模式识别与机器学习

06 概率图模型

大连理工大学 人工智能学院
School of Artificial Intelligence, Dalian University of Technology

目录

- 概率图模型简介
- 贝叶斯网
- 隐马尔可夫模型

简介

- 机器学习最重要的任务,是根据一些已观察到的证据（observation）来对感兴趣的未知变量进行估计和推测.
- 生成式模型考虑联合分布P(Y,R,O),判别式模型考虑条件分布P(Y,R|O).
- 概率图模型是一类用图来表达变量关联关系的概率模型.
- 若变量间存在显式的因果关系,常使用贝叶斯网.
- 若变量间存在相关性但难以获取显式的因果关系,常使用马尔可夫网.

贝叶斯网

贝叶斯网

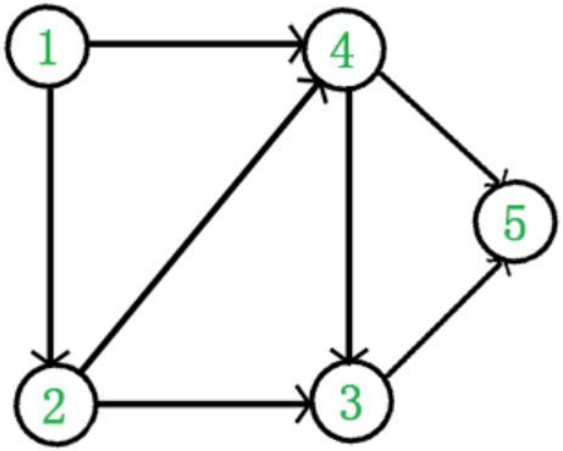
- 贝叶斯网（Bayesian network）亦称“信念网”（brief network），它借助有向无环图（Directed Acyclic Graph, DAG）来刻画属性间的依赖关系，并使用条件概率表（Conditional Probability Table, CPT）来表述属性的联合概率分布。

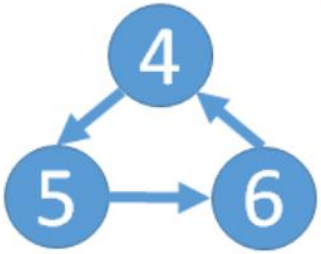
DAG

有向无环图指的是一个无回路的有向图

如果有一个非有向无环图，且A点出发向B经C可回到A，形成一个环

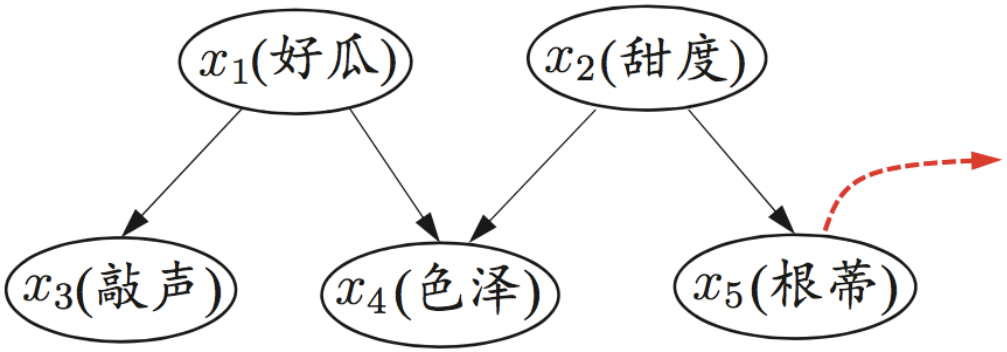
应用：操作系统、区块链等





贝叶斯网

- 贝叶斯网（Bayesian network）亦称“信念网”（brief network），它借助有向无环图（Directed Acyclic Graph, DAG）来刻画属性间的依赖关系，并使用条件概率表（Conditional Probability Table, CPT）来表述属性的联合概率分布。

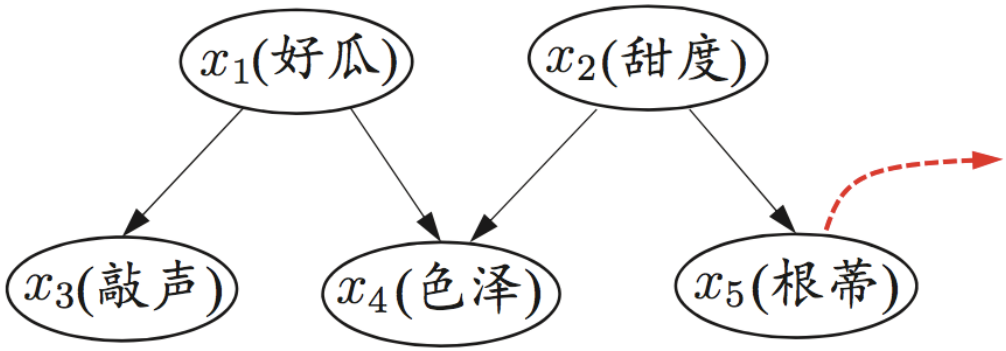


根蒂

甜度	硬挺	蜷缩
高	0.1	0.9
低	0.7	0.3

- 从网络图结构可以看出 -> “色泽”直接依赖于“好瓜”和“甜度”
- 从条件概率表可以得到 -> “根蒂”对“甜度”的量化依赖关系 $P(\text{根蒂}=\text{硬挺}|\text{甜度}=\text{高})=0.1$

贝叶斯网



根蒂

甜度	硬挺	蜷缩
高	0.1	0.9
低	0.7	0.3

- 贝叶斯网B由结构和参数组成：$B = \langle G, \Theta \rangle$
  - G：有向无环图，每个节点对应一个属性，边表示属性间的依赖关系
  - Θ：包含每个属性的条件概率表 $\theta_{x_i|\pi_i}$
    - 假设属性x_i在G中的父节点集为π_i，则

