝分类。不难看出，由上面的 $g(\mathbf{x}^*) = 0$ 确定的决策面为超平面，而$\mathbf{W}$是该决策面的法向量。

对这类分类器的学习过程可以表述如下：1）获取训练样本集，即一组具有类别标志的样本集；2）确定一个准则函数$J(\mathbf{x}^*, \mathbf{w}^*)$:$J$的值反映分类器的性能,它的极值解则对应于最好的分类决策，例如J为分类误差；3）最优化方法求出准则函数的极值解$\widetilde{\mathbf{w}^*}$ ；4）对未知样 y，只要计算$g(\mathbf{y})$，然后根据决策规则判定$\mathbf{y}$所属类别。

将前面常用的线性决策面方程，$W^TX+w_0=0$ 改成$a^TY=0$,其中

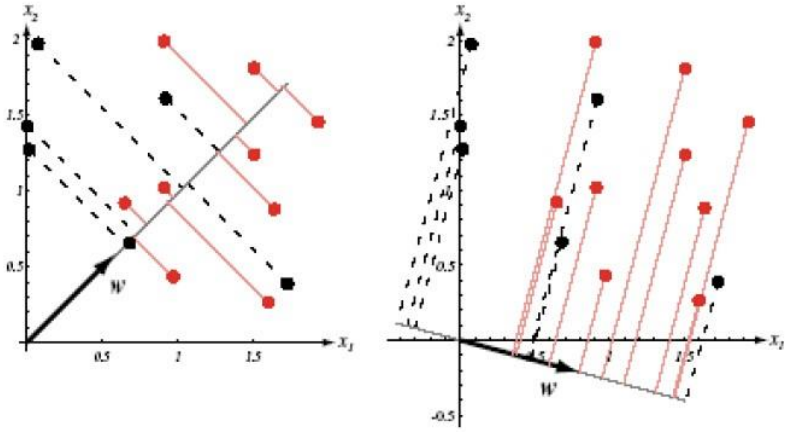
$$a = \begin{pmatrix} w_0 \\ w_1 \\ \vdots \\ w_d \end{pmatrix}, \quad Y = \begin{pmatrix} 1 \\ x_1 \\ \vdots \\ x_d \end{pmatrix}$$

PCA 方法寻找的是用来有效表示同一类样本共同特点的主轴方向，这对于表示同一类数据样本的共同特征是非常有效的。但 PCA 不适合用于区分不同的样本类。Fisher 线性判别分析（FDA）是用于寻找最有效地对不同样本类进行区分的方向。其主要思想是考虑将 d 维空间中的点投影到一条直线上。通过适当地选择直线的方向，有可能找到能够最大限度地区分各类样本数据点的投影方向。

设一组由 n 个 d 维样本组成的数据集合{x1,x2,…,xn}，它们分属于两个不同的类 ω1 和 ω2，其中 ω1 类的样本子集为 D_1，包含 n1 个样本点； ω2 类的样本子集为 D2，包含 n2 个样本点。将每一个样本点投影到方向为 w 的直线上，其中||w||=1，有：

$$y = w^t x$$

得到 n 个投影标量 y1,y2,…,yn，对应于 ω1 和 ω2，分别属于集合 Y1 和 Y2，下图给出同一组样本点在不同方向投影后可分性不同：



除了难以训练之外，感知器也不能够产生概率化的结果。但它最大的限制是基于基函数的线性组合（也是之前讨论几个模型的限制!）。当然，还有其他的一些缺点导致感知器无法应用于复杂分类问题，这些都需要去阅读相关论文。然而，感知器也被发展过，比如适应机(adaline,adaptive linear element),它的模型形式和感知器一样，只是采用了不同的训练方式。

这里考虑构造线性不等式 $\boldsymbol{a^t y_i > 0}$ 的准则函数的问题，令准则函数 J(.)为：

$$J_p(\boldsymbol{a}) = \sum_{\boldsymbol{y}\in Y}(-\boldsymbol{a}^t \boldsymbol{y}_i)$$

其中 Y 是被权向量 a 错分的样本集。当且仅当 JP(a*) = min JP(a) = 0 时，a*是解向量。这就是感知器（Perceptron）准则函数。

基本的感知器设计：感知器准则函数的最小化可以使用梯度下降迭代算法求解：

$$a(k+1) = a(k) + \eta(k) * \nabla J_p(a)$$

其中，k 为迭代次数， η 为调整的步长。即下一次迭代的权向量是把当前时刻的权向量向目标函数的负梯度方向调整一个修正量。

$$\nabla J_p(\boldsymbol{a}) = \sum_{\boldsymbol{y}\in Y}(\boldsymbol{y}_i)$$

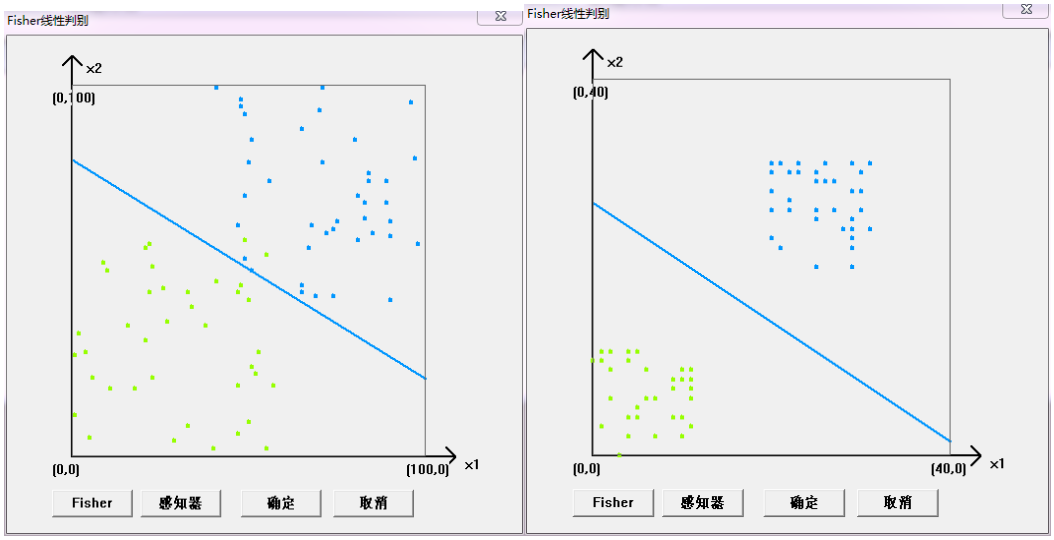
即在每一步迭代时把错分的样本按照某个系数叠加到权向量上。这样就得到了感知算法。

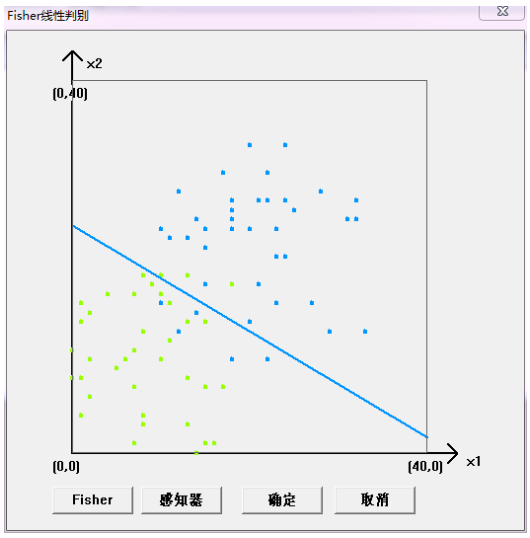
算法解释：Fisher 线性判别函数的基本原理：是将 d 维空间的样本投影到一直线上，形成一维空间，寻求最好的，最易于分类的投影线。1) 计算样本均值 m1, m2; 2) 样本的类内离散度矩阵

s1，s2 和总类内离散度矩阵 sw；3）样本的类间离散度 Sb；4）计算 fisher 准则函数取极大值时的解 w；5）计算阈值 w0.

