Lecture 6-7: ASR — 隐马尔科夫模型

Hidden Markov Models

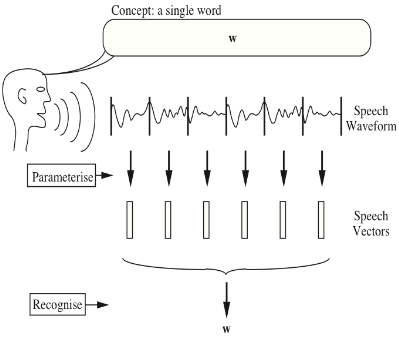
Kai Yu and Yanmin Qian

Cross Media Language Intelligence Lab (X-LANCE)
Department of Computer Science & Engineering
Shanghai Jiao Tong University

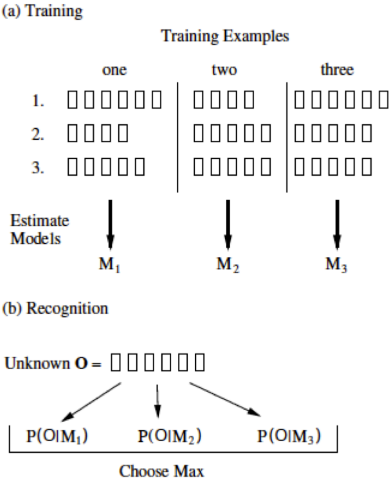
Spring 2021

目录

- 孤立词语音识别
  - 确定化方法：动态时间规整（Dynamic Time Warping）
  - 似然模型方法：随机过程（Stochastic process）
- 马尔科夫链
- 隐马尔科夫模型 (I)
  - 评估：前向算法（Forward algorithm）
  - 解码：维特比算法（Viterbi algorithm）
- 隐马尔科夫模型 (II)
  - HMM 参数估计
  - 期望最大化（EM）算法用于 HMM 参数估计
    - Baum-Welch (forward-backward) 前后向算法
    - Gaussian v.s. GMM



Concept: a single word
w
Speech
Waveform
Parameterise
Speech
Vectors
Recognise
w



(a) Training
Training Examples
one
two
three
1.
2.
3.
Estimate
Models
M1
M2
M3
(b) Recognition
Unknown O =
P(O|M1)
P(O|M2)
P(O|M3)
Choose Max

确定化方法

整个过程完全基于从波形中提取的特征向量序列 $\mathbf{O} = [\mathbf{o}_1, \cdots, \mathbf{o}_T]$

1. 提取待识别词的特征向量序列
2. 计算待识别词的特征向量序列和所有已知词的参考向量序列 $\mathbf{V}_k$ 之间的距离 $D(\mathbf{O}, \mathbf{V}_k)$
3. 找到和待识别词距离最近的词作为识别结果

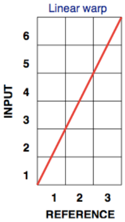
$$\hat{k} = \arg\min_k D(\mathbf{O}, \mathbf{V}_k)$$

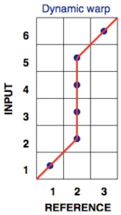
难点: 如何去计算这些变长序列之间的距离？
（类比 WER 的计算）

通过 DTW 算法计算序列 $\mathbf{X}=[\mathbf{x}_1,\cdots,\mathbf{x}_M]$ 和序列 $\mathbf{Y}=[\mathbf{y}_1,\cdots,\mathbf{y}_N]$ 的代价，其中，定义一个 $M\times N$ 维的距离矩阵 **D**，有

$$D(i,j)=\min\begin{cases} D(i-1,j)+C_I & \texttt{insertion} \\ D(i,j-1)+C_D & \texttt{deletion} \\ D(i-1,j-1)+d(\mathbf{x},\mathbf{y}) & \texttt{substitution} \end{cases}$$

其中 C_I 和 C_D 分别插入和删除的代价，$d(\mathbf{x},\mathbf{y})=\sqrt{(\mathbf{x}-\mathbf{y})^\top(\mathbf{x}-\mathbf{y})}$ 是替代错误的代价.





