Hidden Markov Model 01 Background

Chen Gong

07 January 2020

机器学习大致可以分为两个派别，也就是频率派和贝叶斯派的方法，这个之前，我们都有过详细的说明。这里再大致的回顾一下。

频率派的思想就衍生出了统计学习方法，说白了统计学习方法的重点在于优化，找 loss function。频率派的方法可以分成三步，1. 定义 Model，比如 $f(w) = w^T x + b$；2. 寻找策略 strategy，也就是定义 Loss function；3. 求解，也就是优化的方法，比如梯度下降 (GD)，随机梯度下降 (SGD)，牛顿法，拟牛顿法等等。

贝叶斯派的思想也就衍生出了概率图模型。概率图模型重点研究的是一个 Inference 的问题，我们要求的是一个后验概率分布 $P(Z|X)$，其中 X 为观测变量，Z 为隐变量。实际上就是一个积分问题，为什么呢？因为贝叶斯框架中的归一化因子需要对整个状态空间进行积分，非常的复杂。代表性的有前面讲到的 MCMC，MCMC 的提出才是彻底的把贝叶斯理论代入到实际的运用中。

1 概率图模型回顾

概率图模型，如果不考虑时序的关系，我们可以大致的分为：有向图的 Bayesian Network 和无向图的 Markov Random Field (Markov Network)。这样，我们根据分布获得的样本之间都是 iid (独立同分布) 的。比如 Gaussian Mixture Model (GMM)，我们从 $P(X|\theta)$ 的分布中采出 N 个样本 $\{x_1, x_2, \cdots, x_n\}$。N 个样本之间都是独立同分布的。也就是对于隐变量 Z，观测变量 X 之间，我们可以假设 $P(X|Z) = \mathcal{N}(\mu, \Sigma)$，这样就可以引入我们的先验信息，从而简化 X 的复杂分布。

如果引入了时间的信息，也就是 x_i 之间不再是 iid 的了，我们称之为 Dynamic Model。模型如下所示：

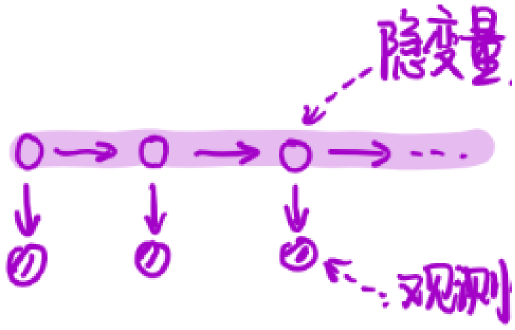

图 1: Dynamic Model 拓扑结构图

Dynamic Model 可以从两个层面来看，横着看就是 time 的角度，如果是竖着看就可以表达为 $P(X|Z)$ 的形式，也就是 Mixture 的形式。概率系统根据状态与状态之间的关系，可以分为两类。

如果是离散的则有 HMM 算法。

如果是连续的，按照线性和非线性可以分为 Kalman Filter 和 Paricle Filter。

2 HMM 算法简介

Hidden Markov Model 的拓扑结构图如下所示：

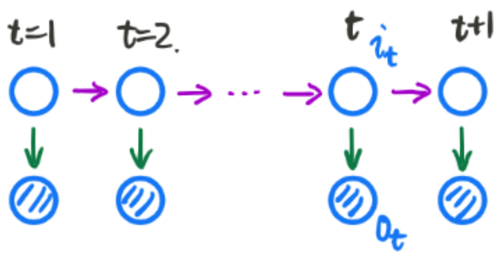

图 2: Hidden Markov Model 拓扑结构图

大家看到这个模型就会觉得和上一讲提到的，MCMC 模型方法有点类似。HMM 可以看做一个三元组 $\lambda = (\pi, \mathcal{A}, \mathcal{B})$。其中：

π：是初始概率分布。

$\mathcal{A}$：状态转移矩阵。

$\mathcal{B}$：发射矩阵。

拓扑结构图的第二行为观测变量，观测变量 o：$o_1, o_2, \cdots, o_t, \cdots \leftarrow \mathcal{V} = v_1, v_2, \cdots, v_M$。其中 $\mathcal{V}$ 是观察变量 o 的值域，代表每一个观测变量 o_i 可能有 M 个状态。

拓扑结构图的第一行为状态变量，状态变量 i：$i_1, i_2, \cdots, i_t, \cdots \leftarrow \mathcal{Q} = q_1, q_2, \cdots, q_N$。其中 $\mathcal{Q}$ 是状态变量 i 的值域，代表每一个状态变量 i 可能有 N 个状态。

$\mathcal{A} = [a_{ij}]$ 表示状态转移矩阵，$a_{ij} = P(i_{t+1} = q_j | i_t = q_i)$。

$\mathcal{B} = [b_j(k)]$ 表示发射矩阵，$b_j(k) = P(o_t = V_k | i_t = q_j)$。

而 π 是什么意思呢？假设当 t 时刻的隐变量 i_t，可能有 $\{q_1, q_2, \cdots, q_N\}$ 个状态，而这些状态出现的概率分别为 $\{p_1, p_2, \cdots, p_N\}$。这就是一个关于 i_t 隐变量的离散随机分布。

$\mathcal{A}$ 表示为各个状态转移之间的概率。

$\mathcal{B}$ 表示为观测变量和隐变量之间的关系。

2.1 两个假设

这是有关 Hidden Markov Model 的两个假设：

1. 齐次 Markov 假设 (无后向性)；2. 观察独立假设。

齐次马尔可夫假设：未来与过去无关，只依赖与当前的状态。也就是：

$$P(i_{t+1}|i_t, i_{t-1}, \cdots, i_1, o_t, \cdots, o_1) = P(i_{t+1}|i_t) \tag{1}$$

