神经网络
Neural Networks
第五章

递归网络

史忠植

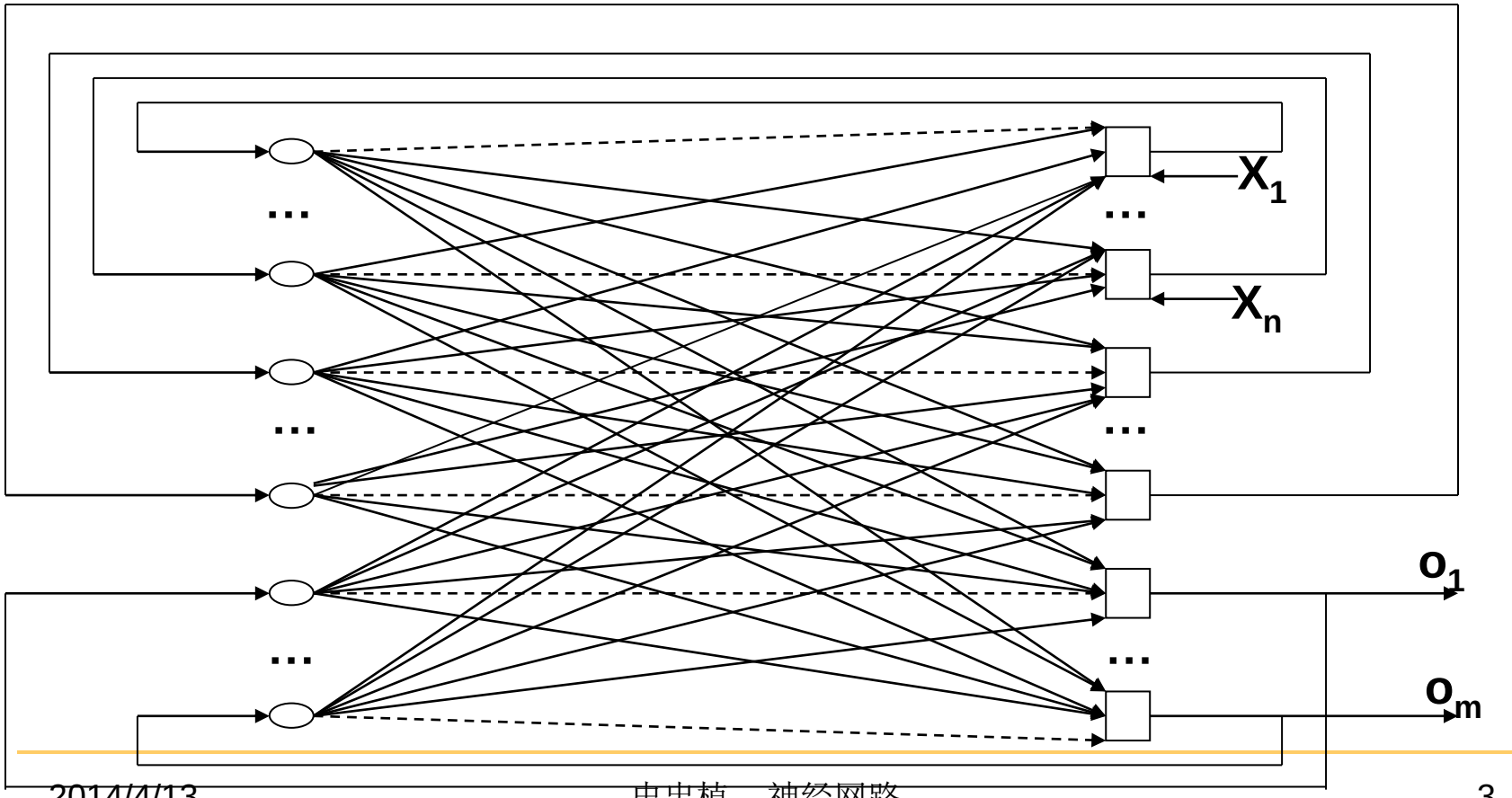
中国科学院计算技术研究所
http://www.intsci.ac.cn/

内容提要

- 5.1 概述
- 5.2 递归网络体系结构
- 5.3 状态空间模型
- 5.4 Hopfield 网络
- 5.5 双向联想记忆模型
- 5.6 模拟退火算法
- 5.7 Boltzmann机

