Deep Learning Technology and Application

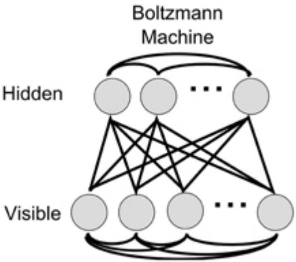
Ge Li

Peking University

玻尔兹曼机（BM）

Boltzmann Machine (BM)

- 1985年由Geoffrey Hinton和Terry Sejnowski提出；
- 一个基于统计力学模型的随机循环神经网络(Stochastic Recurrent Neural Network)
- 随机神经网络随机神经元，神经元只有两种状态（未激活、激活）一般用二进制0和1表示，状态的取值根据概率统计法则决定。
- BM包括一个可见层和一个隐藏层，全对称连接，无自反馈；



玻尔兹曼机（BM）

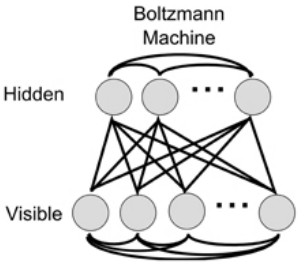
- Boltzmann Machine (BM)

  优点：

  - ◆ 具有强大的无监督学习能力；
  - ◆ 能够学习数据中复杂的规则；

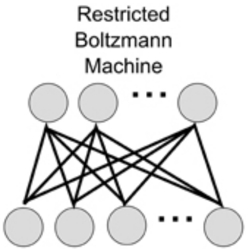
  缺点：

  - ◆ 训练学习时间非常长；
  - ◆ 不仅无法确切地计算BM所表示的分布，甚至得到服从BM所表示分布的随机样本也很困难。



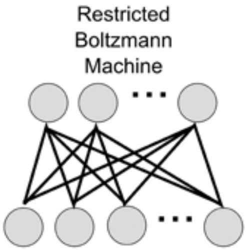
受限的玻尔兹曼机（RBM）

- Restricted Boltzmann Machine, RBM
  - 为克服BM的问题，1986年Smolensky引入RBM。
  - 2002年Hinton提出RBM的快速学习算法-对比散度（Contrastive Divergence, CD）；
  - 理论方面，促进了随机近似理论、基于能量的模型、未归一化的统计模型的研究。
  - 应用方面，RBM已被成功地应用于分类、回归、降维、高维时间序列建模、图像特征提取、协同过滤等问题。



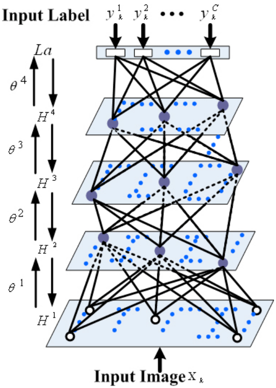
受限的玻尔兹曼机（RBM）

- RBM的性质：
  - RBM具有一个可见层，一个隐层，层内无连接。
  - 在给定可见层单元状态(输入数据)时，各隐单元的激活条件独立，反之，在给定隐单元状态时，可见层单元的激活亦条件独立。
  - 通过Gibbs采样可以得到服从RBM所表示分布的随机样本。
  - Roux和Bengio从理论上证明，只要隐单元的数目足够多，RBM能够拟合任意离散分布。



深度信念网（DBN）

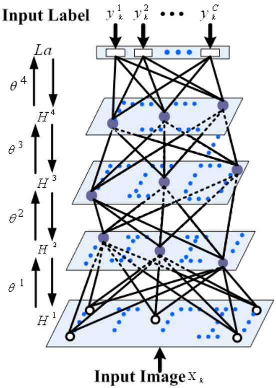
- 2006年Hinton等人提出深度信念网络（Deep Belief Nets, DBN）并给出了该模型的一个高效学习算法。
- DBN模型被视为由若干个RBM堆叠在一起，可通过由低到高逐层训练这些RBM来实现：
  - （1）底部RBM以原始输入数据训练；
  - （2）将底部RBM抽取的特征作为顶部RBM的输入训练；
  - （3）过程（1）和（2）可以重复来训练所需要的尽可能多的层数。



深度信念网（DBN）

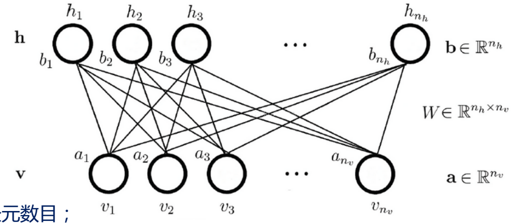
■ 优点：

- ◆ 通过CD快速训练，绕过了直接从整体上训练DBN的高复杂度，从而化简为对多个RBM的训练问题。
- ◆ 经过这种方式训练后，可以再通过传统的全局学习算法对网络进行微调，从而使模型收敛到局部最优点。
- ◆ 本质上等同于先通过逐层RBM训练将模型的参数初始化为较优的值，再通过少量的传统学习算法进一步训练。
- ◆ 实验表明，这种方式能够产生非常好的参数初始值，从而大大提升了模型的建模能力。