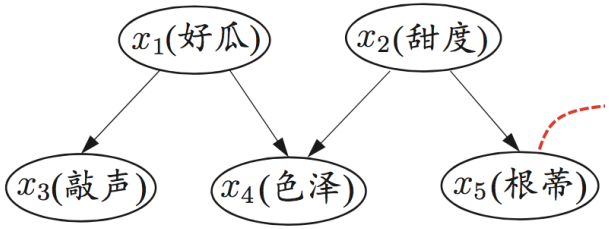
$$\theta_{x_i|\pi_i} = P_B(x_i \mid \pi_i)$$

贝叶斯网：结构

- 贝叶斯网有效地表达了属性间的**条件独立性**。给定父结集，贝叶斯网假设每个属性与他的非后裔属性独立。

- $B = \langle G, \Theta \rangle$ 将属性 $x_1, x_2, \cdots, x_d$ 的联合概率分布定义为

$$P_B(x_1, x_2, \cdots, x_d) = \prod_{i=1}^{d} P_B(x_i \mid \pi_i) = \prod_{i=1}^{d} \theta_{x_i|\pi_i}$$



右图的联合概率分布定义为：

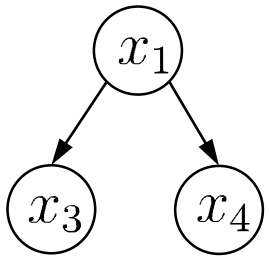
$$P(x_1, x_2, x_3, x_4, x_5) = P(x_1)P(x_2)P(x_3 \mid x_1)P(x_4 \mid x_1, x_2)P(x_5 \mid x_2)$$

$$P(x_1 x_2 x_3 x_4 x_5) = P(x_5 \mid x_1 x_2 x_3 x_4) P(x_1 x_2 x_3 x_4)$$
$$= P(x_5 \mid x_1 x_2 x_3 x_4) P(x_4 \mid x_1 x_2 x_3) P(x_1 x_2 x_3)$$
$$= P(x_5 \mid x_1 x_2 x_3 x_4) P(x_4 \mid x_1 x_2 x_3) P(x_3 \mid x_1 x_2) P(x_1 x_2)$$
$$= P(x_5 \mid x_1 x_2 x_3 x_4) P(x_4 \mid x_1 x_2 x_3) P(x_3 \mid x_1 x_2) P(x_2 \mid x_1) P(x_1)$$

- 显然，x_3 和 x_4 在给定 x_1 的取值时独立，x_4 和 x_5 在给定 x_2 的取值时独立，记为 $x_3 \perp x_4 \mid x_1$ 和 $x_4 \perp x_5 \mid x_2$

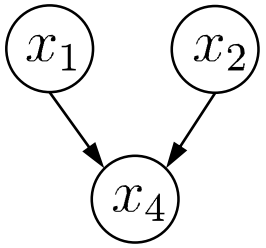
贝叶斯网：结构

- 贝叶斯网中三个变量之间的典型依赖关系：



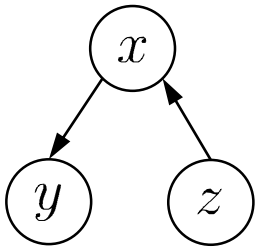
同父结构

$x_3 \perp x_4 \mid x_1$



V型结构

冲撞结构



顺序结构

$y \perp z \mid x$

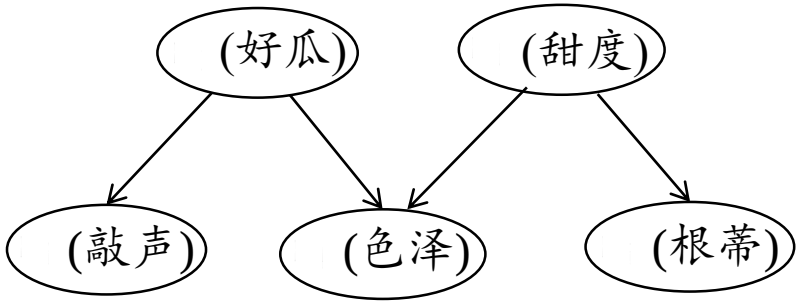
- 给定x_4的取值， x_1与x_2必不独立
- 若x_4的取值完全未知， x_1与x_2则是相互独立的

- 边际独立性，记为$x_1 \perp\!\!\!\perp x_2$

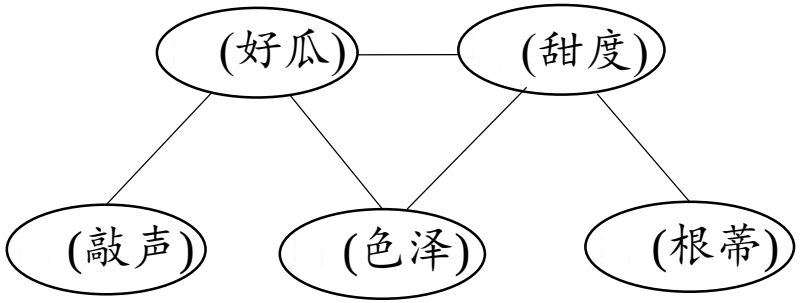
贝叶斯网：结构

- 分析有向图中变量间的条件独立性，可使用“有向分离”把有向图变为无向图
  - V型结构父结点相连
  - 有向边变成无向边

  由此产生的图称为道德图



- 若将x_1去掉，x_3和x_4分属两个连通分支，则称x_3和x_4被x_1有向分离，$x_3 \perp x_4 \mid x_1$ 成立。
- 从右图中找出所有条件独立关系。



隐马尔可夫模型

马尔可夫模型

马尔可夫模型 → 隐马尔可夫模型

- 将随机变量作为结点，若两个随机变量相关或者不独立，则将二者连接一条边；若给定若干随机变量，则形成一个有向图，即构成一个网络。
- 如果该网络是DAG图（有向无环图），则这个网络称为贝叶斯网络。

