厦门大学
XIAMEN UNIVERSITY

第4章 隐马尔可夫模型

洪青阳

纲要

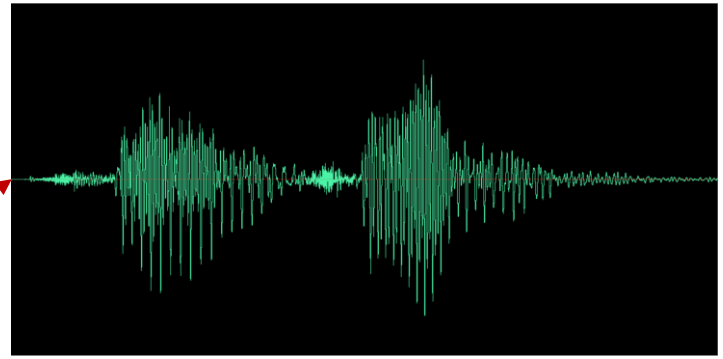
4.1 HMM的基本概念

4.2 HMM三个基本问题

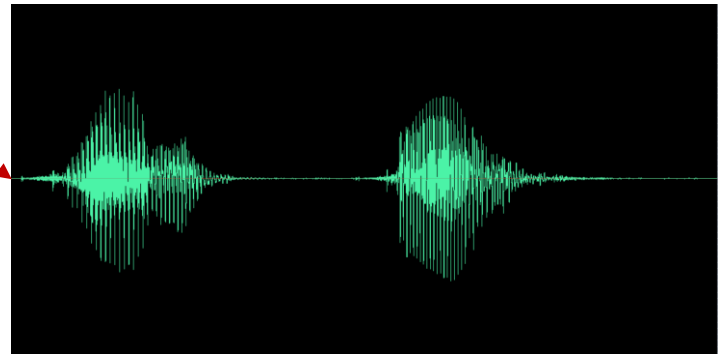
4.3 本章小结

4.1 HMM的基本概念

同样说“前进”



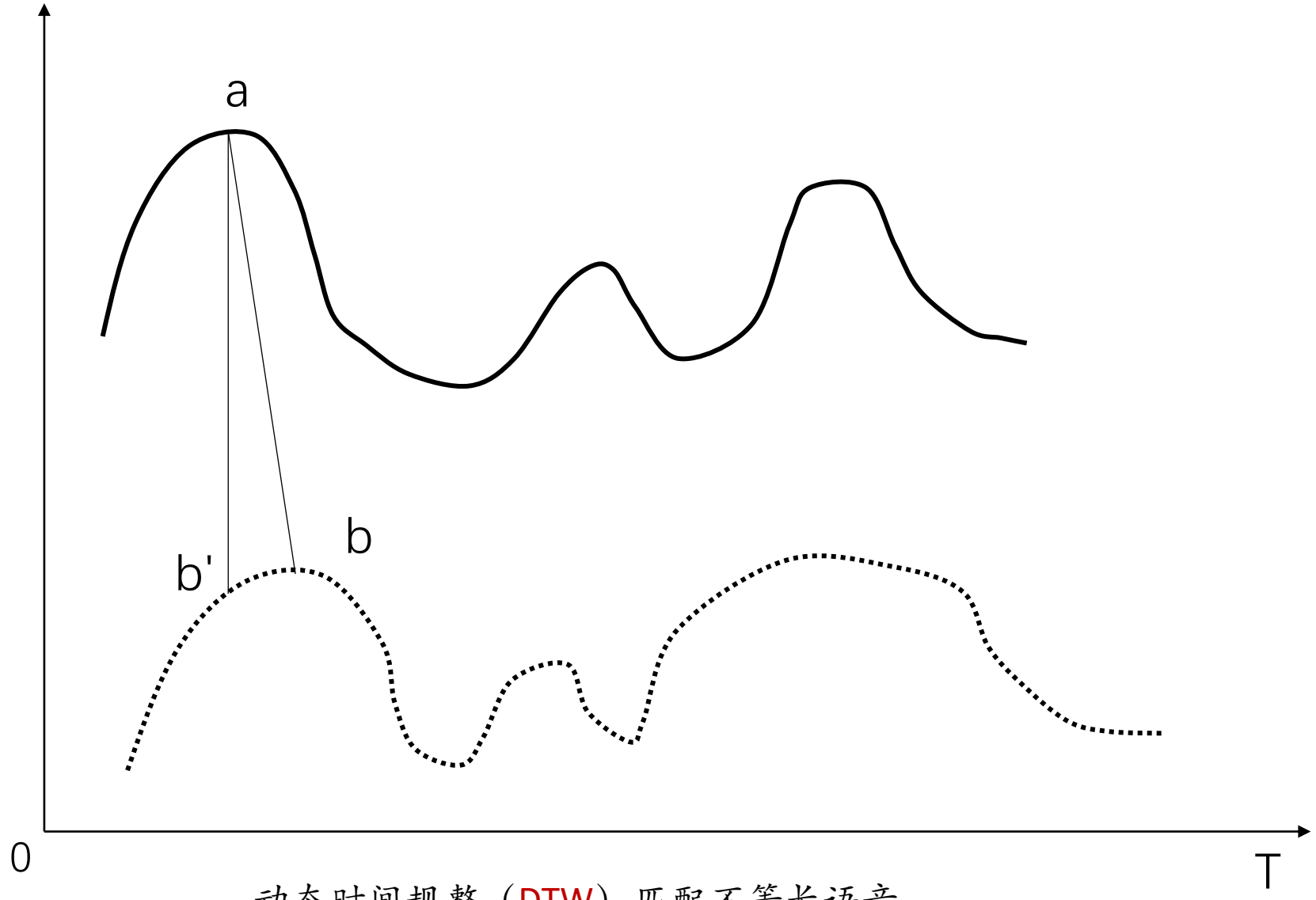
（a）正常语速



（b）慢语速

如何表示两段不同长度语音的相似性？

4.1 HMM的基本概念



动态时间规整（DTW）匹配不等长语音

4.1 HMM的基本概念

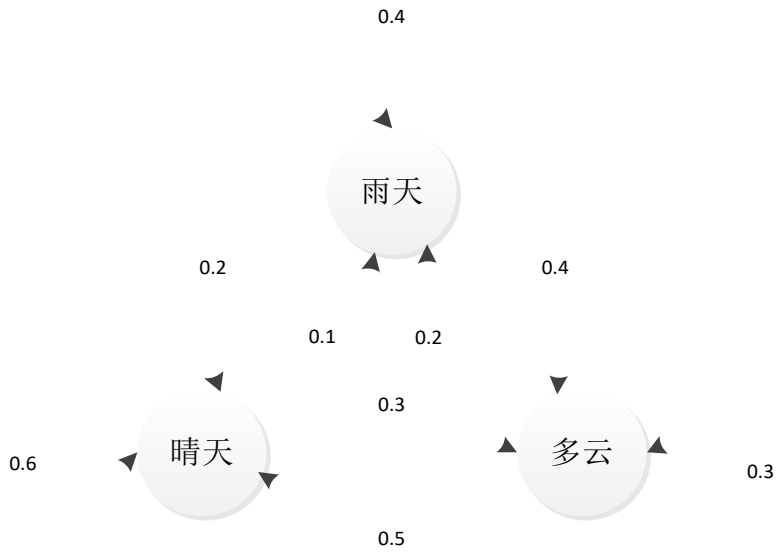
- 人的发音包含双重随机过程：
  - **想说什么不确定：**即说话内容，具体是哪些符号（音素或字词）？
  - **怎么说不确定：**同样内容发音的观察值差异很大。说快还是说慢？其它差异还包括大声小声？在什么环境说？等等。