受限玻尔兹曼机

■ RBM网络结构



- n_v 可见层神经元数目；n_h 隐藏层神经元数目；
- $v = (v_1, v_2, \ldots, v_{n_v})^T$ 可见层的状态向量，v_i表示可见层中第i个神经元的状态；
- $h = (h_1, h_2, \ldots, h_{n_h})^T$ 隐藏层的状态向量，h_j表示隐藏层中第j个神经元的状态；
- $a = (a_1, a_2, \ldots, a_{n_v})^T \in R^{n_v}$ 可见层的偏置向量，a_i表示可见层中第i个神经元的偏置；
- $b = (b_1, b_2, \ldots, b_{n_h})^T \in R^{n_h}$ 隐藏层的偏置向量，b_j表示隐藏层中第j个神经元的偏置；
- $W = (w_{i,j}) \in R^{n_v \times n_h}$ 隐藏层与可见层之间的权值矩阵，$w_{i,j}$表示可见层第i个神经元与隐藏层第j个神经元之间的连接权重；
- 假设所有神经元均为二值，即对$\forall i, j$有$v_i, h_j \in \{0,1\}$

受限玻尔兹曼机

■ 运算过程

- ◆ 对于一个训练输入状态 $v=\{v_1, v_2, \ldots, v_{n_v}\}$，可以得到一个输出状态 $h=\{h_1, h_2, \ldots, h_{n_h}\}$满足如下的关系：

Step 1：求出$p(h_i=1|v)=sigmoid\left(\sum_j^{n_v} w_{i,j}\times v_j+b_i\right)$

Step 2：产生一个0到1之间的随机数，

如果它小于$p(h_i=1|v)$，h_i的取值就是1，否则就是0

- ◆ 需要确定的参数：$\theta=(W, b)$

受限玻尔兹曼机

- RBM的目标：
  - RBM是一种概率图模型；
  - 找到一组模型参数，使得在这组模型参数下，由RBM所表示的概率分布，应尽可能的与训练数据的概率分布相吻合；
- 即：

对训练样本集合：$S = \{\mathbf{v}^1, \mathbf{v}^2, \cdots, \mathbf{v}^{n_s}\}$ n_s 为训练样本的数目

且各个训练样本之间满足“独立同分布”的条件，则RBM训练的目标为：找到一组参数，在该组参数下，RBM所表示的概率分布与训练数据所体现的概率分布最为吻合；

即：求解一组参数θ使 $\prod_{i=1}^{n_s} P(\mathbf{v}^i)$ “发生的可能性最大”

受限玻尔兹曼机

- 即RBM的训练目标为：
  - 最大化似然：
    $$\mathcal{L}_{\theta,\mathrm{S}} = \prod_{i=1}^{n_s} P(\mathbf{v}^i)$$
  - 转化为更易计算的对数形式：
    $$\ln \mathcal{L}_{\theta,\mathrm{S}} = \ln \prod_{i=1}^{n_s} P(\mathbf{v}^i) = \sum_{i=1}^{n_s} \ln P(\mathbf{v}^i)$$
  - 接下来的问题：$P(\mathbf{v}^i)$ 是什么？

受限玻尔兹曼机

- 根据全概率公式：

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

- 问题 $P_\theta(\mathbf{v}, \mathbf{h})$ 是什么？
- 统计热力学结论：
  - 当系统和它周围的环境处于热平衡时，一个基本的结果是状态i发生的概率如下面的公式

$$\mathrm{p}_i = \frac{1}{Z} \times e^{-\frac{E_i}{k_b \times T}}$$

  - 其中，E_i表示系统在状态i时的**能量**
  - T为开尔文绝对温度， k_{B} 为Boltzmann常数，Z为与状态无关的常数。

受限玻尔兹曼机

- 什么是能量？
  - 能量是描述整个系统状态的一种测度。
  - 系统越有序或者概率分布越集中，系统的能量越小。反之，系统越无序或者概率分布越趋于均匀分布，则系统的能量越大。能量函数的最小值，对应于系统的最稳定状态。
- 按照这个原理，
  - 可以定义：在一组给定状态(v, h)下，能量函数为：

$$E_\theta(\mathrm{v},\mathrm{h}) = -\sum_{i=1}^{n_v} a_i v_i - \sum_{j=1}^{n_h} b_j h_j - \sum_{i=1}^{n_v}\sum_{j=1}^{n_h} h_j w_{j,i} v_i$$

  - 写为矩阵形式：$E_\theta(\mathrm{v},\mathrm{h}) = -\mathrm{a}^T\mathrm{v} - \mathrm{b}^T\mathrm{h} - \mathrm{h}^T W \mathrm{v}$

受限玻尔兹曼机

- 依据刚刚的 统计热力学结论：
  - 当系统和它周围的环境处于热平衡时，一个基本的结果是状态i发生的概率如下面的公式

$$p_i = \frac{1}{Z} \times e^{-\frac{E_i}{k_b \times T}}$$

  - 这里的E_i为刚刚定义的能量函数：$E(v, h)$；
  - 参数T和k_B跟求解无关，设置为1；
  - 小问题：如何构造联合概率分布的分母Z？

受限玻尔兹曼机

- 如何构造联合概率分布的分母 Z ？
  - $p(v,h)$是一个概率，其所有情况下的和应为 1；
  - 因此，可以定义其为： $Z_\theta = \sum_{\mathbf{v},\mathbf{h}} e^{-E_\theta(\mathbf{v},\mathbf{h})}$
- 由此，可以得出 状态(v,h) 发生的联合概率分布为：

$$P_\theta(\mathbf{v},\mathbf{h}) = \frac{1}{Z_\theta} e^{-E_\theta(\mathbf{v},\mathbf{h})}$$