递归网路的组织

- 网络结构



递归网路的组织

- **联接：**神经元之间都是互联的w_{ij}，每个神经元都没有到自身的联接$w_{ii}=0$。
- 神经元个数**h**，输入向量维数**n**，输出向量维数**m**。**h**≥n，h≥m，n≥1，m≥1。
- **神经元：**输入、输出、隐藏
- **状态变化：**非同步、同步
- **输入向量：** $X=(x_1, x_2, \ldots, x_n)$
- **输出向量：** $O=(o_1, o_2, \ldots, o_m)$

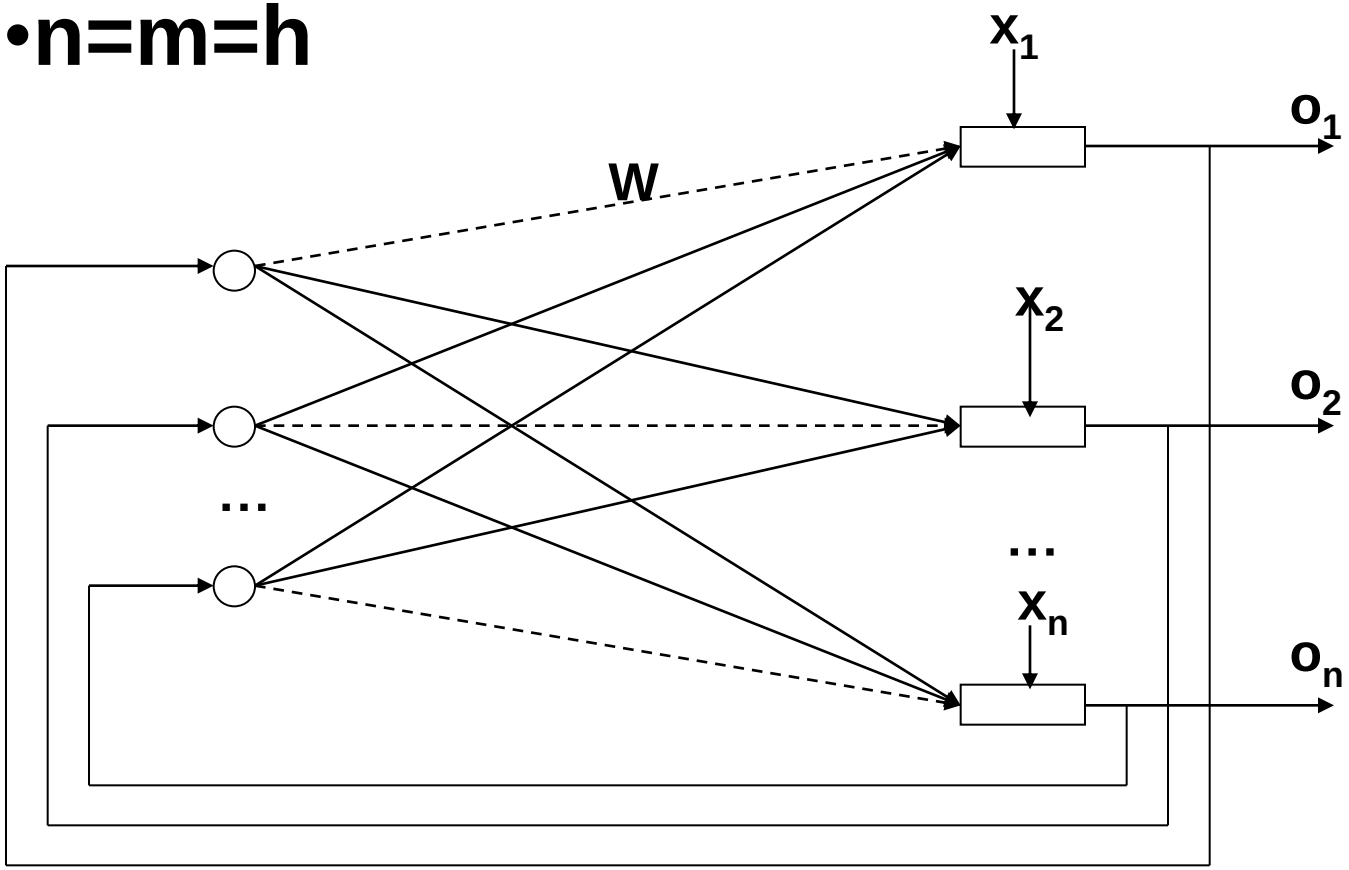
递归网路的组织

神经元的网络输入： $net_j = \sum_{i=1\&i\neq j}^{n} w_{ij} o_i + x_j$

阈值函数： $o_j = \begin{cases} 1 & \text{if } net_j > \theta_j \\ 0 & \text{if } net_j < \theta_j \\ o_j & \text{if } net_j = \theta_j \end{cases}$

最基本的Hopfield网

- n=m=h



最基本的Hopfield网

- 希望网络的联接矩阵存放的是一组这样的样本，在联想过程中实现对信息的“修复”和“加强”，要求：它的输入向量和输出向量是相同的向量，即，**X=Y**
- **样本集**：$S=\{X_1, X_2, \ldots, X_s\}$

最基本的Hopfield网

- **权矩阵：** $w_{ij}=\sum_{k=1}^{s} x_{ik}x_{jk} \qquad i \neq j$

