HMM hidden markov model

Definition

Markov Chain

$Q = q_1, q_2, \cdots q_N$ a set of N states

$A = a_{11}\ a_{12} \cdots a_{n1}\ a_{nn}$ transition probability matrix

$\pi = \pi_1, \pi_2, \cdots \pi_N$ initial probability distribution

Markov Assumption: $P(q_i = a \mid q_1, q_2, \cdots q_{i-1}) = P(q_i = a \mid q_{i-1})$

Hidden Markov model

$Q = q_1, q_2, \cdots q_N$ a set of N states

$A = a_{11}\ a_{12} \cdots a_{n1}\ a_{nn}$ transition probability matrix 状态转移矩阵

$\pi = \pi_1, \pi_2, \cdots \pi_N$ initial probability distribution 初始状态概率向量

$O = o_1 o_2 \cdots o_T$ a sequence of T observations

$B = b_i(o_t)$ emission probability, each expressing the prob of an observation o_t being generated from a state i 观测概率矩阵

- Markov Assumption: $P(q_i = a \mid q_1, q_2, \cdots q_{i-1}) = P(q_i = a \mid q_{i-1})$

  HMM 在位置时刻 t 的状态只依赖于其前一时刻状态，与其他时刻的状态和观测无关！

- Output independence: $P(O_i \mid q_1, \cdots q_i, \cdots q_T, O_1, \cdots O_{i-1}, \cdots O_T) = P(O_i \mid q_i)$

  HMM 任意时刻的观测只依赖于该时刻的状态，与其他观测和状态无关

HMM 的三个重要问题

1. likelihood: Given an HMM $\lambda = (A, B, \pi)$ and an observation sequence O, determine the likelihood $P(O \mid \lambda)$

2. Decoding: Given an observation sequence O and an HMM $\lambda = (A, B, \pi)$, discover the best hidden state sequence Q

3. learning: Given an observation sequence O and the set of states in the HMM, learn the HMM parameters A and B (MLE)

Question 1. likelihood → forward & backward algorithm

问题：给定 HMM：$\lambda = (A, B, \pi)$，求 $P(O|\pi)$

定义：

前向：

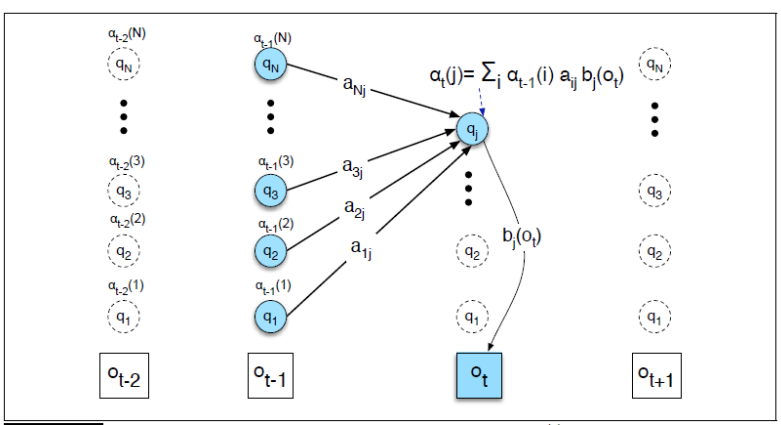
$\alpha_t(j)$: The prob of being in the state j after seeing the first t observations, the value of each cell $\alpha_t(j)$ is computed by summing over the probs of each path that could lead to this cell.

