玻尔兹曼机

刘家锋

哈尔滨工业大学

玻尔兹曼机

玻尔兹曼机

随机神经元

- 神经元的净输入：

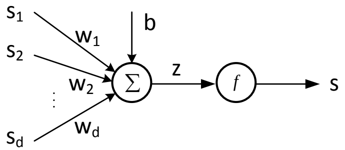
$$z = \sum_{i=1}^{d} w_i s_i + b$$

- 确定神经元：输出确定的值s

$$s = f(z) = \frac{1}{1 + e^{-\alpha z}}$$

- 随机神经元：输出随机变量s

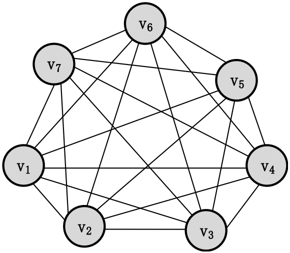
$$P(s = 1) = f(z) = \frac{1}{1 + e^{-z/T}}$$



Boltzmann Machine

- **玻尔兹曼机的结构**
  - 玻尔兹曼机由若干随机神经元组成
  - 神经元之间对称全连接，无自连接：

$$w_{ii} = 0, \quad w_{ij} = w_{ji}$$



玻尔兹曼机的工作过程

- **初始时刻**
  - 每个神经元的状态（输出）随机
- **状态转移**
  - 顺序依照概率改变每个神经元的状态
  - 其它神经元的状态作为输入
- **稳态**
  - 经过足够长的时间，玻尔兹曼机将达到一个稳态
  - 神经元的状态仍会发生变化，但状态发生的概率是稳定的
    - 令$\mathcal{S}$为玻尔兹曼机所有状态的集合，$P(s \in \mathcal{S})$是稳定的
    - 玻尔兹曼机描述了集合$\mathcal{S}$的概率分布，稳态时可以依据此概率产生集合$\mathcal{S}$的抽样

玻尔兹曼机的稳态概率

- **稳态概率**
  - 状态矢量：$\mathbf{v} = (s_1, \cdots, s_d)^t \in \mathcal{S}$
  - 稳态概率服从玻尔兹曼分布：

$$P(\mathbf{v}) = \frac{e^{-E(\mathbf{v})}}{\sum_{\mathbf{u}\in\mathcal{S}} e^{-E(\mathbf{u})}}$$

- **能量函数**
  - 稳态概率只与能量函数$E(\mathbf{v})$有关，而与初始状态无关：

$$E(\mathbf{v}) = -\sum_i b_i s_i^{\mathbf{v}} - \frac{1}{2}\sum_{i,j,i\neq j} w_{ij} s_i^{\mathbf{v}} s_j^{\mathbf{v}}$$

其中，$s_i^{\mathbf{v}}$表示状态$\mathbf{v}$中第i个节点的状态。

玻尔兹曼机与马尔科夫随机场

- **Pairwise Markov Network**
  - Pairwise Markov Network由两组特殊的势函数模型化：
    - 节点势函数(node potentials)：$\{\phi(X_i) : i = 1, \cdots, n\}$
    - 边势函数(edge potentials)：$\{\phi(X_i, X_j) : (X_i, X_j) \in \mathcal{H}\}$
- **玻尔兹曼机**
  - 本质上是一个Pairwise Markov Network
    - 节点势函数：$\phi(s_i) = e^{b_i s_i}$
    - 边势函数：$\phi(s_i, s_j) = e^{w_{ij} s_i s_j}$
  - 节点状态的联合概率：

$$P(\mathbf{v}) = \frac{1}{Z} \prod_i \phi(s_i^{\mathbf{v}}) \prod_{i,j} \phi(s_i^{\mathbf{v}}, s_j^{\mathbf{v}}) = \frac{1}{Z} \prod_i e^{b_i s_i} \prod_{i,j} e^{w_{ij} s_i s_j}$$

  - 能量函数：$E(\mathbf{v}) = -\ln P(\mathbf{v}) - \ln Z$

玻尔兹曼机的学习

- **学习的目的**
  - 训练样本集$\mathcal{V} = \{\mathbf{v}_1, \cdots, \mathbf{v}_n\}$
  - 学习玻尔兹曼机的参数$\mathbf{b}, \mathbf{W}$，使得稳态概率接近于$\mathcal{V}$的抽样分布
- **极大似然估计**
  - 最大化对数似然函数：

$$\max_{\mathbf{b},\mathbf{W}} \sum_{k=1}^{n} \ln P(\mathbf{v}_k) = -\sum_{k=1}^{n} \left[E(\mathbf{v}_k) + \ln \sum_{\mathbf{u}\in\mathcal{S}} e^{-E(\mathbf{u})} \right]$$

无隐含神经元的学习

- **无隐含神经元玻尔兹曼机**
  - 神经元的数量与观察矢量**v**的维数相同
  - 玻尔兹曼机的能量函数：

$$E(\mathbf{v}) = -\sum_i b_i s_i^{\mathbf{v}} - \frac{1}{2}\sum_{i,j,i\neq j} w_{ij} s_i^{\mathbf{v}} s_j^{\mathbf{v}}$$

  - 能量函数关于权值和偏置的导数：

$$\frac{\partial E(\mathbf{v})}{\partial w_{ij}} = -s_i^{\mathbf{v}} s_j^{\mathbf{v}}, \quad \frac{\partial E(\mathbf{v})}{\partial b_i} = -s_i^{\mathbf{v}}$$