感知器准则函数：1）初始权向量 α(0)，置 t=0；2）考查样本 yi，若 $\alpha(t)^T yi<=0$,则进行加量校正，否则继续。3）考查另一样本，重复 2），知道所有的样本都成为 pos 即可。

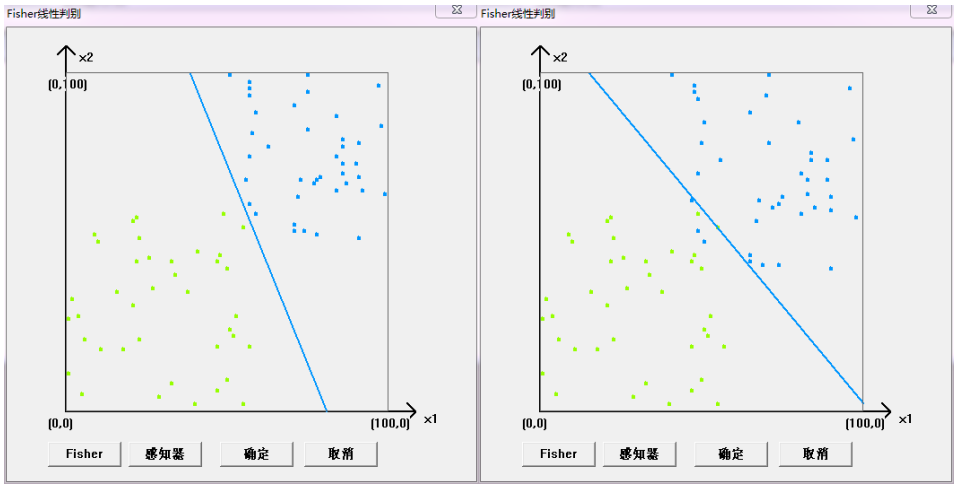
结果分析： Fisher 线性判别不适合投影后仍不是线性可分的情况，且计算量较大。投影之后，类间距离越大，分类效果越好。感知器函数难以训练，不能产生概率化的结果。

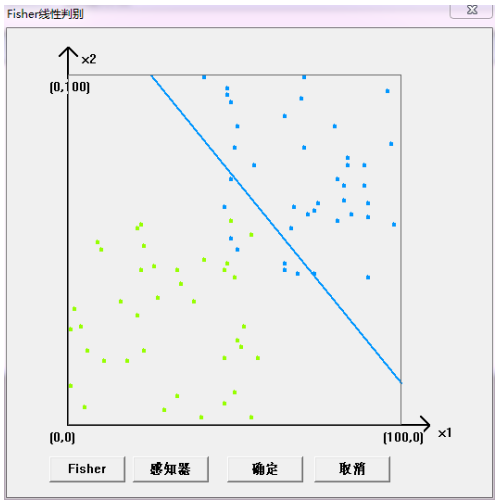




对于可以找到最佳投影方案的分类问题，Fisher 可以很好地将样本分开，而对于无法找到的投影之后线性可分的这种情况，Fisher 难以分开两类。

在感知器准则函数中，迭代的次数和步长会影响分类的结果和错误率。感知器中，只要训练样本集是线性可分的，对于任意的初值 a(1)，经过有限次迭代运算，算法必定收敛。而当样本线性不可分时，感知器算法无法收敛。下图分别为线性可分样本，线性不可分样本以及线性不可分样本在迭代次数分别为 10,10，100 的分类结果。



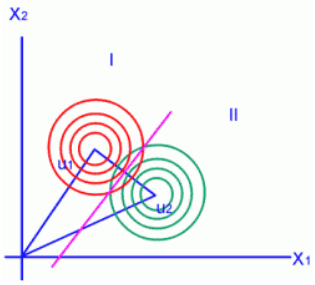


第五章　非线性判别函数

实验目的：随机产生多类二维样本，编写程序用最小距离分类器和 K 近邻分类器对其进行分类。

实验原理：最小距离分类器的基本原理非常简单：把测试样本 X 分类为与其最近的训练样本 Xi 所在的类 i。

最小距离分类器的优点在于原理非常简单，在维数比较低的情况下分类边界可以使非线性的，非常灵活。而缺点在于由于需要一一进行距离计算并比较，速度会很慢，并且当维数过高的时候，将会陷入维数诅咒中。



(a)

距离函数的定义：D(,)是距离函数，如果：1)D(X,Z)>=0 and D(X,X)=0 非负性；2)D(X,Z)=D(Z,X) 对称性； 3) D(X,Z)<=D(X,V)+D(V,Z) 类似三角形边长关系的特性；

A. 欧氏距离 (Euclidian distance)：

对称，球面，将所有维度公平对待，但是对某一点或几个点处巨大的差异相当敏感。也就是说如果两个特征大致是相似的，但是某几个点处差异巨大，被求和之后的距离会比其实际距离大。

B. 明氏距离 (Minkowski distance)： for p-norm:

和欧式距离不同，将二次方换成了 p 次方， p=1 时，就是 Manhattan distance，p=2 时， 就是欧氏距离。 p 趋向于 0 的时候，相当于逻辑“与”， p 趋向于无穷时，相当于逻辑“或”。

将与测试样本最近邻样本的类别作为决策的方法称为最近邻法。对一个 C 类别问题，每类有 N_i 个样本，i＝1，…，C，则第 i 类 ω_i 的判别函数

$$g_i(X) = \min_k \left\| X - X_i^k \right\|, \quad k=1, \ \ldots, \ N_i$$

，k＝1，…，Ni

其中 X_i^k 表示是 ω_i 类的第 k 个样本。决策规则为：如果

$$g_j(X) = \min_i g_i(X), \quad i=1, \ \ldots, \ C,$$

则决策 X∈ωj 。

由此可见最近邻法在原理上最直观，方法上也十分简单，只要对所有样本(共 N＝∑N_i)进行 N 次距离运算，然后以最小距离者的类别作决策。

最近邻法可以扩展成找测试样本的 k 个最近样本作决策依据的方法。其基本规则是，在所有 N 个样本中找到与测试样本的 k 个最近邻者，其中各类别所占个数表示成 k_i, i＝1，…，co 则决策规划是：如果

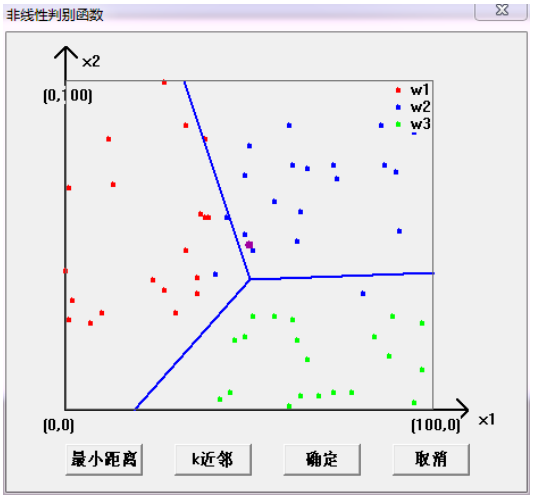
$$k_j(X) = \max_i k_i(X), \quad i=1, \ \ldots, \ c$$

则决策 X∈ω_j

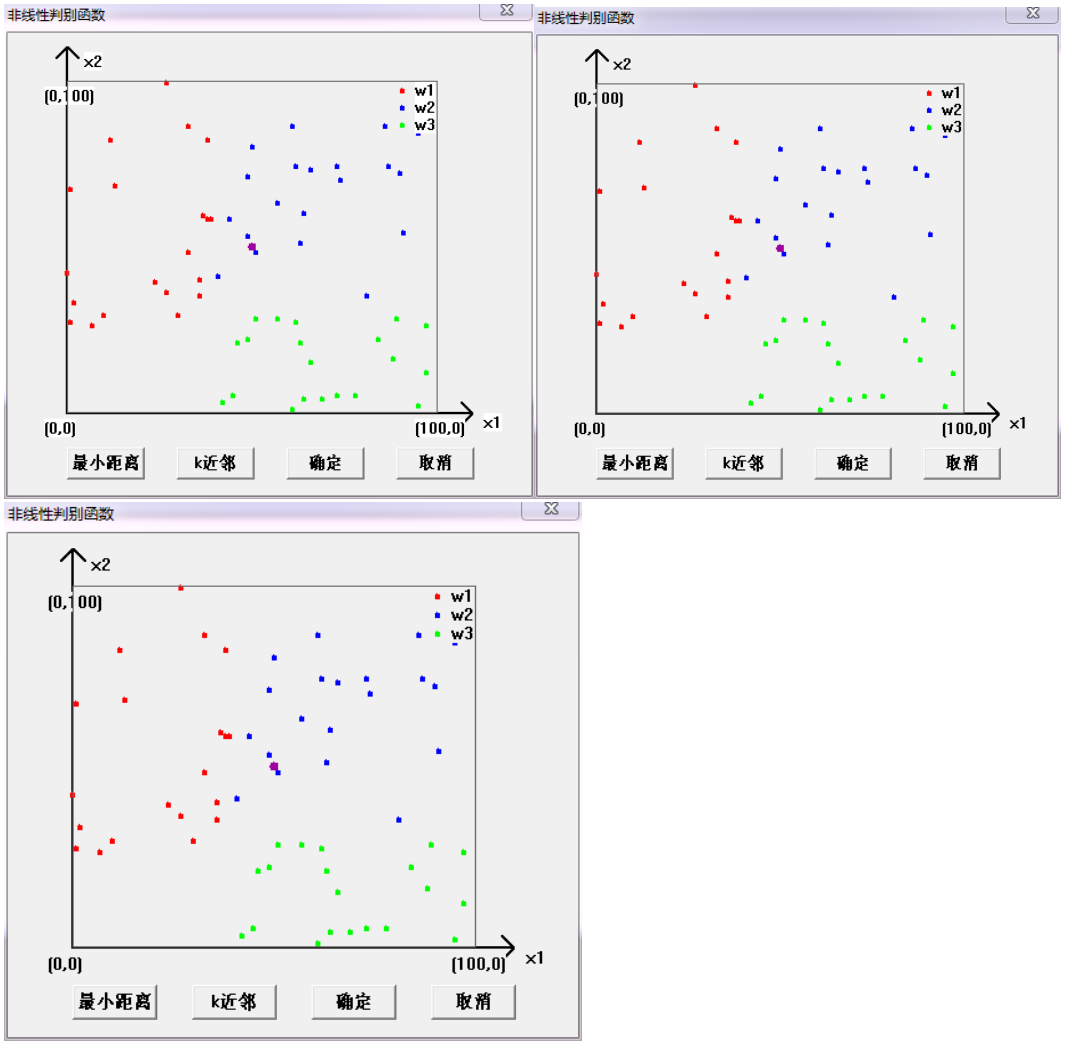
k 近邻一般采用 k 为奇数，跟投票表决一样，避免因两种票数相等而难以决策。

算法解释：k 近邻：1）计算已知类别数据集合汇总的点与当前点的距离； 2）按照距离递增次序排序；3）选取与当前点距离最近的 K 个点；4、确定距离最近的前 K 个点所在类别的出现频率；5）返回距离最近的前 K 个点中频率最高的类别作为当前点的预测分类。