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

$$\alpha_t(j) = \sum_{i=1}^{N} \alpha_{t-1}(i)\, a_{ij}\, b_j(o_t)$$

a_{ij} : the transition prob from previous state q_i to current state q_j

$b_j(o_t)$: the state observation likelihood of the observation symbol o_t given the current state j



前向算法：

算法 10.2（观测序列概率的前向算法）

输入：隐马尔可夫模型 λ，观测序列 O；

输出：观测序列概率 $P(O|\lambda)$。

（1）初值

$$\alpha_1(i) = \pi_i b_i(o_1), \quad i = 1, 2, \cdots, N \tag{10.15}$$

（2）递推　对 $t = 1, 2, \cdots, T-1$，

$$\alpha_{t+1}(i) = \left[\sum_{j=1}^{N} \alpha_t(j) a_{ji}\right] b_i(o_{t+1}), \quad i = 1, 2, \cdots, N \tag{10.16}$$

（3）终止

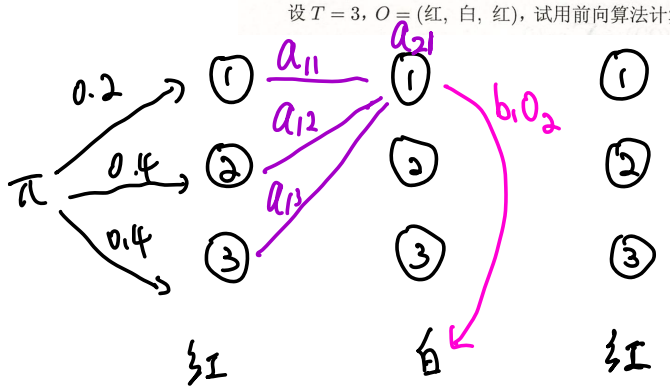
$$P(O|\lambda) = \sum_{i=1}^{N} \alpha_T(i) \tag{10.17}$$

例题：

例 10.2 考虑盒子和球模型 $\lambda = (A, B, \pi)$，状态集合 $Q = \{1, 2, 3\}$，观测集合 $V = \{红, 白\}$，

$$A = \begin{bmatrix} 0.5 & 0.2 & 0.3 \\ 0.3 & 0.5 & 0.2 \\ 0.2 & 0.3 & 0.5 \end{bmatrix}, \quad B = \begin{bmatrix} 0.5 & 0.5 \\ 0.4 & 0.6 \\ 0.7 & 0.3 \end{bmatrix}, \quad \pi = \begin{bmatrix} 0.2 \\ 0.4 \\ 0.4 \end{bmatrix}$$

设 $T = 3$，$O = (红, 白, 红)$，试用前向算法计算 $P(O|\lambda)$。



初始

$a_1(1) = \pi_1 b_1(O_1) = 0.2 \times 0.5 = 0.1$

$a_1(2) = \pi_2 b_2(O_1) = 0.4 \times 0.4 = 0.16$

$a_1(3) = \pi_3 b_3(O_1) = 0.4 \times 0.7 = 0.28$

递推 $a_2(1) = \left(\sum_{j=1}^{3} a_1(j)\, a_{j1}\right) b_1(O_2)$

$= (0.1 \times 0.5 + 0.16 \times 0.3 + 0.28 \times 0.2) \times 0.5$

$= (0.05 + 0.048 + 0.056) \times 0.5$

$= 0.077$

$a_2(2) = 0.1104$

$a_2(3) = 0.0606$

$a_3(1) = 0.04187$

$a_3(2) = 0.03551$

$a_3(3) = 0.05284$

终止：

$P(O|\lambda) = \sum_{i=1}^{3} a_3(i) = 0.04187 + 0.03551 + 0.05284 = 0.13022$

后向算法：

后向概率（backward prob）：

定义 10.3（后向概率） 给定隐马尔可夫模型 λ，定义在时刻 t 状态为 q_i 的条件下，从 $t+1$ 到 T 的部分观测序列为 $o_{t+1}, o_{t+2}, \cdots, o_T$ 的概率为后向概率，记作

$$\beta_t(i) = P(o_{t+1}, o_{t+2}, \cdots, o_T | i_t = q_i, \lambda) \tag{10.18}$$