$$w_{ii}=0 \qquad 1 \leqslant i \leqslant n$$

- **W是一个对角线元素为0的对称矩阵：**
- $\mathbf{W= X_1^T \times X_1+X_2^T \times X_2+\ldots+X_s^T \times X_s - W_0}$
- **W是各个样本向量自身的外积的和——网络实现的是自相联映射。**

最基本的Hopfield网

- **激活函数：**

  改为S形函数后，系统就成为一个**连续系统**

- **多级递归网路**

  除输出向量被反馈到输入层外，其它各层之间的信号传送均执行如下规定：第**i-1**层神经元的输出经过第**i**个连接矩阵被送入第**i**层。

  一般不考虑越层的信号传送、中间的信号反馈和同层的神经元之间进行信号的直接传送

稳定性分析

- 网络的稳定性是与收敛性不同的问题
- **Cohen和Grossberg[1983年]:Hopfield网络的稳定性定理**

  如果Hopfield网络的联接权矩阵是对角线为0的对称矩阵，则它是稳定的
- 用著名的Lyapunov函数作为Hopfield网络的能量函数

Lyapunov函数——能量函数

$$E = -\frac{1}{2}\sum_{i=1}^{h}\sum_{j=1}^{h} w_{ij} o_i o_j - \sum_{j=1}^{n} x_j o_j + \sum_{j=1}^{h} \theta_j o_j$$

- **作为网络的稳定性度量**
  - $w_{ij}o_io_j$：网络的一致性测度。
  - x_jo_j：神经元的输入和输出的一致性测度。
  - $\theta_j o_j$：神经元自身的稳定性的测度。

当AN_k的状态从o_k变成o_k'