↓

- 如果这个图退化成线性链的方式，则得到马尔可夫模型；

  因为每个结点都是随机变量，将其看成各个时刻(或空间)的相关变化，以随机过程的视角，则可以看成是马尔可夫过程

- 最简单的马尔科夫过程是一阶模型，它的状态选择仅与前一个状态有关

马尔可夫模型

(1) 关键概念——状态空间 $S = \{s_1, s_2, \ldots, s_N\}$

所有状态可能取值的集合，被称为“状态空间”

(2) 关键概念——状态序列 $q_1, q_2, \ldots, q_k, \ldots \quad q_i \in S$

马尔可夫链从一个状态转移到另一个状态，构成一个状态序列

(3) 关键概念——状态转移概率

有N个状态的一阶马尔科夫模型，共有N^2个状态转移可能性，因为任何一个状态都有可能是下一个转移状态。每一个状态转移都有一个概率值，称为**状态转移概率**——即**从一个状态转移到另一个状态的概率**。

马尔科夫链性质：$P(q_k \mid q_1, q_2, \ldots, q_{k-1}) = P(q_k \mid q_{k-1})$

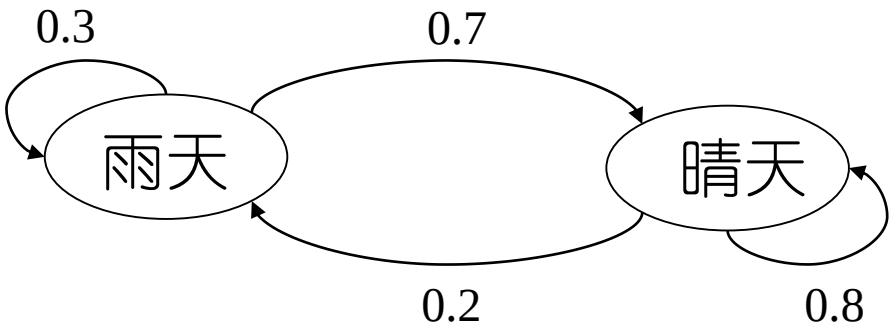
当前状态**仅与前一个状态相关有关**

马尔可夫模型

(4) 马尔科夫模型: $M=(A,\pi)$ $A=(a_{ij})$, $\pi=(\pi_i)$

转移概率矩阵 $a_{ij}=P(s_i \mid s_j)$

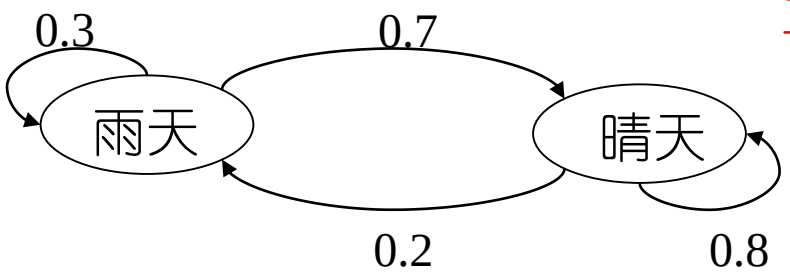
初始概率向量 $\pi_i=P(s_i)$



$$A=\begin{array}{c} \begin{matrix} \text{晴} & \text{雨} \end{matrix} \\ \begin{bmatrix} 0.8 & 0.2 \\ 0.7 & 0.3 \end{bmatrix} \end{array} \begin{matrix} \text{晴} \\ \text{雨} \end{matrix} \qquad \pi=\begin{bmatrix} 0.6 \\ 0.4 \end{bmatrix}$$

- 两个状态: 晴天、雨天.
- 转移概率: P(‘雨天’|‘雨天’)=0.3 , P(‘晴天’|‘雨天’)=0.7 ,
  P(‘雨天’|‘晴天’)=0.2, P(‘晴天’|‘晴天’)=0.8
- 初始概率: P(‘雨天’)=0.4 , P(‘晴天’)=0.6 .

马尔可夫模型



- 两个状态: 晴天、雨天.
- 转移概率: P(‘雨天’|‘雨天’)=0.3 ,
  P(‘晴天’|‘雨天’)=0.7 ,
  P(‘雨天’|‘晴天’)=0.2,
  P(‘晴天’|‘晴天’)=0.8
- 初始概率: P(‘雨天’)=0.4 ,
  P(‘晴天’)=0.6 .

马尔科夫链性质： $q_i \in S \quad S=\{s_1, s_2, \ldots, s_N\}$

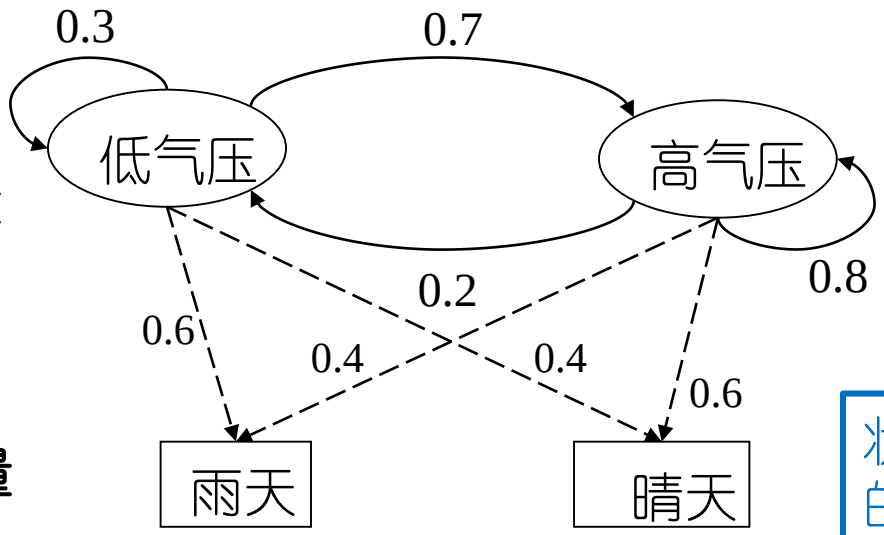
$$
\begin{aligned}
&P(q_1, q_2, \ldots, q_{k-1}, q_k) = \\
&P(q_k \mid q_1, q_2, \ldots, q_{k-1}) P(q_1, q_2, \ldots, q_{k-1}) \\
&= P(q_k \mid q_{k-1}) P(q_1, q_2, \ldots, q_{k-1}) = \ldots \\
&= P(q_k \mid q_{k-1}) P(q_{k-1} \mid q_{k-2}) P(q_1)
\end{aligned}
$$

