隐式马尔科夫模型指南

1. 马尔科夫模型

我们经常会处理由不相互独立的随机变量组成的序列，序列中的每个变量依赖于出现在他前面的元素的集合。对于这样的序列，从直观上来看，我们并不需要了解序列中所有过去的随机变量的值，例如预测图书馆明天有多少本书，我们可能只需要知道今天图书馆有多少本书，而不需要知道上周图书馆有多少本书，更不必考虑上一年的情况。也就是说，在给定当前元素的情况下，序列中未来的元素和序列中过去的元素是独立的。

当一个随机序列 X 满足有限视野性时，我们称其为马尔科夫链。

- 有限视野性： $\mathrm{P}(x_{t+1}|x_1 \dots x_t) = P(x_{t+1}|x_t)$，即序列中下一个元素仅仅与当前元素相关。

（文中讨论的马尔科夫模型均指一阶马尔科夫模型）

这时候，X 可以通过状态转移矩阵描述：

$$a_{i,j} = P(x_{t+1} = s_j | x_t = s_i)$$

同时，我们需要指定不同初始状态的概率

$$\pi_i = P(X_i = s_i)$$

在马尔科夫模型中，状态序列的概率很容易计算，就是计算转移矩阵中的概率乘积：

$$\mathrm{P}(X_1, \dots, X_i) = P(X_1)P(X_1|X_2) \dots P(X_i|X_{i-1})$$

由于现在模型的状态是已知的，它的状态序列或者状态序列的函数就是模型的输出，这样的模型被称为显马尔科夫模型。

2. 隐马尔科夫模型

在隐马尔科夫模型（下称 HMM）中，模型的状态是未知的，我们知道的只是模型的一些概率性的输出。如课上讲的饮料机，它有可乐和冰茶两个状态，但状态是未知的，我们能观测到的只是不同状态下的输出（可乐，冰茶，柠檬汽水）。

HMM 能够有效地解决系统中底层事件引发表层事件的问题：例如分词或者词性标注问

题，我们可以认为文本中的词（相当于 HMM 的输出）是由底层的分词标注或词性标注的序列（相当于 HMM 的状态）产生的。

HMM 模型可以通过 EM 算法进行有效的训练，即在足够的训练数据下，可以通过自动学习得到描述观测数据的模型参数。

HMM 模型可以用一个五元组（S, K, π, A, B）来表示。S 表示状态的集合，K 表示输出的集合，π表示初始状态的概率，A 表示状态转移矩阵（a_{ij}表示从状态 i 转移到状态 j 的概率），B 表示发射概率（b_{ijk}表示从状态 i 转移到状态 j 时，发射输出 k 的概率）。HMM 的输出发射分为两类：

a. T 时刻发射的输出依赖于 T 时刻和 T+1 时刻的状态，这类 HMM 称为弧发射 HMM。
b. T 时刻发射的输出只依赖于 T 时刻的状态，这类 HMM 称为状态发射 HMM。

通过简单建立发射概率矩阵，使得$\forall j, j'$, 都有$b_{ijk} = b_{ij'k}$，我们就能把状态发射 HMM 看作是弧发射 HMM。因此，我们讨论 HMM 时的输出发射时，一般指弧发射 HMM。