- 要从语音信号识别出对应的文本内容，要解决如下问题：
  - 如何描述双重随机过程？
  - 如何通过特征序列识别出符号序列？

- **隐马尔可夫模型（HMM）**
  - **马尔可夫链**：状态转移的随机性
  - 依存于状态的观察事件的随机性

4.1 HMM的基本概念—马尔可夫链

- **马尔可夫（Markov）链**最早由俄国数学家安德雷·马尔可夫提出，用于描述随机过程。
- 一个典型例子是天气模型。



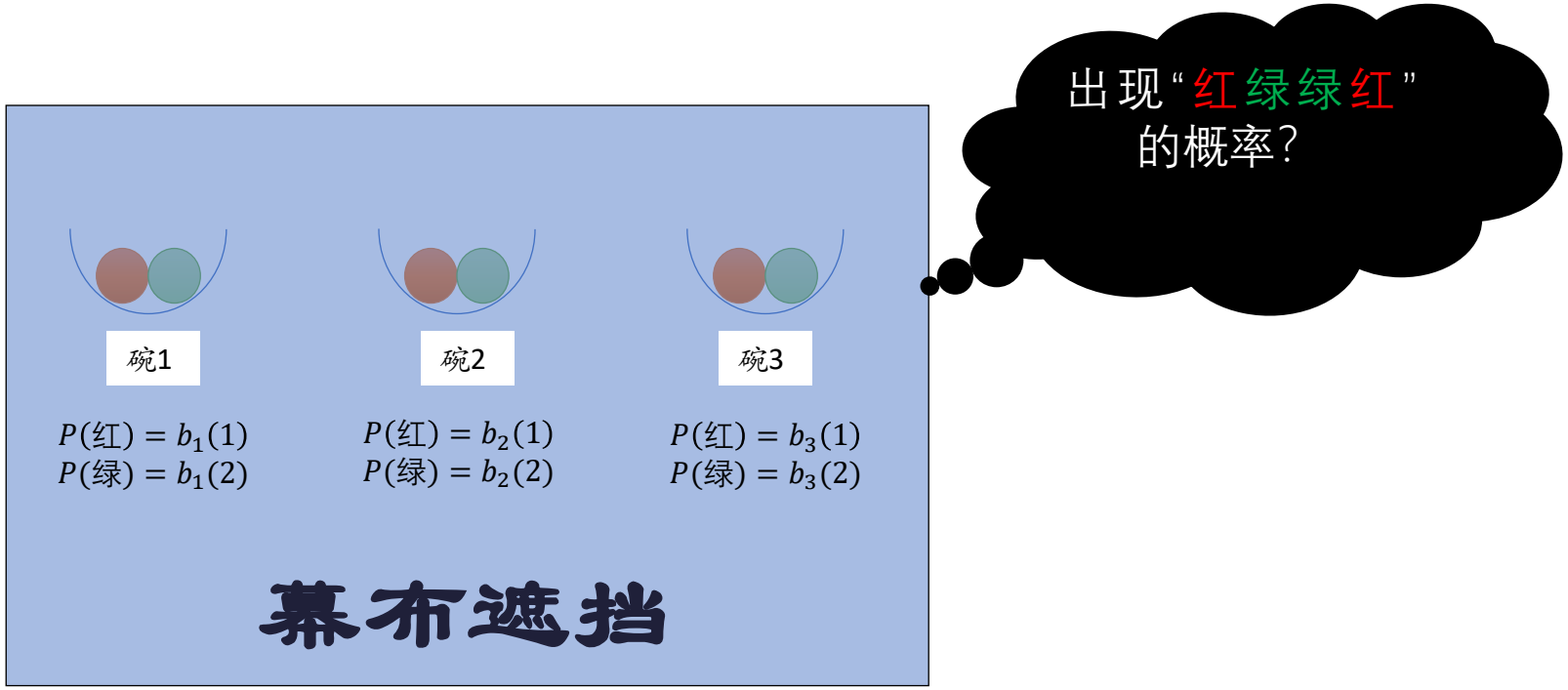
时间和事件都是离散的马尔可夫过程称为马尔可夫链。

状态与观察事件之间一一对应，不存在随机性。

$$P(\text{多云}|\text{晴天})P(\text{晴天}|\text{多云})P(\text{雨天}|\text{晴天}) = 0.3 \times 0.5 \times 0.1 = 0.015$$

4.1 双重随机过程—颜色球例子

- 假设有3个碗，每个碗中按不同比例存放红绿两种颜色的球。
- 根据初始概率分布，随机地选择其中的一个碗，并随机取出一球，记为 o_1 。把球放回原碗，根据碗之间的转移概率，随机选择下一个碗，再随机取出一球，记为o_2。如此反复，可以得到一个描述球的颜色的序列$o_1, o_2 \cdots$，可称之为观察值序列。在此过程中，碗之间的转移不确定，一种球的颜色出自哪个碗也不确定。