状态序列 {‘晴天’,‘晴天’, ‘雨天’, ‘雨天’} 出现的概率 ？

$$
\begin{aligned}
&s_1 = \text{晴天}, s_2 = \text{雨天} \\
&q_1 = s_1, q_2 = s_1, q_3 = s_2, q_4 = s_2
\end{aligned}
$$

P({‘晴天’,‘晴天’, ‘雨天’, ‘雨天’}) = P(‘雨天’|‘雨天’)P (‘雨天’|‘晴天’) P (‘晴天’|‘晴天’) P (‘晴天’) = 0.3*0.2*0.8*0.6

隐马尔可夫模型



隐变量
状态变量

观测变量

- 状态空间

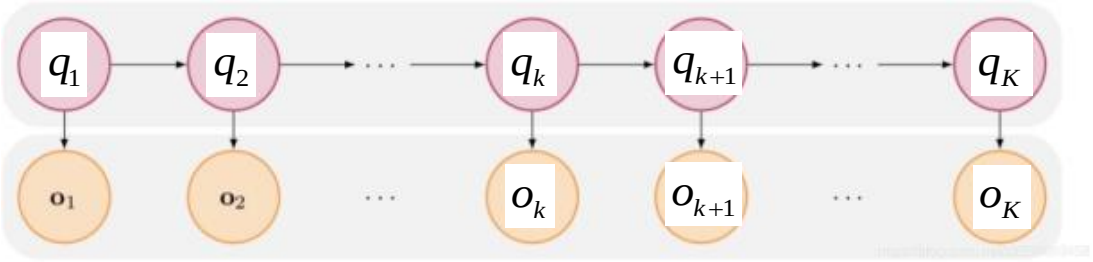
$$S=\{s_1,s_2,\ldots,s_N\}$$

- 观察变量集合

$$V=\{v_1,v_2,\ldots,v_M\}$$

状态不直接可见，但受状态影响的某些变量（观测变量）可见

- 由一个隐藏的马尔可夫链生成不可观测的状态序列，状态序列中的每个状态生成一个观测，从而产生观测序列



隐

- 状态序列 $Q=q_1\ q_2\ \ldots\ q_K$
- 观测序列 $O=o_1\ o_2\ \ldots\ o_K$

隐马尔可夫模型

隐马尔科夫模型：$M = (A, B, \pi)$

$A=[aij]_{N\times N}$, $B=[bi(vm)]_{N\times M}$, $\pi=[\pi_i]$

状态转移概率矩阵 $a_{ij} = P(s_i \mid s_j)$

观测概率矩阵 $b_i(v_m) = P(v_m \mid s_i)$

初始状态概率向量 $\pi_i = P(s_i)$

- 齐次马尔可夫性假设，即假设隐藏的马尔可夫链在任意时刻的状态只依赖于其前一时刻的状态，与其他时刻的状态及观测无关，也与时刻无关；

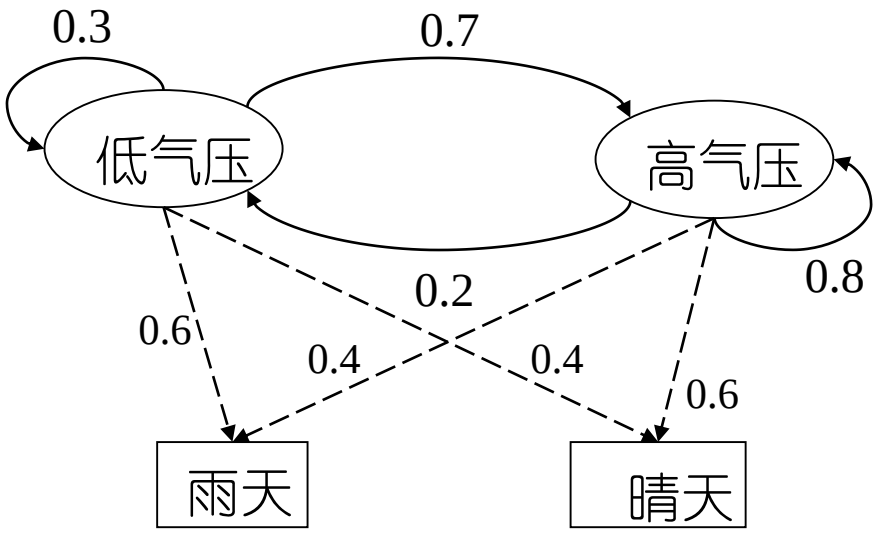
$$P(q_k \mid q_{k-1}, q_{k-2}, \ldots, q_K, o_1, o_k) = P(q_k \mid q_{k-1})$$

- 观测独立性假设，即假设任意时刻的观测只依赖于该时刻的马尔可夫链的状态，与其他观测及状态无关

$$P(o_k \mid o_{k-1}, \ldots, o_1, q_K, \ldots q_k, q_{k-1}, \ldots, q_1) = P(o_k \mid q_k)$$