2. **观测独立假设：**

$$P(o_t|i_t, i_{t-1}, \cdots, i_1, o_t, \cdots, o_1) = P(o_t|i_t) \tag{2}$$

2.2 三个问题

1. Evaluation 的问题，我们要求的问题就是 $P(O|\lambda)$。也就是前向后向算法，给定一个模型 λ，求出观测变量的概率分布。

2. Learning 的问题，λ 如何求的问题。也就是 $\lambda_{MLE} = \arg\max_\lambda P(O|\lambda)$。求解的方法是 EM 算法和 Baum Welch 算法。

3. Decoding 的问题，状态序列为 I，也就是隐变量序列，$\hat{I} = \arg\max_I P(I|O, \lambda)$。也就是在在观测变量 O 和 λ 的情况下使隐变量序列 I 出现的概率最大。而这个问题大致被分为预测和滤波。

预测问题为：$P(i_{t+1}|o_1, \cdots, o_t)$；也就是在已知当前观测变量的情况下预测下一个状态，也就是 Viterbi 算法。

滤波问题为：$P(i_t|o_1, \cdots, o_t)$；也就是求 t 时刻的隐变量。

Hidden Markov Model，可以被我们总结成一个模型 $\lambda = (\pi, \mathcal{A}, \mathcal{B})$，两个假设，三个问题。而其中我们关注得最多的就是 Decoding 的问题。

Hidden Markov Model 02 Evaluation

Chen Gong

08 January 2020

Evaluation 的问题可以被我们描述为：给定一个 λ，如何求得 $P(O|\lambda)$。也就是在给定模型 λ 的情况下，求某个观测序列出现的概率。

1 模型求解

对于 $P(O|\lambda)$ 我们利用概率的基础知识进行化简可以得到：

$$P(O|\lambda) = \sum_I P(O, I|\lambda) = \sum_I P(O|I, \lambda)P(I|\lambda) \tag{1}$$

其中 $\sum_I$ 表示所有可能出现的隐状态序列；$\sum_I P(O|I,\lambda)$ 表示在某个隐状态下，产生某个观测序列的概率；$P(I|\lambda)$ 表示某个隐状态出现的概率。

那么：

$$\begin{aligned} P(I|\lambda) =& P(i_1, \cdots, i_T|\lambda) \\ =& P(i_T|i_1, \cdots, i_{T-1}, \lambda) \cdot P(i_1, \cdots, i_{T-1}|\lambda) \end{aligned} \tag{2}$$

根据 Hidden Markov Model 两个假设中的,齐次马尔可夫假设,我们可以得到:$P(i_T|i_1, \cdots, i_{T-1}, \lambda) = P(i_T|i_{T-1}) = a_{i_{T-1}, i_T}$。后面按照一样的思路进行迭代就可以了。那么我们继续对公式 (2) 进行化简可以得到：

$$\begin{aligned} P(i_T|i_1, \cdots, i_{T-1}, \lambda) \cdot P(i_1, \cdots, i_{T-1}|\lambda) =& P(i_T|i_{T-1}) \cdot P(i_1, \cdots, i_{T-1}|\lambda) \\ =& a_{i_{T-1}, i_T} \cdot a_{i_{T-2}, i_{T-1}} \cdots a_{i_1, i_2} \cdot \pi(a_{i_1}) \\ =& \pi(a_{i_1}) \prod_{t=2}^{T} a_{i_{t-1}, i_t} \end{aligned} \tag{3}$$

然后，运用观察独立假设，我们可以知道：

$$\begin{aligned} P(O|I, \lambda) =& P(o_1, o_2, \cdots, o_T|I, \lambda) \\ =& \prod_{t=1}^{T} P(o_t|I, \lambda) \\ =& \prod_{t=1}^{T} b_{i_t}(o_t) \end{aligned} \tag{4}$$

那么，结合公式 (2-5)，我们可以得到：

$$\begin{aligned} P(O|\lambda) &= \sum_{I} \pi(a_{i_1}) \prod_{t=2}^{T} a_{i_{t-1},i_t} \prod_{t=1}^{T} b_{i_t}(o_t) \\ &= \sum_{i_1} \cdot \sum_{i_2} \cdots \sum_{i_T} \pi(a_{i_1}) \prod_{t=2}^{T} a_{i_{t-1},i_t} \prod_{t=1}^{T} b_{i_t}(o_t) \end{aligned} \tag{5}$$

因为一共有 T 个状态，每个状态有 N 种可能，所以算法复杂度为 $\mathcal{O}(N^T)$。既然这样直接求太困难了，我们就需要另外想办法。

2 Forward Algorithm

下面，我们首先展示一下 Hidden Markov Model 的拓扑结构图。

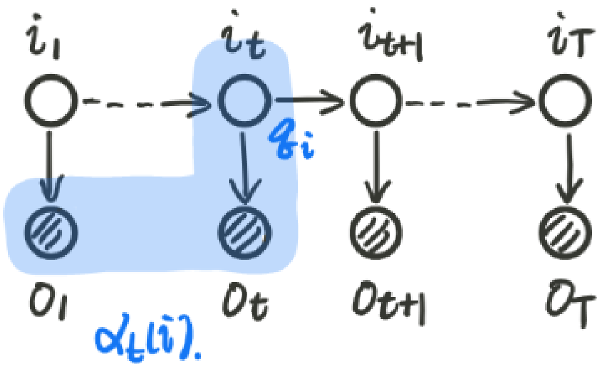

图 1: 矩阵与列向量的乘法

我们记，$\alpha_t(i) = P(o_1, \cdots, o_t, i_t = q_i|\lambda)$，这个公式表示的是在之前所有的观测变量的前提下求出当前时刻的隐变量的概率。那么：

$$P(O|\lambda) = \sum_{i=1}^{N} P(O, i_t = q_i|\lambda) = \sum_{i=1}^{N} \alpha_t(i) \tag{6}$$