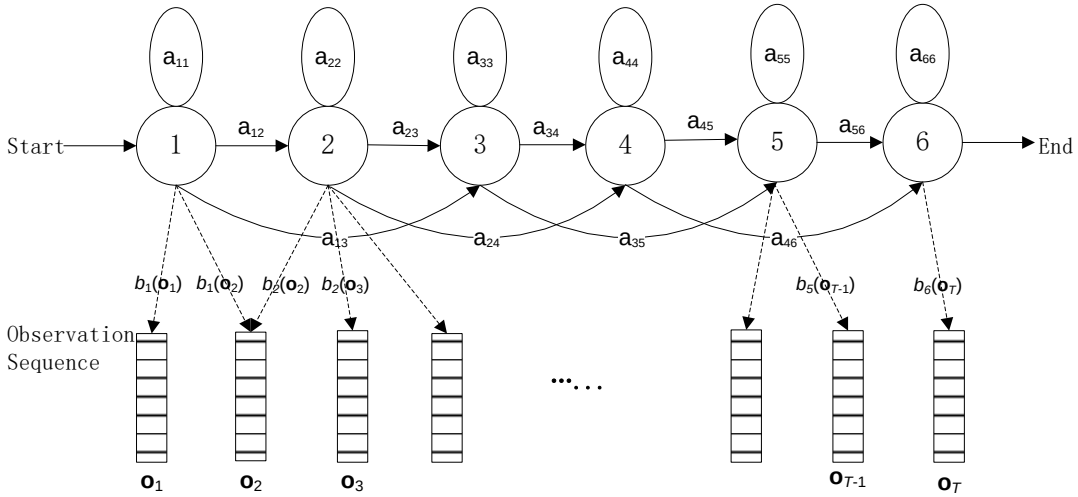


4.1 HMM的基本概念

- 隐含Markov模型

  包含6个状态（不可见），

  观察值$O = o_1, o_2, \cdots, o_T$可见。



- HMM的模型要素包括：

  N：模型中的状态数目

  M：每个状态可能输出的观察符号的数目

  $A = \{a_{ij}\}$：状态转移矩阵

  $B = \{b_j(k)\}$：输出概率矩阵

  $\pi = \{\pi_i\}$：初始状态概率分布

4.2 HMM的三个基本问题

- 模型评估问题(如何求：$P(O|\lambda)$)
- 最佳路径问题(如何求：$Q = q_1 q_2 \cdots q_T$)
- 模型训练问题(如何求：π, A, B)

模型评估问题

- 穷举法
- 前向-后向(Forward-Backward)算法

模型评估问题—穷举法

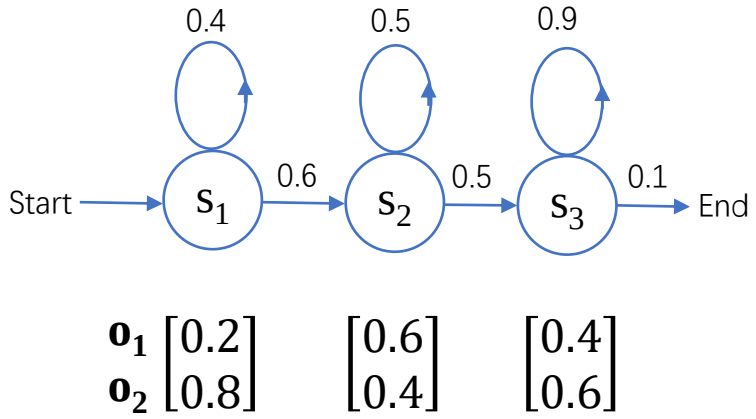
- 当给定模型$\lambda = (\pi, A, B)$以及观察序列$O = o_1, o_2, \cdots, o_T$时，计算模型λ对观察序列$O$的$P(O|\lambda)$概率的思路是(穷举法):

(1)对长度为T的观察序列O，找出所有可能产生该观察序列O的状态转移序列$Q^j = q_1^j q_2^j q_3^j \cdots q_T^j (j = 1,2,\cdots,J)$;

(2)分别计算Q^j与观察序列O的联合概率$P(O, Q^j|\lambda)$;

(3)取各联合概率$P(O, Q^j|\lambda)$的和，即:

$$P(O|\lambda) = \sum_{j=1}^{J} P(O|\lambda)$$



模型评估问题—穷举法

- $P(O|\lambda)$的一般解法：

$\because$ $P(O,Q^j|\lambda)=P(Q^j|\lambda)\ P(O|Q^j,\lambda)$

$$P(Q^j|\lambda)=P\left(q_1^j\right)P\left(q_2^j\middle|q_1^j\right)P\left(q_3^j\middle|q_2^j\right)\cdots P\left(q_T^j\middle|q_{T-1}^j\right)=a_{0,1}^j a_{1,2}^j\ a_{2,3}^j\cdots a_{T-1,T}^j$$

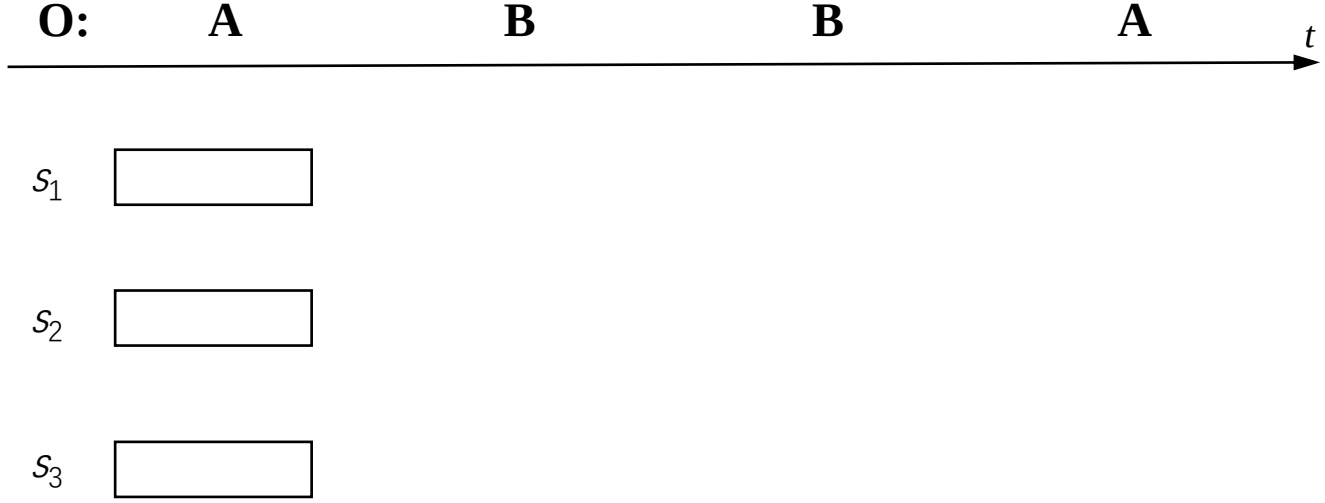
$$P(O|Q^j,\lambda)=P\left(o_1\middle|q_1^j\right)P\left(o_2\middle|q_2^j\right)\cdots P\left(o_T\middle|q_T^j\right)=b_1^j(o_1)b_2^j(o_2)b_3^j(o_3)\cdots b_T^j\,(o_T)$$