动态时间规整 — Dynamic Time Warping

Recap: Levenshtein Distance

```
int DTWDistance(s: array [1..n], t: array [1..m]) {
    DTW := array [0..n, 0..m]

    for i := 0 to n
        for j := 0 to m
            DTW[i, j] := infinity
    DTW[0, 0] := 0

    for i := 1 to n
        for j := 1 to m
            cost := d(s[i], t[j])
            DTW[i, j] := cost + minimum(DTW[i-1, j  ],    // insertion
                                        DTW[i  , j-1],    // deletion
                                        DTW[i-1, j-1])    // match

    return DTW[n, m]
}
```

DTW 用于语音识别

DTW 可以找到两个数据序列之间的最佳对齐序列，从而计算得到两者之间的实际距离。它可以作为一种简单的模板匹配方法来得到语音识别的结果:

1. 训练: 为每一个词录制一个样例语音，来作为这个词的模板
2. 解码: 录制测试语音，然后通过 DTW 算法计算和各模板之间的距离

优势: 训练简单, 仅需要一个训练样本

劣势: 不可靠。对模板的依赖程度很高，推广性较差。

Q: 如何有效地处理多训练样本的问题？

用于孤立词语音识别的似然模型方法

统计语音识别:

$$\hat{\mathbf{W}} = \arg\max_{\mathbf{W}} P(\mathbf{W}|\mathbf{O}) = \arg\max_{\mathbf{W}} p(\mathbf{O}|\mathbf{W})P(\mathbf{W})$$

孤立词识别: 假定所有候选词都是等似然的，目标是找到声学似然最高的词。

$$k = \arg\max_{i} p(\mathbf{O}|w_i)$$

其中 $p(\mathbf{O}|w_i)$ 需要完成特征向量长度的归一化。

隐马尔科夫模型 (Hidden Markov Model, HMM) 被广泛地用于表示/得到 $p(\mathbf{O}|w_i)$。

马尔科夫链

一阶的情况

马尔科夫链是一个**随机过程**，它允许产生一个在**时间**和**取值**上都是离散的一个随机序列。这个随机过程的输出是一个长度为 T 的状态的序列，可以表示为

$$\mathbf{x} = [x_1, x_2, \cdots, x_T]$$

其中，状态 $x \in \{q_1, \cdots, q_M\}$，

$$P(\mathbf{x}) = P(x_1) \prod_{t=2}^{T} P(x_t | x_1, \cdots, x_{t-1})$$

一阶马尔科夫近似: x_t 仅仅依赖于它的前一时刻 x_{t-1}

$$P(\mathbf{x}) = P(x_1) \prod_{t=2}^{T} P(x_t | x_{t-1})$$

马尔科夫链的参数集合

以下符号和公式定义了所有状态，而随机序列是通过从一个状态转移到另一个状态产生的。其中，状态的选择通过转移概率来确定。

- **初始状态似然**

$$\pi_i = P(q_i) \quad \sum_i P(q_i) = 1 \quad 1 \leqslant i \leqslant M$$

- **状态转移概率**

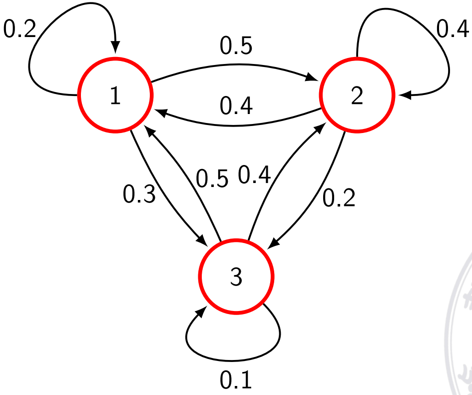
$$\begin{aligned} a_{ij} &= P(x_t = q_j | x_{t-1} = q_i) & 1 \leqslant i \leqslant M, \quad 1 \leqslant j \leqslant M \\ \sum_j P(x_t = q_j | x_{t-1} &= q_i) = 1 & 1 \leqslant j \leqslant M \end{aligned}$$

稳态马尔科夫链: 参数是常数（不是时变的）

马尔科夫链的一个例子（I）

马尔科夫链可以用图示形式表示出来，e.g.