隐马尔可夫模型

隐马尔科夫模型：$M = (A, B, \pi)$



- 两个状态（隐）：低气压、高气压.
- 两个观测值：雨天、晴天

$$A = \begin{bmatrix} 0.3 & 0.7 \\ 0.2 & 0.8 \end{bmatrix} \begin{matrix} \text{低} \\ \text{高} \end{matrix} \qquad B = \begin{bmatrix} 0.6 & 0.4 \\ 0.7 & 0.3 \end{bmatrix} \begin{matrix} \text{低} \\ \text{高} \end{matrix}$$

$$\pi = \begin{bmatrix} 0.4 \\ 0.6 \end{bmatrix} \begin{matrix} \text{低} \\ \text{高} \end{matrix}$$

示例：计算观测序列{‘晴天’, ‘雨天’}出现的概率？

P({‘晴’,’雨’}) = P({‘晴’,’雨’} , {‘低’,’低’}) + P({‘晴’,’雨’} , {‘低’,’高’}) +
P({‘晴’,’雨’} , {‘高’,’低’}) + P({‘晴’,’雨’} , {‘高’,’高’})

P ({‘晴’,’雨’} , {‘低’,’低’}) = P ({‘晴’,’雨’} | {‘低’,’低’}) P({‘低’,’低’})
= P(‘晴’|’低’)P(‘雨’|’低’) P(低 | 低) P(‘低’)
= 0.4*0.6*0.3*0.4

隐马尔可夫模型

三个基本问题（Given To-Do）

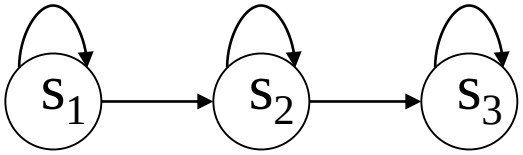
1.评价问题： 给定HMM模型 $M=(A, B, \pi)$和观测序列$O=o_1 o_2 \ldots o_K$
（Evaluation） 计算模型M产生此观测序列O的概率

2.解码问题： 给定HMM模型 $M=(A, B, \pi)$和观测序列$O=o_1 o_2 \ldots o_K$
（Decoding） 计算产生此观测序列O的最匹配的状态序列Q

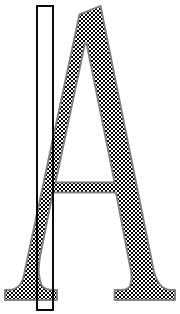
3.学习问题： 已知观测序列$O=o_1 o_2 \ldots o_K$，估计模型$M=(A, B, \pi)$
（Learning） 参数，使得在该模型下观测序列概率最大，
即用极大似然估计的方法估计参数

Character recognition with HMM example.

- The structure of hidden states is chosen.（给定隐状态结构）

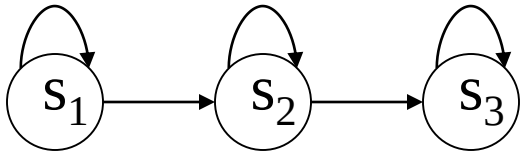


- Observations are feature vectors extracted from vertical slices.



Exercise: character recognition with HMM(1)

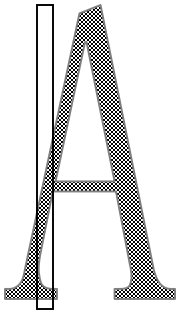
- The structure of hidden states:



- Observation = number of islands（“岛”数） in the vertical slice.
- HMM for character ‘A’ :

Transition probabilities: $\{a_{ij}\} = \begin{pmatrix} .8 & .2 & 0 \\ 0 & .8 & .2 \\ 0 & 0 & 1 \end{pmatrix}$

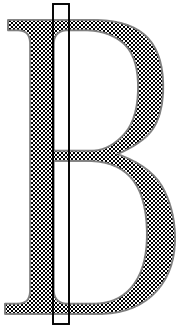
Observation probabilities: $\{b_{jk}\} = \begin{pmatrix} .9 & .1 & 0 \\ .1 & .8 & .1 \\ .9 & .1 & 0 \end{pmatrix}$



- HMM for character ‘B’ :

Transition probabilities: $\{a_{ij}\} = \begin{pmatrix} .8 & .2 & 0 \\ 0 & .8 & .2 \\ 0 & 0 & 1 \end{pmatrix}$

Observation probabilities: $\{b_{jk}\} = \begin{pmatrix} .9 & .1 & 0 \\ 0 & .2 & .8 \\ .6 & .4 & 0 \end{pmatrix}$



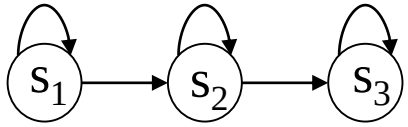
Exercise: character recognition with HMM(2)

- 问题：若观测序列为 { 1, 3, 2, 1}　（“岛”数）
  评估模型与观测序列之间的匹配程度。

  What HMM is more likely to generate this observation sequence , HMM for ‘A’ or HMM for ‘B’ ?