受限玻尔兹曼机

- 基于如上的结论，可以得到：

$$P_\theta(\mathbf{v}) = \sum_{\mathbf{h}} P_\theta(\mathbf{v}, \mathbf{h}) = \frac{1}{Z_\theta} \sum_{\mathbf{h}} e^{-E_\theta(\mathbf{v},\mathbf{h})}$$

- 因此，似然函数 $\ln \mathcal{L}_{\theta,\mathrm{S}} = \ln \prod_{i=1}^{n_s} P(\mathbf{v}^i) = \sum_{i=1}^{n_s} \ln P(\mathbf{v}^i)$

  成为关于θ的函数；
- 接下来：在给定(v, h)的条件下，求解能够最大化上述函数的θ

受限玻尔兹曼机

- 梯度上升法：

$$\theta := \theta + \eta \frac{\partial \ln \mathcal{L}_{\mathrm{S}}}{\partial \theta}.$$

- 变换一下表示：

$$\begin{aligned}
\ln \mathcal{L}_{\mathrm{S}} &= \ln P(\mathbf{v}) \\
&= \ln \left(\frac{1}{Z} \sum_{\mathbf{h}} e^{-E(\mathbf{v},\mathbf{h})} \right) \\
&= \ln \sum_{\mathbf{h}} e^{-E(\mathbf{v},\mathbf{h})} - \ln Z \\
&= \ln \sum_{\mathbf{h}} e^{-E(\mathbf{v},\mathbf{h})} - \ln \sum_{\mathbf{v},\mathbf{h}} e^{-E(\mathbf{v},\mathbf{h})}
\end{aligned}$$

受限玻尔兹曼机

- 因而，

$$
\begin{aligned}
&\frac{\partial \ln P(\mathrm{v})}{\partial \theta} \\
&= \frac{\partial}{\partial \theta}\left(\ln \sum_{\mathbf{h}} e^{-E(\mathbf{v},\mathbf{h})}\right) - \frac{\partial}{\partial \theta}\left(\ln \sum_{\mathbf{v},\mathbf{h}} e^{-E(\mathbf{v},\mathbf{h})}\right) \\
&= -\frac{1}{\sum_{\mathbf{h}} e^{-E(\mathbf{v},\mathbf{h})}} \sum_{\mathbf{h}} e^{-E(\mathbf{v},\mathbf{h})} \frac{\partial E(\mathbf{v},\mathbf{h})}{\partial \theta} + \frac{1}{\sum_{\mathbf{v},\mathbf{h}} e^{-E(\mathbf{v},\mathbf{h})}} \sum_{\mathbf{v},\mathbf{h}} e^{-E(\mathbf{v},\mathbf{h})} \frac{\partial E(\mathbf{v},\mathbf{h})}{\partial \theta} \\
&= -\boxed{\sum_{\mathbf{h}} P(\mathbf{h}|\mathbf{v}) \frac{\partial E(\mathbf{v},\mathbf{h})}{\partial \theta}} + \sum_{\mathbf{v},\mathbf{h}} P(\mathbf{v},\mathbf{h}) \frac{\partial E(\mathbf{v},\mathbf{h})}{\partial \theta},
\end{aligned}
$$

$$
\begin{aligned}
\sum_{\mathbf{v},\mathbf{h}} P(\mathbf{v},\mathbf{h}) \frac{\partial E(\mathbf{v},\mathbf{h})}{\partial \theta} &= \sum_{\mathbf{v}} \sum_{\mathbf{h}} P(\mathbf{v}) P(\mathbf{h}|\mathbf{v}) \frac{\partial E(\mathbf{v},\mathbf{h})}{\partial \theta} \\
&= \sum_{\mathbf{v}} P(\mathbf{v}) \boxed{\sum_{\mathbf{h}} P(\mathbf{h}|\mathbf{v}) \frac{\partial E(\mathbf{v},\mathbf{h})}{\partial \theta}}
\end{aligned}
$$

受限玻尔兹曼机

由于θ包含三个参数(W, a, b)，分布进行拆解计算：

$$\frac{\partial \ln p(v)}{\partial w_{ij}} = \sum_h p(h|v)\left(-\frac{\partial E(v,h)}{\partial w_{ij}}\right) - \sum_{v,h} p(v,h)\left(-\frac{\partial E(v,h)}{\partial w_{ij}}\right)$$