其中，$\sum_{i=1}^{N}$ 表示对所有可能出现的隐状态情形求和，而 $\alpha_t(i)$ 表示对所有可能出现的隐状态情形求和。我们的想法自然就是寻找 $\alpha_t(i)$ 和 $\alpha_t(i+1)$ 之间的关系，这样通过递推，我们就可以得到整个观测序列出现的概率。那么，下面我们来进行推导：

$$\alpha_t(i+1) = P(o_1, \cdots, o_t, o_{t+1}, i_{t+1} = q_j|\lambda) \tag{7}$$

因为 $\alpha_t(i)$ 里面有 $i_t = q_j$，我们就要想办法把 i_t 给塞进去，所以：

$$\begin{aligned} \alpha_t(i+1) &= P(o_1, \cdots, o_t, o_{t+1}, i_{t+1} = q_j|\lambda) \\ &= \sum_{i=1}^{N} P(o_1, \cdots, o_t, o_{t+1}, i_t = q_i, i_{t+1} = q_j|\lambda) \\ &= \sum_{i=1}^{N} P(o_{t+1}|o_1, \cdots, o_t, i_t = q_i, i_{t+1} = q_j, \lambda) \cdot P(o_1, \cdots, o_t, i_t = q_i, i_{t+1} = q_j|\lambda) \end{aligned} \tag{8}$$

又根据观测独立性假设,我们可以很显然的得到 $P(o_{t+1}|o_1,\cdots,o_t,i_t=q_i,i_{t+1}=q_j,\lambda)=P(o_{t+1}|i_{t+1}=q_j)$。所以:

$$
\begin{aligned}
\alpha_t(i+1) &= \sum_{i=1}^{N} P(o_{t+1}|o_1,\cdots,o_t,i_t=q_i,i_{t+1}=q_j,\lambda)\cdot P(o_1,\cdots,o_t,i_t=q_i,i_{t+1}=q_j|\lambda) \\
&= \sum_{i=1}^{N} P(o_{t+1}|i_{t+1}=q_j)\cdot P(o_1,\cdots,o_t,i_t=q_i,i_{t+1}=q_j|\lambda)
\end{aligned} \tag{9}
$$

看到这个化简后的公式，我们关注一下和 $\alpha_t(i)$ 相比，好像还多了一项 $i_{t+1}=q_j$，我们下一步的工作就是消去它。所以:

$$
P(o_1,\cdots,o_t,i_t=q_i,i_{t+1}=q_j|\lambda)=P(i_{t+1}=q_j|o_1,\cdots,o_t,i_t=q_i,\lambda)\cdot P(o_1,\cdots,o_t,i_t=q_i|\lambda) \tag{10}
$$

根据齐次马尔可夫性质，我们可以得到 $P(i_{t+1}=q_j|o_1,\cdots,o_t,i_t=q_i,\lambda)=P(i_{t+1}=q_j=i_t=q_i)$。所以根据以上的推导，我们可以得到:

$$
\begin{aligned}
\alpha_{t+1}(j) = &\sum_{i=1}^{N} P(o_{t+1}|i_{t+1}=q_j)\cdot P(i_{t+1}=q_j|i_t=q_i)\cdot P(o_1,\cdots,o_t,i_t=q_i|\lambda) \\
=& b_j(o_{t+1})\cdot a_{ij}\cdot \alpha_t(i)
\end{aligned} \tag{11}
$$

经过上述的推导，我们就成功的得到了 $\alpha_{t+1}(j)$ 和 $\alpha_t(i)$ 之间的关系。通过这个递推关系，就可以遍历整个 Markov Model 了。这个公式是什么意思呢? 它可以被我们表达为，所有可能出现的隐变量状态乘以转移到状态 j 的概率，乘以根据隐变量 i_{t+1} 观察到 o_{t+1} 的概率，乘上根据上一个隐状态观察到的观察变量的序列的概率。

我们可以用一个图来进行表示:

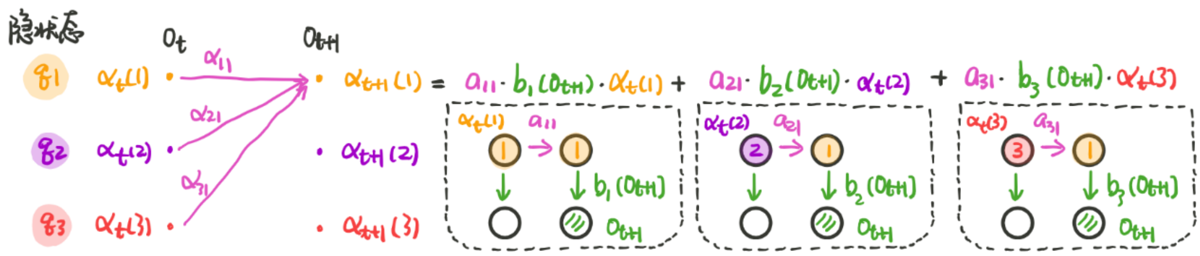

图 2: Hidden Markov Model 前向传播示意图

其实读神经网络了解的同学就会发现，这实际上和前向传播神经网络非常的像，实际上就是状态的值乘以权重。也就是对于上一个隐状态的不同取值分别计算概率之后再求和。这样每次计算，有隐状态的状态空间数为 N，序列的长度为 T，那么总的时间复杂度为 $\mathcal{O}(TN^2)$。