Exercise: character recognition with HMM(3)

列出生成观测序列的所有可能隐状态序列：



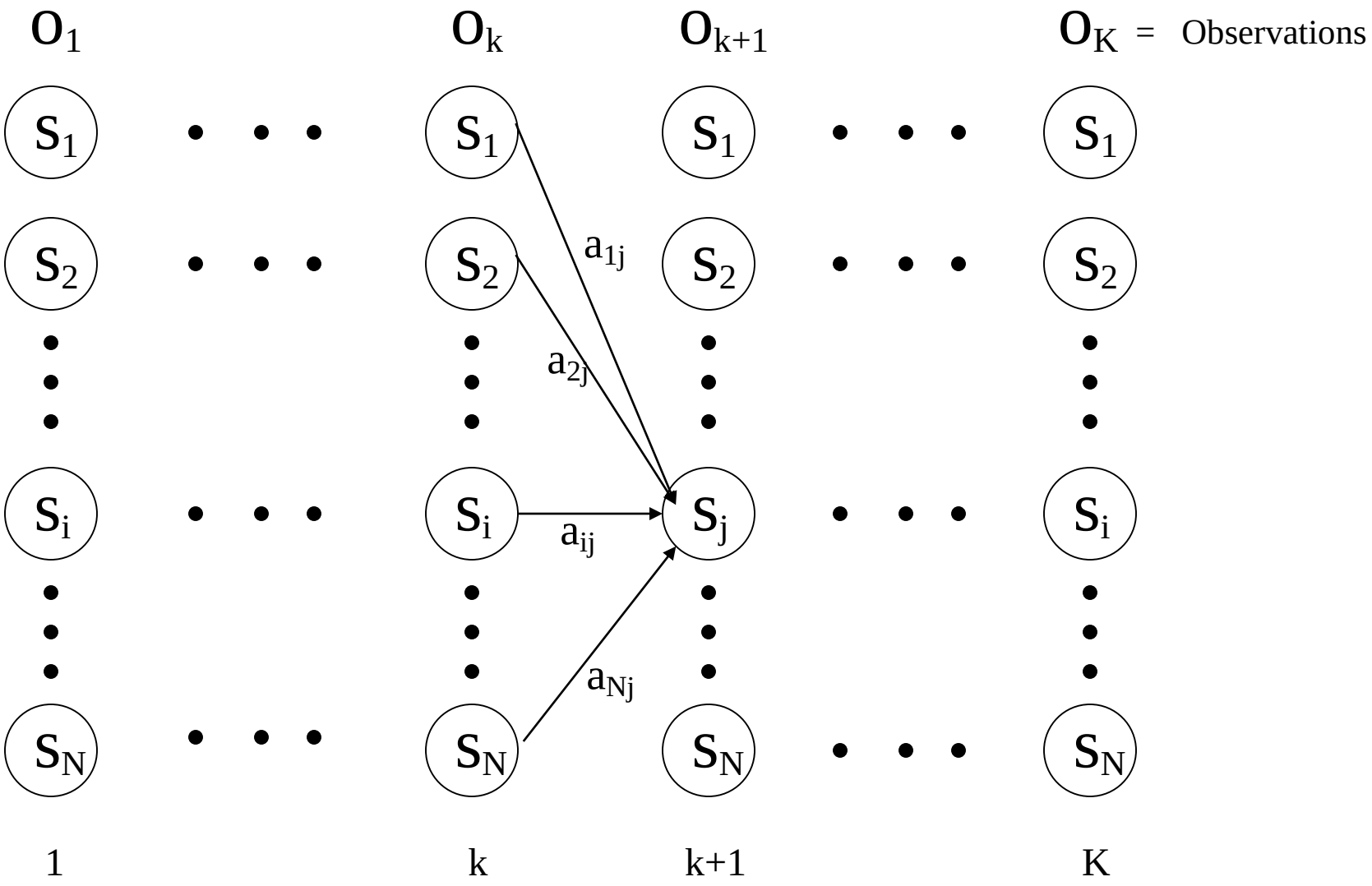
- HMM for character ‘A’:

Hidden state sequence	Transition probabilities		Observation probabilities	
$s_1 \rightarrow s_2 \rightarrow s_2 \rightarrow s_2$	$.2 * .8 * .8$	$*$	$.9 * 0.1 * .8 * .1$	$= 0$
$s_1 \rightarrow s_2 \rightarrow s_2 \rightarrow s_3$	$.2 * .8 * .2$	$*$	$.9 * .1 * .8 * .9$	$= 0.0020736$
$s_1 \rightarrow s_2 \rightarrow s_3 \rightarrow s_3$	$.2 * .2 * 1$	$*$	$.9 * .1 * .1 * .9$	$= 0.000324$
			Total	$= t1$

- HMM for character ‘B’:

Hidden state sequence	Transition probabilities		Observation probabilities	
$s_1 \rightarrow s_2 \rightarrow s_2 \rightarrow s_2$	$.2 * .8 * .8$	$*$	$.9 * .8 * .2 * 0$	$= 0$
$s_1 \rightarrow s_2 \rightarrow s_2 \rightarrow s_3$	$.2 * .8 * .2$	$*$	$.9 * .8 * .2 * .6$	$= 0.0027648$
$s_1 \rightarrow s_2 \rightarrow s_3 \rightarrow s_3$	$.2 * .2 * 1$	$*$	$.9 * .8 * .4 * .6$	$= 0.006912$
			Total	$= t2$

Trellis representation of a HMM



Evaluation Problem.

•**Evaluation problem.** Given the HMM $M=(A, B, \pi)$ and the observation sequence $O=o_1 o_2 \ldots o_K$, calculate the probability that model M has generated sequence O.

• Trying to find probability of observations $O=o_1 o_2 \ldots o_K$ by means of considering all hidden state sequences (as was done in example) is impractical:

N^K hidden state sequences - exponential complexity.

• Use **Forward-Backward HMM algorithms** for efficient calculations.