无隐含神经元的学习

- **无隐含神经元玻尔兹曼机**
  - 对数似然函数关于权值的导数：

$$\sum_{k=1}^{n}\frac{\partial \ln P(\mathbf{v}_k)}{\partial w_{ij}} = -\sum_{k=1}^{n}\left[\frac{\partial E(\mathbf{v}_k)}{\partial w_{ij}} - \frac{\sum_{\mathbf{u}\in\mathcal{S}} e^{-E(\mathbf{u})}\frac{\partial E(\mathbf{u})}{\partial w_{ij}}}{\sum_{\mathbf{u}\in\mathcal{S}} e^{-E(\mathbf{u})}}\right]$$

$$= \sum_{k=1}^{n} s_i^{\mathbf{v}_k} s_j^{\mathbf{v}_k} - n\sum_{\mathbf{u}\in\mathcal{S}} P(\mathbf{u}) s_i^{\mathbf{u}} s_j^{\mathbf{u}}$$

$$= \langle s_i s_j\rangle_{data} - \langle s_i s_j\rangle_{model}$$

  - 对数似然函数关于偏置的导数：

$$\sum_{k=1}^{n}\frac{\partial \ln P(\mathbf{v}_k)}{\partial b_i} = \sum_{k=1}^{n} s_i^{\mathbf{v}_k} - n\sum_{\mathbf{u}\in\mathcal{S}} P(\mathbf{u}) s_i^{\mathbf{u}}$$

$$= \langle s_i\rangle_{data} - \langle s_i\rangle_{model}$$

无隐含神经元的学习

Algorithm 1 无隐含神经元玻尔兹曼机学习

Input: 训练样本集：$\mathcal{V} = \{\mathbf{v}_1, \cdots, \mathbf{v}_n\}$

1: 随机初始化权值$\mathbf{W}$和偏置$\mathbf{b}$

2: **repeat**

3: 计算训练样本集$\mathcal{V}$的数据项：

$$\langle s_i s_j \rangle_{data} = \sum_{k=1}^{n} s_i^{\mathbf{v}_k} s_j^{\mathbf{v}_k}, \quad \langle s_i \rangle_{data} = \sum_{k=1}^{n} s_i^{\mathbf{v}_k}$$

4: 玻尔兹曼机运行到稳态，抽样近似计算模型项：

$$\langle s_i s_j \rangle_{model} = n \sum_{\mathbf{u} \in \mathcal{S}} P(\mathbf{u}) s_i^{\mathbf{u}} s_j^{\mathbf{u}}, \quad \langle s_i \rangle_{model} = n \sum_{\mathbf{u} \in \mathcal{S}} P(\mathbf{u}) s_i^{\mathbf{u}}$$

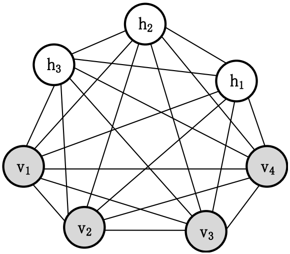
5: 梯度法修正权值和偏置：

$$w_{ij} = w_{ij} + \eta \left(\langle s_i s_j \rangle_{data} - \langle s_i s_j \rangle_{model} \right)$$
$$b_i = b_i + \eta \left(\langle s_i \rangle_{data} - \langle s_i \rangle_{model} \right)$$

6: **until** 达到收敛条件

有隐含神经元玻尔兹曼机

- **隐含神经元**
  - 观察矢量**v**只对应部分（可见）神经元的状态
  - 其它神经元为隐含神经元
- **隐含神经元的作用**
  - 提高玻尔兹曼机对于稳态分布的描述能力



有隐含神经元的学习

- **对数似然函数的梯度**

$$\sum_{k=1}^{n} \frac{\partial \ln P(\mathbf{v}_k)}{\partial w_{ij}} = \langle s_i s_j \rangle_{clamped} - \langle s_i s_j \rangle_{model}$$

$$\sum_{k=1}^{n} \frac{\partial \ln P(\mathbf{v}_k)}{\partial b_i} = \langle s_i \rangle_{clamped} - \langle s_i \rangle_{model}$$

- **钳制状态：** 计算$\langle s_i s_j \rangle_{clamped}$，$\langle s_i \rangle_{clamped}$
  - 钳制可见神经元的状态为训练样本相应的元素，不可改变
  - 计算隐含神经元的稳态分布
- **稳态：** 计算$\langle s_i s_j \rangle_{model}$，$\langle s_i \rangle_{model}$
  - 隐含神经元和可见神经元的状态均可改变
  - 计算玻尔兹曼机的稳态分布

玻尔兹曼机的应用

- **产生式模型**
  - 学习和描述训练样本集的复杂概率分布
- **产生随机样本**
  - 在模型稳态时，可以抽样与训练样本集同分布的样本
- **计算观察概率**
  - 计算模型产生某个观察矢量$\mathbf{v}$的概率
  - 无隐含神经元模型，可以直接计算：

$$P(\mathbf{v}) = \frac{e^{-E(\mathbf{v})}}{\sum_{\mathbf{u}\in\mathcal{S}} e^{-E(\mathbf{u})}}$$

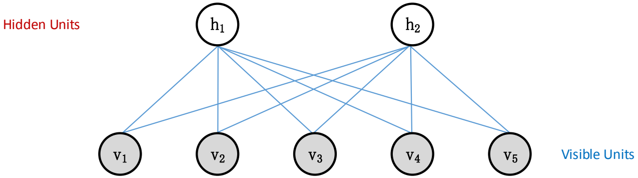
  - 有隐含神经元模型，钳制可见神经元的状态，在稳态时抽样近似计算边缘概率：

$$P(\mathbf{v}) = \sum_{\mathbf{h}} P(\mathbf{v},\mathbf{h})$$

限制玻尔兹曼机

RBM: Restricted Boltzmann Machines

- **RBM的结构**
  - 本质上仍然是玻尔兹曼机，但在结构上做出了一些限制
  - 描述的仍然是训练样本集$\mathcal{V} = \{\mathbf{v}_1, \cdots, \mathbf{v}_n\}$的抽样分布
  - 结构上的限制：
    1. 两层结构：可见节点层和隐含节点层，两层节点之间对称连接，神经元有各自的偏置
    2. 可见节点层和隐含节点层内没有连接