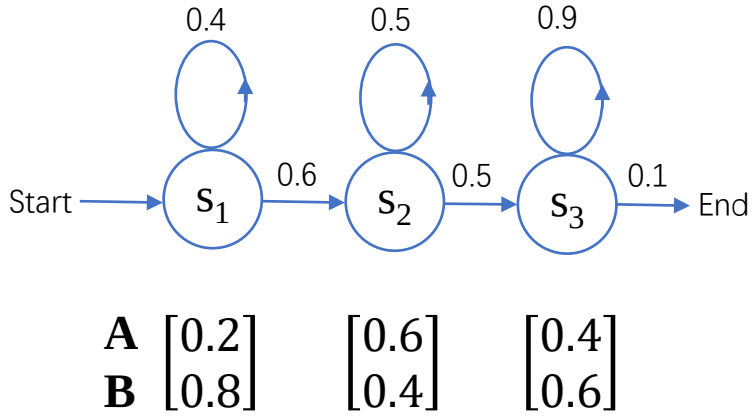
$\therefore$ $P(O,Q^j|\lambda)=a_{0,1}^j b_1^j(o_1)a_{1,2}^j b_2^j(o_2)\cdots a_{T-1,T}^j b_T^j\,(o_T)$

$$P(O|\lambda)=\sum_{j=1}^{J}P(O,Q^j|\lambda)=\sum_{j=1}^{J}\prod_{t=1}^{T}a_{t-1,t}^j b_t^j(o_t)$$

重复计算，复杂度太高！

前向算法

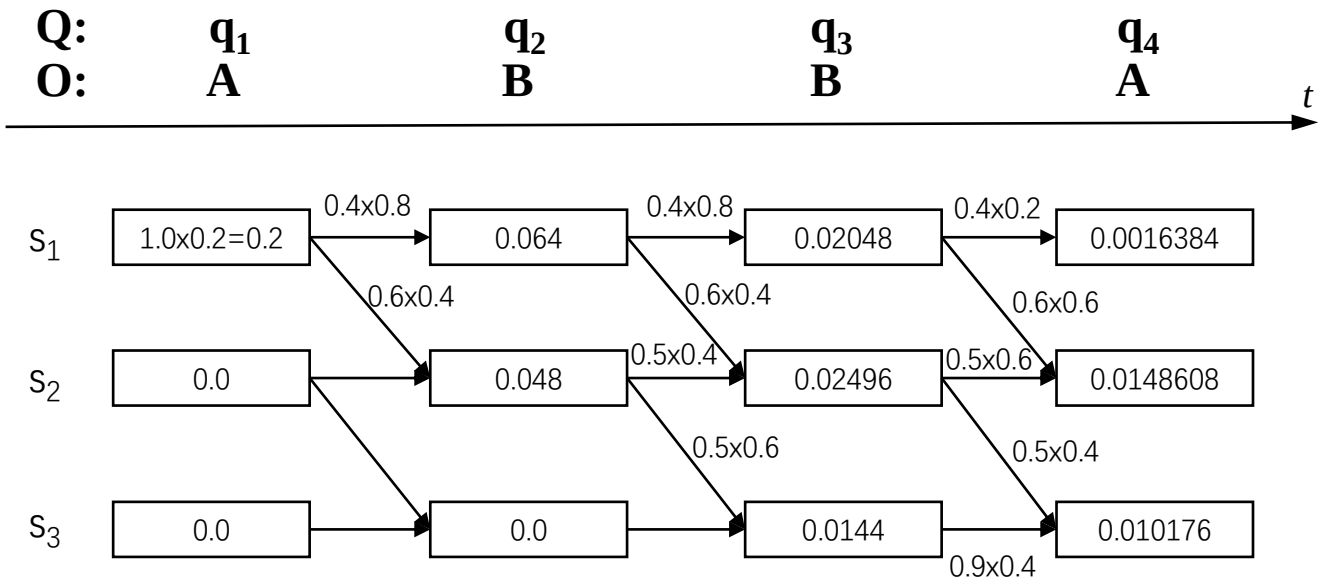




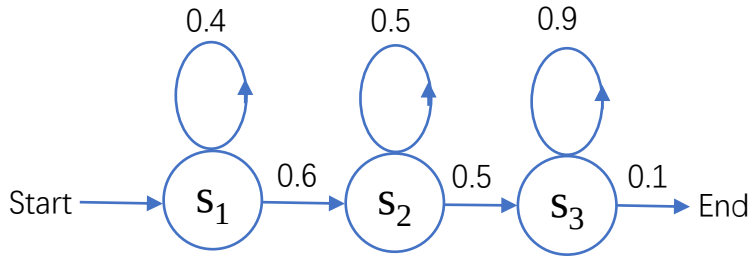
$$\begin{matrix}\mathbf{A}\\ \mathbf{B}\end{matrix}\begin{bmatrix}0.2\\0.8\end{bmatrix}\quad\begin{bmatrix}0.6\\0.4\end{bmatrix}\quad\begin{bmatrix}0.4\\0.6\end{bmatrix}$$

采用前向算法求解P（**ABBA**|λ）概率的格型图

前向算法



采用前向算法求解P(**ABBA**|λ)概率的格型图



$$\begin{matrix}\mathbf{A}\\ \mathbf{B}\end{matrix}\begin{bmatrix}0.2\\0.8\end{bmatrix}\quad\begin{bmatrix}0.6\\0.4\end{bmatrix}\quad\begin{bmatrix}0.4\\0.6\end{bmatrix}$$

总概率=0.0016384+0.0148608+0.010176=0.0266752

前向-后向算法

前向-后向(Forward-Backward)算法用来解决高效计算$P(O|\lambda)$的问题。该算法分为两部分：

◆前向算法

前向算法按输出观察值序列的时间，从前向后地推计算输出概率。

此算法用$\alpha_t(j)$表示已经输出部分观察值$o_1, o_2, \cdots, o_t$，并且到达状态为s_j的概率：

$$\alpha_t(j) = P(o_1, o_2, \cdots, o_t, q_t = s_j|\lambda)$$

1、初始化

$$\alpha_1(i) = \pi_i b_i(o_1), 1 \le i \le N$$

2、迭代计算

$$\alpha_{t+1}(j) = \left[\sum_{i=1}^{N} \alpha_t(i)a_{ij}\right] b_j(o_{t+1}), 1 \le t \le T-1, 1 \le j \le N$$

3、终止计算

$$P(O|\lambda) = \sum_{i=1}^{N} \alpha_T(i)$$

前向-后向算法

$$o_1, o_2, \cdots, o_t, o_{t+1}, \cdots, o_T$$

◆后向算法

后向算法由后向前推算输出概率。

用$\beta_t(i)$表示时刻t时的状态为s_i，输出结束时的状态为s_N，且输出观察值序列为$o_t o_{t+1}, \cdots, o_T$的概率：

$$\beta_t(i) = P(o_t, o_{t+1}, \cdots, o_T | q_t = s_i, q_T = s_N, \lambda)$$

1、初始化

$$\beta_T(i) = 1, i = N$$

2、迭代计算

$$\beta_t(i) = \sum_{j=1}^{N} a_{ij} b_j(o_{t+1}) \beta_{t+1}(j), 1 \le t \le T-1, 1 \le j \le N$$