3 Backward Algorithm

后向概率的推导实际上比前向概率的理解要难一些，前向算法实际上是一个联合概率，而后向算法则是一个条件概率，所以后向的概率实际上比前向难求很多。

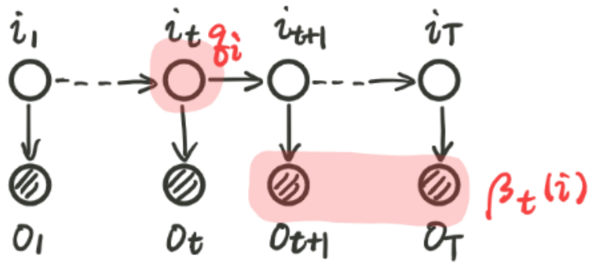

图 3: Hidden Markov Model 后向算法示意图

我们设 $\beta_t(i) = P(o_{t+1}, \cdots, o_T | i_t = q_i, \lambda)$，以此类推，$\beta_t(1) = P(o_2, \cdots, o_T | i_1 = q_i, \lambda)$。我们的目标是计算 $P(O|\lambda)$ 的概率，我们首先来推导一下这个公式：

$$
\begin{aligned}
P(O|\lambda) =& P(o_1, o_2, \cdots, o_N|\lambda) \\
=& \sum_{i=1}^{N} P(o_1, o_2, \cdots, o_N, i_1 = q_i|\lambda) \\
=& \sum_{i=1}^{N} P(o_1, o_2, \cdots, o_N|i_1 = q_i, \lambda) P(i_1 = q_i|\lambda) \\
=& \sum_{i=1}^{N} P(o_1|o_2, \cdots, o_N, i_1 = q_i, \lambda) \cdot P(o_2, \cdots, o_N, i_1 = q_i|\lambda) \cdot \pi_i \\
=& \sum_{i=1}^{N} P(o_1|i_1 = q_i, \lambda) \cdot \beta_1(i) \cdot \pi_i \\
=& \sum_{i=1}^{N} b_i(o_1) \cdot \pi_i \cdot \beta_1(i)
\end{aligned} \tag{12}
$$

现在我们已经成功的找到了 $P(O|\lambda)$ 和第一个状态之间的关系。其中，π_i 为某个状态的初始状态的概率，$b_i(o_1)$ 表示为第 i 个隐变量产生第 1 个观测变量的概率，$\beta_1(i)$ 表示为第一个观测状态确定以后生成后面观测状态序列的概率。结构图如下所示：

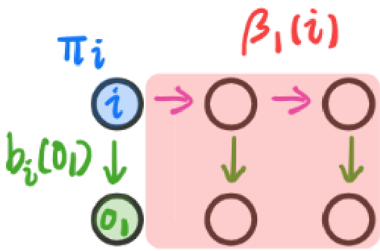

图 4: $P(O|\lambda)$ 与第一个状态之间的关系结构图

那么，我们下一步要通过递推，找到最后一个状态与第一个状态之间的关系。下面做如下的推导：

$$\begin{aligned}\beta_t(i) =& P(o_{t+1},\cdots,o_T|i_t=q_i) \\ =& \sum_{j=1}^{N} P(o_{t+1},\cdots,o_T,i_{t+1}=q_j|i_t=q_i) \\ =& \sum_{j=1}^{N} P(o_{t+1},\cdots,o_T|i_{t+1}=q_j,i_t=q_i)\cdot\underbrace{P(i_{t+1}=q_j|i_t=q_i)}_{a_{ij}} \\ =& \sum_{j=1}^{N} P(o_{t+1},\cdots,o_T|i_{t+1}=q_j)\cdot a_{ij} \\ =& \sum_{j=1}^{N} P(o_{t+1}|o_{t+2}\cdots,o_T,i_{t+1}=q_j)\cdot\underbrace{P(o_{t+2}\cdots,o_T|i_{t+1})}_{\beta_{t+1}(j)}\cdot a_{ij} \\ =& \sum_{j=1}^{N} P(o_{t+1}|i_{t+1}=q_j)\cdot\beta_{t+1}(j)\cdot a_{ij} \\ =& \sum_{j=1}^{N} b_j(o_{t+1})\cdot\beta_{t+1}(j)\cdot a_{ij}\end{aligned} \tag{13}$$

其中第三行到第四行的推导 $P(o_{t+1},\cdots,o_T|i_{t+1}=q_j,i_t=q_i)=P(o_{t+1},\cdots,o_T|i_{t+1}=q_j)$ 使用的马尔可夫链的性质，每一个状态都是后面状态的充分统计量，与之前的状态无关。通过这样的迭代从后往前推，我们就可以得到 $\beta_i(1)$ 的概率，从而推断出 $P(O|\lambda)$。整体的推断流程图如下图所示：