可以用递推的方法求得后向概率 $\beta_t(i)$ 及观测序列概率 $P(O|\lambda)$。

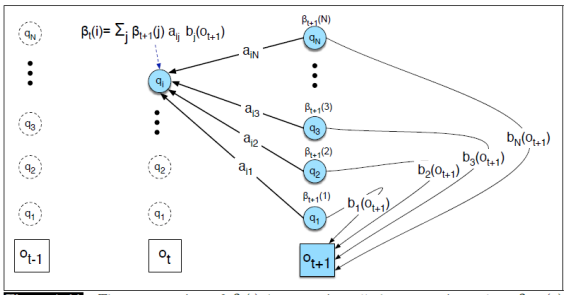


Figure A.11 The computation of $\beta_t(i)$ by summing all the successive values $\beta_{t+1}(j)$ weighted by their transition probabilities a_{ij} and their observation probabilities $b_j(o_{t+1})$.

算法 10.3（观测序列概率的后向算法）

输入：隐马尔可夫模型 λ，观测序列 O；

输出：观测序列概率 $P(O|\lambda)$。

（1）

$$\beta_T(i) = 1, \quad i = 1, 2, \cdots, N \tag{10.19}$$

（2）对 $t = T-1, T-2, \cdots, 1$

化简：

$$\beta_t(i) = \sum_{j=1}^{N} a_{ij} b_j(o_{t+1}) \beta_{t+1}(j), \quad i = 1, 2, \cdots, N \tag{10.20}$$

（3）

$$P(O|\lambda) = \sum_{i=1}^{N} \pi_i b_i(o_1) \beta_1(i) \tag{10.21}$$

Question 2: Decoding - Viterbi alg (最优路径问题)

Decoding: Given an observation sequence O and an HMM $\lambda = (A, B, \pi)$, discover the best hidden state sequence $Q = q_1 q_2 \cdots q_T$

(给定观测序列，求最有可能的对应的状态序列) $P(Q|O)$

定义：

1. 定义在时刻 t 状态为 i 的所有单个路径中概率最大值为

$$\delta_t(i) = \max_{q_1, q_2, \cdots, q_{t-1}} P(q_t = i, q_{t-1}, \cdots, q_1, O_t, \cdots, O_1 | \lambda) \quad i = 1, 2, \cdots, N$$

δ 的递推公式

$$\delta_{t+1}(i) = \max_{q_1, \cdots q_t} P(q_{t+1} = i, \cdots q_1, O_{t+1}, \cdots, O_1 | \lambda)$$

$$= \max_{1 \le j \le N} [\delta_t(j) a_{ji}] b_i(O_{t+1}) \quad \begin{matrix} i = 1, 2, \cdots, N \\ t = 1, 2, \cdots, T-1 \end{matrix}$$

2. 定义在时刻 t 状态为 i 的所有单个路径中概率最大的路径的第 $t-1$ 个结点为：

$$\Psi_t(i) = \arg\max_{1 \le j \le N} [\delta_{t-1}(j) a_{ji}] \quad i = 1, 2, \cdots, N$$

Viterbi Alg:

输入：模型 $\lambda = (A, B, \pi)$ 和观测 $O = (o_1, o_2, \cdots, o_T)$；

输出：最优路径 $I^* = (i_1^*, i_2^*, \cdots, i_T^*)$。

(1) 初始化

$$\delta_1(i) = \pi_i b_i(o_1), \quad i = 1, 2, \cdots, N$$

$$\Psi_1(i) = 0, \quad i = 1, 2, \cdots, N$$

(2) 递推。对 $t = 2, 3, \cdots, T$

$$\delta_t(i) = \max_{1 \le j \le N} [\delta_{t-1}(j) a_{ji}] b_i(o_t), \quad i = 1, 2, \cdots, N$$

$$\Psi_t(i) = \arg\max_{1 \le j \le N} [\delta_{t-1}(j) a_{ji}], \quad i = 1, 2, \cdots, N$$