$$= \sum_h p(h|v)h_i v_j - \sum_{v,h} p(v,h)h_i v_j$$

$$= \sum_h p(h|v)h_i v_j - \sum_v p(v)\sum_h p(h|v)h_i v_j$$

$$= \boxed{p(h_i = 1|v)v_j} - \sum_v p(v)\boxed{p(h_i = 1|v)v_j}$$

受限玻尔兹曼机

$$\sum_h p(h|v)(-\frac{\partial E(v,h)}{\partial w_{ij}})$$ **的推导：**

$$\begin{aligned}
&\sum_{\mathbf{h}} P(\mathbf{h}|\mathbf{v})\frac{\partial E(\mathbf{v},\mathbf{h})}{\partial w_{ij}} \\
&= -\sum_{\mathbf{h}} P(\mathbf{h}|\mathbf{v})h_i v_j \quad \left(\text{利用 } \frac{\partial E(\mathbf{v},\mathbf{h})}{\partial w_{ij}} = -h_i v_j\right) \\
&= -\sum_{\mathbf{h}} \prod_{k=1}^{n_h} P(h_k|\mathbf{v}) h_i v_j \quad (\text{利用 (3.32) 式}) \\
&= -\sum_{\mathbf{h}} P(h_i|\mathbf{v}) P(h_{-i}|\mathbf{v}) h_i v_j \quad (\text{独立性: } h_{-i} \text{ 的定义见 (3.26) 式}) \\
&= -\sum_{h_i} \sum_{h_{-i}} P(h_i|\mathbf{v}) P(h_{-i}|\mathbf{v}) h_i v_j \quad \left(\text{对 } \sum_{\mathbf{h}} \text{ 进行拆分}\right) \\
&= -\sum_{h_i} P(h_i|\mathbf{v}) h_i v_j \sum_{h_{-i}} P(h_{-i}|\mathbf{v}) \quad (\text{求和号下的项进行分组}) \\
&= -\sum_{h_i} P(h_i|\mathbf{v}) h_i v_j \quad \left(\because \sum_{h_{-i}} P(h_{-i}|\mathbf{v}) = 1\right) \\
&= -\left(P(h_i=0|\mathbf{v}) \cdot 0 \cdot v_j + P(h_i=1|\mathbf{v}) \cdot 1 \cdot v_j\right) \quad (\because h_i = 0 \text{ 或 } 1) \\
&= -P(h_i=1|\mathbf{v}) v_j.
\end{aligned}$$

受限玻尔兹曼机

■ 同理：

$$\frac{\partial \ln p(v)}{\partial a_j} = \sum_h p(h|v)\left(-\frac{\partial E(v,h)}{\partial a_j}\right) - \sum_{v,h} p(v,h)\left(-\frac{\partial E(v,h)}{\partial a_j}\right)$$

$$= \sum_h p(h|v) v_j - \sum_{v,h} p(v,h) v_j = \sum_h p(h|v) v_j - \sum_v p(v) \sum_h p(h|v) v_j$$

$$= v_j - \sum_v p(v) v_j$$

$$\frac{\partial \ln p(v)}{\partial b_i} = \sum_h p(h|v)\left(-\frac{\partial E(v,h)}{\partial b_i}\right) - \sum_{v,h} p(v,h)\left(-\frac{\partial E(v,h)}{\partial b_i}\right)$$

$$= \sum_h p(h|v) h_i - \sum_{v,h} p(v,h) h_i = \sum_h p(h|v) h_i - \sum_v p(v) \sum_h p(h|v) h_i$$

$$= \boxed{p(h_i = 1|v)} - \sum_v p(v) \boxed{p(h_i = 1|v)}$$

激活函数Sigmoid推导

■ 其中：$\boxed{P(h_k = 1 \mid \mathbf{v})}$

$$
\begin{aligned}
&= P(h_k = 1 \mid h_{-k}, \mathbf{v}) \\
&= \frac{P(h_k = 1, h_{-k}, \mathbf{v})}{P(h_{-k}, \mathbf{v})} \\
&= \frac{P(h_k = 1, h_{-k}, \mathbf{v})}{P(h_k = 1, h_{-k}, \mathbf{v}) + P(h_k = 0, h_{-k}, \mathbf{v})} \\
&= \frac{\frac{1}{Z} e^{-E(h_k=1, h_{-k}, \mathbf{v})}}{\frac{1}{Z} e^{-E(h_k=1, h_{-k}, \mathbf{v})} + \frac{1}{Z} e^{-E(h_k=0, h_{-k}, \mathbf{v})}} \\
&= \frac{1}{1 + e^{-E(h_k=0, h_{-k}, \mathbf{v}) + E(h_k=1, h_{-k}, \mathbf{v})}} \\
&= sigmoid(b_k + \sum_{i=1}^{n_v} w_{k,i} v_i)
\end{aligned}
$$

受限玻尔兹曼机

- 由上可知，

$$\frac{\partial \ln P(\mathrm{v})}{\partial w_{i,j}} = P(h_i = 1|\mathrm{v})v_j - \sum_{\mathbf{v}} P(\mathbf{v})P(h_i = 1|\mathbf{v})v_j$$

$$\frac{\partial \ln P(\mathrm{v})}{\partial a_i} = v_i - \sum_{\mathbf{v}} P(\mathbf{v})v_i$$

$$\frac{\partial \ln P(\mathrm{v})}{\partial b_i} = P(h_i = 1|\mathrm{v}) - \sum_{\mathbf{v}} P(\mathbf{v})P(h_i = 1|\mathbf{v})$$

- 现在的问题：$\sum_v P(v)$ ...的计算复杂度是 $O(2^{n_v+n_h})$，如何计算？

从MC到Gibbs采样

- 蒙特卡罗方法（Monte Carlo，MC）
  - 给定函数$h(x)$，求解积分$\int_a^b h(x)dx$
  - 若，无法直接求出其解析解，则可以：将积分 $\int_a^b h(x)dx$看作某个函数$f(x)$与在(a,b)上按照某个概率密度函数$p(x)$分布的均值，即：

$$\int_a^b h(x)dx = \int_a^b f(x)p(x)dx = \mathbb{E}_{p(x)}[f(x)]$$

  - 则，如果我们按照分布密度$p(x)$采集x的大量样本点$x_1, x_2, \ldots, x_n$，则我们可以利用这些样本点来逼近这个均值，即：

$$\int_a^b h(x)dx = \mathbb{E}_{p(x)}[f(x)] \approx \frac{1}{n}\sum_{i=1}^{n} f(x_i)$$