结果分析：在最小距离分类器中，计算距离的方法（相似度矩阵的选取），以及样本的分布都很重要。



在 kNN 中，k 值的选取，计算距离的方法，以及数据的选取都十分的重要。



第七章　聚类分析

实验目的：随机产生多类二维样本，编写程序用 C--均值聚类和 ISODATA 聚类算法对其进行分类。

实验原理：C 均值算法是一种很常用的聚类算法，其基本思想是通过迭代寻找 c 个聚类的一种划分方案，使得用这 c 个聚类的均值来代表相应各类样本时所得到的总体误差最小。也称 k 均值。

具体原理如下：这个准则函数是以计算各类均值 m_i，与计算各类样本到其所属类均值点误差平方和为准则，若各类均值表示成

$$m_i = \frac{1}{N_i}\sum_{y\in\Gamma_i} y$$

其中第 i 类集合为 Γ_i，其样本数目为 N_i 是样本特征向量。

此时误差平方和准则可表示成

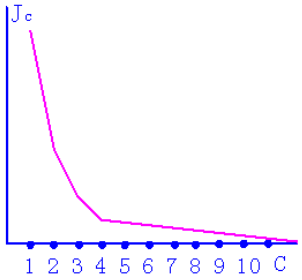
$$J_c = \sum_{i=1}^{c}\sum_{y\in\Gamma_i} \| y - m_i \|^2$$

其含义是各类样本与其所属样本均值间误差平方之总和。对于样本集的不同分类，导致不同的样本子集Γ_i及其均值m_i，从而得到不同的j_c值，而最佳的聚类是使j_c为最小的分类。这种类型的聚类通常称为最小方差划分。

动态聚类算法原理上就是通过迭代求函数极值的方法，因此在精神上与牛顿法等都是相通的。所不同的是要解决的问题是数据的聚类，也就是将现有的数据集进行划分。因此要构造一个函数，这个函数的值与数据划分有关，从而调整数据的划分使该函数达到极值。

c - 均值算法的准则函数式表示了相似度量是以数据到数据子集均值的模的平方来度量，这是用欧氏距离的度量方法。

随着迭代次数的增加，误差平方和变化如下图。



ISOData 算法步骤：

第一步：输入 N 个模式样本{xi,i=1,2,…,N}，预选 Nc 个初始聚类中心{z1,z2,…zNc}，它可以不等于所要求的聚类中心的数目，其初始位置可以从样本中任意选取。预选：K= 预期的聚类中心数目； θN = 每一聚类域中最少的样本数目，若少于此数即不作为一个独立的聚类； θS= 一个聚类域中样本距离分布的标准差； θc= 两个聚类中心间的最小距离，若小于此数，两个聚类需进行合并； L= 在一次迭代运算中可以合并的聚类中心的最多对数； I= 迭代运算的次数。

第二步：将 N 个模式样本分给最近的聚类 Sj，假若 Dj=min{∥x-zi∥,i=1,2,…Nc}，即||x-zj||的距离最小，则 x∈Sj。

第三步：如果 Sj 中的样本数目 $S_j<\theta_N$，则取消该样本子集，此时 N_c减去 1。（以上各步对应基本步骤（1））

第四步：修正各聚类中心 zj=1Nj∑x∈Sjx,j=1,2,…,Nc

第五步:计算各聚类域S_j中模式样本与各聚类中心间的平均距离 $\bar{D}j=1Nj\sum x\in Sj\text{//}\text{//}x-zj\text{//}\text{//}$,j=1,2,…,Nc

第六步：计算全部模式样本和其对应聚类中心的总平均距离 $\bar{D}=1N\sum j=1NNj\bar{D}j$

第七步：判别分裂、合并及迭代运算。1）若迭代运算次数已达到 I 次，即最后一次迭代，则置 θ_c =0，转至第十一步。2）若 Nc≤K2，即聚类中心的数目小于或等于规定值的一半，则转至第八步，对已有聚类进行分裂处理；3）若迭代运算的次数是偶数次，或 Nc≥2K，不进行分裂处理，转至第十一步；否则（即既不是偶数次迭代，又不满足 Nc≥2K），转至第八步，进行分裂处理。

第八步：计算每个聚类中样本距离的标准差向量 σj=(σ1j,σ2j,…,σnj)T，其中向量的各个分量为 σij=1Nj∑k=squrt（1Nj(xik-zij)exp2）式中，i = 1, 2, …, n 为样本特征向量的维数，j = 1, 2, …, N_c为聚类数，N_j为 S_j中的样本个数。

第九步：求每一标准差向量{σ_j, j = 1, 2, …, N_c}中的最大分量，以{σ_{jmax}, j = 1, 2, …, N_c}代表。

第十步：在任一最大分量集{σ_{jmax}, j = 1, 2, …, N_c}中，若有 $\sigma_{jmax}>\theta_S$ ，同时又满足如下两个条件之一：1）$\bar{D}j>\bar{D}$和 N_j > 2(θ_N + 1)，即 S_j中样本总数超过规定值一倍以上，2）Nc≤K2，则将 z_j 分裂为两个新的聚类中心和,且 N_c加 1。中对应于 σ_{jmax}的分量加上 kσ_{jmax},其中;中对应于 σ_{jmax}的分量减去 kσ_{jmax}。如果本步骤完成了分裂运算，则转至第二步，否则继续。

第十一步：计算全部聚类中心的距离。Dij=||zi-zj||，i=1,2,…,Nc-1，j=i+1,…,Nc。

第十二步：比较 D_{ij} 与 θ_c 的值，将 D_{ij} <θ_c 的值按最小距离次序递增排列，即

{Di1j1,Di2j2,…,DiLjL}，式中 Di1j1<Di2j2<…<DiLjL。

第十三步：将距离为 Dikjk 的两个聚类中心 Zik 和 Zjk 合并，得新的中心为：

z*k=1Nik+Njk[Nikzik+Njkzjk],k=1,2,…,L

式中，被合并的两个聚类中心向量分别以其聚类域内的样本数加权，使 Z*k 为真正的平均向量。

第十四步：如果是最后一次迭代运算（即第 I 次），则算法结束；否则，若需要操作者改变输入参数，转至第一步；若输入参数不变，转至第二步。在本步运算中，迭代运算的次数每次应加 1。

算法解释：C 均值聚类：1）选择某种方法把 N 个样本分成 C 个聚类的初始划分，计算每个聚类的均值 $m_1, m_2, \ldots, m_c$ 和 j_c；2）选择一个备选样本 y，设其在 Γ_i 中；3）若 $N_i = 1$，则转(2)，否则继续 4）计算

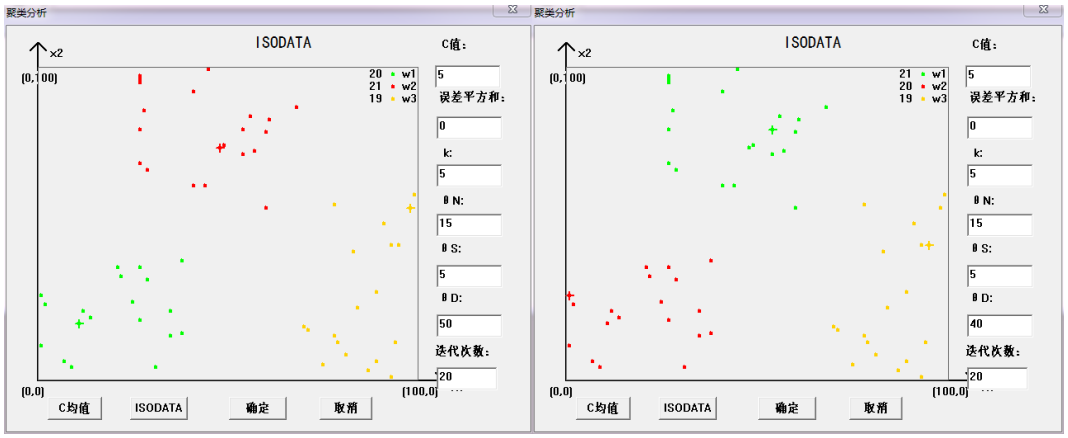
$$e_j = \begin{cases} \dfrac{N_j}{N_j + 1} \| y - m_j \|^2, j \neq i \\ \dfrac{N_j}{N_j - 1} \| y - m_j \|^2, j = i \end{cases}$$