1、AN_k是输入神经元

$$E' = -\frac{1}{2}\sum_{i=1\&i\neq k}^{h}\sum_{j=1\&j\neq k}^{h} w_{ij}o_i o_j - \sum_{j=1\&j\neq k}^{n} x_j o_j + \sum_{j=1\&j\neq k}^{h}\theta_j o_j -$$

$$-\frac{1}{2}\sum_{j=1\&j\neq k}^{h} w_{kj}o_k' o_j - \frac{1}{2}\sum_{j=1\&j\neq k}^{h} w_{jk}o_j o_k' - \frac{1}{2}w_{kk}o_k' o_k' -$$

$$-x_k o_k' + \theta_k o_k'$$

$$= -\frac{1}{2}\sum_{i=1\&i\neq k}^{h}\sum_{j=1\&j\neq k}^{h} w_{ij}o_i o_j - \sum_{j=1\&j\neq k}^{n} x_j o_j + \sum_{j=1\&j\neq k}^{h}\theta_j o_j -$$

$$-\sum_{j=1\&j\neq k}^{h} w_{kj}o_k' o_j - x_k o_k' + \theta_k o_k'$$

当AN_k的状态从o_k变成o_k'

$$\Delta E = E' - E$$

$$= -[\sum_{j=1\&\, j\neq k}^{h} w_{kj}(o_k' - o_k)o_j] - x_k(o_k' - o_k) + \theta_k(o_k' - o_k)$$

$$= -[\sum_{j=1\&\, j\neq k}^{h} w_{kj}o_j + x_k - \theta_k](o_k' - o_k) \qquad w_{kk}=0$$

$$= -[\sum_{j=1}^{h} w_{kj}o_j + x_k - \theta_k](o_k' - o_k)$$

$$= -(net_k - \theta_k)\Delta o_k$$

$\Delta E=-(net_k-\theta_k)\Delta o_k$

- AN_k状态的变化： $\Delta o_k=(o_k'-o_k)$
- $\Delta o_k=0$， $\Delta E=0$
- $\Delta o_k>0$，$o_k'=1$& $o_k=0$，o_k由0变到1，

  $net_k>\theta_k$，$net_k-\theta_k>0$

  所以，$-(net_k-\theta_k)\Delta o_k<0$故$\Delta E<0$
- $\Delta o_k<0$, $o_k'=0$& $o_k=1$，o_k由1变到0

  $net_k<\theta_k$，$net_k-\theta_k<0$

  $-(net_k-\theta_k)\Delta o_k<0$故$\Delta E<0$

结论：网络的目标函数总是下降

当AN_k的状态从o_k变成o_k'

2、AN_k不是输入神经元

$$E' = -\frac{1}{2}\sum_{i=1\&i\neq k}^{h}\sum_{j=1\&j\neq k}^{h} w_{ij}o_i o_j - \sum_{j=1}^{n} x_j o_j + \sum_{j=1\&j\neq k}^{h} \theta_j o_j -$$

$$-\frac{1}{2}\sum_{j=1\&j\neq k}^{h} w_{kj}o'_k o_j - \frac{1}{2}\sum_{j=1\&j\neq k}^{h} w_{jk}o_j o'_k - \frac{1}{2}w_{kk}o'_k o'_k + \theta_k o'_k$$

$$= -\frac{1}{2}\sum_{i=1\&i\neq k}^{h}\sum_{j=1\&j\neq k}^{h} w_{ij}o_i o_j - \sum_{j=1}^{n} x_j o_j + \sum_{j=1\&j\neq k}^{h} \theta_j o_j -$$

$$-\sum_{j=1\&j\neq k}^{h} w_{kj}o'_k o_j + \theta_k o'_k$$

当AN_k的状态从o_k变成o_k'