从MC到Gibbs采样

- 接下来的问题是：
  - 如何按照概率密度$p(x)$采集x的无偏样本点？
  - 且条件是： $p(x)$可能是任意的概率密度函数

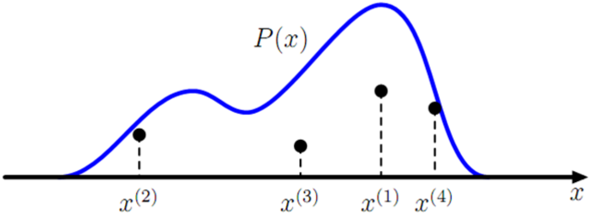


Table of contents

Hinge Loss

Hinge loss is usually used to train large margin classifiers such as Support Vector Machine (SVM).

$$L_{hinge} = \frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{K}\left[max(0, 1 - \delta(y^{(i)}, j)w^T x_i)\right]^p$$

- 其中， if $y^{(i)} = j$, $\delta(y^{(i)}, j) = 1$, otherwise $\delta(y^{(i)}, j) = -1$;
- if$p = 1$, it will be Hinge-Loss(L_1 Loss), if$p = 2$, it will be Squared Hinge-Loss(L_2 Loss).

[1]Y. Tang, Deep learning using linear support vector machines, in: ICML Workshop, 2013.

Softmax Loss

Hinge loss is usually used to train large margin classifiers such as Support Vector Machine (SVM).

$$L_{hinge} = \frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{K}\left[max(0, 1-\delta(y^{(i)}, j)w^T x_i)\right]^p$$

- 其中， if $y^{(i)} = j$, $\delta(y^{(i)}, j) = 1$, otherwise $\delta(y^{(i)}, j) = -1$;
- if$p = 1$, it will be Hinge-Loss(L_1 Loss), if$p = 2$, it will be Squared Hinge-Loss(L_2 Loss).

[1]Y. Tang, Deep learning using linear support vector machines, in: ICML Workshop, 2013.

Softmax Loss

设问题 P 共包含 N 个词，每个词的向量为 P_n
设 app A 共包含 M 个词（设定为 M=10），每个词的向量为 A_m
1. 计算：P_n 与 A_m 的相似度 sim_{mn}，然后计算：
$SIM_n = \sum_{m=1}^{M} sim_{mn}$，即得到问题 P 中的每个词 P_n 与各 app A 的相似度 SIM_n
2. 选取问题 p 的 N 个词中 SIM 最大的 2 个词的和，即
$SIM_{n_1} + SIM_{n_2}$ 作为问题 P 与 app A 之间的相似度；
3. 列出一个问题与所有 app 之间的上述相似度

马尔可夫链

- 设 X_t 表示随机变量 X 在离散时间 t 时刻的取值。
- 若该变量随时间变化的转移概率仅依赖于它的当前值，即：

$$
\begin{aligned}
&P(X_{t+1}=s_j|X_0=s_{i_0},X_1=s_{i_1},\dots,X_t=s_i)\\
=&P(X_{t+1}=s_j|X_t=s_i)
\end{aligned}
$$

- 则变量 X 称为：马尔可夫变量；
- 变量 X 所具备的这种性质，称为：马尔可夫性质；
- 具有马尔可夫性质的随机过程，称为：马尔可夫过程
- 一段时间内随机变量 X 的取值序列 $(X_0,X_1,\dots,X_m)$ 称为：马尔科夫链

马尔可夫链

- 一个马尔可夫链可以通过其对应的转移概率来定义。
- 转移概率是指：随机变量从一个时刻到下一个时刻，从状态 s_i 转移到 s_j 的概率，即：

$$Pi \to j := P_{i,j} = P(X_{t+1} = s_j)X_t = s_i).$$

- 设 $\pi_k^{(t)}$ 表示随机变量 X 在 t 时刻取值为 s_k 的概率；
- 那么，X 在 $t+1$ 时刻取值为 s_i 的概率为：

$$\begin{aligned}
\pi_k^{(t)} &= P(X_{t+1} = s_i) \\
&= \sum_k P(X_{t+1} = s_i | X_t = s_k) P(X_t = s_k) \\
&= \sum_k P_{k,i} \cdot \pi_k^{(t)}
\end{aligned}$$

马尔可夫链

设状态的数目为 n 个，则基于上文的定义：

$$(\pi_1^{(t+1)}, \ldots, \pi_n^{(t+1)}) = (\pi_1^t, \ldots, \pi_n^t) \begin{bmatrix} P_{1,1} & P_{1,2} & \ldots & P_{1,n} \\ P_{2,1} & P_{2,2} & \ldots & P_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ P_{n,1} & P_{n,2} & \ldots & P_{n,n} \end{bmatrix}$$

写成矩阵形式：

$$\pi^{(t+1)} = \pi^{(t+1)} \cdot P$$

其中，$P = (P_{i,j})_{n \times n}$ 为转移概率矩阵.