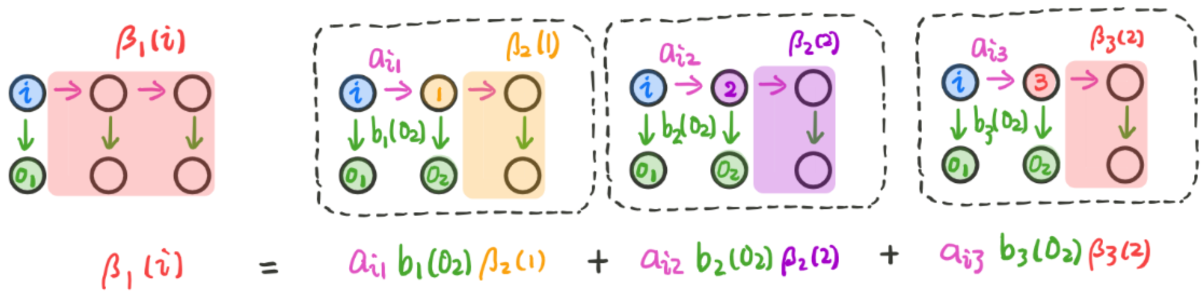

图 5: Hidden Markov Model 后向算法拓扑结构图

Hidden Markov Model 03 Learning

Chen Gong

09 January 2020

首先我们回顾一下，上一节讲的有关 Evaluation 的问题。Evaluation 可以被我们描述为在已知模型 λ 的情况下，求观察序列的概率。也就是：

$$P(O|\lambda) = \sum_I P(O, I|\lambda) = \sum_{i_1} \cdots \sum_{i_T} \pi_{i_1} \prod_{t=2}^T a_{i_{t-1},i_t} \prod_{t=1}^T b_{i_1}(o_t) \tag{1}$$

此时的算法复杂度为 $\mathcal{O}(N^T)$。算法的复杂度太高了，所以，就有了后来的 forward 和 backward 算法。那么就有如下定义：

$$\begin{aligned}
&\alpha_t(i) = P(o_1, \cdots, o_t, i_t = q_i|\lambda) \\
&\beta_t(i) = P(o_{t+1}, \cdots, o_T|i_t = q_i, \lambda) \\
&\alpha_T(i) = P(O, i_T = q_i) \rightarrow P(O|\lambda) = \sum_{i=1}^N \alpha_T(i) \\
&\beta_1(i) = P(o_2, \cdots, o_T|i_1 = q_i, \lambda) \rightarrow P(O|\lambda) = \sum_{i=1}^N \pi_i b_i(o_1)\beta_1(i)
\end{aligned} \tag{2}$$

而使用 forward 和 backward 算法的复杂度为 $\mathcal{O}(TN^2)$。这一节，我们就要分析 Learning 的部分，Learning 就是要在已知观测数据的情况下求参数 λ，也就是：

$$\lambda_{MLE} = \arg\max_\lambda P(O|\lambda) \tag{3}$$

1 Learning

我们需要计算的目标是：

$$\lambda_{MLE} = \arg\max_\lambda P(O|\lambda) \tag{4}$$

又因为：

$$P(O|\lambda) = \sum_{i_1} \cdots \sum_{i_T} \pi_{i_1} \prod_{t=2}^T a_{i_{t-1},i_t} \prod_{t=1}^T b_{i_1}(o_t) \tag{5}$$

对这个方程的 λ 求偏导，实在是太难算了。所以，我们考虑使用 EM 算法。我们先来回顾一下 EM 算法：

$$\theta^{(t+1)} = \arg\max_\theta \int_z \log P(X, Z|\theta) \cdot P(Z|X, \theta^{(t)}) dZ \tag{6}$$

而 $X \rightarrow O$ 为观测变量；$Z \rightarrow I$ 为隐变量，其中 I 为离散变量；$\theta \rightarrow \lambda$ 为参数。那么，我们可以将公式改写为：

$$\lambda^{(t+1)} = \arg\max_{\lambda} \sum_{I} \log P(O, I|\lambda) \cdot P(I|O, \lambda^{(t)}) \tag{7}$$

这里的 $\lambda^{(t)}$ 是一个常数，而：

$$P(I|O, \lambda^{(t)}) = \frac{P(I, O|\lambda^{(t)})}{P(O|\lambda^{(t)})} \tag{8}$$

并且 $P(O|\lambda^{(t)})$ 中 $\lambda^{(t)}$ 是常数，所以这项是个定量，与 λ 无关，所以 $\frac{P(I,O|\lambda^{(t)})}{P(O|\lambda^{(t)})} \propto P(I, O|\lambda^{(t)})$。所以，我们可以将等式 (7) 改写为：

$$\lambda^{(t+1)} = \arg\max_{\lambda} \sum_{I} \log P(O, I|\lambda) \cdot P(I, O|\lambda^{(t)}) \tag{9}$$

这样做有什么目的呢？很显然这样可以把 $\log P(O, I|\lambda)$ 和 $P(I, O|\lambda^{(t)})$ 变成一种形式。其中，$\lambda^{(t)} = (\pi^{(t)}, \mathcal{A}^{(t)}, \mathcal{B}^{(t)})$，而 $\lambda^{(t+1)} = (\pi^{(t+1)}, \mathcal{A}^{(t+1)}, \mathcal{B}^{(t+1)})$。

我们定义：

$$Q(\lambda, \lambda^{(t)}) = \sum_{I} \log P(O, I|\lambda) \cdot P(O, I|\lambda^{(t)}) \tag{10}$$

而其中，

$$P(O|\lambda) = \sum_{i_1} \cdots \sum_{i_T} \pi_{i_1} \prod_{t=2}^{T} a_{i_{t-1}, i_t} \prod_{t=1}^{T} b_{i_1}(o_t) \tag{11}$$

所以，

$$Q(\lambda, \lambda^{(t)}) = \sum_{I} \left[\left(\log \pi_{i_1} + \sum_{t=2}^{T} \log a_{i_{t-1}, i_t} + \sum_{t=1}^{T} \log b_{i_t}(o_t) \right) \cdot P(O, I|\lambda^{(t)}) \right] \tag{12}$$