前向-后向算法

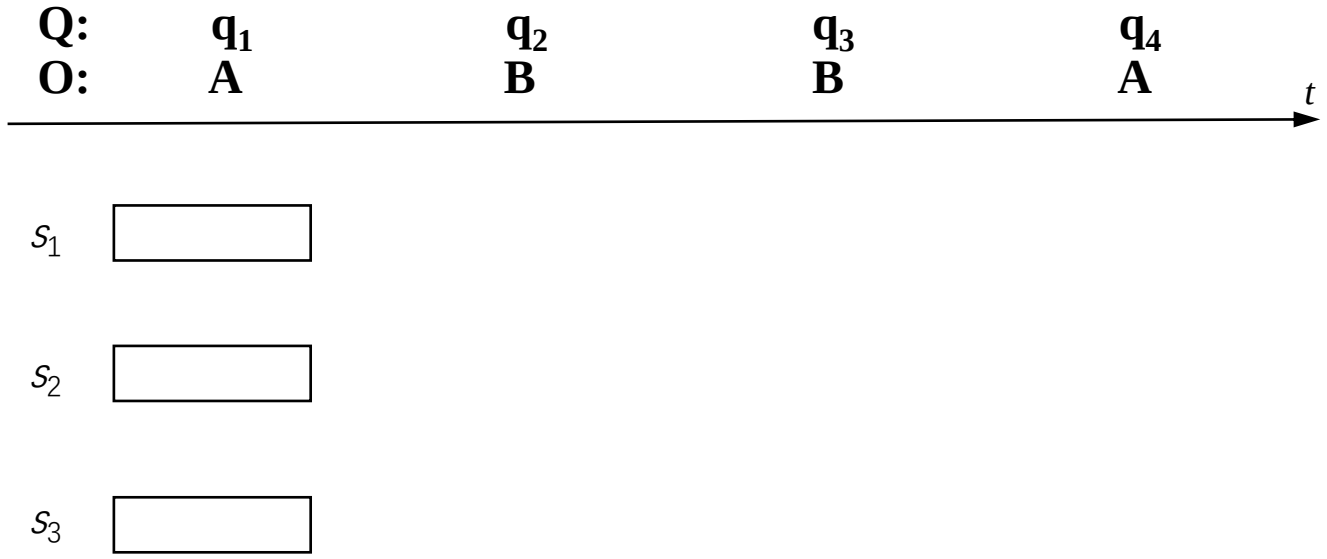
HMM的前向、后向概率估计：

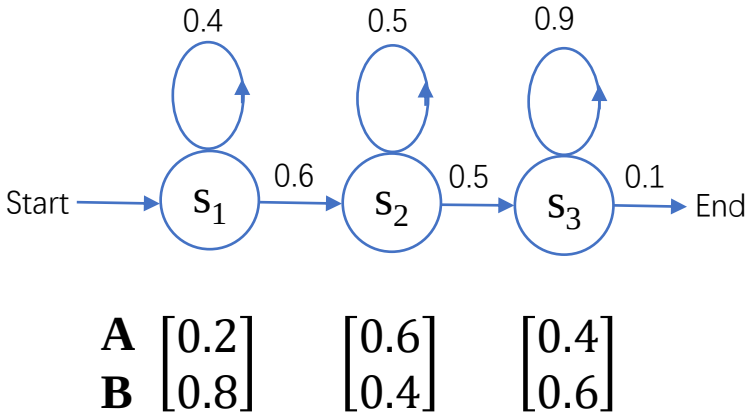
$$P(O|\lambda)=\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_t(i)a_{ij}b_j(o_{t+1})\beta_{t+1}(j),1\le t\le T-1$$

$$P(O|\lambda)=\sum_{i=1}^{N}\alpha_t(i)\beta_t(i)=\sum_{i=1}^{N}\alpha_T(i),1\le t\le T-1$$