马尔可夫链

- 如果存在某个取值，从它出发转移回自身所需要的转移次数总是整数 d 的整数倍，则称这个马尔科夫过程具有周期性；
- 如果任意两个取值之间总能以非零的概率互相转移，则称该马尔科夫过程不可约。即，每一个状态都可能来自任意的其他状态。
- 如果一个马尔科夫过程既没有周期性，又不可约，则称该过程是各态遍历的；

对于各态遍历的马尔可夫过程，不论 $\pi^{(0)}$ 取何值，随着转移次数的增多，随机变量的取值分布最终都会收敛于唯一的平稳分布 π^*，即：

$$\lim_{t\to\infty} \pi^{(0)} P^t = \pi^*$$

并且，这个平稳分布 π^* 满足：

$$\pi^* P = \pi^*$$

马尔可夫链

如果我们能构造一个转移矩阵为 P 的马氏链，使得该马氏链的平稳分布恰好是 $\pi(x)$，那么我们从任何一个初始状态 X_0 出发沿着马氏链转移，得到一个转移序列：

$$X_0, X_1, X_2, \ldots, X_n, X_{n+1}, \ldots$$

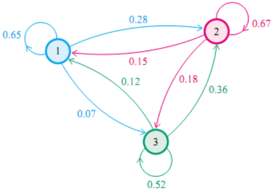
如果马氏链在第 n 步已经收敛了，于是我们就得到了 $\pi(x)$ 的样本：

$$X_n, X_{n+1}, \ldots$$

现在的关键是：
如果我们想从分布$\pi(x)$上采集样本，我们应该如何构造状态转移矩阵？

马尔科夫链的收敛特性

	State	子代 1	子代 2	子代 3
父代	1	0.65	0.28	0.07
	2	0.15	0.67	0.18
	3	0.12	0.36	0.52



第n代人	下层	中层	上层
0	0.210	0.680	0.110
1	0.252	0.554	0.194
2	0.270	0.512	0.218
3	0.278	0.497	0.225
4	0.282	0.490	0.226
5	0.285	0.489	0.225
6	0.286	0.489	0.225
7	0.286	0.489	0.225
8	0.289	0.488	0.225
9	0.286	0.489	0.225
10	0.286	0.489	0.225
...	...	...	...

第n代人	下层	中层	上层
0	0.75	0.15	0.1
1	0.522	0.347	0.132
2	0.407	0.426	0.167
3	0.349	0.459	0.192
4	0.318	0.475	0.207
5	0.303	0.482	0.215
6	0.295	0.485	0.220
7	0.291	0.487	0.222
8	0.289	0.488	0.225
9	0.286	0.489	0.225
10	0.286	0.489	0.225
...	...	...	...

Metropolis 采样算法

- 1953 年，Metropolis 首次提出了基于马尔科夫链的蒙特卡罗方法，即 Metropolis 算法；
- 论文被收录在"统计学中的重大突破"中，被遴选为二十世纪的十个最重要的算法之一；
- MCMC 算法是 Metropolis 算法的一个改进变种，即常用的 Metropolis-Hastings 算法;

细致平稳条件 (detailed balance condition)：如果非周期马尔科夫链的转移矩阵 P 和分布 $\pi(x)$ 满足：

$$\pi(i)P_{ij} = \pi(j)P_{ji} \text{for all i, j}$$

则 $\pi(x)$ 是马尔可夫链的平稳分布。
数学证明如下：

$$\sum_{i=1}^{\infty} \pi(i)P_{ij} = \sum_{i=1}^{\infty} \pi(j)P_{ji} = \pi(j)\sum_{i=1}^{\infty} P_{ji} = \pi(j) \Rightarrow \pi P = \pi$$

Metropolis 采样算法

那么，如果我们想从分布$\pi(x)$上采集样本，我们应该如何构造状态转移矩阵呢？
假设已经有一个状态转移矩阵 Q，其基本元素 $q(i,j)$ 表示从状态 i 转移到状态 j 的概率，且假设 Q 不满足细致平衡条件。(这当然是大大大概率事件。)
我们的思路是：对状态转移矩阵Q进行改造，使其满足细致平衡条件。
引入一个新的概率 $\alpha(i,j)$，凑出细致平衡条件：

$$\pi(i)q(i,j)\alpha(i,j) = \pi(j)q(j,i)\alpha(j,i)$$

但是，什么样的 $\alpha(i,j)$ 才能满足上述条件呢？很简单，取：

$$\alpha(i,j) = \pi(j)q(j,i), \alpha(j,i) = \pi(i)q(i,j)$$

其中，$\alpha(i,j)$ 称为“接受概率”.