3. HMM 的基本问题

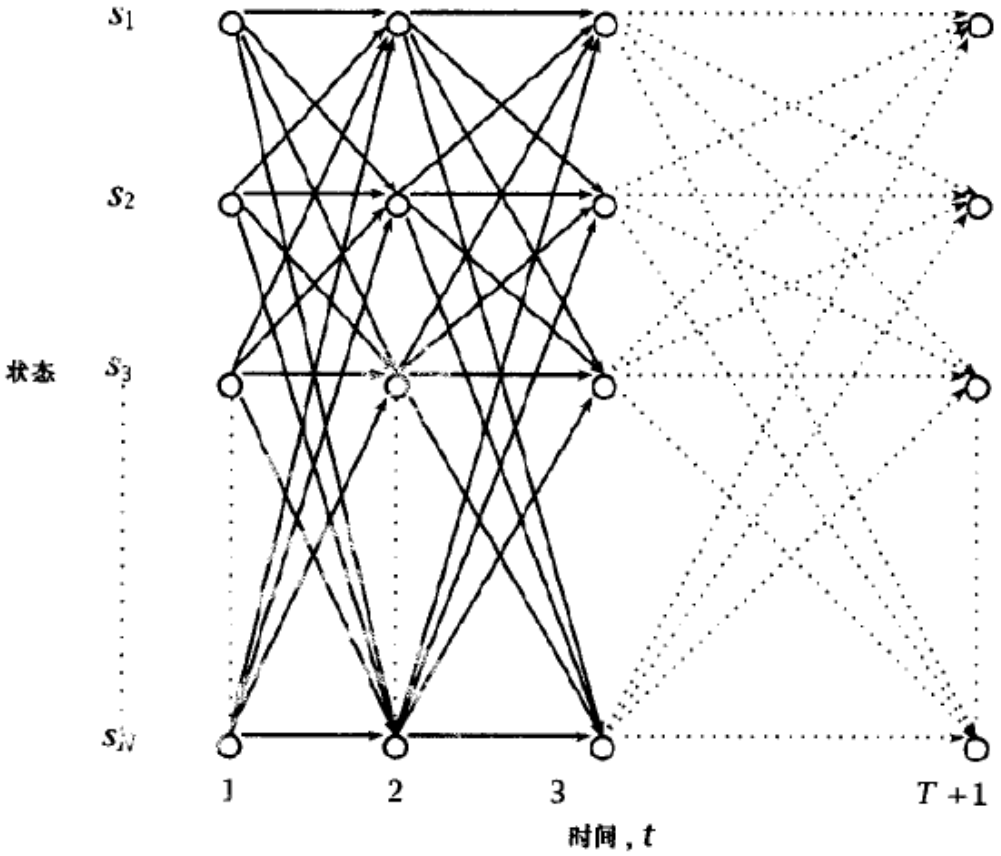
HMM 有三个基本问题：

1. 给出一个模型 μ，计算出某个观测序列发生的概率。
2. 给出观测序列和模型 μ，选择最能够解释观测序列的状态序列。
3. 给出观测序列，找到最能够解释这个观测序列的模型 μ。

下面依次介绍解决这三个问题的方法。

1. 计算观测序列的概率

直接对可能发生的状态序列发生的观测概率求和的效率很低（需要 $(2T+1) * N^{T+1}$）次乘法），我们可以利用 DP 的思想，记住部分之前得到的结果而不是每次都重新计算，这里我们使用格路算法。



这里，我们构造一个状态-时间的方阵（格路），根据上一时刻处于某个状态的概率来计算下一时刻处于某个状态的概率。基于这个方阵，格路算法可以通过前向或后向两个过程之一进行：

- **前向过程**

我们在格路的每个点(S_i, t)中存储在t时刻以状态S_i结束的概率，这个概率称为前向变量$\alpha_i(t)$，利用以下的前向算法计算。这里O表示观测序列: $o_1 o_2 o_3 \ldots o_T$

a. 初始化

$$\alpha_i(1) = \pi_i$$

b. 推导

$$\alpha_j(t+1) = \sum_{i=1}^{N} \alpha_i(t) a_{ij} b_{ij o_t}$$

c. 求和

$$\mathrm{P}(\mathrm{O} \mid \mu) = \sum_{i=1}^{N} \alpha_i(T+1)$$

得到给定观测序列O的概率。这个过程只需要$2N^2T$次乘法。

- **后向过程**

我们也可以在格路中的每个点记录给定当前状态S_i和时间t时观测序列中剩余部分的概率之和，即

$$\beta_i(\mathrm{t}) = P(o_t \dots o_T | X_t = i, \mu)$$

从右到左计算格路中的后向变量:

a. 初始化

$$\beta_i(T+1) = 1$$

b. 推导

$$\beta_i(t) = \sum_{j=1}^{N} a_{ij} b_{ij o_t} \beta_j(t+1)$$

c. 求和

$$\mathrm{P(O \mid \mu)} = \sum_{i=1}^{N} \pi_i \beta_i(1)$$

- **前向 – 后向结合**

更一般地，观察到：

$$\mathrm{P(O}, x_t = \mathrm{i \mid \mu)}$$

$= \mathrm{P}(o_1 o_2 o_3 \dots o_T, x_t = \mathrm{i \mid \mu})$

$= \mathrm{P}(o_1 o_2 o_3 \dots o_{t-1}, x_t = \mathrm{i \mid \mu}) * \mathrm{P}(o_t \dots o_T \mid o_1 o_2 o_3 \dots o_{t-1}, x_t = \mathrm{i}, \mu)$

$= \mathrm{P}(o_1 o_2 o_3 \dots o_{t-1}, x_t = \mathrm{i \mid \mu}) * \mathrm{P}(o_t \dots o_T \mid x_t = \mathrm{i}, \mu)$