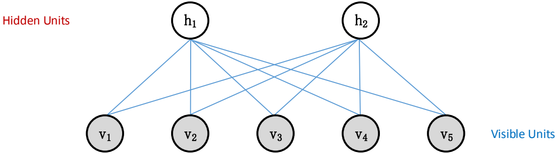
RBM计算上的简化

- **钳制状态**
  - 一步达到稳定，隐含节点的稳态输出概率可以直接计算
  - 可见节点上输入训练样本**v**，计算隐含节点s_j的状态概率：

$$P(s_j = 1)_{clamped} = \frac{1}{1 + e^{-\sum_i w_{ij} v_i - b_j}}$$

  - 学习时：

$$\langle s_i, s_j \rangle_{clamped} = \sum_{s_j \in \{0,1\}} P(s_j) v_i s_j = v_i P(s_j = 1)_{clamped}$$



RBM计算上的简化

- **自由状态**
  - 在给定隐含节点状态，可见节点相互独立，可直接计算可见节点s_i的概率：

$$P(s_i = 1)_{free} = \frac{1}{1 + e^{-\sum_j w_{ij} s_j - b_i}}$$

依此概率，随机抽样节点s_i的状态s_i^{free}。

  - 在给定可见节点状态，隐含节点相互独立，可直接计算隐含节点s_j的概率：

$$P(s_j = 1)_{free} = \frac{1}{1 + e^{-\sum_i w_{ij} s_i^{free} - b_j}}$$

依此概率，随机抽样节点s_j的状态s_j^{free}。

RBM计算上的简化

- **模型的稳态**
  - 经过可见层→隐含层→可见层→隐含层的若干次迭代，模型可以达到稳态
  - Contrastive Divergence（对比散度）：
    - 训练样本对应的可见节点状态已经接近于稳态
    - 迭代次数不需要很多，几次即可，甚至只须一次
  - 学习时：

$$
\begin{aligned}
\langle s_i, s_j \rangle_{free} &= \sum_{s_i \in \{0,1\}} \sum_{s_j \in \{0,1\}} P(s_i) P(s_j) s_i s_j \\
&= P(s_i = 1)_{free} P(s_j = 1)_{free}
\end{aligned}
$$

限制玻尔兹曼机的学习

Algorithm 2 RBM的学习

Input: 训练样本集：$\mathcal{V} = \{\mathbf{v}_1, \cdots, \mathbf{v}_n\}$

1: 随机初始化权值$\mathbf{W}$和偏置$\mathbf{b}$

2: **repeat**

3: 从训练集$\mathcal{V}$中随机抽取一个样本$\mathbf{v}$

4: 可见层→隐含层：计算隐含层概率$P(s_j = 1)_{clamped}$，抽样$s_j^{clamped}$

5: 隐含层→可见层：计算可见层概率$P(s_i = 1)_{free}$，抽样s_i^{free}

6: 可见层→隐含层：计算隐含层概率$P(s_j = 1)_{free}$，抽样s_j^{free}

7: 梯度法修正权值和偏置：

$$w_{ij} = w_{ij} + \eta \left[v_i P(s_j = 1)_{clamped} - s_i^{free} P(s_j = 1)_{free} \right]$$

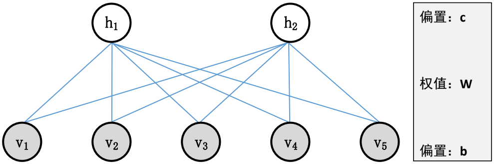
$$b_i = b_i + \eta (v_i - s_i^{free})$$

$$b_j = b_j + \eta \left[P(s_j = 1)_{clamped} - P(s_j = 1)_{free} \right]$$

8: **until** 达到一定的迭代次数

重新定义符号

- **重新定义RBM的符号**
  - 可见节点的状态（矢量）：$\mathbf{v}$
  - 隐含节点的状态（矢量）：$\mathbf{h}$
  - 可见节点的偏置（矢量）：$\mathbf{b}$
  - 隐含节点的偏置（矢量）：$\mathbf{c}$
  - 连接权值（矩阵）：$\mathbf{W}$，$\mathbf{w}_i$为第i行矢量



非二值输入

- **非二值特征的理解**
  - 将每一维特征x_i归一化为$[0, 1]$之间的实数
  - 以此作为相应输入节点取值为1的概率：

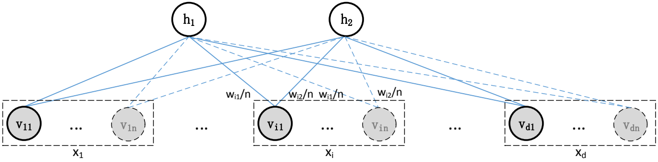
$$P(v_i = 1) = x_i$$

- **处理方法1**：抽样法
  - 依照每个样本的特征对输入矢量抽样，产生大量的样本
  - 学习时：直接以抽样之后的样本集学习模型参数
  - 分类时：对所有抽样样本的识别结果取平均
  - 缺点：计算量大

非二值输入

- 处理方法2：直接计算法
  - 每个可见节点复制n个副本，副本节点的输入$v_{i_1}, \cdots, v_{i_n}$以相应特征x_i为概率抽样
  - 第i个特征副本节点与第j个隐含节点的连接权值均为w_{ij}/n
  - $n \to \infty$时，可以直接计算隐含节点的概率：

$$P(h_j = 1) = \lim_{n\to\infty} \frac{1}{1 + \exp\left(-\sum_{i=1}^{d}\sum_{k=1}^{n} \frac{w_{ij}}{n} v_{ik} - b_j\right)} = \frac{1}{1 + \exp\left(-\sum_{i=1}^{d} w_{ij} x_i - b_j\right)}$$