2 以 $\pi^{(t+1)}$ 为例

这小节中我们以 $\pi^{(t+1)}$ 为例，在公式 $Q(\lambda, \lambda^{(t)})$ 中，$\sum_{t=2}^{T} \log a_{i_{t-1}, i_t}$ 与 $\sum_{t=1}^{T} \log b_{i_t}(o_t)$ 与 π 无关，所以，

$$\begin{aligned}
\pi^{(t+1)} &= \arg\max_{\pi} Q(\lambda, \lambda^{(t)}) \\
&= \arg\max_{\pi} \sum_{I} [\log \pi_{i_1} \cdot P(O, I|\lambda^{(t)})] \\
&= \arg\max_{\pi} \sum_{i_1} \cdots \sum_{i_T} [\log \pi_{i_1} \cdot P(O, i_1, \cdots, i_T|\lambda^{(t)})]
\end{aligned} \tag{13}$$

我们观察 $\{i_2, \cdots, i_T\}$ 就可以知道，联合概率分布求和可以得到边缘概率。所以：

$$\begin{aligned}
\pi^{(t+1)} &= \arg\max_{\pi} \sum_{i_1} [\log \pi_{i_1} \cdot P(O, i_1|\lambda^{(t)})] \\
&= \arg\max_{\pi} \sum_{i=1}^{N} [\log \pi_i \cdot P(O, i_1 = q_i|\lambda^{(t)})] \qquad (s.t. \sum_{i=1}^{N} \pi_i = 1)
\end{aligned} \tag{14}$$

2.1 拉格朗日乘子法求解

根据拉格朗日乘子法，我们可以将损失函数写完：

$$\mathcal{L}(\pi,\eta) = \sum_{i=1}^{N} \log \pi_i \cdot P(O, i_1 = q_i|\lambda^{(t)}) + \eta(\sum_{i=1}^{N} \pi_i - 1) \tag{15}$$

使似然函数最大化，则是对损失函数 $\mathcal{L}(\pi,\eta)$ 求偏导，则为：

$$\frac{\mathcal{L}}{\pi_i} = \frac{1}{\pi_i} P(O, i_1 = q_i|\lambda^{(t)}) + \eta = 0 \tag{16}$$

$$P(O, i_1 = q_i|\lambda^{(t)}) + \pi_i \eta = 0 \tag{17}$$

又因为 $\sum_{i=1}^{N} \pi_i = 1$，所以，我们将公式 (17) 进行求和，可以得到：

$$\sum_{i=1}^{N} P(O, i_1 = q_i|\lambda^{(t)}) + \pi_i \eta = 0 \Rightarrow P(O|\lambda^{(t)}) + \eta = 0 \tag{18}$$

所以，我们解得 $\eta = -P(O|\lambda^{(t)})$，从而推出：

$$\pi_i^{(t+1)} = \frac{P(O, i_1 = q_i|\lambda^{(t)})}{P(O|\lambda^{(t)})} \tag{19}$$

进而，我们就可以推导出 $\pi^{(t+1)} = (\pi_1^{(t+1)}, \pi_2^{(t+1)}, \cdots, \pi_N^{(t+1)}$。而 $\mathcal{A}^{(t+1)}$ 和 $\mathcal{B}^{(t+1)}$ 也都是同样的求法。这就是大名鼎鼎的 Baum Welch 算法，实际上思路和 EM 算法一致。不过在 Baum Welch 算法诞生之前，还没有系统的出现 EM 算法的归纳。所以，这个作者还是很厉害的。

Hidden Markov Model 04 Decoding

Chen Gong

10 January 2020

Decoding 问题可被我们描述为:

$$\hat{I} = \arg\max_{I} P(I|O, \lambda) \tag{1}$$

也就是在给定观察序列的情况下，寻找最大概率可能出现的隐概率状态序列。也有人说 Decoding 问题是预测问题,但是实际上这样说是并不合适的。预测问题应该是,$P(o_{t+1}|o_1, \cdots, o_t)$ 和 $P(i_{t+1}|o_1, \cdots, o_t)$，这里的 $P(i_1, \cdots, i_t|o_1, \cdots, o_t)$ 看成是预测问题显然是不合适的。

1 Decoding Problem

下面我们展示一下 Hidden Markov Model 的拓扑模型:

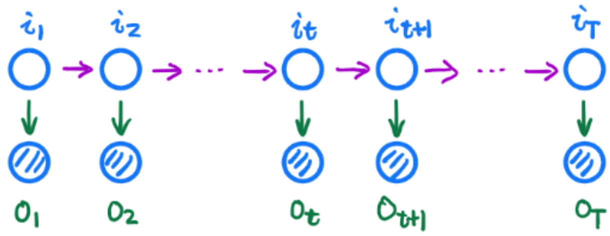

图 1: Hidden Markov Model 的拓扑模型

这里实际上就是一个动态规划问题，这里的动态规划问题实际上就是最大概率问题，只不过将平时提到的最大距离问题等价于最大概率问题，理论上都是一样的。每个时刻都有 N 个状态，所有也就是从 N^T 个可能的序列中找出概率最大的一个序列，实际上就是一个动态规划问题，如下图所示:

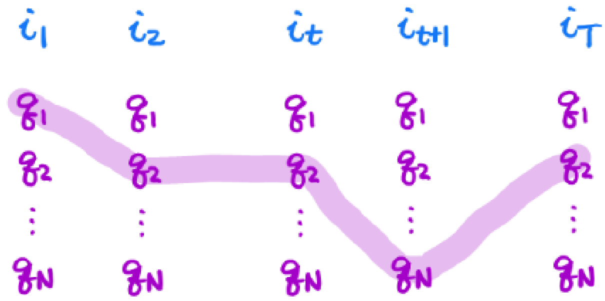

图 2: Decoding 的动态规划问题

我们假设：

$$\delta_t(i) = \max_{i_1,\cdots,i_{t-1}} P(o_1,\cdots,o_t,i_1,\cdots,i_{t-1},i_t=q_i) \tag{2}$$