5）对于所有的 j，若 $e_i \leq e_j$，则将 y 从 Γ_i 移到 Γ_j 中；6）重新计算 m_i 和 m_j 的值，并修改 j_c。

7）若连续迭代 N 次(即所有样本都运算过）不变，则停止，否则转到 2。

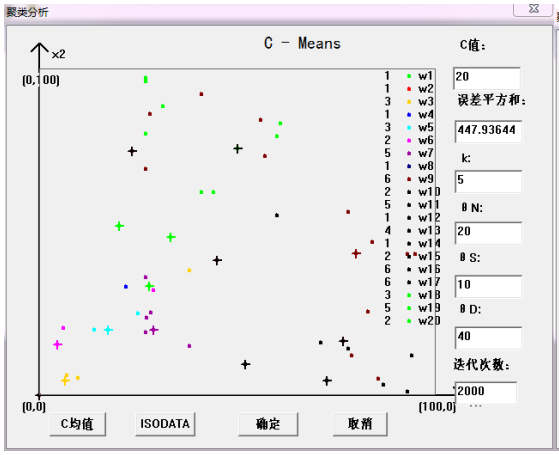
ISODATA 聚类：1）选择某些初始值。可选不同的参数指标，也可在迭代过程中人为修改，以将 N 个模式样本按指标分配到各个聚类中心中去；2）计算各类中诸样本的距离指标函数；（3）~（5）按给定的要求，将前一次获得的聚类集进行分裂和合并处理（4）为分裂处理，5）为合并处理），从而获得新的聚类中心；6）重新进行迭代运算，计算各项指标，判断聚类结果是否符合要求。经过多次迭代后，若结果收敛，则运算结束。

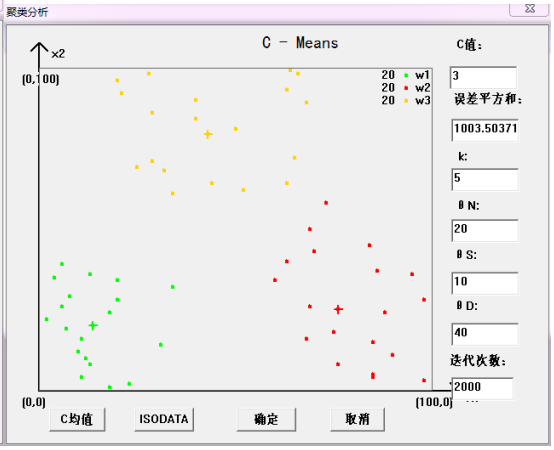
结果分析：ISOdata 中，可以用类心间的距离，类中的样本数目，类内样本的距离方差来评价聚类结果的好坏。参数较多，初始化的条件对结果影响很大。类心移动很大。

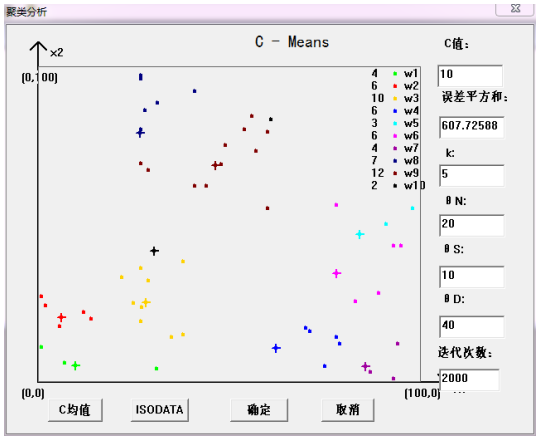


C 均值聚类算法中，从 K-means 里我们可以看出它其实就是 EM 的体现，E 步是确定隐含类别变量 c，M 步更新其他参数 μ 来使 J 最小化。这里的隐含类别变量指定方法比较特殊，属于硬指定，从 k 个类别中硬选出一个给样例，而不是对每个类别赋予不同的概率。总体思想还是一个迭代优化过程，有目标函数，也有参数变量，只是多了个隐含变量，确定其他参数估计隐含变量，再确定

隐含变量估计其他参数，直至目标函数最优。







第八章　模糊模式识别

实验目的：随机产生多类二维样本，编写程序用模糊 C--均值聚类算法对其进行分类。

实验原理：

算法解释：FCM 的价值函数（或目标函数）就是所有个点隶属度乘以该点与中心的欧氏距离之和。

在批处理方式运行时，FCM 用下列步骤确定聚类中心 ci 和隶属矩阵 U[1]：

步骤 1：用值在 0，1 间的随机数初始化隶属矩阵 U，使其满足式（6.9）中的约束条件

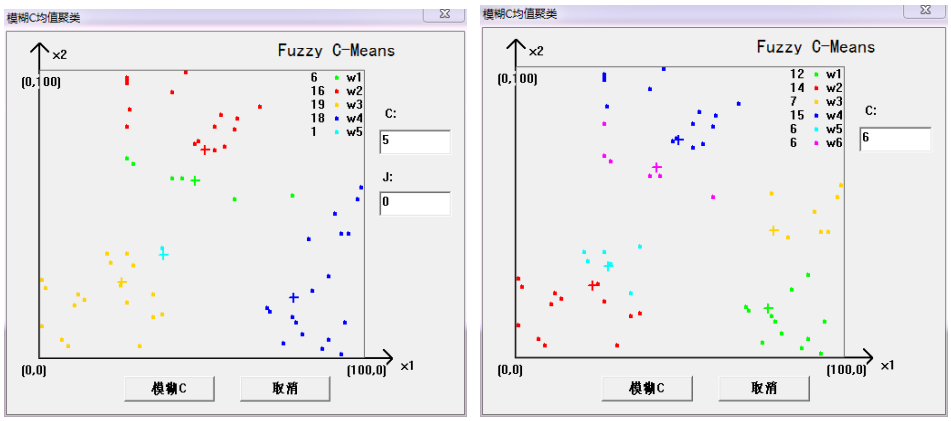
步骤 2：用式（6.12）计算 c 个聚类中心 ci，i=1,…,c。

步骤 3：根据式（6.10）计算价值函数。如果它小于某个确定的阀值，或它相对上次价值函数值的改变量小于某个阀值，则算法停止。

步骤 4：用（6.13）计算新的 U 矩阵。返回步骤 2。

上述算法也可以先初始化聚类中心，然后再执行迭代过程。由于不能确保 FCM 收敛于一个最优解。算法的性能依赖于初始聚类中心。因此，我们要么用另外的快速算法确定初始聚类中心，要么每次用不同的初始聚类中心启动该算法，多次运行 FCM。

结果分析：缺点在于迭代速度慢，没有考虑模式特征的重要性描述。

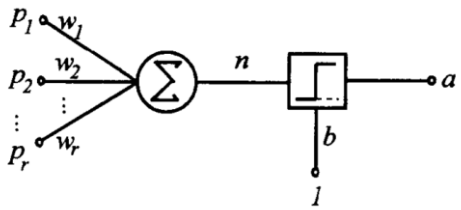


神经网络

第二章　感知器网络

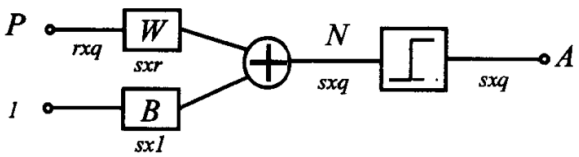
实验目的：用三层感知器网络，解决异或分类问题。

实验原理：感知器是由美国计算机科学家罗森布拉特（F.Roseblatt）于 1957 年提出的。感知器可谓是最早的人工神经网络。单层感知器是一个具有一层神经元、采用阈值激活函数的前向网络。通过对网络权值的训练，可以使感知器对一组输人矢量的响应达到元素为 0 或 1 的目标输出，从而实现对输人矢量分类的目的。下图给出了单层感知器神经元模型图。



其中，每一个输入分量严 pj(j＝1，2…，r)通过一个权值分量 wj，进行加权求和，并作为阈值函数的输人。偏差 b 的加入使得网络多了一个可调参数，为使网络输出达到期望的目标矢量提供了方便。感知器特别适合解决简单的模式分类问题。F.Roseblatt 已经证明，如果两类模式是线性可分的（指存在一个超平面将它们分开），则算法一定收敛。

感知器的网络是由单层的 s 个感知神经元，通过一组权值{ωij}(i＝1，2…s；j＝1，2…r)与 r 个输入相连组成。对于具有输入矢量 Pr×q 和目标矢量 Ts×q 的感知器网络的简化结构，如下图所示。



根据网络结构，可以写出第 i 个输出神经元(i＝1，2，…，s)的加权输入和 ni 及其输出 ai 为：

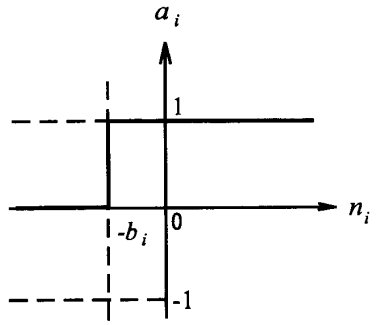
$$n_i = \sum_{j=1}^{r} w_{ij} p_j$$

$$a_i = f(n_i + b_i)$$

感知器的输出值是通过测试加权输入和值落在阈值函数的左右来进行分类的，即有：

$$a_i = \begin{cases} 1 & n_i + b_i \geqslant 0 \\ 0 & n_i + b_i < 0 \end{cases}$$