RBM生成样本

- **Gibbs抽样**
  - 随机矢量$\mathbf{s} = (s_1, \cdots, s_m)^t \sim P(s_1, \cdots, s_m)$，依照联合概率$P(s_1, \cdots, s_m)$抽样存在困难
  - 但可以很容易地依据条件概率$P(s_i|\mathbf{s}_{-i})$，抽样其中的一个随机变量s_i，$\mathbf{s}_{-i}$表示除s_i之外的其它随机变量
  - 抽样过程：
    1. 随机初始化$\mathbf{s}^{(0)}$
    2. 依次依照概率$P(s_i^{(t)}|s_1^{(t)}, \cdots, s_{i-1}^{(t)}, s_{i+1}^{(t-1)}, \cdots, s_m^{(t-1)})$抽样$m$个随机变量，得到新的抽样矢量$\mathbf{s}^{(t)}$
    3. 重复上述过程$T+n$次，放弃前T个抽样，保留后n个抽样矢量
  - 可以证明，Gibbs抽样得到的n个矢量，近似服从联合分布$P(s_1, \cdots, s_m)$

RBM生成样本

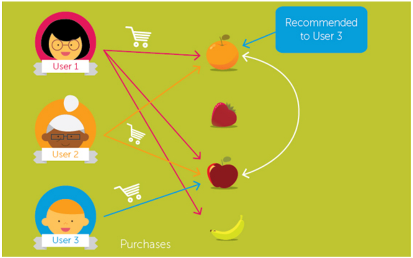
- **Gibbs抽样法**
  - 使用训练集$\mathcal{V}$学习RBM的参数
  - RBM描述了$\mathcal{V}$的抽样分布$p(\mathbf{v})$，希望抽样n个新的满足同样分布的样本
  - 利用RBM输入节点之间和隐含节点之间的条件独立性，可以采用Gibbs抽样的方法生成新的样本：
    1. 随机初始化RBM的输入$\mathbf{v}^{(0)}$
    2. 计算隐含层节点的概率$P(\mathbf{h}|\mathbf{v}^{(t-1)})$，抽样$\mathbf{h}^{(t)}$
    3. 计算输入层节点的概率$P(\mathbf{v}|\mathbf{h}^{(t)})$，抽样$\mathbf{v}^{(t)}$
  - 经过若干次迭代之后，对输入层节点抽样，生成新的样本

RBM生成样本

- **Contrastive Divergence抽样法**
  - 使用训练集$\mathcal{V}$学习RBM的参数
  - 认为训练样本作为输入，已经接近于RBM的稳态
    1. 随机选择样本$\mathbf{v} \in \mathcal{V}$，初始化RBM的输入$\mathbf{v}^{(0)} = \mathbf{v}$
    2. 计算隐含层节点的概率$P(\mathbf{h}|\mathbf{v}^0)$，抽样$\mathbf{h}^{new}$
    3. 计算输入层节点的概率$P(\mathbf{v}|\mathbf{h}^{new})$，抽样$\mathbf{v}^{new}$

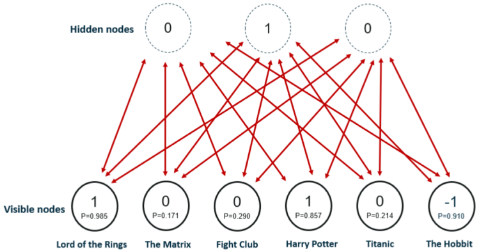
RBM用于协同滤波

- **Collaborative Filtering**
  - 通过用户的历史行为数据发现用户对商品或内容的喜好，并对这些喜好进行度量和打分
  - 常用的方法：
    1. 根据用户的历史数据，对用户分类
    2. 根据同类用户的偏好，推荐商品或内容



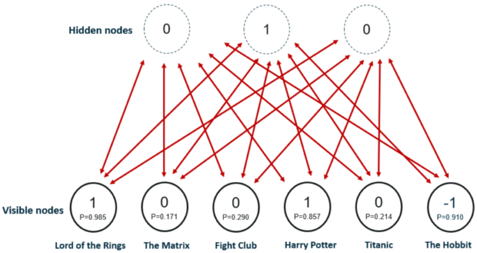
RBM用于协同滤波

- **电影推荐**
  - 网站上有很多用户的观影记录，表达了对不同电影的喜好
  - 使用大量的用户数据，学习RBM：
    - 输入节点对应不同的影片
    - 隐含节点描述了用户的不同偏好



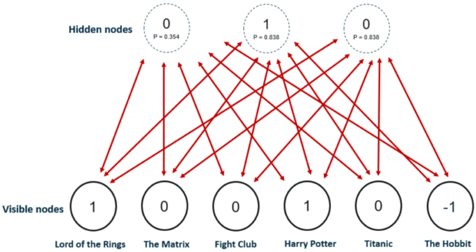
RBM用于协同滤波

- **用户推荐**
  - 在RBM的输入层，输入某个用户对不同电影的喜好：
    - 1：表示喜欢
    - 0：表示不喜欢
    - -1：表示不知道（随机抽样输入）
  - RBM达到稳态，输入节点的概率表示了用户对相应电影的喜好程度



RBM用于协同滤波

- **隐层节点**
  - 经过学习之后，隐含层节点反映了电影的某种潜在的属性：
    - 例如电影的类型：Drama, Fantasy和Science Fiction
  - 输入用户对不同影片的评价，可以在隐层节点反映出所喜好的电影类型