• Define the forward variable $\alpha_k(i)$ as the joint probability of the partial observation sequence $o_1 o_2 \ldots o_k$ and that the hidden state at time k is s_i : $\alpha_k(i)= P(o_1 o_2 \ldots o_k, q_k= s_i)$

Forward recursion for HMM

- Initialization:

$$\alpha_1(i) = P(o_1, q_1 = s_i) = \pi_i \, b_i(o_1), \quad 1 <= i <= N.$$

- Forward recursion:

$$\alpha_{k+1}(i) = P(o_1 o_2 \ldots o_{k+1}, q_{k+1} = s_i) =$$
$$\Sigma_j \, P(o_1 o_2 \ldots o_{k+1}, q_k = s_j, q_{k+1} = s_i) =$$
$$\Sigma_j \, P(o_1 o_2 \ldots o_k, q_k = s_j) \, a_{ji} \, b_i(o_{k+1}) =$$
$$[\Sigma_j \, \alpha_k(j) \, a_{ji}] \, b_i(o_{k+1}), \quad 1 <= j <= N, \; 1 <= k <= K-1.$$

- Termination:

$$P(o_1 o_2 \ldots o_K) = \Sigma_j \, P(o_1 o_2 \ldots o_K, q_K = s_j) = \Sigma_j \, \alpha_K(j)$$

- Complexity :

N^2K operations.

Backward recursion for HMM

• Define the backward variable $\beta_k(i)$ as the joint probability of the partial observation sequence $o_{k+1}\, o_{k+2} \ldots o_K$ given that the hidden state at time k is s_i : $\beta_k(i) = P(o_{k+1}\, o_{k+2} \ldots o_K \mid q_k = s_i)$

• Initialization:

$$\beta_K(i) = 1 \quad , \quad 1 <= i <= N.$$

• Backward recursion:

$$\beta_k(i) = P(o_{k+1}\, o_{k+2} \ldots o_K \mid q_k = s_i) =$$

$$\Sigma_j\, P(o_{k+1}\, o_{k+2} \ldots o_K\, , q_{k+1} = s_j \mid q_k = s_i) =$$

$$\Sigma_j\, P(o_{k+2}\, o_{k+3} \ldots o_K \mid q_{k+1} = s_j)\, a_{ij}\, b_j(o_{k+1}) =$$

$$\Sigma_j\, \beta_{k+1}(j)\, a_{ij} b_j(o_{k+1}) \quad , \qquad 1 <= j <= N,\ 1 <= k <= K-1.$$

• Termination:

$$P(o_1\, o_2 \ldots o_K) = \Sigma_j\, P(o_1\, o_2 \ldots o_K\, , q_1 = s_j) =$$

$$\Sigma_j\, P(o_2 \ldots o_K \mid q_1 = s_j)\, P(q_1 = s_j)\, b_j(o_1) = \Sigma_j\, \beta_1(j)\, b_j(o_1)\, \pi_j$$

最终目标： $P(O_1 O_2 \cdots O_K) = \sum_j P(O_1 O_2 \cdots O_K, q_1 = S_j)$

根据贝叶斯网络性质 $P(ABC) = P(A|BC)P(B|C)P(C)$

$$P(O_1 O_2 \cdots O_K) = \sum_j P(q_1 = S_j) P(O_1 | q_1 = S_j) P(O_2 \cdots O_K | q_1 = S_j, O_1)$$

$$= P(O_2 \cdots O_K | q_1 = S_j)$$

$$= \sum_j \pi_j b_j(O_1) \beta_1(j)$$

Decoding problem

•**Decoding problem.** Given the HMM $M=(A, B, \pi)$ and the observation sequence $O=o_1 o_2 \dots o_K$, calculate the most likely sequence of hidden states s_i that produced this observation sequence.

• We want to find the state sequence $Q= q_1 \dots q_K$ which maximizes $P(Q \mid o_1 o_2 \dots o_K)$, or equivalently $P(Q , o_1 o_2 \dots o_K)$.

• Brute force consideration of all paths takes exponential time. Use efficient **Viterbi algorithm** instead.

• Define variable $\delta_k(i)$ as the maximum probability of producing observation sequence $o_1 o_2 \dots o_k$ when moving along any hidden state sequence $q_1 \dots q_{k-1}$ and getting into $q_k= s_i$.

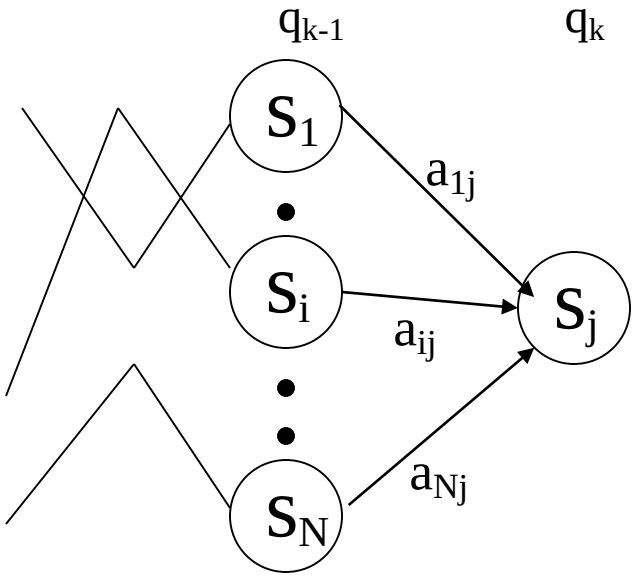
$$\delta_k(i) = \max P(q_1 \dots q_{k-1} , q_k= s_i , o_1 o_2 \dots o_k)$$

where max is taken over all possible paths $q_1 \dots q_{k-1}$.

Viterbi algorithm (1)

- General idea:

  if best path ending in $q_k = s_j$ goes through $q_{k-1} = s_i$ then it should coincide with best path ending in $q_{k-1} = s_i$.



维特比算法实际是用动态规划求概率最大路径

- $\delta_k(j) = \max P(q_1 \ldots q_{k-1}, q_k = s_j, o_1 o_2 \ldots o_k) =$

$\max_i [\, a_{ij}\, b_j(o_k)\ \max P(q_1 \ldots q_{k-1} = s_i, o_1 o_2 \ldots o_{k-1}) \,]$

- To backtrack best path keep info that predecessor of s_j was s_i.