这个等式是什么意思呢？也就是当 t 个时刻是 q_i，前面 $t-1$ 个随便走，只要可以到达 q_i 这个状态就行，而从中选取概率最大的序列。我们下一步的目标就是在知道 $\delta_t(i)$ 的情况下如何求 $\delta_t(i+1)$，那么这样就能通过递推来求得知道最后一个状态下概率最大的序列。$\delta_t(i+1)$ 的求解方法如下所示：

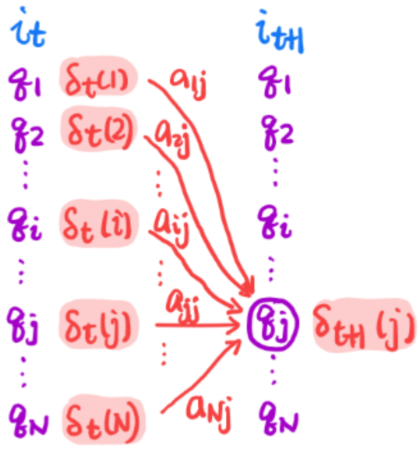

图 3: 根据 $\delta_t(i)$ 求解 $\delta_t(i+1)$ 的方法示意图

所以，

$$\begin{aligned}\delta_{t+1}(j) &= \max_{i_1,\cdots,i_t} P(o_1,\cdots,o_{t+1},i_1,\cdots,i_t,i_{t+1}=q_j) \\ &= \max_{i_1,\cdots,i_t} \delta_t(i)\cdot a_{ij}\cdot b_j(o_{t+1})\end{aligned} \tag{3}$$

这就是 Viterbi 算法，但是这个算法最后求得的是一个值，没有办法求得路径，如果要想求得路径，我们需要引入一个变量：

$$\varphi_{t+1}(j) = \arg\max_{1\leq i\leq N} \delta_t(i)\cdot a_{ij}\cdot b_j(o_{t+1}) \tag{4}$$

这个函数用来干嘛的呢？他是来记录每一次迭代过程中经过的状态的 index。这样我们最终得到的 $\{\varphi_1,\varphi_2,\cdots,\varphi_T\}$，就可以得到整个路径了。

Hidden Markov Model 05 Conclusion

Chen Gong

11 January 2020

Hidden Markov Model 实际上是一个 Dynamic Model。我们以 Guassian Mixture Model (GMM) 为例。对于一个观测状态，在隐变量状态给定的情况下，是符合一个 Gaussian Distribution，也就是 $D(O|i_1) \sim \mathcal{N}(\mu, \Sigma)$。如果，加入了 time 的因素就是 Hidden Markov Model，而其中 $\{i_1, i_2, \cdots, i_T\}$ 是离散的就行，这些我们在第一章的背景部分有过讨论。而观测变量 o_1 是离散的还是连续的都不重要。

1 Hidden Markov Model 简述

Hidden Markov Model，可以用一个模型，两个假设和是三个问题来描述。一个模型就是指 $\lambda = (\pi, \mathcal{A}, \mathcal{B})$。其中，$\pi$：指的是初始概率分布；$\mathcal{A}$：指的是状态转移矩阵；$\mathcal{B}$：指的是发射矩阵，也就是在已知隐变量的情况下，得到观测变量的概率分布。

两个假设：1. 齐次马尔可夫模型，马尔科夫性质中非常重要的一条。2. 观测独立假设，也就是观测变量只和当前的隐变量状态有关。

三个问题：1. Evaluation：$P(O|\lambda)$，也就是在在已知模型的情况下，求观测变量出现的概率。2. Learning：$\hat{\lambda} = \arg\max_\lambda P(O|\lambda)$，在已知观测变量的情况下求解隐马尔可夫模型的参数。3. Decoding：$P(I|O) = P(i_1, \cdots, i_t|o_1, \cdots, o_t)$，用公式的语言描述就是 $\hat{I} = \arg\max_I P(I, O|\lambda)$。

2 Dynamic Model

Dynamic Model 实际上就是一个 State Space Model，通常我们可以将 Dynamic Model 的问题分成两类。第一类为 Learning 问题，即为，参数 λ 是未知的，通过数据来知道参数是什么；第二类就是 Inference 问题，也就是在 λ 未知的情况下，推断后验概率。实际上，我们需要求的就是 $P(Z|X)$，其中 $X = \{x_1, x_2, \cdots, x_N\}$ 而数据之间是非 i.i.d 的。

Inference 问题大概可以被我们分成四类，Filtering Problem；Smoothing；Prediction Problem；Decoding。

2.1 Learning

Learing 问题中 λ 是已知的，$\lambda_{MLE} = \arg\max_\lambda P(X|\lambda)$。我们采用的是 Baum Welch Algorithm，算法思想上和 EM 算法类似，实际上也是 Forward-Backward 算法。

2.2 Inference

这一小节中，我们将分别来介绍 Filtering；Smoothing；Prediction；Decoding，四个问题。

2.2.1 Decoding

这里前面已经做出过详细的描述了，这里就不再展开进行描述了，主要可以概括为：在已知观测数据序列的情况下，求得出现概率最大的隐变量序列，被我们描述为：$Z = \arg\max_z P(z_1, \cdots, z_t|x_1, \cdots, x_t)$。我们使用的一种动态规划的算法，被称为 Viterbi Algorithm。