$= \alpha_i(t)\beta_i(t)$

即

$$\mathrm{P(O \mid \mu)} = \sum_{i=1}^{N} \alpha_i(t)\beta_i(t)$$

此时 t 可以取任意值，前面两个过程都是前向-后向结合过程的特例。

2. 确定最佳状态序列

给定观测序列的前提下，计算最可能的完整状态路径的方法是 Viterbi 算法：

- Viterbi 算法

我们在格路中每个节点存储到达这个节点最可能路径 Q 的概率$\delta_j(t)$，同时用变量$\varphi_j(t)$记录导致这条最可能路径的入弧节点。这样，我们就可以用动态规划的方法求解出通过整条格路的最可能路径。

a. 初始化

$$\delta_j(1) = \pi_j$$

b. 推导

$$\delta_j(t+1) = max_{1\le i\le N}\delta_i(t)a_{ij}b_{ijo_t}$$

即计算从上一时刻的不同状态i到当前时刻状态j并发射出输出o_t的概率，并选取其最大值，即为到达这个节点最可能路径

$$\varphi_j(t+1) = argmax_{1\le i\le N}\delta_i(t)a_{ij}b_{ijo_t}$$

将最可能路径的上一时刻状态节点存储起来

c. 回溯出路径

$$X_{T+1} = argmax_{1\le i\le N}\delta_i(t+1)$$

$$X_t = \varphi_j(t+1)$$

从最后时刻概率最大的节点开始，依次读取当前节点的入弧节点，形成状态序列。

3. HMM 的参数估计

给定一个特定观测序列 O，我们希望确定模型$\mu = (A, B, \pi)$的参数值，并且要求这个模型能够给出观测序列的最佳解释，即

$$argmax_{\mu} P(O_{training}|\mu)$$

目前没有已知的解析方法来选择模型μ，但我们可以通过迭代使其局部最大化，这种算法被称为 Baum-Welch 算法，它是 EM（Expectation Maximum）算法的一个特例。这个算法的工作方式如下：

- 最初我们不知道模型的信息，但我们用某个可能是预先选取或随机选取的模型算出观测序列的概率;
- 查看计算过程，发现哪个状态转移或者符号发射可能出现的次数最多;
- 增加它们的概率，使得模型可以为观测序列给出更高的概率;

算法具体执行方式如下：

- 定义$P_t(i, j)$为给定观测序列 O 的情况下，在 t 时刻经过弧 ij 的概率。则

$$\begin{aligned} P_t(i, j) &= P(X_t = i, X_{t+1} = j|O, \mu) \\ &= \frac{P(X_t = i, X_{t+1} = j, O|\mu)}{P(O|\mu)} \\ &= \frac{\alpha_i(t) * a_{ij} b_{ijo_t} * \beta_j(t+1)}{\sum_{m=1}^{N} \alpha_m(t)\beta_m(t)} \\ &= \frac{\alpha_i(t) * a_{ij} b_{ijo_t} * \beta_j(t+1)}{\sum_{m=1}^{N}\sum_{n=1}^{N} \alpha_m(t) a_{mn} b_{mno_t} \beta_n(t+1)} \end{aligned}$$

- 令$\gamma_i(t) = \sum_{j=1}^{N} P_t(i, j)$,，则有

  在 O 中从状态 i 发出的转移的期望数目 $= \sum_{t=1}^{T} \gamma_i(t)$

  在 O 中从状态 i 到 j 的转移的期望数目 $= \sum_{t=1}^{T} P_t(i, j)$

- 我们从模型μ开始，通过运行 O 来估计每个模型参数值的期望，接下来改变模型来最大化使用较多的路径的值。**重估公式如下**：

$$\pi'_i = 时间t = 1\ 时状态i的期望频率 = \gamma_i(1)$$

$$a'_{ij} = \frac{状态i到状态j的期望转移数}{状态i出发的期望转移数} = \frac{\sum_{t=1}^{T} P_t(i, j)}{\sum_{t=1}^{T} \gamma_i(t)}$$

$$b'_{ijk} = \frac{状态i到状态k的转移发出输出k的期望}{状态i到状态j的期望转移数} = \frac{\sum_{\{t:o_t=k, 1\le t\le T\}} P_t(i, j)}{\sum_{t=1}^{T} P_t(i, j)}$$

- 这样，我们就生成了一个新的模型μ′,Baum 证明了这个新模型的一个结论：

$$P(O \mid \mu') \geq P(O \mid \mu)$$

- 我们可以迭代重复这个过程，直到重估的结果没有显著的提高为止。但是，这个参数重估的方法并不能保证得到最好的模型，因为重估过程可能会停留在一个局部极值点上。当然，Baum-Welch 重估法对 HMM 通常是很有效的。

4. 参数值的初始化

重估过程只能保证我们找到一个局部极值，如果我们想要找到一个全局极值，就要使得 HMM 的重估尽量在全局附近的参数空间开始，我们可以通过预先估计参数的最佳值来实现这一点。