校内停留位置的一个三状态的一阶马尔科夫链，状态编号为 $S = \{1, 2, 3\}$，分别表示宿舍、教室、食堂：

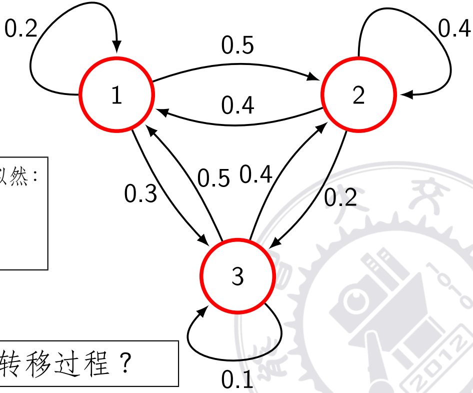


马尔科夫链的一个例子（I）

马尔科夫链也可以用转移矩阵（transition matrix）表示，e.g.

校内停留位置的一个三状态的一阶马尔科夫链，状态编号为 $S = \{1, 2, 3\}$，分别表示宿舍、教室、食堂：

$$\mathbf{P} = \begin{array}{c} \\ 1 \\ 2 \\ 3 \end{array} \begin{array}{c} \begin{array}{ccc} 1 & 2 & 3 \end{array} \\ \begin{pmatrix} 0.2 & 0.5 & 0.3 \\ 0.4 & 0.4 & 0.2 \\ 0.5 & 0.4 & 0.1 \end{pmatrix} \end{array}$$



从状态 i 经过 n 步后到达状态 j 的似然：

$$a_{ij}^{(n)} = (\mathbf{P}^n)_{ij} = \big(\underbrace{\mathbf{P} \times \mathbf{P} \times \cdots \times \mathbf{P}}_{n\text{个}}\big)_{ij}$$

Q: 如何画高阶马尔科夫链的转移过程？

马尔科夫链的一个例子（I）

马尔科夫链也可以用转移矩阵（transition matrix）表示，e.g.

校内停留位置的一个三状态的二阶马尔科夫链，状态编号为 $S = \{1, 2, 3\}$，分别表示宿舍、教室、食堂：

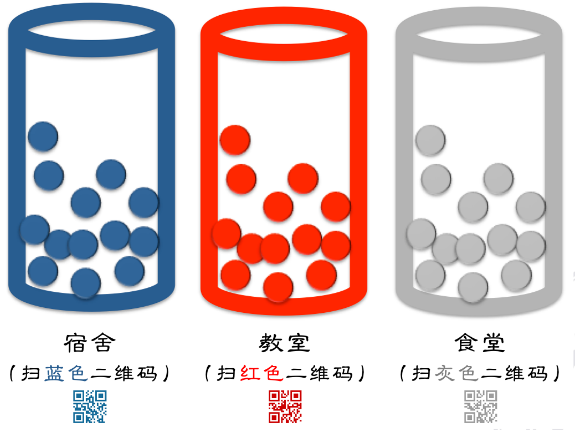
$$
\mathbf{P} = \begin{array}{c} \\ 11 \\ 21 \\ 31 \\ 12 \\ 22 \\ 32 \\ 13 \\ 23 \\ 33 \end{array}
\begin{array}{c} \begin{array}{ccc} 1 & 2 & 3 \end{array} \\
\begin{pmatrix}
0.1 & 0.5 & 0.4 \\
0.2 & 0.5 & 0.3 \\
0.1 & 0.6 & 0.3 \\
0.4 & 0.4 & 0.2 \\
0.5 & 0.2 & 0.3 \\
0.6 & 0.2 & 0.2 \\
0.5 & 0.4 & 0.1 \\
0.4 & 0.5 & 0.1 \\
0.7 & 0.2 & 0.1
\end{pmatrix} \end{array}
$$