$$\Delta E = E' - E$$

$$= -[\sum_{j=1\&j\neq k}^{h} w_{kj}(o'_k - o_k)o_j] + \theta_k(o'_k - o_k)$$

$$= -[\sum_{j=1\&j\neq k}^{h} w_{kj}o_j - \theta_k](o'_k - o_k)$$

$$= -[\sum_{j=1}^{h} w_{kj}o_j - \theta_k](o'_k - o_k)$$

$$= -(net_k - \theta_k)\Delta o_k$$

无论AN_k的状态是如何变化的，总有$\Delta E \leqslant 0$

统计Hopfield网与Boltzmann机

- **统计Hopfield网**
  - 在网络运行中，神经元状态与 “人工温度”确定的概率相关
  - 网络运行模拟金属退火过程

$$p_i = \frac{1}{1+\exp(-\frac{net_i - \theta_i}{T})}$$

p_i： AN_i的状态取1的概率

net_i： AN_i所获网络输入；

θ_i：AN_i的阈值；

T：系统的人工温度。

统计Hopfield网运行算法

1 取一个很大的值作为人工温度T的初值；

2 对网络中每一个神经元AN_i，

2.1 按照相应式子计算相应的概率p_i；

2.2 按照均匀分布，在[0，1]中取一个随机数r；

2.3 如果 $p_i>r$ 则使AN_i的状态为1，

否则使AN_i的状态为0；

3 逐渐降低温度T，如果温度足够低，则算法结束。否则，重复2

Boltzmann机的训练

- **Boltzmann机是多级递归网路，是Hopfield网的一种扩展。**
- **神经元AN_i实际输出状态$o_i=1$的概率为：**

$$p_i = \frac{1}{1+\exp(-\frac{net_i - \theta_i}{T})}$$

- T趋近于0时，神经元的状态不再具有随机性，Boltzmann机退化成一般Hopfield网。

Boltzmann机的训练

- Boltzmann机的能量函数(一致性函数)

$$E = -\sum_{i<j} w_{ij} o_i o_j + \sum_{j=1}^{h} \theta_j o_j$$

- 神经元AN_i在运行中状态发生了变化

$$\Delta E_i = E(o_i = 0) - E(o_i = 1)$$
$$= \sum_j w_{ij} o_j - \theta_i$$

Boltzmann机的训练

- 如果$\Delta E_i>0$，则应该选AN_i输出为1，否则，应该选AN_i输出为0。
- ΔE_i的值越大，神经元AN_i应该处于状态1的概率就应该越大。反之，ΔE_i的值越小，神经元AN_i应该处于状态1的概率就应该越小。从而，$o_i=1$的概率为：

$$p_i = \frac{1}{1+\exp(-\frac{\Delta E_i}{T})}$$

Boltzmann机的训练

- 处于状态a，b的概率P_a和P_b，对应于o_i=1和o_i=0，其它的神经元在a，b状态下不变
- $P_a = \gamma\, p_i$
- $P_b = \gamma\ (1-p_i)$

$$\frac{P_a}{P_b} = \exp\left(-\frac{E_a - E_b}{T}\right)$$

Boltzmann机的训练

- 网络进行足够多次迭代后，处于某状态的概率与此状态下的能量和此时系统的温度有关。
- 由于高温时网络的各个状态出现的概率基本相同，这就给它逃离局部极小点提供了机会。
- 当系统的温度较低时，如果$E_a<E_b$，则$P_a>P_b$：网络处于较低能量状态的概率较大

Boltzmann机的训练

- **1986年，Hinton和Sejnowski训练方法**
  - **自由概率P_{ij}^{-}：** 没有输入时AN_i和AN_j同时处于激发状态的概率。
  - **约束概率P_{ij}^{+}：** 加上输入后AN_i和AN_j同时处于激发状态的概率。
  - **联接权修改量：** $\Delta w_{ij} = \alpha (P_{ij}^{+} - P_{ij}^{-})$