由下图可知：当输入 ni 十 bi 大于等于 0，即有 ni≥-bi 时，感知器的输出为 1，否则输出 ai 为 0。利用偏差 bi 的使用，使其函数可以左右移动，从而增加了一个自由调整变量和实现网络特性的可能性。



学习规则是用来计算新的权值矩阵 W 及新的偏差 B 的算法。感知器利用其学习规则来调整网络的权值，以便使该网络对输人矢量的响应达到数值为 0 或 1 的目标输出。

对于输入矢量 P，输出矢量 A，目标矢量为 T 的感知器网络，感知器的学习规则是根据以下输出矢量可能出现的几种情况来进行参数调整的。

1) 如果第 i 个神经元的输出是正确的，即有：ai＝ti，那么与第 i 个神经元联接的权值 wij 和偏差值 bi 保持不变；

2) 如果第 i 个神经元的输出是 0，但期望输出为 1，即有 ai＝0，而 ti＝1，此时权值修正算法为：新的权值 wij 为旧的权值 wij 加上输人矢量 pj；类似的，新的偏差 bi 为旧偏差 bi 加上它的输入 1；

3) 如果第 i 个神经元的输出为 1，但期望输出为 0，即有 ai＝1，而 ti＝0，此时权值修正算法为：新的权值 wij 等于旧的权值 wij 减去输入矢量 pj；类似的，新的偏差 bi 为旧偏差 bi 减去 1。

由上面分析可以看出感知器学习规则的实质为：权值的变化量等于正负输入矢量。具体算法总结如下。

对于所有的 i 和 j，i＝1，2，…，s；j＝1，2，…，r，感知器修正权值公式为：

$$\Delta w_{ij} = (t_i - y_i) \times p_j$$

$$\Delta b_i = (t_i - y_i) \times 1$$

用矢量矩阵来表示为

$$W = W + EP^T$$

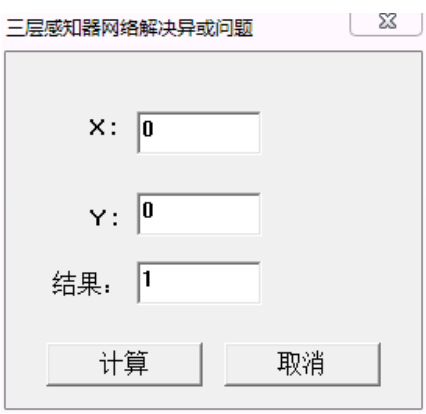
$$B = B + E$$

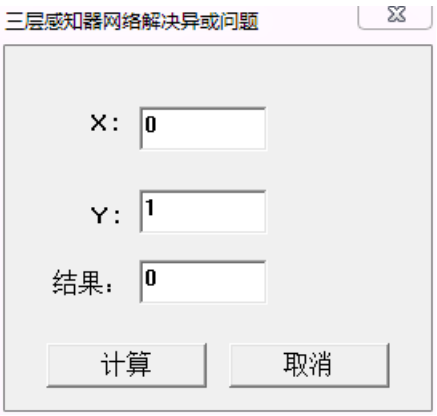
感知器的学习规则属于梯度下降法，该法则已被证明：如果解存在，则算法在有限次的循环迭代

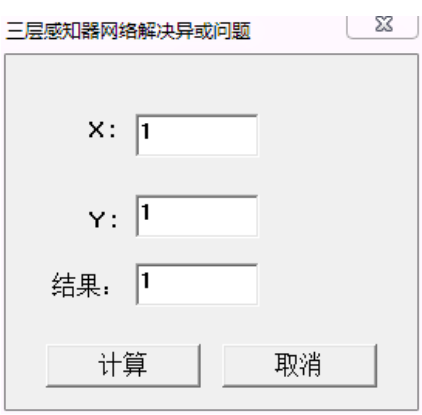
后可以收敛到正确的目标矢量。
要使前向神经网络模型实现某种功能，必须对它进行训练，让它逐步学会要做的事情，并把所学到的知识记忆在网络的权值中。人工神经网络权值的确定不是通过计算，而是通过网络的自身训练来完成的。这也是人工神经网络在解决问题的方式上与其他方法的最大不同点。借助于计算机的帮助，几百次甚至上千次的网络权值的训练与调整过程能够在很短的时间内完成。
感知器的训练过程如下：
在输入矢量 P 的作用下，计算网络的实际输出 A，并与相应的目标矢量 T 进行比较，检查 A 是否等于 T，然后用比较后的误差量，根据学习规则进行权值和偏差的调整；重新计算网络在新权值作用下的输入，重复权值调整过程，直到网络的输出 A 等于目标矢量 T 或训练次数达到事先设置的最大值时训练结束。
若网络训练成功，那么训练后的网络在网络权值的作用下，对于被训练的每一组输入矢量都能够产生一组对应的期望输出；若在设置的最大训练次数内，网络未能够完成在给定的输入矢量 P 的作用下，使 A＝T 的目标，则可以通过改用新的初始权值与偏差，并采用更长训练次数进行训练，或分析一下所要解决的问题是否属于那种由于感知器本身的限制而无法解决的一类。
感知器设计训练的步骤可总结如下：
1）对于所要解决的问题，确定输入矢量 P，目标矢量 T，并由此确定各矢量的维数以及确定网络结构大小的神经元数目：r，s 和 q；
2）参数初始化：
  a）赋给权矢量 w 在(—1，1)的随机非零初始值；
  b）给出最大训练循环次数 max_epoch；
3）网络表达式：根据输人矢量 P 以及最新权矢量 W，计算网络输出矢量 A；
4）检查：检查输出矢量 A 与目标矢量 T 是否相同，如果是，或已达最大循环次数训练结束，否则转入；
5）学习：根据感知器的学习规则调整权矢量，并返回 3)。

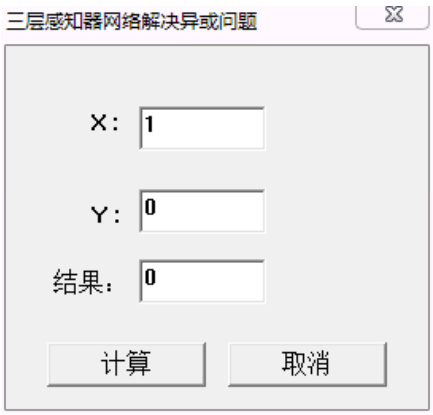
算法解释：1）确定一组异或训练数据，前两位表示输入，后一位表示输出，即(0,0,1),(1,1,1),(1,0,0)，(0,1,0)。初始化权向量 w。2）网络表达式：根据输人矢量 P 以及最新权矢量 W，计算网络输出矢量 A；3）检查：检查输出矢量 A 与目标矢量 T 是否相同，如果是，或已达最大循环次数训练结束，否则转入；4）学习：根据感知器的学习规则调整权矢量，并返回 2)。

结果分析：结果界面如下所示。输入 X,Y，计算出结果。在弹出对话框之前已经训练好网络，输入数值即可得到结果。









单层感知器网络无法解决异或问题，三层的成功解决。初始的权值以及梯度下降法求解最小值的算法会影响到最终训练结果的权值。

第三章　反向传播网络（BPN)

实验目的：设计一个三层 BP 网络，实现非线性函数的拟合。

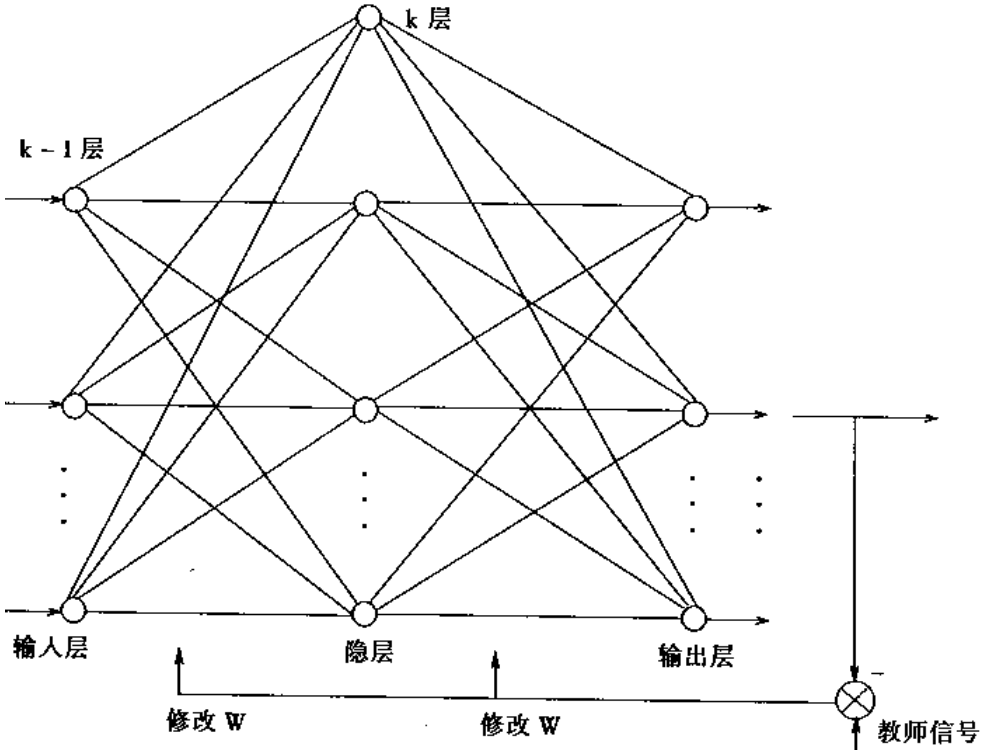
实验原理：反向传播算法也称 BP 算法。由于这种算法在本质上是一种神经网络学习的数学模型，所以，有时也称为 BP 模型。BP 算法是为了解决多层前向神经网络的权系数优化而提出来的；所以，BP 算法也通常暗示着神经网络的拓扑结构是一种无反馈的多层前向网络。故而．有时也称无反馈多层前向网络为 BP 模型。