每一行 ij 表示前两个状态依次为 i 和 j 时转移到第 $1, 2, 3$ 个状态的似然。

马尔科夫链的一个例子（II）

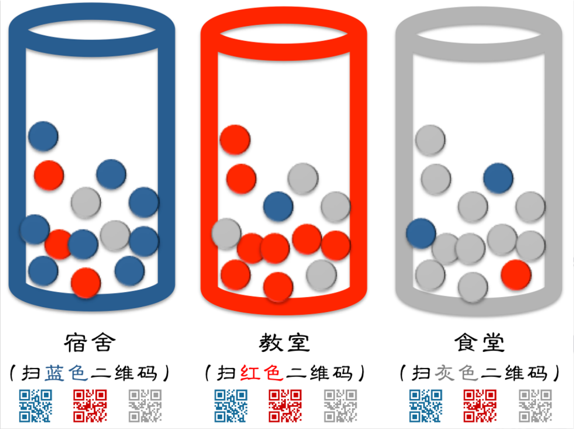
桶中取球

随机选择一个桶，并从桶中随机取一个球拿出来。



马尔科夫链过程可以用来对拿出来的球组成的序列（状态输出是**确定的**序列）进行建模。

随机选择一个桶，并从桶中随机取一个球拿出来。



隐马尔科夫模型可以用来对这种状态输出是不确定的序列进行建模。

- **马尔科夫链:** 每一个状态的输出是确定的：状态序列 = 观测序列

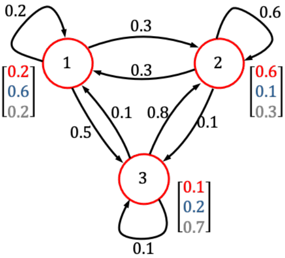
$$\mathbf{o} = s$$

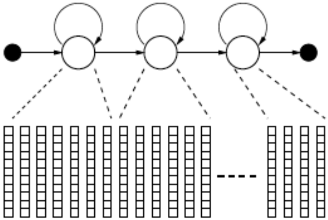
- **隐马尔科夫模型 (HMM):** 每一个状态的输出是随机的，它由一个似然分布来决定：状态序列 $\neq$ 观测序列

$$b(\mathbf{o}) = P(\mathbf{o}|s)$$

HMM 可以认为是一个有限状态转录机，它将一个特征向量序列 (表示语音波形)，通过状态机状态之间的转移，得到一个状态序列，来表示音素，音节，或词序列。

HMM 的一个示例





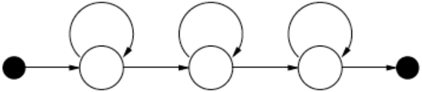
按照**观测序列**的特性，HMM 可以分为：

- **离散：** $b(\mathbf{o}) = P(\mathbf{o}|\mathrm{s})$, e.g.: 天气情况
- **连续：** $b(\mathbf{o}) = p(\mathbf{o}|\mathrm{s})$, e.g.: Speech

HMM 中有两种类型的**序列**：

- **状态序列：** 隐含的，潜在的，不可见的状态序列
- **观测序列：** 实际观测到的特征序列

假定用一个 3 状态 自左至右的 HMM 模型去建模一个由 $(\{a, b, c\})$ 三个观察值组成的长度为 15 的离散符号序列

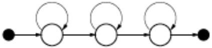


	State 1	State 2	State 3	Scores
PDF	[0.8 0.1 0.1]	[0.1 0.8 0.1]	[0.1 0.1 0.8]	
Seq 1	a a a a a	b b b b b	c c c c c	0.8^{15}
Seq 2	a a c b a	b b a b b	c c b c c	$0.8^{11} \times 0.1^{4}$
Seq 3	a a a a c	b b c a b	c c b b c	$0.8^{10} \times 0.1^{5}$
Seq 4	c c c c c	b b b b b	a a a a a	$0.8^{5} \times 0.1^{10}$