Viterbi算法

- Viterbi算法用于解决如何寻找与给定观察值序列对应的最佳状态序列的问题。

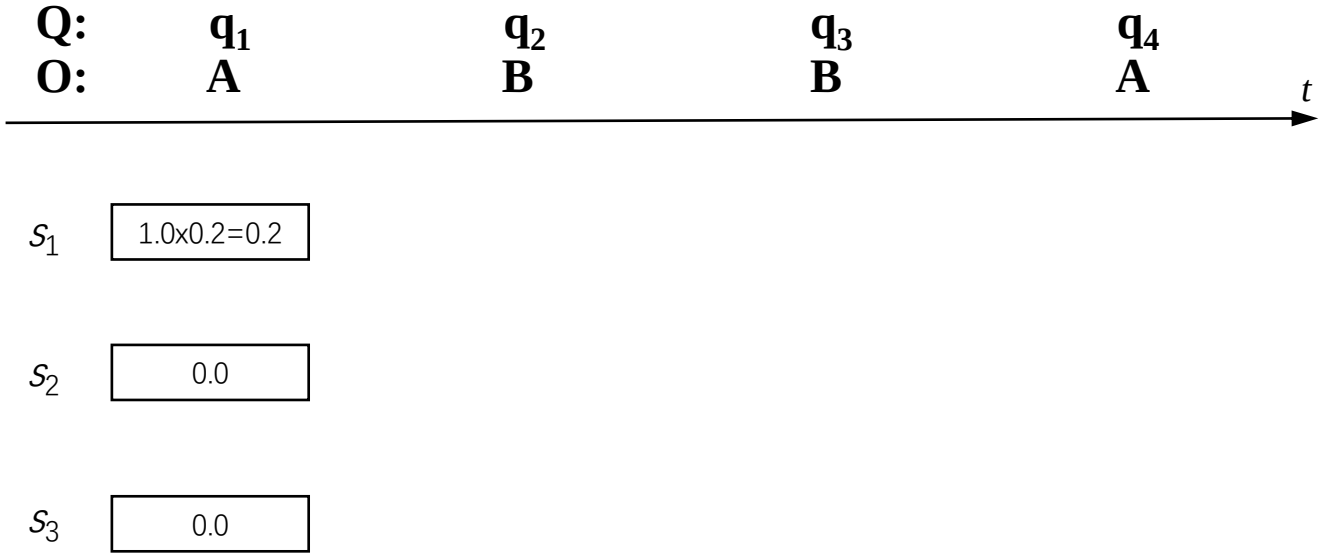


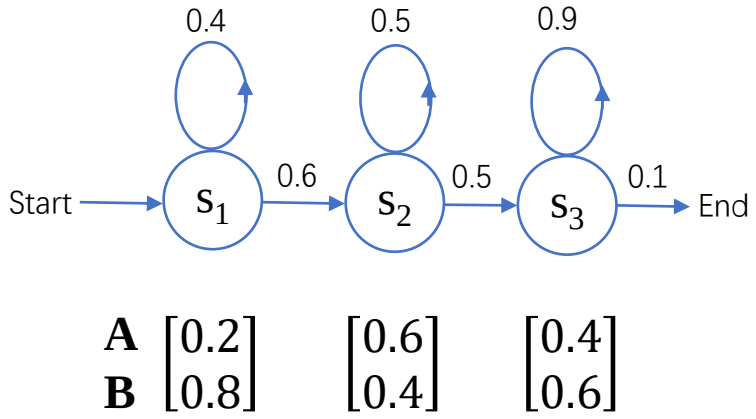


采用Viterbi算法求解产生观察序列**ABBA**最佳路径的格型图

Viterbi算法

- Viterbi算法用于解决如何寻找与给定观察值序列对应的最佳状态序列的问题。



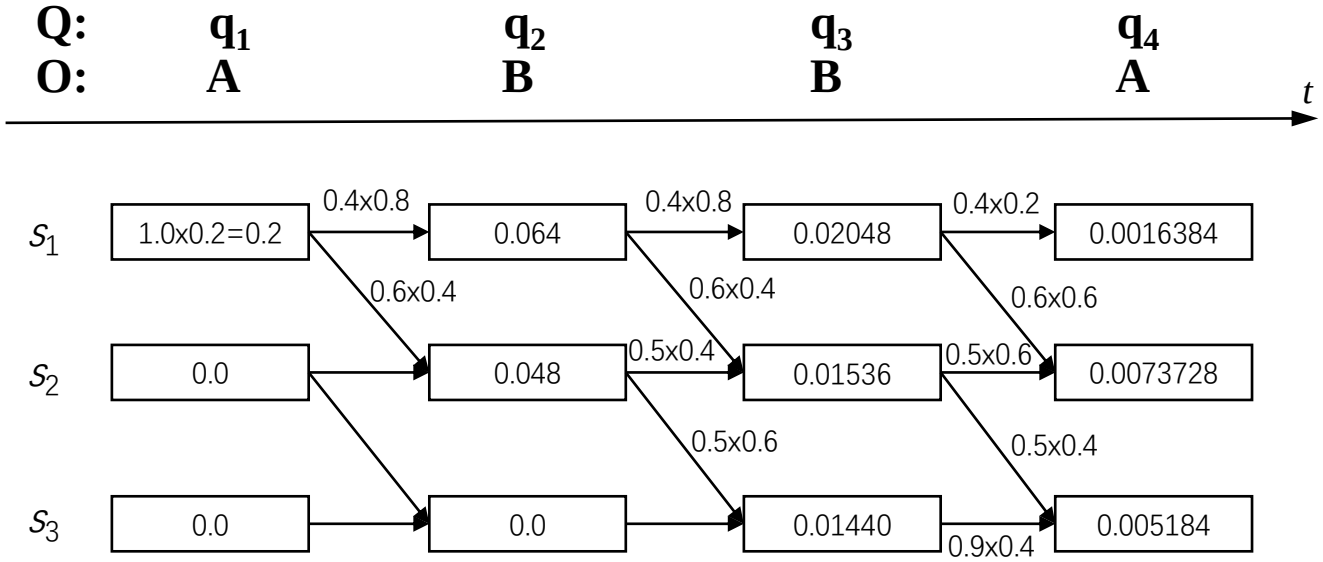


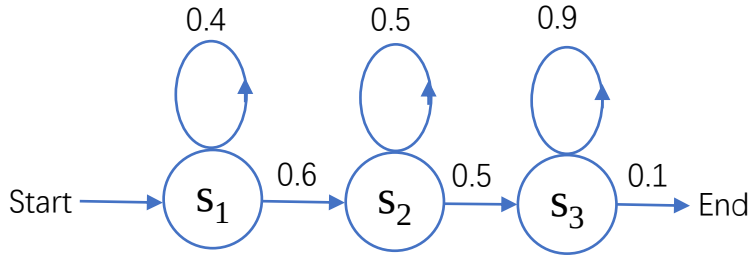
$$\begin{matrix} \mathbf{A} \\ \mathbf{B} \end{matrix} \begin{bmatrix} 0.2 \\ 0.8 \end{bmatrix} \quad \begin{bmatrix} 0.6 \\ 0.4 \end{bmatrix} \quad \begin{bmatrix} 0.4 \\ 0.6 \end{bmatrix}$$