RBM计算样本的生成概率

- 可见节点和隐含节点的联合概率

$$P(\mathbf{v},\mathbf{h}) = \frac{e^{-E(\mathbf{v},\mathbf{h})}}{Z} = \frac{e^{-E(\mathbf{v},\mathbf{h})}}{\sum_{\mathbf{u},\mathbf{h}} e^{-E(\mathbf{u},\mathbf{h})}}$$

其中，能量函数：

$$E(\mathbf{v},\mathbf{h}) = -\sum_i b_i v_i - \sum_j c_j h_j - \frac{1}{2}\sum_{i,j} w_{ij} v_i h_j = -\mathbf{b}^t\mathbf{v} - \mathbf{c}^t\mathbf{h} - \mathbf{h}^t\mathbf{W}\mathbf{v}$$

- 可见节点的边缘概率

$$P(\mathbf{v}) = \sum_{\mathbf{h}} P(\mathbf{v},\mathbf{h}) = \frac{\sum_{\mathbf{h}} e^{-E(\mathbf{v},\mathbf{h})}}{Z} = \frac{e^{-F(\mathbf{v})}}{\sum_{\mathbf{u}} e^{-F(\mathbf{u})}}$$

其中，$F(\mathbf{v})$为自由能量函数：

$$F(\mathbf{v}) = -\ln \sum_{\mathbf{h}} e^{-E(\mathbf{v},\mathbf{h})}$$

RBM计算样本的生成概率

- **自由函数**
  - 可以证明：

$$F(\mathbf{v}) = -\ln \sum_{\mathbf{h}} e^{-E(\mathbf{v},\mathbf{h})} = -\mathbf{b}^t\mathbf{v} - \sum_i \ln\left(1 + e^{c_i + \mathbf{w}_i\mathbf{v}}\right)$$

  自由函数可以直接计算，无需RBM达到稳态

- **边缘概率**
  - 计算RBM生成矢量$\mathbf{v}$的概率，需要RBM稳态抽样近似计算分母求和式：

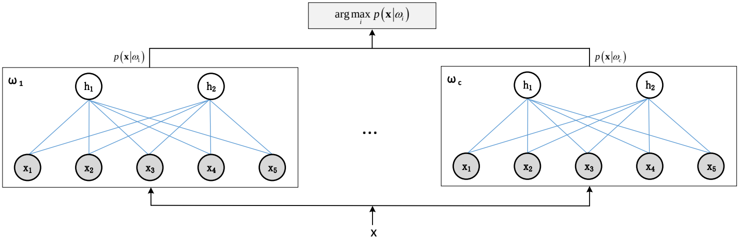
$$P(\mathbf{v}) = \sum_{\mathbf{h}} P(\mathbf{v},\mathbf{h}) = \frac{e^{-F(\mathbf{v})}}{\sum_{\mathbf{u}} e^{-F(\mathbf{u})}}$$

  - 比较RBM生成不同矢量$\mathbf{v}_1, \mathbf{v}_2$的概率，无须稳态抽样：

$$P(\mathbf{v}_1) > P(\mathbf{v}_2) \quad \Longleftrightarrow \quad F(\mathbf{v}_1) < F(\mathbf{v}_2)$$

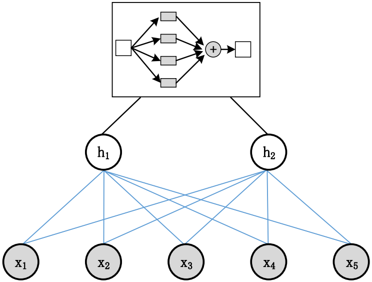
RBM构建分类器

- 方案一
  - RBM作为一个生成式模型
  - **学习：**每个类别学习一个RBM，描述类别的类条件概率
  - **识别：**分别计算每个类别产生识别样本的概率$p(\mathbf{x}|\omega_i)$
  - 缺点：
    - 每个类别的样本比较少，模型学习不充分
    - RBM产生样本的概率计算困难



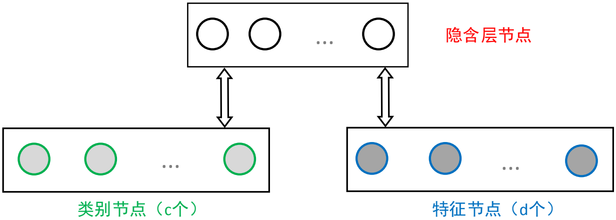
RBM构建分类器

- 方案二
  - RBM用于特征提取，使用其它方法进行分类
  - RBM无监督学习，隐含层的输出（概率）作为提取的特征
  - 使用其它分类器方法，在这些特征上构建分类器



RBM构建分类器

- 方案三
  - 统一的产生式模型，类别标签和特征一起作为可见节点，RBM描述联合概率$P(\mathbf{x}, \omega_i)$
  - **学习**：输入训练样本$\mathbf{x}$，相应类别节点置1，其它节点置0
  - **识别**：依次将c个类别节点置1，计算类别与特征同时发生的联合概率
  - **计算**：比较联合概率的大小，只需计算F值，不需要计算Z

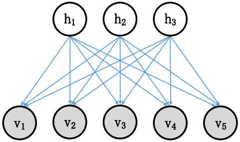


深度信念网络

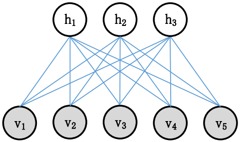
Belief Networks

- 信念网络
  - 两层的贝叶斯网络
  - 节点为二值状态，状态概率只与父辈节点有关
- **Logistic Belief Network**
  - 节点状态概率为Logistic函数：

$$P(v_i = 1|\{h_j\}) = \frac{1}{1 + e^{-\sum_j w_{ji}h_j - b_i}}$$



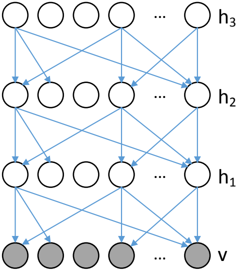
Belief Networks



Restricted Boltzmann Machine

Deep Belief Networks

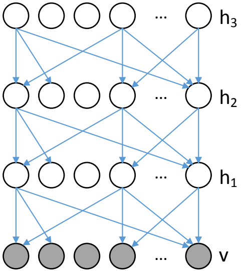
- **深度信念网络**
  - 增加信念网络的隐含层数量
  - 提高网络的描述能力
  - 能够在可见节点层描述数据集更复杂的概率分布
- **DBN的困难**
  - 如何描述最高层节点的先验概率$P(\mathbf{h}_3)$?