注意：Seq 4 的似然度是最低的，跳转序列是 $c \rightarrow b \rightarrow a$，它正好是模型描述的实际序列 $a \rightarrow b \rightarrow c$ 的逆过程。

HMM 拓扑结构示例

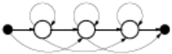
- 自左至右



→

Possible state sequences:
1, 1, 1, 2, 2, 3, 3, 3
1, 1, 1, 1, 2, 3, 3, 3, 3

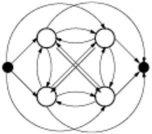
- 有跨越状态的跳转



→

Possible state sequences:
1, 1, 1, 3, 3
2, 2, 2
2, 2, 2, 3, 3

- 全连接



→

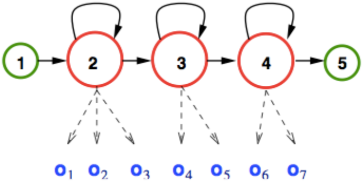
Possible state sequences:
2, 2, 3, 1, 3, 4, 2, 1
3, 3, 2, 4, 2, 2, 3, 3
4, 2, 4, 3, 4

用 HMM 来建模语音

语音信号

- 由准平稳的语音片段组成
- 具有有限的长度
- 可以用小的声学单元（“音素”）的连接来进行较好的描述

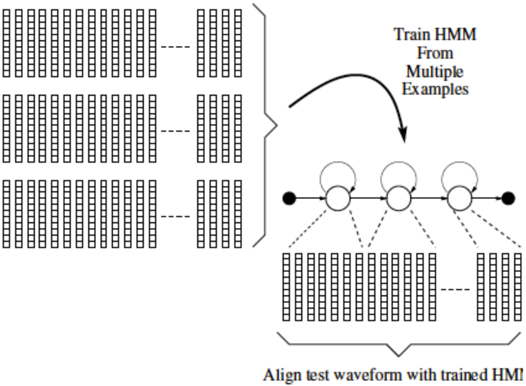
自左至右的 HMM 结构可以用于对以上这些特性进行建模：



注意：图示中的 HMM 初始状态和结束状态是非发射 (non-emitting) 状态，可以认为是一个虚状态。

HMM 用于模式匹配

和 DTW 的对比



- 对应不同语音内容的似然模型作为模板
- 代价函数是似然度
- 状态是隐含的，允许多个对齐序列

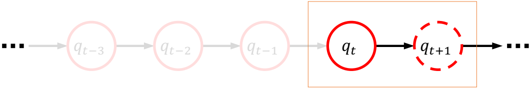
HMM 参数集和模型假设

- **HMM 参数集 θ 包括:**
  - 隐含状态: $Q = \{q_i,\ 1 \le i \le S\}$
  - 转移概率: $A = \{a_{ij},\ 1 \le i \le S\} 1 \le j \le S\}$
  - 状态输出分布: $B(\mathbf{o}) = \{b_j(\mathbf{o}),\ 1 \le j \le S\}$

$$a_{ij} = P(q_t = j | q_{t-1} = i) \quad b_j(\mathbf{o}_t) = p(\mathbf{o}_t | q_t = j)$$

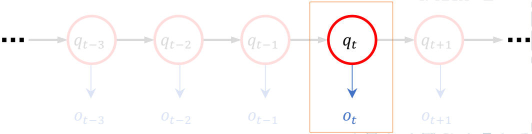
- **模型假设:**
  - **马尔科夫假设: 即时转移**

    给定当前状态 q_t 的情况下, 到下一状态 q_{t+1} 的转移与历史状态无关。



  - **条件独立性假设:**

    给定当前状态 q_t 的情况下, 观测向量 $\mathbf{o}_t$ 与历史状态和历史观测向量无关。



HMM 用于孤立词识别中的关键问题

将 HMM 用于语音识别中，需要回答如下一些问题：

- **似然分数计算：** 如何计算一个给定观测序列的整体似然 $\mathbf{O} = [\mathbf{o}_1, \cdots, \mathbf{o}_T]$?
$$p(\mathbf{O}|\theta) = ?$$
- **解码：** 如何找到给定观测序列的最可能的状态序列？
$$\hat{Q} = \max_{Q} p(\mathbf{O}, Q|\theta)$$
- **参数估计：** 如何找到最优的模型参数？
$$\hat{\theta} = \max_{\theta} \mathcal{L}(\theta)$$