采用Viterbi算法求解产生观察序列**ABBA**最佳路径的格型图

Viterbi算法

- Viterbi算法用于解决如何寻找与给定观察值序列对应的最佳状态序列的问题。



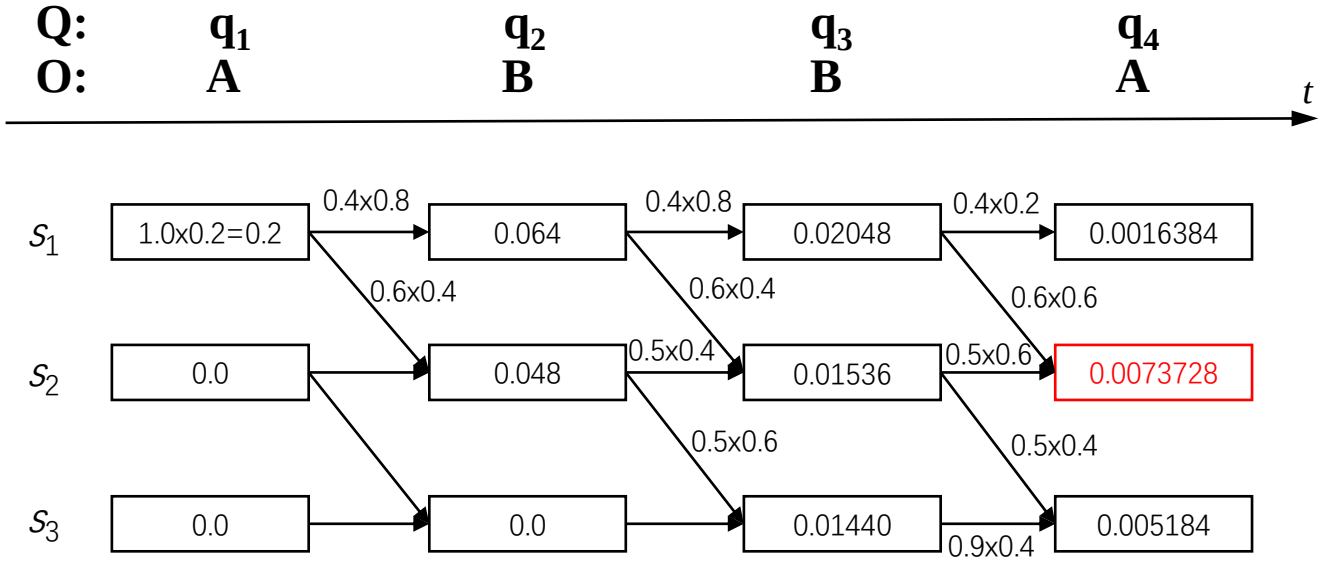


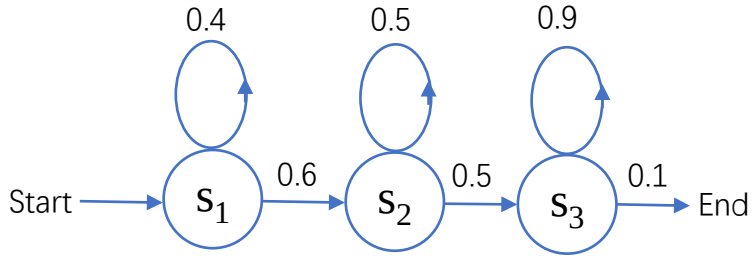
$$\begin{matrix} \mathbf{A} \\ \mathbf{B} \end{matrix} \begin{bmatrix} 0.2 \\ 0.8 \end{bmatrix} \quad \begin{bmatrix} 0.6 \\ 0.4 \end{bmatrix} \quad \begin{bmatrix} 0.4 \\ 0.6 \end{bmatrix}$$

采用Viterbi算法求解产生观察序列**ABBA**最佳路径的格型图

Viterbi算法

- Viterbi算法用于解决如何寻找与给定观察值序列对应的最佳状态序列的问题。



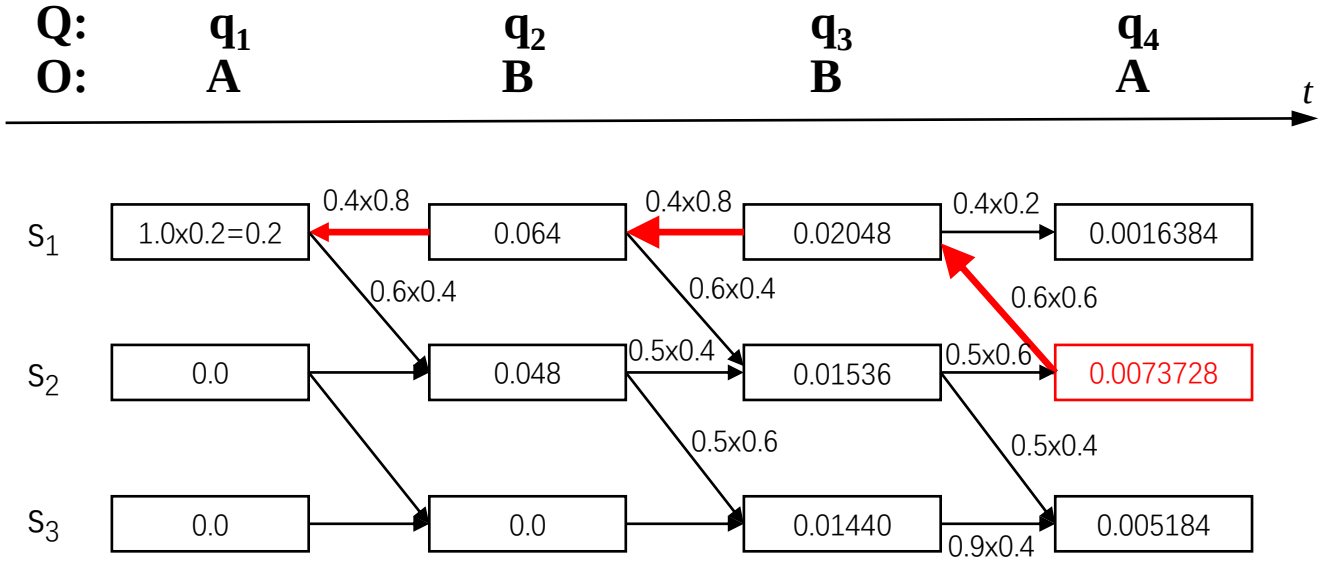


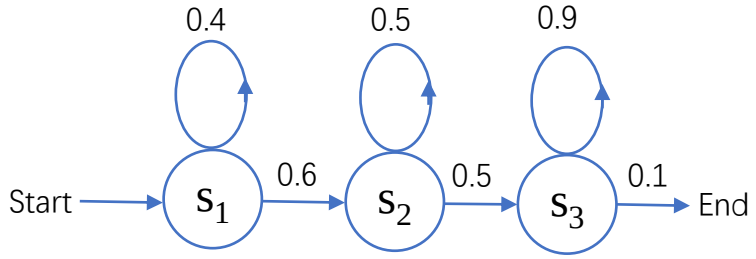
$$\mathbf{A}\ \begin{bmatrix}0.2\\0.8\end{bmatrix}\quad\begin{bmatrix}0.6\\0.4\end{bmatrix}\quad\begin{bmatrix}0.4\\0.6\end{bmatrix}$$

采用Viterbi算法求解产生观察序列**ABBA**最佳路径的格型图

Viterbi算法

- Viterbi算法用于解决如何寻找与给定观察值序列对应的最佳状态序列的问题。





$$\mathbf{A}\begin{bmatrix}0.2\\0.8\end{bmatrix}\quad\begin{bmatrix}0.6\\0.4\end{bmatrix}\quad\begin{bmatrix}0.4\\0.6\end{bmatrix}$$

最佳状态序列：S_1-S_1-S_1-S_2

采用Viterbi算法求解产生观察序列**ABBA**最佳路径的格型图

Viterbi算法

定义最佳状态序列$Q^* = q_1^*, q_2^*, \cdots, q_T^*$， $\varphi_t(j)$记录局部最佳状态序列。

定义$\delta_t(i)$为截止到时刻t，依照状态转移序列$q_1, q_2, \cdots, q_t$，产生出观察值$o_1, o_2, \cdots, o_t$的最大概率，且最终状态为s_i。

$$\delta_t(i) = \max_{q_1, q_2, \cdots, q_{t-1}} P(q_1, q_2, \cdots, q_t, q_t = s_i | \lambda)$$

1、初始化

$$\delta_0(1) = 1, \delta_0(j) = 0 \ \ (j \neq 1)$$
$$\varphi_1(j) = q_1$$

2、递推

$$\delta_t(j) = b_j(o_t) \max_{1 \leq i \leq N} \delta_{t-1}(i) a_{ij} \ \ (1 \leq t \leq T; 1 \leq i \leq N)$$
$$\varphi_t(j) = \operatorname{argmax}_{1 \leq i \leq N} \delta_{t-1}(i) a_{ij}$$

3、终止

$$P_{max}(S, O|\lambda) = \max_{1 \leq i \leq N} \delta_T(i)$$
$$\varphi_T(N) = arg \max_{1 \leq i \leq N} \delta_{T-1}(i) a_{ij}$$

算法终止时$\varphi_t()$记录的数据便是Q^*。

Viterbi算法

Q的不同取值使概率值$P(Q,O|\lambda)$差别很大，而$P(Q^*,O|\lambda)$是$\sum_s P_{max}(Q,O|\lambda)$的各个分量中占比最大的路径概率。