隐含层的先验概率

- 生成样本
  1. 依据概率 $P(\mathbf{h}_3)$，抽样 $\mathbf{h}_3$
  2. 计算 $P(\mathbf{h}_2|\mathbf{h}_3)$，抽样 $\mathbf{h}_2$
  3. 计算 $P(\mathbf{h}_1|\mathbf{h}_2)$，抽样 $\mathbf{h}_1$
  4. 计算 $P(\mathbf{v}|\mathbf{h}_1)$，抽样 $\mathbf{v}$
- 计算边缘概率

$$
\begin{aligned}
P(\mathbf{v}) &= \sum_{\mathbf{h}_1,\mathbf{h}_2,\mathbf{h}_3} P(\mathbf{v},\mathbf{h}_1,\mathbf{h}_2,\mathbf{h}_3) \\
&= \sum_{\mathbf{h}_1,\mathbf{h}_2,\mathbf{h}_3} P(\mathbf{v}|\mathbf{h}_1)P(\mathbf{h}_1|\mathbf{h}_2)P(\mathbf{h}_2|\mathbf{h}_3)P(\mathbf{h}_3)
\end{aligned}
$$



互补先验：Complementary Prior

- **互补先验：是一个Markov随机过程的概念**
  - 平稳Markov链，具有状态$x_1, \cdots, x_K$，状态转移概率：

$$P_{ij} = P(X_t = x_j | X_{t-1} = x_i)$$

  - 如果Markov链是各态历经的（遍历的），则当$t \to \infty$时，状态有唯一的平稳分布：

$$P_{t\to\infty}(X_t = x_j) = \pi_j = \sum_{i=1}^{K} \pi_i P_{ij}$$

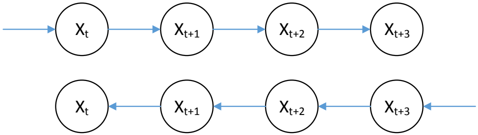
  - 达到平稳的Markov链会满足细节平衡性(Detailed Balance)：

$$\pi_i P_{ij} = \pi_j P_{ji}$$

互补先验：Complementary Prior

- **互补先验的概念**
  - 这样的Markov链是可逆的：

$$\begin{aligned} P_\infty(X_t, X_{t+1}, X_{t+2}, X_{t+3}) &= P_\infty(X_t)P(X_{t+1}|X_t)P(X_{t+2}|X_{t+1})P(X_{t+3}|X_{t+2}) \\ &= P_\infty(X_{t+3})P(X_{t+2}|X_{t+3})P(X_{t+1}|X_{t+2})P(X_t|X_{t+1}) \end{aligned}$$



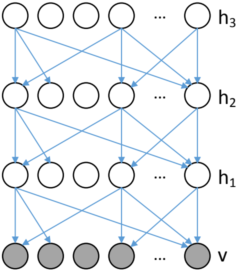
隐含层的互补先验

- **互补先验假设**
  - 假设DBN的最高隐含层具有互补先验：$P_\infty(\mathbf{h}_3)$
- **DBN正方向运行**
  - 每一个隐含层的边缘概率都是互补先验：

$$P_\infty(\mathbf{h}_2) = \sum_{\mathbf{h}_3} P(\mathbf{h}_2|\mathbf{h}_3)P_\infty(\mathbf{h}_3)$$

$$P_\infty(\mathbf{h}_1) = \sum_{\mathbf{h}_2} P(\mathbf{h}_1|\mathbf{h}_2)P_\infty(\mathbf{h}_2)$$

$$P_\infty(\mathbf{v}) = \sum_{\mathbf{h}_1} P(\mathbf{v}|\mathbf{h}_1)P_\infty(\mathbf{h}_1)$$



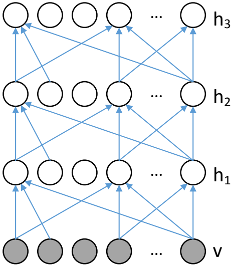
隐含层的互补先验

- **DBN反方向运行**
  - 根据平稳Markov链的可逆性，互补先验也可以通过反向运行得到：

$$P_\infty(\mathbf{h}_1) = \sum_{\mathbf{v}} P(\mathbf{h}_1|\mathbf{v}) P_\infty(\mathbf{v})$$

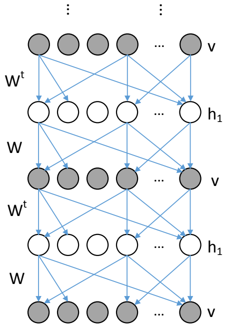
$$P_\infty(\mathbf{h}_2) = \sum_{\mathbf{h}_1} P(\mathbf{h}_2|\mathbf{h}_1) P_\infty(\mathbf{h}_1)$$

$$P_\infty(\mathbf{h}_3) = \sum_{\mathbf{h}_2} P(\mathbf{h}_3|\mathbf{h}_2) P_\infty(\mathbf{h}_2)$$