Metropolis 采样算法

所以，我们可以得到 Metropolis 采样算法：
假设已经有一个状态转移矩阵 Q，其基本元素 $q(i,j)$ 表示从状态 i 转移到状态 j 的概率，则：

1. 初始化马尔科夫链初始状态 $X_0 = x_0$;
2. 对 $t = 0, 1, 2, \ldots,$ 循环以下过程进行采样：
    1. 设第 t 时刻马尔科夫链状态为 $X_t = x_t$
    2. 按照已经定义的状态转移条件计算出 $x_{t+1} = x_t * q(x|x_t)$
    3. 从均匀分布采样 $u \sim Uniform[0,1]$
    4. 如果 $u < \alpha(x_t, x_{t+1}) = \pi(x_{t+1})q(x_t|x_{t+1})$，则接受转移 $x_t \to x_{t+1}$，即 $X_{t+1} = x_{t+1}$;
    5. 否则不接受转移，即 $X_{t+1} = x_t$

Metropolis-Hastings 采样算法

虽然，上述算法已经很完美了，但是，它有个效率问题：
　当接受概率 $\alpha(i,j)$ 比较小时，马尔科夫链的收敛速度比较慢！
那么，有什么办法可以解决这个问题呢？
分析一下：假设 $\alpha(i,j)=0.1, \alpha(j,i)=0.2$ 时满足细致平衡条件，即这时有：

$$\pi(i)q(i,j)0.1=\pi(j)q(j,i)0.2$$

此时，均匀分布采样结果 u 满足 $u<\alpha(i,j)$ 的概率较小；
但如果将上述等式两边同时等比例扩大，即：

$$\pi(i)q(i,j)0.1*5=\pi(j)q(j,i)0.2*5, \text{即:}\alpha(i,j)=5*\alpha(i,j)$$

则，均匀分布采样结果 u 满足 $u<\alpha(i,j)$ 的概率就被放大了！而细致平衡条件没有改变！

Metropolis-Hastings 采样算法

所以，我们可以把细致平衡条件中的 $\alpha(i,j)$ 和 $\alpha(j,i)$ 同时放大，最大可以把两者中最大的一个放大到 1，即取：

$$\alpha(i,j) = min\{1, \frac{\pi(j)q(j,i)}{\pi(i)q(i,j)}\}$$

这是，细致平衡条件仍然成立！证明如下：

$$\begin{aligned}
\pi(i)P_{ij} &= \pi(i)\alpha(i,j)q(i,j) \\
&= \pi(i)min\{1, \frac{\pi(j)q(j,i)}{\pi(i)q(i,j)}\}qi,j \\
&= min\{\pi(j)q(j,i), \pi(i)q(i,j)\} \\
&= \pi(j)min\{1, \frac{\pi(i)q(i,j)}{\pi(j)q(j,i)}\}qj,i \\
&= \pi(j)\alpha(j,i)q(j,i) \\
&= \pi(j)P_{ji}
\end{aligned}$$

Metropolis-Hastings 采样算法

所以，我们可以得到 Metropolis-Hastings 采样算法：

1. 初始化马尔科夫链初始状态 $X_0 = x_0$;
2. 对 $t = 0, 1, 2, \ldots,$ 循环以下过程进行采样：
    1. 设第 t 时刻马尔科夫链状态为 $X_t = x_t$
    2. 按照已经定义的状态转移条件计算出 $x_{t+1} = x_t * q(x|x_t)$
    3. 从均匀分布采样 $u \sim Uniform[0, 1]$
    4. 如果 $u < \alpha(x_t, x_{t+1}) = min\{1, \frac{\pi(x_{t+1})q(x_t|x_{t+1})}{\pi(x_t)q(x_{t+1}|x_t)}\}$，则接受转移 $x_t \rightarrow x_{t+1}$，即 $X_{t+1} = x_{t+1}$;
    5. 否则不接受转移，即 $X_{t+1} = x_t$

Gibbs 采样算法

对于高维的情况下，计算会变得繁琐：

$$\alpha((x_1, y_1), (x_2, y_2)) = min\{1, \frac{\pi(x_2, y_2)q((x_2, y_2), (x_1, y_1))}{\pi(x_1, y_1)q((x_1, y_1), (x_2, y_2))}\}$$

那么，有没有更加简便的计算方法呢？
我们注意到：假设存在概率分布 $p(x, y)$，则状态 (x_1, y_1) 与状态 (x_1, y_2) 之前存在如下转移概率：

$$\pi(x_1, y_1)\pi(y_2|x_1) = \pi(x_1, y_2)\pi(y_1|x_1)$$

这是因为：

$$\pi(x_1, y_1)\pi(y_2|x_1) = \pi(x_1)\pi(y_1|x_1)\pi(y_2|x_1)$$
$$\pi(x_1, y_2)\pi(y_1|x_1) = \pi(x_1)\pi(y_2|x_1)\pi(y_1|x_1)$$

所以，如果使用边缘概率分布 $\pi(y|x_1)$ 作为状态 (x_1, y_1) 与状态 (x_1, y_1) 之间的转移概率，那么状态 (x_1, y_i) 与状态 (x_1, y_j) 之前满足细致平衡条件！

Gibbs 采样算法

因为任意两种状态 (x_i, y_i) 与状态 (x_j, y_j) 之间的转换，可以看作：先完成 $(x_i, y_i) \to (x_i, y_j)$，再完成 $(x_i, y_j) \to (x_j, y_j)$

因此可以定义：

1. $Q((x_i, y_i) \to (x_i, y_j))$ 之间的转换概率为：$\pi(y|x_i)$；
2. $Q((x_i, y_j) \to (x_j, y_j))$ 之间的转换概率为：$\pi(x|y_j)$；
3. 且禁止两种状态 (x_i, y_i) 与状态 (x_j, y_j) 之间的转换，即：$Q((x_i, y_i) \to (x_j, y_j))$ 之间的转换概率为：0；

则得到任意状态 $X = (x_i, y_i)$ 与 $Y = (x_j, y_j)$ 之间的细致平衡条件：

$$\pi(X)Q(X \to Y) = \pi(Y)Q(Y \to X)$$

于是这个二维空间上的马尔可夫链将收敛到平稳分布 $\pi(x, y)$.