其中 $\mathcal{L}(\theta)$ 是一个特定的准则函数。

HMM 似然度计算

O: 观测序列
q: 状态序列
θ: 模型参数

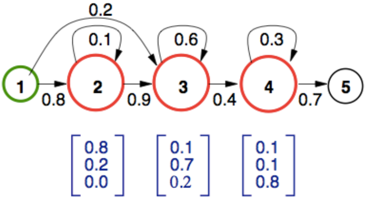
状态序列已知的情况下，应用条件独立假设:

$$p(\mathbf{O}|\mathbf{q},\theta) = \prod_{t=1}^{T} p(\mathbf{o}_t|q_t,\theta)$$

状态序列未知的情况下，应用马尔科夫假设:

$$p(\mathbf{O}|\theta) = \sum_{\mathbf{q}} p(\mathbf{O}|\mathbf{q},\theta)P(\mathbf{q}|\theta) = \sum_{\mathbf{q}} \left(\prod_{t=1}^{T} p(\mathbf{o}_t|q_t,\theta)P(q_t|q_{t-1},\theta) \right)$$

其中 $P(\mathbf{q}|\theta) = \prod_{t=1}^{T} P(q_t|q_{t-1},\theta)$ 是每一个状态序列的先验，且 $P(q_0) = 1$.



假定观测序列是 $\mathbf{o} = [a, a, b, c]$，则可能存在的状态序列是

$$\mathbf{q}_1 = [1,2,2,3,4,5],\ \mathbf{q}_2 = [1,2,3,3,4,5],\ \mathbf{q}_3 = [1,2,3,4,4,5],$$
$$\mathbf{q}_4 = [1,3,3,3,4,5],\ \mathbf{q}_5 = [1,3,3,4,4,5],\ \mathbf{q}_6 = [1,3,4,4,4,5].$$

这个观测序列的总似然可以表示为：

$$p(\mathbf{O}|\theta) = \sum_{i=1}^{6} p(\mathbf{O}|\mathbf{q}_i, \theta) P(\mathbf{q}_i|\theta)$$

前向算法

高效的 HMM 似然度计算

观测序列 $\mathbf{O}_1^T = [\mathbf{o}_1, \cdots, \mathbf{o}_T]$ 的似然度为

$$p(\mathbf{O}_1^T|\theta) = \sum_{\mathbf{q}} p(\mathbf{O}_1^T, \mathbf{q}, q_0 = 1, q_{T+1} = N|\theta)$$

其中，初始和结束的非发射状态分别是 $q_0 = 1$ 和 $q_{T+1} = N$。以上似然度可以有效地通过前向似然的**前向递归**算法计算

$$\begin{aligned}
\alpha_j(t) &= p(\mathbf{O}_1^t, q_t = j|\theta) = \sum_{i=1}^{N-1} p(\mathbf{O}_1^t, q_t = j, q_{t-1} = i|\theta) \\
&= p(\mathbf{o}_t|q_t = j, \theta) \sum_{i=1}^{N-1} P(q_t = j|q_{t-1} = i, \theta) p(\mathbf{O}_1^{t-1}, q_{t-1} = i|\theta) \\
&= b_j(\mathbf{o}_t) \sum_{i=1}^{N-1} a_{ij}\alpha_i(t-1)
\end{aligned}$$

前向似然 $\alpha_j(t)$ 的定义域为 $1 \le t \le T$ 和 $1 \le j \le N$。

边界条件:

- 初始状态固定为 $q_0 = 1$

$$\alpha_j(0) = \begin{cases} 1 & j = 1 \\ 0 & \texttt{otherwise} \end{cases}$$

- 结束状态必须是非发射状态 $q_{T+1} = N$

$$\begin{aligned} p(\mathbf{O}_1^T|\theta) &= p(\mathbf{O}_1^T, q_{T+1} = N|\theta) \\ &= \alpha_N(T+1) = \sum_{i=1}^{N-1} a_{iN}\alpha_i(T) \end{aligned}$$

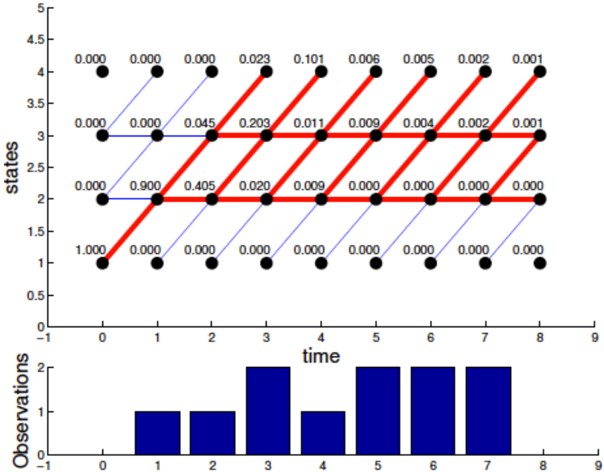
前向算法

伪代码

```
float forward(float a[][], float b[][]) {
    int N = a.length, T= b.length;
    float alpha[][] = new float[N][T+1];
    for (int j = 0; j < N; j++) alpha[j][0] = (j == 0) ? 1 : 0;
    for (int j = 0; j < N; j++)
        for (int t = 1; t <= T; t++) {
            alpha[j][t] = 0;
            for (int i = 0; i < N; i++) alpha[j][t] += a[i][j] * alpha[i][t-1];
            alpha[j][t] *= b[t][j];
        }
    float likelihood = 0;
    for (int j = 0; j < N; j++) likelihood += alpha[j][T];
    return likelihood;
}
```

前向算法

示例



前向算法可以直接用来进行孤立词识别。

HMM 用于孤立词识别中的关键问题

将 HMM 用于语音识别中，需要回答如下一些问题：

- **似然度计算**：如何计算一个给定观测序列的整体似然 $\mathbf{O} = [\mathbf{o}_1, \cdots, \mathbf{o}_T]$?

$$p(\mathbf{O}|\theta) = ?$$

- **解码**：如何找到给定观测序列的最可能的状态序列？

$$\hat{Q} = \max_{Q} p(\mathbf{O}, Q|\theta)$$

- **参数估计**：如何找到最优的模型参数？

$$\hat{\theta} = \max_{\theta} \mathcal{L}(\theta)$$

其中 $\mathcal{L}(\theta)$ 是一个特定的准则函数。

使用 HMM 进行解码

计算最可能（似然度最大）的状态序列

- **似然度计算:**
  - **目标:** 在所有可能状态序列下的期望的似然度

$$p(\mathbf{O}|\theta) = \sum_{\mathbf{q}} p(\mathbf{O}, \mathbf{q}|\theta)$$

  - 一般认为特征向量序列对应的词 (序列) 是已知的
  - 对未知的词 (序列) 的计算代价非常昂贵
- **解码:**
  - 使用最优状态对齐序列来近似得到整体的似然度

$$p(\mathbf{O}|\theta) \approx \max_{\mathbf{q}} p(\mathbf{O}, \mathbf{q}|\theta)$$

  - 高效性，同时可以利用动态规划算法
  - **目标:** 找到最相似的状态序列

$$\hat{\mathbf{q}} = \arg\max_{\mathbf{q}} p(\mathbf{O}, \mathbf{q}, q_0 = 1, q_{T+1} = N|\theta)$$