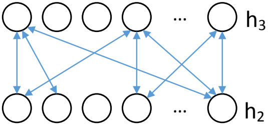
无限层同权值Belief Network

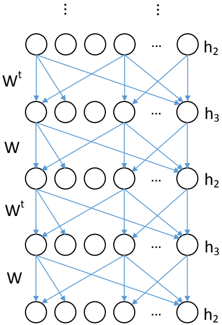
- **平稳的Markov链**
  - 无论$P_\infty(\mathbf{h}_3)$还是$P_\infty(\mathbf{v})$都需要Markov链达到平稳状态才能够得到
  - 只有当$t \to \infty$时，Markov链才能够达到平稳
- **构造无限层同权值信念网络**
  - 以可见层与第1隐含层为例
  - 在无穷远处随机采样可见层（或隐含层）
  - 达到平稳后，可以依据互补先验抽样可见层（或隐含层）



无限层同权值Belief Network

- **等价性**
  - 无限层同权值网络同RBM是等价的
  - 可以利用RBM来解决DBN顶层节点的先验概率问题

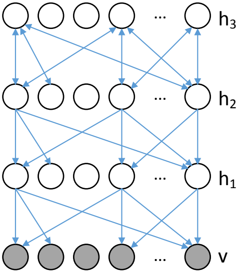




Deep Belief Networks

- **DBN生成样本**
  1. 随机抽样$\mathbf{h}_3$的状态
  2. RBM方式，$\mathbf{h}_3 \leftrightarrow \mathbf{h}_2$迭代至稳态
  3. 依据稳态概率抽样$\mathbf{h}_2$
  4. 计算$P(\mathbf{h}_1|\mathbf{h}_2)$，抽样$\mathbf{h}_1$
  5. 计算$P(\mathbf{v}|\mathbf{h}_1)$，抽样$\mathbf{v}$
- **计算边缘概率**

$$P(\mathbf{v}) = \sum_{\mathbf{h}_1,\mathbf{h}_2,\mathbf{h}_3} P(\mathbf{v},\mathbf{h}_1,\mathbf{h}_2,\mathbf{h}_3)$$
$$= \sum_{\mathbf{h}_1,\mathbf{h}_2,\mathbf{h}_3} P(\mathbf{v}|\mathbf{h}_1)P(\mathbf{h}_1|\mathbf{h}_2)P(\mathbf{h}_2,\mathbf{h}_3)$$



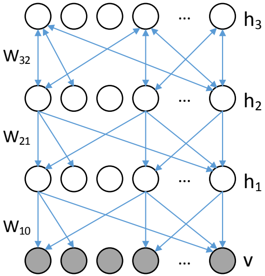
DBN的学习

- 学习的目标
  - 给定训练集$\mathcal{V} = \{\mathbf{v}_1, \cdots, \mathbf{v}_n\}$
  - 学习DBN的权值参数$\mathbf{W}$，描述样本集$\mathcal{V}$的抽样分布
- 学习方法
  - 整体学习：构造似然函数，梯度法学习参数

$$L(\mathbf{W}) = \prod_{i=1}^{n} \sum_{\mathbf{h}_1, \mathbf{h}_2, \mathbf{h}_3} P(\mathbf{v}_i|\mathbf{h}_1)P(\mathbf{h}_1|\mathbf{h}_2)P(\mathbf{h}_2, \mathbf{h}_3)$$

  - 逐层学习：

$$\mathbf{W}_{10} \rightarrow \mathbf{W}_{21} \rightarrow \mathbf{W}_{32}$$

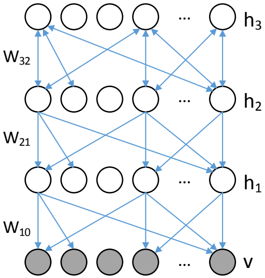


DBN的学习

- **底层权值参数的学习**
  - 在已知参数$\mathbf{W}_{21}, \mathbf{W}_{32}$的条件下，可以计算概率：

$$P(\mathbf{h}_1) = \sum_{\mathbf{h}_2, \mathbf{h}_3} P(\mathbf{h}_1|\mathbf{h}_2) P(\mathbf{h}_2, \mathbf{h}_3)$$

  - $P(\mathbf{h}_1)$是$\mathbf{h}_1$的互补先验概率
  - 互补先验概率还可以由无限层同权值BN来实现
  - 权值参数$\mathbf{W}_{10}$，可以由$\mathbf{v}$和$\mathbf{h}_1$按照RBM的方式学习，而不需要考虑$\mathbf{W}_{21}, \mathbf{W}_{32}$



DBN的学习

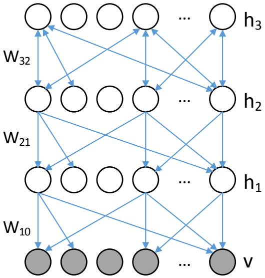
- 上层权值参数的学习
  - 固定$\mathbf{W}_{10}$
  - 以$\mathbf{v}-\mathbf{h}_1$的RBM稳态概率近似互补先验概率
  - Contrastive Divergence：

$$P(\mathbf{h}_1) \leftarrow \frac{1}{1+e^{-\mathbf{W}_{10}^t\mathbf{v}-\mathbf{b}}}$$

  - 映射$\mathcal{V}$为：

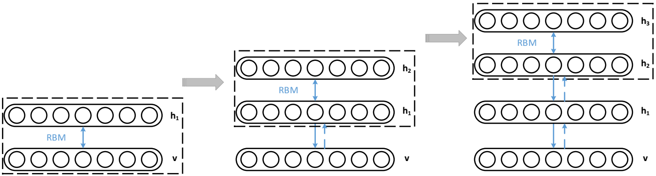
$$\mathcal{H}_1 = \{P(\mathbf{h}_1^1), \cdots, P(\mathbf{h}_1^n)\}$$