Viterbi algorithm (2)

- Initialization:

$$\delta_1(i) = \max P(q_1 = s_i , o_1) = \pi_i \, b_i(o_1) , \quad 1<=i<=N.$$

- Forward recursion:

$$\delta_k(i) = \max P(q_1 \ldots q_{k-1} , q_k = s_i , o_1 o_2 \ldots o_k) =$$

$$\max_j [a_{ji} \, b_i(o_k) \max P(q_1 \ldots q_{k-1} = s_j , o_1 o_2 \ldots o_{k-1})] =$$

$$\max_j [a_{ji} \, b_i(o_k) \, \delta_{k-1}(j)] , \quad 1<=j<=N, \; 2<=k<=K.$$

- Termination: choose best path ending at time K

$$\max_i [\delta_K(i)]$$

- **Backtrack best path**.

This algorithm is similar to the forward recursion of evaluation problem, with Σ replaced by max and additional backtracking.

Learning problem (1)

•**Learning problem.** Given some training observation sequences $O=o_1 o_2 \ldots o_K$ and general structure of HMM (numbers of hidden and visible states), determine HMM parameters $M=(A, B, \pi)$ that best fit training data, that is maximizes $P(O \mid M)$.

• There is no algorithm producing optimal parameter values.

• Use iterative expectation-maximization algorithm to find local maximum of $P(O \mid M)$ - **Baum-Welch algorithm.**

Learning problem (2)

• If training data has information about sequence of hidden states (as in word recognition example), then use maximum likelihood estimation of parameters:

$$a_{ij} = P(s_j \mid s_i) = \frac{\text{Number of transitions from state } \mathbf{S}_i \text{ to state } \mathbf{S}_j}{\text{Number of transitions out of state } \mathbf{S}_i}$$

$$b_i(v_m) = P(v_m \mid s_i) = \frac{\text{Number of times observation } \mathbf{V}_m \text{ occurs in state } \mathbf{S}_i}{\text{Number of times in state } \mathbf{S}_i}$$

Baum-Welch algorithm

General idea:

$$a_{ij} = P(s_j \mid s_i) = \frac{\text{Expected number of transitions from state } S_i \text{ to state } S_j}{\text{Expected number of transitions out of state } S_i}$$

$$b_i(v_m) = P(v_m \mid s_i) = \frac{\text{Expected number of times observation } V_m \text{ occurs in state } S_i}{\text{Expected number of times in state } S_i}$$

$$\pi_i = P(s_i) = \text{Expected frequency in state } S_i \text{ at time } k=1.$$

Baum-Welch algorithm: expectation step(1)

• Define variable $\xi_k(i,j)$ as the probability of being in state s_i at time k and in state s_j at time k+1, given the observation sequence $o_1 o_2 \ldots o_K$.

$$\xi_k(i,j) = P(q_k = s_i , q_{k+1} = s_j \mid o_1 o_2 \ldots o_K)$$

$$\xi_k(i,j) = \frac{P(q_k = s_i , q_{k+1} = s_j , o_1 o_2 \ldots o_K)}{P(o_1 o_2 \ldots o_K)} =$$

$$\frac{P(q_k = s_i , o_1 o_2 \ldots o_k)\, a_{ij}\, b_j(o_{k+1})\, P(o_{k+2} \ldots o_K \mid q_{k+1} = s_j)}{P(o_1 o_2 \ldots o_K)} =$$

$$\frac{\alpha_k(i)\, a_{ij}\, b_j(o_{k+1})\, \beta_{k+1}(j)}{\sum_i \sum_j \alpha_k(i)\, a_{ij}\, b_j(o_{k+1})\, \beta_{k+1}(j)}$$

Baum-Welch algorithm: expectation step(2)

• Define variable $\gamma_k(i)$ as the probability of being in state s_i at time k, given the observation sequence $o_1 o_2 \ldots o_K$.

$$\gamma_k(i) = P(q_k = s_i \mid o_1 o_2 \ldots o_K)$$

$$\gamma_k(i) = \frac{P(q_k = s_i , o_1 o_2 \ldots o_K)}{P(o_1 o_2 \ldots o_K)} = \frac{\alpha_k(i)\, \beta_k(i)}{\sum_i \alpha_k(i)\, \beta_k(i)}$$

Baum-Welch algorithm: expectation step(3)

- We calculated $\xi_k(i,j) = P(q_k = s_i , q_{k+1} = s_j \mid o_1 o_2 \ldots o_K)$
  and $\gamma_k(i) = P(q_k = s_i \mid o_1 o_2 \ldots o_K)$

- Expected number of transitions from state s_i to state s_j =
  $= \sum_{k=1}^{K-1} \xi_k(i,j)$
- Expected number of transitions out of state s_i $= \sum_{k=1}^{K-1} \gamma_k(i)$

- Expected number of times observation v_m occurs in state s_i =
  $= \sum_{k=1}^{K} \gamma_k(i)$, k is such that $o_k = v_m$
- Expected frequency in state s_i at time k= $\gamma_k(i)$.

Baum-Welch algorithm: maximization step

$$a_{ij} = \frac{\text{Expected number of transitions from state } s_i \text{ to state } s_j}{\text{Expected number of transitions out of state } s_i} = \frac{\sum_{k=1}^{K-1} \xi_k(i,j)}{\sum_{k=1}^{K-1} \gamma_k(i)}$$

$$b_i(v_m) = \frac{\text{Expected number of times observation } v_m \text{ occurs in state } s_i}{\text{Expected number of times in state } s_i}$$

$$= \frac{\sum_{k=1, o_k = v_m}^{K} \gamma_k(i)}{\sum_{k=1}^{K} \gamma_k(i)}$$

$\pi_i = ($Expected frequency in state s_i at time k=1) $= \gamma_1(i)$.

用最大似然估计的推导见
https://blog.csdn.net/qq_37334135/article/details/86302735