Boltzmann机训练算法

1　计算约束概率

1.1 对样本集中每个样本，执行如下操作：

1.1.1 将样本加在网络上（输入向量及其对应的输出向量）；

1.1.2 让网络寻找平衡；

1.1.3 记录下所有神经元的状态；

1.2 计算对所有的样本，AN_i和AN_j的状态同时为1的概率P_{ij}^{+}；

Boltzmann机训练算法

2 计算自由概率

2.1 从一个随机状态开始，不加输入、输出，让网络自由运行，并且在运行过程中多次纪录网络的状态；

2.2 对所有的AN_i和AN_j，计算它们的状态同时为1的概率P_{ij}^{-}；

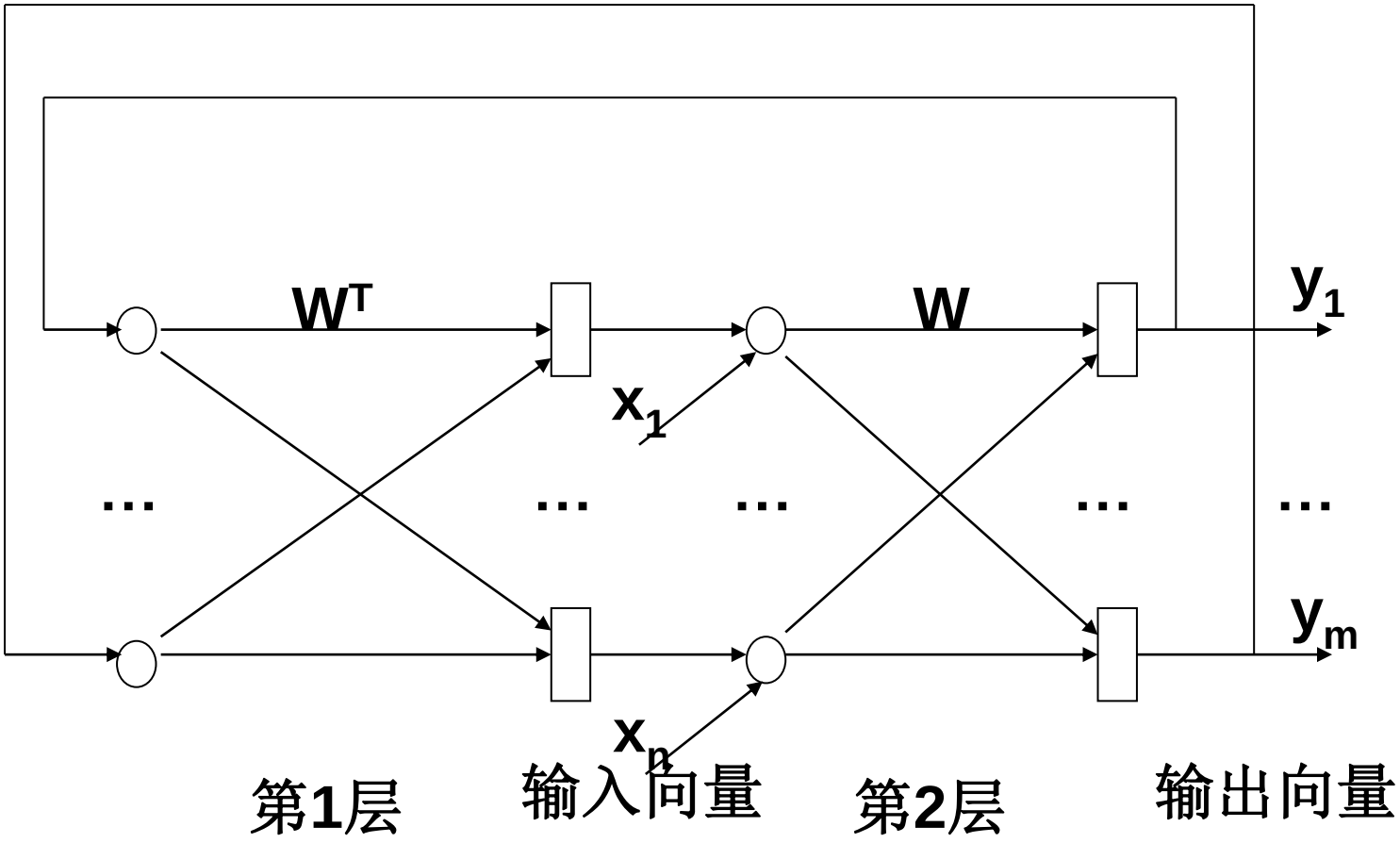
3 对权矩阵进行调整

$$\Delta w_{ij} = \alpha (P_{ij}^{+} - P_{ij}^{-})$$

双联存储器的结构

- **智力链**
  - 从一件事想到另一件事，“唤回失去的记忆”。
- **自相联**
- **异相联**
  - 双联存储器（**Bidirectional Associative Memory—BAM**）。
- 双联存储器具有一定的**推广能力**
  - 它对含有一定缺陷的输入向量，通过对信号的不断变换、修补，最后给出一个正确的输出。