(3) 终止

$$P^* = \max_{1 \le i \le N} \delta_T(i)$$

$$i_T^* = \arg\max_{1 \le i \le N} [\delta_T(i)]$$

(4) 最优路径回溯。对 $t = T-1, T-2, \cdots, 1$

$$i_t^* = \Psi_{t+1}(i_{t+1}^*)$$

求得最优路径 $I^* = (i_1^*, i_2^*, \cdots, i_T^*)$。

forward alg:

2) 递推 对 $t = 1, 2, \cdots, T-1$,

$$\alpha_{t+1}(i) = \left[\sum_{j=1}^{N} \alpha_t(j) a_{ji}\right] b_i(o_{t+1}), \quad i = 1, 2, \cdots, N \qquad (10.16)$$

Figure A.9 shows pseudocode for the Viterbi algorithm. Note that the Viterbi algorithm is identical to the forward algorithm except that it takes the **max** over the previous path probabilities whereas the forward algorithm takes the **sum**. Note also that the Viterbi algorithm has one component that the forward algorithm doesn't have: **backpointers**. The reason is that while the forward algorithm needs to produce an observation likelihood, the Viterbi algorithm must produce a probability and also the most likely state sequence. We compute this best state sequence by keeping track of the path of hidden states that led to each state, as suggested in Fig. A.10, and then at the end backtracing the best path to the beginning (the Viterbi **backtrace**).

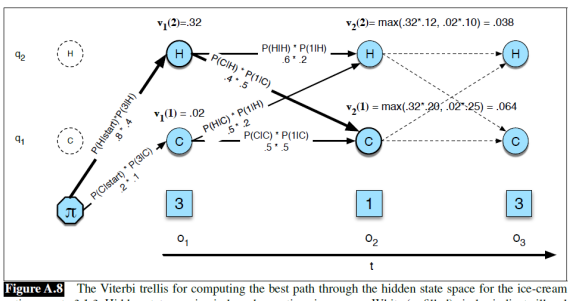


Figure A.8 The Viterbi trellis for computing the best path through the hidden state space for the ice-cream eating events *3 1 3*. Hidden states are in circles, observations in squares. White (unfilled) circles indicate illegal transitions. The figure shows the computation of $v_t(j)$ for two states at two time steps. The computation in each cell follows Eq. A.14: $v_t(j) = \max_{1 \le i \le N-1} v_{t-1}(i)\, a_{ij}\, b_j(o_t)$. The resulting probability expressed in each cell is Eq. A.13: $v_t(j) = P(q_0, q_1, \ldots, q_{t-1}, o_1, o_2, \ldots, o_t, q_t = j|\lambda)$.

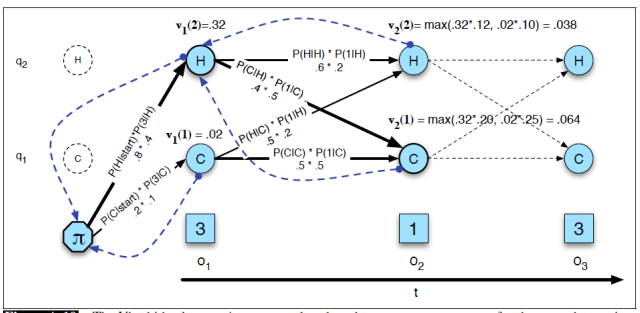


Figure A.10 The Viterbi backtrace. As we extend each path to a new state account for the next observation, we keep a backpointer (shown with broken lines) to the best path that led us to this state.

Question 3: learning (Forward-backward alg and Baum-Welch alg)

问题：给定 观测序列O 和对应状态序列Q，推断参数 A, B, π

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

$\lambda = (A, B, \pi)$

Baum-Welch algorithm

Review.