2.3 Evaluation

在还有大家应该见得比较多的 Prob of Evidence 问题，也就是：$P(X|\theta) = P(x_1, \cdots, x_t\theta)$。我们通俗的称之为证据分布，实际上就是我们前面讲到的 Evaluation 方法。也就是在已知参数的情况下，求观测数据序列出现的概率，用公式描述即为：$P(X|\theta) = P(x_1, x_2, \cdots, x_t|\theta)$。

2.3.1 Filtering

实际上是一个 Online-Learning 的过程，也就是如果不停的往模型里面喂数据，我们可以得到概率分布为：$P(z_t|x_1, \cdots, x_t)$。所以 Filtering 非常的适合与 on-line update。我们要求的这个就是隐变量的边缘后验分布。为什么叫滤波呢？这是由于我们求的后验是 $P(z_t|x_1, \cdots, x_t)$，运用到了大量的历史信息，比 $P(z_t|x_t)$ 的推断更加的精确，可以过滤掉更多的噪声，所以被我们称为"过滤"。求解过程如下所示：

$$P(z_t|x_{1:t}) = \frac{P(z_t, x_1, \cdots, x_t)}{P(x_1, \cdots, x_t)} = \frac{P(z_t, x_1 : x_t)}{\sum_{z_t} P(z_t, x_1 : x_t)} \propto P(z_t, x_1 : x_t) \tag{1}$$

2.3.2 Smoothing

Smoothing 问题和 Filtering 问题的性质非常的像，不同的是，Smoothing 问题需要观测的是一个不变的完整序列。对于 Smoothing 问题的计算，前面的过程和 Filtering 一样，都是：

$$P(z_t|x_{1:T}) = \frac{P(z_t, x_1, \cdots, x_T)}{P(x_1, \cdots, x_T)} = \frac{P(z_t, x_1 : x_T)}{\sum_{z_t} P(z_t, x_1 : x_T)} \propto P(z_t, x_1 : x_T) \tag{2}$$

同样因为 $\sum_{z_t} P(z_t, x_1 : x_T)$ 是一个归一化常数，我们这里不予考虑。下面的主要问题是关于 $P(z_t, x_1 : x_T)$ 如何计算，我们来进行推导：

$$\begin{aligned} P(x_{1:T}, z_t) =& P(x_{1:t}, x_{t+1:T}, z_t) \\ =& P(x_{t+1:T}|x_{1:t}, z_t) \cdot \underbrace{P(x_{1:t}, z_t)}_{\alpha_t} \end{aligned} \tag{3}$$

推导到了这里就是要对 $P(\underbrace{x_{t+1:T}}_{C} | \underbrace{x_{1:t}}_{A}, \underbrace{z_t}_{B})$ 进行分析，在这个概率图模型中，符合如下结构：

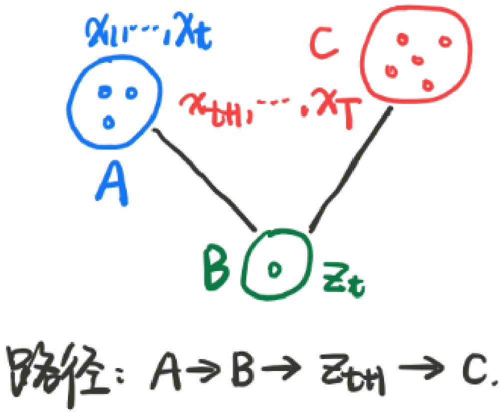

图 1: A，B，z_{t+1}，C，概率图结构图

根据概率图模型中提到 D-Separation 中，我们可以很简单的得出，$A \perp C|B$。所以，$P(x_{t+1:T}|x_{1:t}, z_t) = P(x_{t+1:T}|x_{1:t}, z_t = \beta_t)$。所以，我们可以得到:

$$P(x_{1:T}, z_t) = \alpha_t \cdot \beta_t \tag{4}$$

那么，最终得到的就是:

$$P(z_t|x_{1:T}) \propto P(x_{1:T}, z_t) = \alpha_t \beta_t \tag{5}$$

所以，我们需要同时用到 Forward Algorithm 和 Backward Algorithm，所以，被我们称为 Forward-Backward Algorithm。

2.3.3 Prediction

预测问题，大体上被我们分成两个方面:

$$\begin{aligned} P(z_{t+1}|x_1, \cdots, x_t) &= \sum_{z_t} P(z_{t+1}, z_t|x_1, \cdots, x_t) \\ &= \sum_{z_t} \underbrace{P(z_{t+1}|z_t, x_1, \cdots, x_t)}_{P(z_{t+1}|z_t)} \underbrace{P(z_t|z_t, x_1, \cdots, x_t)}_{Filtering} \end{aligned} \tag{6}$$

$$\begin{aligned} P(x_{t+1}|x_1, \cdots, x_t) &= \sum_{z_{t+1}} P(x_{t+1}, z_{t+1}|x_1, \cdots, x_t) \\ &= \underbrace{P(x_{t+1}|z_{t+1}, x_1, \cdots, x_t)}_{P(x_{t+1}|z_{t+1})} \cdot \underbrace{P(z_{t+1}|x_1, \cdots, x_t)}_{Formula(6)} \end{aligned} \tag{7}$$

公式 (7) 选择从 z_{t+1} 进行积分的原因是因为想利用齐次马尔科夫性质。实际上求解的过程大同小异都是缺什么就补什么。

其实，我们已经大致的介绍了 Dynamic Model 的几种主要模型，后面我们会详细的来解释线性动态系统。