基本的双联存储器结构



网络运行

$Y=F(XW)$

$X=F(YW^T)$

$X=(x_1,\ x_2,\ \ldots,\ x_n)$

$Y=(y_1,\ y_2,\ \ldots,\ y_m)$

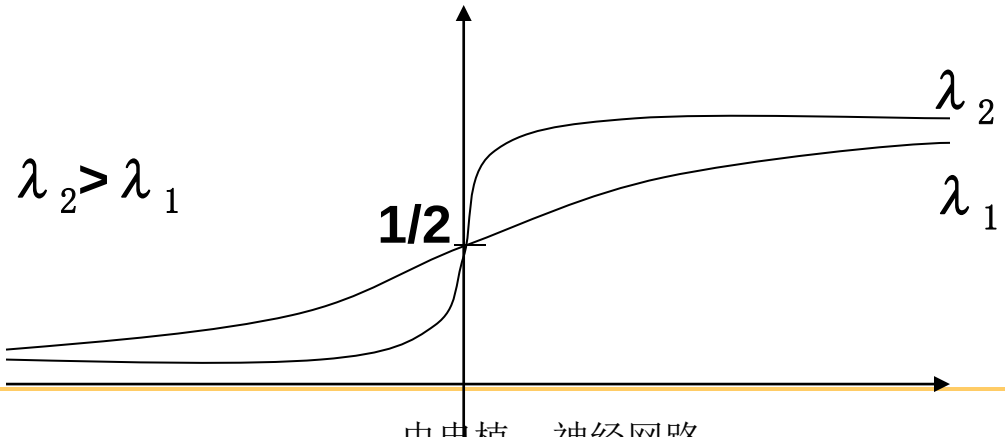
F为神经元的激活函数，一般可采用S形函数

$$y_i = \frac{1}{1+\exp(-\lambda net_i)}$$

激活函数——阈值函数

- 随着 λ 的增加，该函数趋近于阈值为0的阈值函数。

$$y_i=\begin{cases}1 & \text{if } net_i>0\\ 0 & \text{if } net_i<0\\ y_i & \text{if } net_i=0\end{cases}$$



基本BAM的稳定

- **Kosko(1987)：**
  - **基本的双联存储器无条件稳定**——联接权矩阵是互为转置矩阵。
- 当输入向量的维数与输出向量的维数相同时，**W**为方阵，此时如果联接矩阵**W**是对称的，则基本的双联存储器退化成一个**Hopfield**网

异相联存储

- **样本集：** $S=\{(X_1,Y_1),(X_2,Y_2)\ldots,(X_s,Y_s)\}$
- **权矩阵**

$$W=\sum_{i=1}^{s}X_i^T Y_i$$

- 网络需要对输入向量进行循环处理的情况
  - 当输入向量中含有“噪音”
  - 样本集所含的信息超出网络的容量

容量

- **Kosko（1987），一般情况下，相联存储器的容量不会超过网络最小层神经元的个数min**
- **Haines和Hecht-Nielson（1988），“非均匀”网络的容量最多可以达到2^{min}**
- **R. J. McEliece、E. C. Posner、E. R. Rodemich**
  - 用户随机地选择L个状态
  - 每个向量中有$4+\log_2 min$个分量为1，其它为-1
  - **98%的向量成为稳定状态**

$$L < \frac{0.68\,\min^2}{(\log_2 \min + 4)^2}$$

其它的双联存储器

- **具有竞争的双联存储器**
  - 可通过附加侧联接实现竞争。这些权构成另一个主对角线元素为正值，其它元素为负值的权矩阵。
  - **Cohen-Grossberg**定理指出，如果权矩阵是对称的，则网络是稳定。
  - 即使权矩阵不对称，网络通常也是稳定的。但是目前还不知道哪一类权矩阵会引起不稳定