因此，常等价地用$P(Q^*,O|\lambda)$近似$\sum_Q P_{max}(Q,O|\lambda)$，那么实际上Viterbi算法就能用来计算$P(O|\lambda)$。

模型训练问题

- 模型训练问题可定义为：给定一个观察值序列$O = o_1, o_2, \cdots, o_T$，确定一个$\lambda = (\pi, A, B)$，使得$P(O|\lambda)$最大。

$$\bar{\lambda} = arg\max_{\lambda} P(O|\lambda)$$

- 实际上，不存在一种方法直接估计最佳的λ。
- 替代的办法：根据观察值序列选取初始模型$\lambda = (\pi, A, B)$，然后依据某种方法求得一组新参数$\bar{\lambda} = (\pi, \bar{A}, \bar{B})$，保证有$P(O|\bar{\lambda}) \geq P(O|\lambda)$。重复这个过程，逐步改进模型参数，直到$P(O|\bar{\lambda})$收敛。

寻求辅助函数：
使得$Q(\lambda, \bar{\lambda}) \geq Q(\lambda, \lambda) \Rightarrow P(O|\bar{\lambda}) \geq P(O|\lambda)$

Baum-Welch算法（简称B-W算法）

基于状态序列Q，有概率公式：

$$P(O,Q|\lambda)=\pi_{q_0}\prod_{t=1}^{T}a_{q_{t-1}q_t}b_{q_t}(o_t)$$

取对数得到：

$$logP(O,Q|\lambda)=log\pi_{q_0}+\sum_{t=1}^{T}loga_{q_{t-1}q_t}+\sum_{t=1}^{T}logb_{q_t}(o_t)$$

定义辅助函数：

$$Q(\lambda,\bar{\lambda})=\sum_{Q}P(O,Q|\lambda)\log P(O,Q|\bar{\lambda})$$

$$=\sum_{i=1}^{N}P(O,q_0=i|\bar{\lambda})\,log\pi_i+\sum_{i=1}^{N}\sum_{j=1}^{N}\sum_{t=1}^{T}P(O,q_{t-1}=i,q_t=j|\bar{\lambda})\log a_{ij}$$

$$+\sum_{i=1}^{N}\sum_{t=1}^{T}P(O,q_t=i|\bar{\lambda})\log b_i(o_t)$$

基于Bayes公式和Jensen不等式，可确保$Q(\lambda,\bar{\lambda})\geq Q(\lambda,\lambda)\Rightarrow P(O|\bar{\lambda})\geq P(O|\lambda)$
因此求$Q(\lambda,\bar{\lambda})$最大值即可。

Baum-Welch算法（简称B-W算法）

模型参数$\pi_i, a_{ij}, b_i(o_t)$均符合如下函数形式：

$$F(x) = \sum_i c_i log x_i$$

并有条件限制$\sum_i x_i = 1$。

该函数可获得全局最优值，当：

$$x_i = \frac{c_i}{\sum_k c_k}$$

因此我们可得到模型参数的重估计公式。

Baum-Welch算法—重估计公式

重估计公式：

$$a_{ij} = \frac{\sum_{t=1}^{T-1} \alpha_t(i) a_{ij} b_j(o_{t+1}) \beta_{t+1}(j)}{\sum_{t=1}^{T-1} \alpha_t(i) \beta_t(i)} = \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)}$$

$$b_j(k) = \frac{\sum_{t=1 \; s.t. o_t = v_k}^{T} \alpha_t(i) \beta_t(i)}{\sum_{t=1}^{T} \alpha_t(i) \beta_t(i)} = \frac{\sum_{t=1 \; s.t. o_t = v_k}^{T} \gamma_t(j)}{\sum_{t=1}^{T} \gamma_t(j)}$$

其中$\xi_t(i,j)$为给定训练序列O和模型λ时，HMM在时刻t处于状态s_i，在$t+1$时刻处于状态s_j的概率，即

$$\xi_t(i,j) = P(q_t = s_i, q_{t+1} = s_j | O, \lambda)$$

根据前向-后向算法，可推导出：

$$\xi_t(i,j) = \frac{\alpha_t(i) a_{ij} b_j(o_{t+1}) \beta_{t+1}(j)}{\sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_t(i) a_{ij} b_j(o_{t+1}) \beta_{t+1}(j)} = \frac{\alpha_t(i) a_{ij} b_j(o_{t+1}) \beta_{t+1}(j)}{P(O|\lambda)}$$

定义$\gamma_t(i)$为在时刻t时处于状态s_i的概率：

$$\gamma_t(i) = \sum_{j=1}^{N} \xi_t(i,j) = \sum_{j=1}^{N} \frac{\alpha_t(i) a_{ij} b_j(o_{t+1}) \beta_{t+1}(j)}{P(O|\lambda)} = \frac{\alpha_t(i) \beta_t(i)}{P(O|\lambda)}$$

Baum-Welch算法—EM训练过程

厦门大学
XIAMEN UNIVERSITY

- Baum-Welch算法的理论基础是（Expectation Maximization）算法，包含两个主要方面：
  - 一是求期望（expectation），用E来表示；
  - 一是最大化（maximization），用M来表示。

Baum-Welch算法—EM训练过程

ALGORITHM 4.4: The Baum-Welch Algorithm

1. **初始化**
   π和A的初值对结果影响不大，只要满足约束条件可随机选取或均值选取。B的初值对参数重估影响较大，选取算法较复杂。

2. **E-Step**
   基于模型参数，计算$\gamma_t(i)$和$\xi_t(i,j)$。

3. **M-Step**
   由重估计公式重新计算a_{ij}和$b_j(k)$，最大化辅助函数。

4. **迭代**
   重复第2步的操作，直到a_{ij}和$b_j(k)$收敛为止，即$P(O|\lambda)$趋于稳定，不再明显增大。

4.3 本章小结

厦门大学
XIAMEN UNIVERSITY

- 描述双重随机过程：HMM
- HMM三大问题
  - 模型评估问题：前向-后向算法
  - 最佳路径问题：Viterbi算法
  - 模型训练问题：Baum-Welch算法
- Baum-Welch算法采用的辅助函数和推导过程比较复杂，可查阅Jensen不等式。

附录：Jensen’s inequality(不等式)

- 凸函数满足：

$$f\left(\sum_{i=1}^{M}\lambda_i x_i\right)\leq\sum_{i=1}^{M}\lambda_i f(x_i)$$

其中$\lambda_i\geq 0$，$\sum_i\lambda_i=1$。

- 该公式称为Jensen’s不等式，对于连续变量，表示如下：

$$f\left(\int xp(x)\mathrm{d}x\right)\leq\int f(x)p(x)\mathrm{d}x$$