在这里，并不要求过于严格去争论和区分算法和模型两者的有关异同。感知机学习算法是一种单层网络的学习算法。在多层网络中．它只能改变最后权系数。因此， 感知机学习算法不能用于多层神经网络的学习。1986 年，Rumelhart 提出了反向传播学习算法，即 BP(backpropagation)算法。这 种算法可以对网络中各层的权系数进行修正，故适用于多层网络的学习。BP 算法是目前最广泛用的神经网络学习算法之一，在自动控制中是最有用的学习算法。

BP 算法是用于前馈多层网络的学习算法，前馈多层网络的结构一般如下图所示。

它含有输人层、输出层以及处于输入输出层之间的中间层。中间层有单层或多层，由于它们和外界没有直接的联系，故也称为隐层。在隐层中的神经元也称隐单元。 隐层虽然和外界不连接．但是，它们的状态则影响输入输出之间的关系。这也是说，改变隐层的权系数，可以改变整个多层神经网络的性能。

设有一个 m 层的神经网络，并在输入层加有样本 X;设第 k 层的 i 神经元的输入总和表示为 Ui k ，输出 Xi k ；从第 k—1 层的第 j 个神经元到第 k 层的第 i 个神经元的权系数为 Wij 各个神经元的激发函数为 f，则各个变量的关系可用下面有关数学式表示：$X_i^k = f(U_i^k)$ $U_i^k = \sum_j W_{ij} X_j^{k-1}$



反向传播算法分二步进行，即正向传播和反向传播。这两个过程的工作简述如下。

1．正向传播

输入的样本从输入层经过隐单元一层一层进行处理，通过所有的隐层之后，则传向输出层；在逐层处理的过程中，每一层神经元的状态只对下一层神经元的状态产生影响。在输出层把现行输出和期望输出进行比较，如果现行输出不等于期望输出，则进入反向传播过程。

2．反向传播

反向传播时，把误差信号按原来正向传播的通路反向传回，并对每个隐层的各个神经元的权系数进行修改，以望误差信号趋向最小。

二、BP 算法的数学表达

BP 算法实质是求取误差函数的最小值问题。这种算法采用非线性规划中的最速下降方法，按误差函数的负梯度方向修改权系数。

为了说明 BP 算法，首先定义误差函数 e。取期望输出和实际输出之差的平方和为误差函数，则有：

$$e = \frac{1}{2}\sum_i (X_i^m - Y_i)^2$$

其中：Yi 是输出单元的期望值；它也在这里用作教师信号；

X_i^m 是实际输出；因为第 m 层是输出层。

由于 BP 算法按误差函数 e 的负梯度方向修改权系数，故权系数 W_{ij} 的修改量 Aw_{ij}，和 e

$\triangle W_{ij} \propto -\frac{\partial e}{\partial W_{ij}}$ 也可写成 $\triangle W_{ij} \propto -\eta \frac{\partial e}{\partial W_{ij}}$，其中：η 为学习速率，即步长。

很明显，根据 BP 算法原则，求 ae/aW_{ij} 最关键的。下面求 ae/aW_{ij} ；有

$\frac{\partial e}{\partial W_{ij}} = \frac{\partial e_k}{\partial U_i^k} \cdot \frac{\partial U_i^k}{\partial W_{ij}}$ 。由于 $\frac{\partial U_i^k}{\partial W_{ij}} = \frac{\partial(\sum_l W_{il} X_l^{k-1})}{\partial W_{ij}} = X_j^{k-1}\Big|_{l=j}$ 故而 $\frac{\partial e}{\partial W_{ij}} = \frac{\partial e}{\partial U_i^k} \cdot X_j^{k-1}$ 。

$$\triangle W_{ij} = -\eta \frac{\partial e}{\partial W_{ij}}$$
$$= -\eta \frac{\partial e}{\partial U_i^k} \cdot X_j^{k-1}$$

令 $d_i^k = \frac{\partial e}{\partial U_i^k}$ 则有学习公式 $\triangle W_{ij} = -\eta \cdot d_i^k \cdot X_j^{k-1}$ 其中：η 为学习速率，即步长，一般取 0-1 间的数。从上面可知，d_i^k 实际仍未给出明显的算法公式，下面求 d_i^k 的计算公式。

$$d_j^k = \frac{\partial e}{\partial U_i^k}$$
$$= \frac{\partial e}{\partial X_i^k} \cdot \frac{\partial X_i^k}{\partial U_i^k}$$
$$\frac{\partial X_i^k}{\partial U_i^k} = f'(U_i^k)$$

为了方便进行求导，取 f 为连续函数。一般取非线性连续函数，例如 Sigmoid 函数。当取 f 为非对称 Sigmoid 函数时，有：

$$f(U_i^k) = \frac{1}{1 + \exp(-U_i^k)}$$

$$f'(U_i^k) = f'(U_i^k)(1 - f(U_i^k)) = X_i^k (1 - X_i^k)$$

再考虑偏微分项 ae / aX_i^k，有两种情况需考虑的：如果 k＝m，则是输出层，这时有 Y_i 是输出期望值，它是常数。有

$$\frac{\partial e}{\partial X_i^k} = \frac{\partial e}{\partial X_i^m}$$
$$= (X_i^m - Y_i)$$

从而有 $d_i^m = X_i^m (1 - X_i^m)(X_i^m - Y_i)$

如果 k<m，则该层是隐层．这时应考虑上一层对它的作用，故有：

$$\frac{\partial e}{\partial X_i^k} = \sum_l \frac{\partial e}{\partial U_l^{k+1}} \cdot \frac{\partial U_l^{k+1}}{\partial X_i^k}$$
$$\frac{\partial e}{\partial U_j^{k+1}} = d_l^{k+1}$$
$$\frac{\partial U_l^{k+1}}{\partial X_i^k} = \frac{\partial (\sum W_{li} X_i^k)}{\partial X_i^k} = W_{li}\Big|_{j=i}$$
$$\frac{\partial e}{\partial X_i^k} = \sum_l W_{li} \cdot d_l^{k+1}$$
$$d_i^k = X_i^k (1 - X_i^k) \cdot \sum_l W_{li} \cdot d_l^{k+1}$$

从上述过程可知：多层网络的训练方法是把一个样本加到输入层，并根据向前传播的规则：$X_i^k = f(U_i^k)$ 不断一层一层向输出层传递，最终在输出层可以得到输出 X_i^m。把 Xi^m 和期望输出 Yi 进行比较．如果两者不等，则产生误差信号 e，接着则按下面公式反向传播修改权系数：

$\triangle W_{ij} = -\eta \cdot d_i^k \cdot X_j^{k-1}$ 其中 $d_i^k = X_i^k (1 - X_i^k) \sum_l W_{li} d_l^{k+1}$，$d_i^m = X_i^m (1 - X_i^m)(X_i^m - Y_i)$

上面公式中，求取本层 d_i^k 时，要用到高一层的 d_i^{k+1}；可见，误差函数的求取是从输出层开始，到输入层的反向传播过程。在这个过程中不断进行递归求误差。

通过多个样本的反复训练，同时向误差渐渐减小的方向对权系数进行修正，以达最终消除误差。从上面公式也可以知道，如果网络的层数较多时，所用的计算量就相当可观，故而收敛速度不快。

为了加快收敛速度，一般考虑上一次的权系数，并以它作为本次修正的依据之一，故而有修正公式：

$$\triangle W_{ij}(t+1) = -\eta d_i^k \cdot X_j^{k-1} + \alpha \triangle W_{ij}(t)$$

其中：η 为学习速率，即步长，η ＝0.1—0.4 左右 α 为权系数修正常数，取 0．7—0．9 左右。

上式也称为一般化的 Delta 法则。对于没有隐层的神经网络，可取

$$\triangle W_{ij} = \eta (Y_j - X_j) \cdot X_i$$

其中:，Y_i 为期望输出；X_j 为输出层的实际输出；X_i 为输入层的输入。

这显然是一种十分简单的情况，上式也称为简单 Delta 法则。在实际应用中，只有一般化的 Delta 法则才有意义。简单 Delta 法则式只在理论推导上有用。

算法解释： 算法的执行的步骤如下：1) 对权系数 Wij 置初值。对各层的权系数 Wij 置一个较小的非零随机数，但其中 Wi ,n+1=-θ 。2) 输入一个样本 X＝(x1 ，x2 ，…，xn ，1)，以及对应期望输出 Y＝(Y1 ，Y2 ，…，Yn)。3．计算各层的输出。对于第 k 层第 i 个神经元的输出 Xi^k，有：$U_i^k = \sum_{j=1}^{n+1} W_{ij} X_j^{k-1}$，　$X_{n+1}^{k-1} = 1, W_{i,n+1} = -\theta$　$X_i^k = f(U_i^k)$ 。4) 求各层的学习误差 d_i^k 。对于输出层有 k＝m，有 $d_i^m = X_i^m (1-X_i^m)(X_i^m - Y_i)$，对于其他各层，有 $d_i^k = X_i^k (1 - X_i^k) \sum_l W_{li} d_l^{k+1}$