其它的双联存储器

- **连续的双联存储器**
  - **Kosko（1987）证明，神经元的状态非同步变换，而且这些神经元使用其他激励函数，仍然是稳定的，且有更强的表达能力**
- **自适应双联存储器**
  - 最简单的方法是使用**Hebb**学习律进行训练。
  - $\Delta w_{ij} = \alpha o_i o_j$

Hopfield网解决TSP问题

- **1985年，J. J. Hopfield和D. W. Tank用循环网求解TSP。试验表明，当城市的个数不超过30时，多可以给出最优解的近似解。而当城市的个数超过30时，最终的结果就不太理想了**
- n个城市间存在n!/(2n)条可能路径
- 设问题中含有**n**个城市, 用**n*n**个神经元构成网络

Hopfield网解决TSP问题

- d_{xy}——城市X与城市Y之间的距离；
- y_{xi}——城市X的第i个神经元的状态：

$$y_{xi}=\begin{cases}1 & \text{城市X在第i个被访问}\\ 0 & \text{城市X不在第i个被访问}\end{cases}$$

- $w_{xi,yj}$——城市X的第i个神经元到城市Y的第j个神经元的连接权。

Hopfield网用于解决TSP问题

例如：四个城市X、Y、Z、W

城市名	访问顺序标示			
	1	2	3	4
X	0	1	0	0
Y	0	0	0	1
Z	1	0	0	0
W	0	0	1	0

Hopfield网用于解决TSP问题

- **联接矩阵**

$$w_{xi,yj} = -A\,\delta_{xy}(1-\delta_{ij}) - B\,\delta_{ij}(1-\delta_{xy}) - C - \zeta\, d_{xy}(\delta_{ji+1} + \delta_{ji-1})$$

$$\delta_{ij} = \begin{cases} 1 & \text{如果 } i=j \\ 0 & \text{如果 } i \neq j \end{cases}$$

网络的能量函数

$$E = \frac{A}{2}\sum_{x}\sum_{i}\sum_{j\neq i} y_{xi}y_{xj}$$

$$+\frac{B}{2}\sum_{i}\sum_{x}\sum_{x\neq z} y_{xi}y_{zi} + \frac{C}{2}\left(\sum_{x}\sum_{i} y_{xi} - n\right)^2$$

$$+\frac{D}{2}\sum_{x}\sum_{z\neq x}\sum_{i} d_{xz}y_{xi}\left(y_{zi+1} + y_{zi-1}\right)$$

网络的能量函数

- A、B、C、D为惩罚因子

第1项 $\frac{A}{2}\sum_{x}\sum_{i}\sum_{j\neq i}y_{xi}y_{xj}$

- 仅当所有的城市最多只被访问一次时取得极小值0。

网络的能量函数

第2项 $$+\frac{B}{2}\sum_{i}\sum_{x}\sum_{x\neq z}y_{xi}y_{zi}$$

- 仅当每次最多只访问一个城市时取得极小值0。

网络的能量函数

第3项 $$+\frac{C}{2}\left(\sum_x\sum_i y_{xi}-n\right)^2$$

- 当且仅当所有的n个城市一共被访问n次时才取得最小值0。

网络的能量函数

第4项 $$+\frac{D}{2}\sum_{x}\sum_{z\neq x}\sum_{i} d_{xz} y_{xi}\left(y_{zi+1}+y_{zi-1}\right)$$

- 表示按照当前的访问路线的安排，所需要走的路径的总长度

Thank You

Intelligence Science

http://www.intsci.ac.cn/