Gibbs 采样算法

所以得到 Gibbs 采样算法：

1. 初始化马尔科夫链初始状态 $X_0 = x_0, Y_0 = y_0$;
2. 对 $t = 0, 1, 2, \dots,$ 循环以下过程进行采样：
   1. 设第 t 时刻马尔科夫链状态为 $X_t = x_t, Y_t = y_t$
   2. 按照已经定义的状态转移条件计算出 $y_{t+1} = y_t * q(y|x_t)$
   3. 按照已经定义的状态转移条件计算出 $x_{t+1} = x_t * q(x|y_{t+1})$

同理，可以扩展到 N 维的情况。

Gibbs采样

Algorithm 8 n维Gibbs Sampling 算法

1: 随机初始化$\{x_i : i = 1, \cdots, n\}$

2: 对$t = 0, 1, 2, \cdots$ 循环采样

1. $x_1^{(t+1)} \sim p(x_1 | x_2^{(t)}, x_3^{(t)}, \cdots, x_n^{(t)})$

2. $x_2^{(t+1)} \sim p(x_2 | x_1^{(t+1)}, x_3^{(t)}, \cdots, x_n^{(t)})$

3. $\cdots$

4. $x_j^{(t+1)} \sim p(x_j | x_1^{(t+1)}, \cdots, x_{j-1}^{(t+1)}, x_{j+1}^{(t)}, \cdots, x_n^{(t)})$

5. $\cdots$

6. $x_n^{(t+1)} \sim p(x_n | x_1^{(t+1)}, x_2^{t}, \cdots, x_{n-1}^{(t+1)})$

对比散度算法

- 对比散度（Contrastive Divergence）
  - 将MCMC的状态一训练样本作为起点
- 基本思想

对于任意的样本数据v，将该样本数据设置为起始值，执行k步Gibbs采样：

- 利用 $P(\mathbf{h}|\mathbf{v}^{(t-1)})$ 采样出 $\mathbf{h}^{(t-1)}$;
- 利用 $P(\mathbf{v}|\mathbf{h}^{(t-1)})$ 采样出 $\mathbf{v}^{(t)}$,

对比散度算法

- 利用k步Gibbs采样后得到的样本数据 $v^{(k)}$ 对期望项进行估计：

$$\frac{\partial \ln P(\mathbf{v})}{\partial w_{i,j}} = P(h_i = 1|\mathbf{v})v_j - \sum_{\mathbf{v}} P(\mathbf{v})P(h_i = 1|\mathbf{v})v_j,$$

$$\frac{\partial \ln P(\mathbf{v})}{\partial w_{i,j}} \approx P(h_i = 1|\mathbf{v}^{(0)})v_j^{(0)} - P(h_i = 1|\mathbf{v}^{(k)})v_j^{(k)},$$

$$\frac{\partial \ln P(\mathbf{v})}{\partial a_i} = v_i - \sum_{\mathbf{v}} P(\mathbf{v})v_i,$$

$$\frac{\partial \ln P(\mathbf{v})}{\partial a_i} \approx v_i^{(0)} - v_i^{(k)},$$

$$\frac{\partial \ln P(\mathbf{v})}{\partial b_i} = P(h_i = 1|\mathbf{v}) - \sum_{\mathbf{v}} P(\mathbf{v})P(h_i = 1|\mathbf{v}),$$

$$\frac{\partial \ln P(\mathbf{v})}{\partial b_i} \approx P(h_i = 1|\mathbf{v}^{(0)}) - P(h_i = 1|\mathbf{v}^{(k)}).$$

受限玻尔兹曼机

Algorithm 1. k-step contrastive divergence

Input: RBM $(V_1, \ldots, V_m, H_1, \ldots, H_n)$, training batch S

Output: gradient approximation Δw_{ij}, Δb_j and Δc_i for $i = 1, \ldots, n$, $j = 1, \ldots, m$

init $\Delta w_{ij} = \Delta b_j = \Delta c_i = 0$ for $i = 1, \ldots, n$, $j = 1, \ldots, m$

forall the $v \in S$ **do**

- $v^{(0)} \leftarrow v$
- **for** $t = 0, \ldots, k-1$ **do**
  - **for** $i = 1, \ldots, n$ **do** sample $h_i^{(t)} \sim p(h_i \mid v^{(t)})$
  - **for** $j = 1, \ldots, m$ **do** sample $v_j^{(t+1)} \sim p(v_j \mid h^{(t)})$
- **for** $i = 1, \ldots, n$, $j = 1, \ldots, m$ **do**
  - $\Delta w_{ij} \leftarrow \Delta w_{ij} + p(H_i = 1 \mid v^{(0)}) \cdot v_j^{(0)} - p(H_i = 1 \mid v^{(k)}) \cdot v_j^{(k)}$
  - $\Delta b_j \leftarrow \Delta b_j + v_j^{(0)} - v_j^{(k)}$
  - $\Delta c_i \leftarrow \Delta c_i + p(H_i = 1 \mid v^{(0)}) - p(H_i = 1 \mid v^{(k)})$

Thanks.