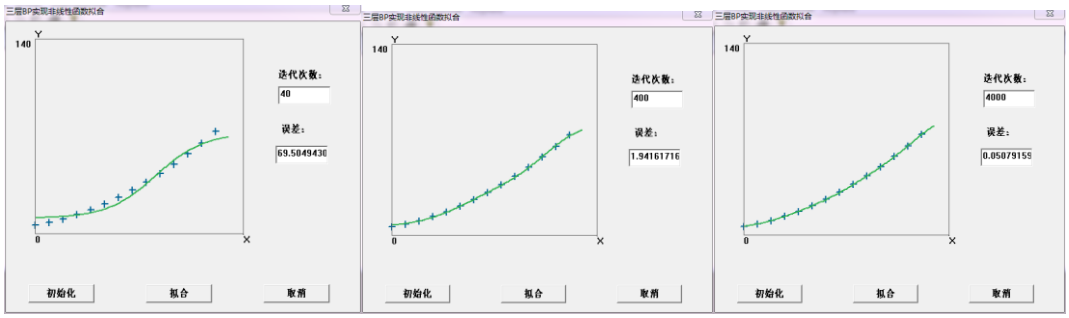
5) 修正权系数 Wij 和阀值 θ. $W_{ij}(t+1) = W_{ij}(t) - \eta \cdot d_i^k \cdot X_j^{k-1}$

$W_{ij}(t+1) = W_{ij}(t) - \eta \cdot d_i^k \cdot X_i^{k-1} + \alpha \triangle W_{ij}(t)$ 其中：

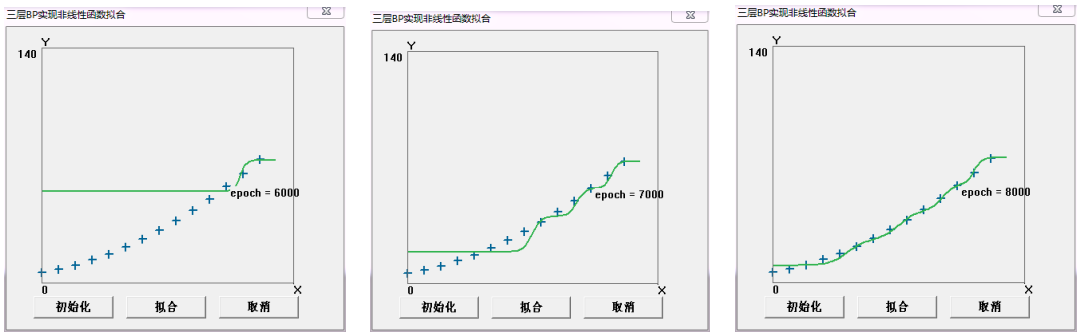
$$\begin{aligned} \triangle W_{ij}(t) &= -\eta \cdot d_i^k \cdot X_j^{k-1} + \alpha \triangle W_{ij}(t-1) \\ &= W_{ij}(t) - W_{ij}(t-1) \end{aligned}$$

6) 当求出了各层各个权系数之后，可按给定品质指标判别是否满足要求。如果满足要求，则算法结束；如果未满足要求，则返回 3)执行。这个学习过程，对于任一给定的样本 Xp ＝(Xp1 ，Xp2 ，…Xpn ，1)和期望输出 Yp＝(Yp1 ，Yp2 ，…，Ypn)都要执行，直到满足所有输入输出要求为止。

结果分析：



上面一排是学习效率为 0.01，迭代次数分别为 40,400，和 4000 的结果，误差分别为 69.6,1.01,0.05。下面一排是学习效率为 0.15 的结果。可见，对指数函数，低学习效率会取得更好的结果，不易过拟合；而对于余弦函数，高学习效率会有比较好的结果，不易欠拟合。

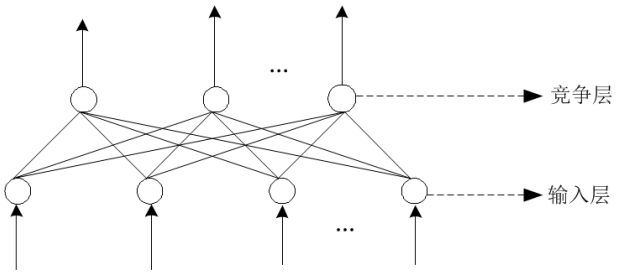


此外，随着迭代次数的增加，曲线拟合的误差越小，但是牺牲了时间，用此方法要综合考虑时间和误差成本，选择合适的限制条件。

第四章　自组织神经网络（SONN)

实验目的：用自组织神经网络实现模式样本的自动聚类。

实验原理：自组织神经网络是一种无导师学习网络。它通过自动寻找样本中的内在规律和本质属性，自组织、自适应地改变网络参数与结构。



SOM 神经网络采用的算法称为 Kohonen 算法，它的基本思想是：网络输出层的各神经元通过竞争来获得对输入层的响应机会，最后只有一个神经元获胜。获胜的神经元对它临近的神经元的影响由近及远，由兴奋逐渐转为抑制，那些与获胜神经元有关的各连接权朝着有利于它竞争的方向转变。

以获胜神经元为中心设定一个邻域半径，该半径圈定的范围称为优胜邻域。在 SOM 网学习算法中，优胜邻域内的所有神经元均按其离开获胜神经元的距离远近不同程度地调整权值。优胜邻域开始定得很大，但其大小随着训练次数的增加不断收缩，最终收缩到半径为零。

相似性测量：1）欧式距离法。两个模式向量的欧式距离越小，两个向量越接近，因此认为这两个模式越相似，当两个模式完全相同时其欧式距离为零。如果对同一类内各个模式向量间的欧式距离作出规定，不允许超过某一最大值 T，则最大欧式距离 T 就成为一种聚类判据，同类模式向量的距离小于 T，两类模式向量的距离大于 T。2）余弦法。两个模式向量越接近，其夹角越小，余弦越大。当两个模式向量完全相同时，其余弦夹角为 1。如果对同一类内各个模式向量间的夹角作出规定，不允许超过某一最大夹角 a，则最大夹角就成为一种聚类判据。同类模式向量的夹角小于 a，两类模式向量的夹角大于 a。余弦法适合模式向量长度相同和模式特征只与向量方向相关的相似性测量。

算法解释：1）初始化。对输出层各权向量赋予较小的随机数并进行归一化处理，得到 $\hat{\mathbf{w}}_j$ (j=1,2,…,m)。建立初始优胜邻域 $N_j^*(0)$；学习率 η 赋初始值。2）接受输入。从训练集中随机选

取一个输入模式并进行归一化处理，得到 $\hat{\mathbf{X}}^p$，p∈{1,2,…,P}。3）寻找获胜节点计算 $\hat{\mathbf{X}}^p$ 与 $\hat{\mathbf{W}}_j$ 的点积，j=1,2,…m，从中选出点积最大的获胜节点。4）定义优胜邻域 $N_{j*}(t)$ 以 j*为中心确定 t 时刻的权值调整域，一般初始邻域 $N_{j*}(0)$ 较大，训练过程中 $N_{j*}(t)$ 随训练时间逐渐收缩。5）调整权值。对优胜邻域 $N_{j*}(t)$ 内的所有节点调整权值：$w_{ij}(t+1)=w_{ij}(t)+\eta(t,N)[x_i{}^p-w_{ij}(t)]$ i=1,2,…n，$j\in N_{j*}(t)$. 式中，$\eta(t,N)$ 是训练时间 t 和邻域内第 j 个神经元与获胜神经元 j*之间的拓扑距离 N 的函数，该函数一般有以下规律：$t\uparrow\rightarrow\eta\downarrow, N\uparrow\rightarrow\eta\downarrow$。6）结束检查。学习率是否衰减到零或某个预定的正小数。

结果分析：未调通。

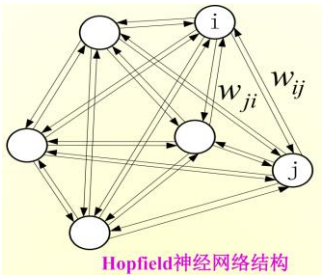
第五章　霍普菲尔德（Hopfield）神经网络

实验目的：用 Hopfield 神经网络实现联想记忆。

实验原理：1982 年，生物物理学家 J.Hopfield 提出了一种新颖的人工神经网络模型——Hopfield 网络模型，引入了能量函数的概念，是一个非线性动力学系统。

(1) 离散的 Hopfield 网络用于联想记忆；

(2) 连续的 Hopfield 网络用于求解最优化问题。



Hopfield神经网络结构

在 Hopfield 网络中，由于反馈的存在，其加权 输 入和 u_i，i=1~n 为网络状态，网络的输出为 y_1~y_n， 则 u,y 的变化过程为一个非线性动力学系统。可用非线性差（微）分方程来描述。一般有如下的几种状态演变形式：1）渐进稳定；2）极限环；3）混沌现象；4）状态轨迹发散。