  - 以$\mathcal{H}_1$为训练集，RBM方式学习$\mathbf{W}_{21}$



DBN的学习

- **顶层权值参数的学习**
  - 顶层本身就是一个RBM
  - 输入训练集$\mathcal{V}$，自底向上逐层计算，得到$\mathbf{h}_2$层的近似“互补先验概率” $\mathcal{H}_2$
  - RBM学习权值参数$\mathcal{W}_{32}$



DBN的学习

- **逐层学习**
  - 是一种近似的“贪心”优化方法
  - 可以证明，学习过程是对似然函数下界的优化
  - 为模型参数学习提供了一个好的初始值
- **精细调整**
  - 将DBN作为一个统一产生式模型学习
  - 极大似然估计，学习参数权值
  - 需要简化计算

Fine Tuning

- **Wake-Sleep学习过程**
  - 除最高层RBM之外，解除上下行参数相同约束
  - 迭代公式：

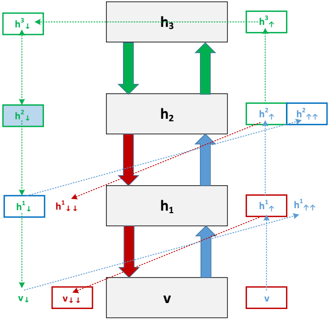
$$\mathbf{W}_{\downarrow}^{10} = \mathbf{W}_{\downarrow}^{10} + \eta \mathbf{h}_{\uparrow}^{1} \left[\mathbf{v} - P(\mathbf{v}_{\downarrow\downarrow} | \mathbf{h}_{\uparrow}^{1}) \right]$$

$$\mathbf{W}_{\downarrow}^{21} = \mathbf{W}_{\downarrow}^{21} + \eta \mathbf{h}_{\uparrow}^{2} \left[\mathbf{h}_{\uparrow}^{1} - P(\mathbf{h}_{\downarrow\downarrow}^{1} | \mathbf{h}_{\uparrow}^{2}) \right]$$

$$\mathbf{W}^{32} = \mathbf{W}^{32} + \eta \left[\mathbf{h}_{\uparrow}^{2} (\mathbf{h}_{\uparrow}^{3})^{t} - \mathbf{h}_{\downarrow}^{2} (\mathbf{h}_{\downarrow}^{3})^{t} \right]$$

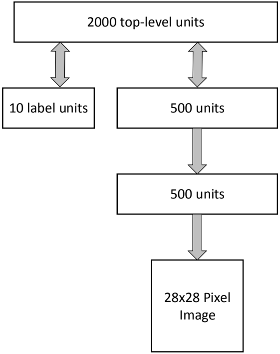
$$\mathbf{W}_{\uparrow}^{12} = \mathbf{W}_{\uparrow}^{12} + \eta \mathbf{h}_{\downarrow}^{1} \left[\mathbf{h}_{\downarrow}^{2} - P(\mathbf{h}_{\uparrow\uparrow}^{2} | \mathbf{h}_{\downarrow}^{1}) \right]$$

$$\mathbf{W}_{\uparrow}^{01} = \mathbf{W}_{\uparrow}^{01} + \eta \mathbf{v}_{\downarrow} \left[\mathbf{h}_{\downarrow}^{1} - P(\mathbf{h}_{\uparrow\uparrow}^{1} | \mathbf{v}_{\downarrow}) \right]$$



DBN的应用(MNIST)

- 类别节点
  - 作为顶层RBM的一部分引入
  - 学习时，根据监督信息相应节点置1，其它置0
  - 自由状态下，类别节点的激活函数为Softmax
  - DBN模型化联合概率$p(\mathbf{x}, y)$

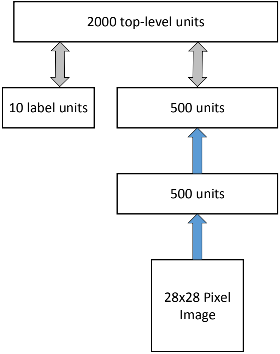


样本生成

- **生成样本**
  - 根据要生成样本的类别，钳制类别节点
  - 顶层RBM多次迭代到稳态（1000次）
  - $\mathbf{h}^2$层节点抽样，向下计算$\mathbf{h}^1$层和$\mathbf{x}$层节点的概率并抽样
  - 以$\mathbf{x}$层节点的采样值作为生成的样本

DBN识别

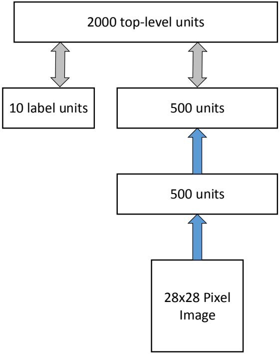
- 识别方法一
  - $\mathbf{x}$层输入识别样本
  - 上行到$\mathbf{h}_2$层
  - 类别节点设置为等概率0.1
  - 顶层RBM迭代到稳态
  - 依照稳态概率，随机抽样类别节点



DBN识别

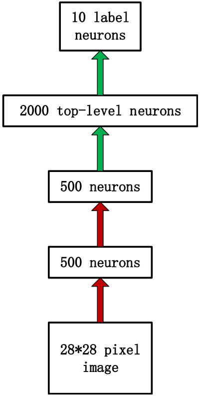
- **识别方法二**
  - $\mathbf{x}$层输入识别样本
  - 上行到$\mathbf{h}_2$层
  - 类别节点依次置1，计算顶层RBM的“自由函数”
  - 选择最大者作为识别结果：

$$y^* = \arg\max_y P(\mathbf{x}, y)$$



DBN识别

- 识别方法三
  - 训练阶段：
    - 无监督逐层学习DBN（不包括类别节点）
    - 将DBN转化为ANN，顶层之上增加类别节点
    - BP算法精细学习
  - 识别阶段：
    - 直接按照ANN的方式识别



End