1. EM algorithm

算法 9.1（EM 算法）

输入：观测变量数据 Y，隐变量数据 Z，联合分布 $P(Y, Z|\theta)$，条件分布 $P(Z|Y, \theta)$；

输出：模型参数 θ。

（1）选择参数的初值 $\theta^{(0)}$，开始迭代；

（2）E 步：记 $\theta^{(i)}$ 为第 i 次迭代参数 θ 的估计值，在第 $i+1$ 次迭代的 E 步，计算

$$\begin{aligned} Q(\theta, \theta^{(i)}) &= E_Z[\log P(Y, Z|\theta)|Y, \theta^{(i)}] \\ &= \sum_Z \log P(Y, Z|\theta) P(Z|Y, \theta^{(i)}) \end{aligned} \tag{9.9}$$

这里，$P(Z|Y, \theta^{(i)})$ 是在给定观测数据 Y 和当前的参数估计 $\theta^{(i)}$ 下隐变量数据 Z 的条件概率分布；

（3）M 步：求使 $Q(\theta, \theta^{(i)})$ 极大化的 θ，确定第 $i+1$ 次迭代的参数的估计值 $\theta^{(i+1)}$

$$\theta^{(i+1)} = \arg\max_{\theta} Q(\theta, \theta^{(i)}) \tag{9.10}$$

（4）重复第（2）步和第（3）步，直到收敛。 ■

式 (9.9) 的函数 $Q(\theta, \theta^{(i)})$ 是 EM 算法的核心，称为 Q 函数（Q function）。

定义 9.1（Q 函数） 完全数据的对数似然函数 $\log P(Y, Z|\theta)$ 关于在给定观测数据 Y 和当前参数 $\theta^{(i)}$ 下对未观测数据 Z 的条件概率分布 $P(Z|Y, \theta^{(i)})$ 的期望称为 Q 函数，即

$$Q(\theta, \theta^{(i)}) = E_Z[\log P(Y, Z|\theta)|Y, \theta^{(i)}] \tag{9.11}$$

下面关于 EM 算法作几点说明：

步骤（1） 参数的初值可以任意选择，但需注意 EM 算法对初值是敏感的。

步骤（2） E 步求 $Q(\theta, \theta^{(i)})$。Q 函数式中 Z 是未观测数据，Y 是观测数据。注意，$Q(\theta, \theta^{(i)})$ 的第 1 个变元表示要极大化的参数，第 2 个变元表示参数的当前估计值。每次迭代实际在求 Q 函数及其极大。

步骤（3） M 步求 $Q(\theta, \theta^{(i)})$ 的极大化，得到 $\theta^{(i+1)}$，完成一次迭代 $\theta^{(i)} \to \theta^{(i+1)}$。后面将证明每次迭代使似然函数增大或达到局部极值。

步骤（4） 给出停止迭代的条件，一般是对较小的正数 $\varepsilon_1, \varepsilon_2$，若满足

$$\|\theta^{(i+1)} - \theta^{(i)}\| < \varepsilon_1 \quad 或 \quad \|Q(\theta^{(i+1)}, \theta^{(i)}) - Q(\theta^{(i)}, \theta^{(i)})\| < \varepsilon_2$$

则停止迭代。

EM: $\theta^{(t+1)} = \arg\max_{\theta} \int_z \log P(X, z|\theta) P(z|X, \theta^{(t)}) dz$

($\theta^{(t)}$: current step)

X: 观测数据 → O

Z: 隐变量 → I

θ: 参数 → λ

HMM中:

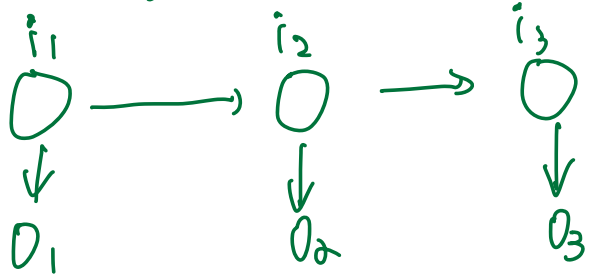
$$\lambda^{(t+1)} = \arg\max_{\lambda} \sum_I \log P(O, I|\lambda) P(I|O, \lambda^{(t)})$$

$$= \arg\max_{\lambda} \sum_I \log P(O, I|\lambda) \frac{P(I, O|\lambda^{(t)})}{P(O, \lambda^{(t)})} \leftarrow C$$

$$= \arg\max_{\lambda} \sum_I \log P(O, I|\lambda) P(I, O|\lambda^{(t)})$$

E步:

Q函数: $\sum_I \log P(O, I|\lambda) P(I, O|\lambda^{(t)})$



$I = i_1 i_2 i_3 \cdots i_T$ 状态序列

$O = O_1 O_2 O_3 \cdots O_T$

$Q = \{q_1, q_2, q_3, \cdots q_N\}$ 状态值

$V = \{v_1, v_2, v_3, \cdots v_N\}$

$P(I|\lambda) = \pi_1 a_{12} a_{23} \cdots a_{T-1,T}$

$P(O|I, \lambda) = b_1(O_1) b_2(O_2) \cdots b_T(O_T)$

$P(O, I|\lambda) = \pi_1 a_{12} b_1(O_1) a_{23} b_2(O_2) \cdots a_{T-1,T} b_T(O_T)$

∴ Q函数:

$$\sum_I \left[\log \pi_{i_1} + \sum_{t=1}^{T} \log a_{i_t i_{t+1}} + \sum_{t=1}^{T} \log (b_t O_t)\right] P(I, O|\lambda^{(t)})$$

Target : $\arg\max_{\lambda} P(O|\lambda), \quad \lambda = (A, B, \pi)$

M-Step:

① $\pi^{(t+1)} = \arg\max_{\pi} \sum_{I} \log \pi_{i_1} P(I, O|\lambda^{(t)})$

$= \arg\max_{\pi} \sum_{I} \log \pi_{i_1} P(i_1, i_2, i_3 \cdots i_n, O|\lambda^{(t)})$

$= \arg\max_{\pi} \sum_{i_1} \sum_{i_2} \sum_{i_3} \cdots \log \pi_{i_1} P(i_1, i_2, \cdots i_n, O|\lambda^{(t)})$

$= \arg\max_{\pi} \sum_{i_1} \log \pi_{i_1} P(i_1, O|\lambda^{(t)})$

$= \arg\max_{\pi} \sum_{i=1}^{N} \left[\log \pi_i \; P(O, i_1 = q_i | \lambda^{(t)}\right]$

Besides, $\sum_{i=1}^{N} \pi_i = 1$

Lagrange: $\sum_{i=1}^{N} \log \pi_i \; P(O, i_1 = q_i | \lambda^{(t)}) + \eta \left(\sum_{i=1}^{N} \pi_i - 1\right)$

$\frac{\partial \lambda}{\partial \pi_i} = \sum_{i=1}^{N} \left[\frac{-P(O, i_1 = q_i|\lambda^t)}{\pi_i} + \eta\right] = 0$

$= \sum_{i=1}^{N} \left[P(O, i_1 = q_i|\lambda^t) + \pi_i \eta\right] = 0$ ($\sum_{i=1}^{N} \pi_i = 1$)

↓ 边缘

$= P(O|\lambda^t) + \eta = 0$ ($\sum_{x_1, x_2 \cdots x_N} P(x, y) = P(y)$)

$\Rightarrow \eta = -P(O|\lambda^t)$

↳ $P(O, i_1 = q_i | x^t) + \pi_i \eta = 0$

$\pi_i = -\frac{1}{\eta} P(O, i_1 = q_i | x^t)$

$\pi_i^{(t+1)} = \frac{P(O, i_1 = q_i | x^t)}{P(O | x^t)}$

$\pi^{(t+1)} = (\pi_1^{(t+1)}, \pi_2^{(t+1)} \cdots \pi_N^{t+1})$

②