Hopfield 网络的稳定性可用能量函数进行分析。目前，人工神经网络常利用渐进稳定点来解决某些问题。例如，如果把系统的稳定点视为一个记忆的话，那么从初态朝这个稳定点的演变过程就是寻找记忆的过程。初态可以认为是给定的有关记忆的部分信息。如果把系统的稳定点视为一个能量函数的极小点，把能量函数视为一个优化问题的目标函数，那么从初态朝这个稳定点的演变过程就是一个求该优化问题的过程。这样的优点在于它的解并不需要真的去计算，而只要构成这种反馈网络，适当的设计其连接值和输入就可达到目的。

下面介绍一下离散型 Hopfield 神经网络。DHNN 主要有以下两种工作方式：1）串行工作方式。在某一时刻只有一个神经元按照上式改变状态，而其它神经元的输出不变。这一变化的神经元可以按照随机的方式或预定的顺序来选择。2）并行工作方式。在某一时刻有 N 个神经元按照上式改变状态，而其它的神经元的输出不变。变化的这一组神经元可以按照随机方式或某种规则来选择。当 N=n 时，称为全并行方式。

能量函数：

$$E=-\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}w_{ij}x_i x_j+\sum_{i=1}^{n}\theta_i x_i$$

能量函数 E 按照迭代一定会下降。证明如下。

$$\text{证明 } \Delta \mathrm{E}=\mathrm{E}(\mathrm{t}+1)-E(t)$$

$$\Delta x_k=x_k(t+1)-x_k(t)\quad x_k=\pm 1$$

$$\therefore\quad \Delta x_k=\begin{cases}0 & x_k(t)=\operatorname{sgn}(u_k(t))\\ -2 & x_k(t)=1,\operatorname{sgn}(u_k(t))=-1\\ 2 & x_k(t)=-1,\operatorname{sgn}(u_k(t))=1\end{cases}$$

$$\Delta E=-\frac{1}{2}\left[\Delta x_k\sum_{i=1}^{n}w_{ik}x_i(t)+\Delta x_k\sum_{j=1}^{n}w_{kj}x_j(t)+w_{kk}(\Delta x_k)^2\right]+\Delta x_k\theta_k$$

无论 x 从-1 变到 1 还是从 1 变到-1，能量都是下降的。结点输出为-1 或+1；用于联想记忆（自联想，互联想）；1）先训练出权值 w2）根据输入进行迭代，回忆出联想的结果。

因为根据定理条件有 $w_{ij}=w_{ji}$，故有

$$\Delta E=-\Delta x_k\left[\sum_{j=1}^{n}w_{kj}x_j(t)-\theta_k\right]-\frac{1}{2}w_{kk}(\Delta k)^2=-\Delta x_k u_k(t+1)-\frac{1}{2}w_{kk}(\Delta k)^2$$

根据*DHNN*的运行规则，$\Delta x_k u_k(t+1)\geq 0$。又因为$w_{kk}\geq 0$，故对任意的神经元$k$有$\Delta E\leq 0$。

另外能量函数是有界的，所以它总能收敛到它的一个局部极小点。

DHNN 网络设计需要考虑两个重要的问题，权值 W 和 θ 的确定；网络给定之后记忆容量的分析。权值的设计有外积法、伪逆法、正交设计法等。记忆容量的分析：当网络只记忆一个稳定的模式时，该模式肯定被网络准确无误的记住。但当所要记忆的模式增加是，情况会发生变化，主要表现在权值移动和交叉干扰两点上。

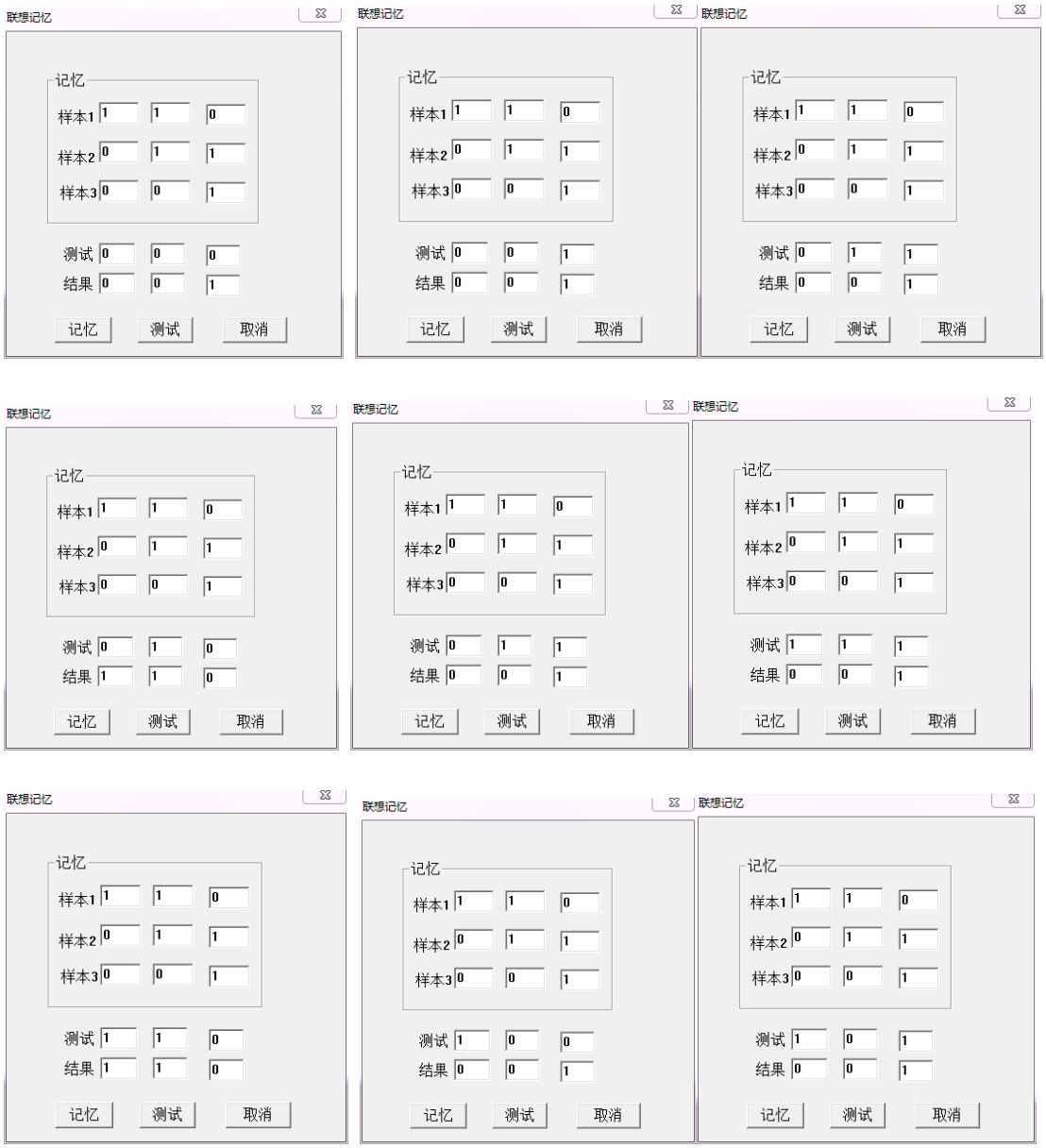
在网络的学习过程中，网络对权值的记忆实际上是逐个实现的。当网络准确的 X^1 时，为了记忆 X^2，需要在记忆样本 X^1 的权值上加上对样本 X^2 的记忆项 $X^2X^{2T}-I$，将权值在原来值的基础上产生了移动。这样网络有可能部分得遗忘了以前以记忆住的模式。从动力学的角度来看，k 值较小时，网络 Hebb 学习规则，可以使输入学习样本成为其吸引子。随着 k 值的增加，不但难以使后来的样本成为网络的吸引子，而且有可能使已记忆住的吸引子的吸引域变小，使原来处于吸引子位置上的样本从吸引子的位置移动。对一记忆的样本发生遗忘，这种现象称为“疲劳”。

网络在学习多个样本后，在回忆阶段即验证该记忆样本时，所产生的干扰，称为交叉干扰。对外积型设计而言，如果输入样本是彼此正交的，n 个神经元的网络其记忆容量的上界为 n。但是在大多数情况下，学习样本不可能是正交的，因而网络的记忆容量要比 n 小得多，一般为(0.13~0.15)n，n 为神经元数。

算法解释：1）初始化权值矩阵为 0.2）输入下一个模式 P，P 为 N 维向量(p1,p2,p3…pn)。3）将 p 乘以 p 的转置得到一个 N*N 的方阵 A。4）将第 3 步的方阵 A 对角线元素设置为 0。5）将方阵 A 回到权值矩阵 W 上。6）如果有更多的模式，则转到第 2 步。7）训练结束。

训练结束后，可以用 Hopfield 网络识别前面的训练模式，或者与它们相近的模式，这体现了 Hopfield 的联想功能。Hopfield 能够自动识别与其训练模式相反的模式。

结果分析：



上面九幅图是用 hopfield 网络训练记忆 110,011,001 三个稳态的九种测试结果，可见 011 并非其稳定状态，记忆样本的选取要合适。