给定 $\mathbf{O}_1^T = [\mathbf{o}_1, \cdots, \mathbf{o}_T]$ 和 $\mathbf{q}_1^T = [q_1, \cdots, q_T]$,
Viterbi 算法通过递归来找到

$$\hat{\mathbf{q}}_1^T = \arg\max_{\mathbf{q}_1^T} p(\mathbf{O}_1^T, \mathbf{q}_1^T, q_0 = 1, q_{T+1} = N|\theta)$$

类似前向算法中的 $\alpha_j(t)$, 我们定义到时刻 t 为止的，似然度最大的部分状态序列的似然度为:

$$\begin{aligned}
\phi_j(t) &= \max_{\mathbf{q}_1^t} p(\mathbf{O}_1^t, q_0 = 1, \mathbf{q}_1^{t-1}, q_t = j|\theta) \\
&= \max_{1\le i\le N} \max_{\mathbf{q}_1^t} p(\mathbf{O}_1^t, q_0 = 1, \mathbf{q}_1^{t-2}, q_{t-1} = i, q_t = j|\theta) \\
&= {\color{magenta} p(\mathbf{o}_t|q_t = j, \theta)} \max_{1\le i\le N} {\color{blue} P(q_t = j|q_{t-1} = i, \theta)} \\
&\quad \max_{\mathbf{q}_1^{t-1}} p(\mathbf{O}_1^{t-1}, q_0 = 1, \mathbf{q}_1^{t-2}, q_{t-1} = i|\theta) \\
&= {\color{magenta} b_j(\mathbf{o}_t)} \max_{1\le i\le N} {\color{blue} a_{ij}} \phi_i(t-1)
\end{aligned}$$

Viterbi 似然 $\phi_j(t)$ 的相关定义为 $1 \le t \le T$ 和 $1 \le j \le N$.

边界条件:

- 初始状态固定为 $q_0 = 1$

$$\phi_j(0) = \begin{cases} 1 & j = 1 \\ 0 & \texttt{otherwise} \end{cases}$$

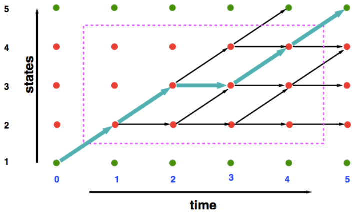
- 有必要去记录在给定最优 部分似然度 的情况下，前一时刻的最优状态

$$q_j^{\texttt{max}}(t) = \arg \max_{1 \le i \le N} a_{ij}\phi_i(t-1)$$

- 结束状态必须是非发射状态 $q_{T+1} = N$

$$\phi_N(T+1) = \max_{1 \le i \le N} a_{iN}\phi_i(T), \quad q_N^{\texttt{max}}(T+1) = \arg \max_{1 \le i \le N} a_{iN}\phi_i(T)$$

维特比 (Viterbi) 算法
回溯



回溯: 在完成 Viterbi 递归之后，找到最优的状态序列。

- 从最终时刻的状态开始 $\hat{q}_{T+1} = N$
- 可以得到前一个状态为 $\hat{q}_T = q^{\max}_{\hat{q}_{T+1}}(T)$
- 类似地，在 t 时刻产生最优似然分数时对应的 $t-1$ 时刻的最优状态是

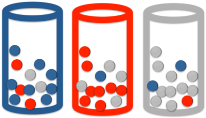
$$\hat{q}_{t-1} = q^{\max}_{\hat{q}_t}(t)$$

- 整个过程可以重复，直到到达 $\hat{q}_0 = 1$

使用 HMM 进行解码 — 例子

计算最可能（似然度最大）的状态序列

考虑以下桶中取球问题：



取出不同球的似然

状态 i	b_i(红球)	b_i(蓝球)	b_i(灰球)
1	0.2	0.6	0.2
2	0.6	0.1	0.3
3	0.1	0.2	0.7

$$\mathbf{P} = \begin{array}{c} \\ 1 \\ 2 \\ 3 \end{array} \begin{array}{c} \begin{array}{ccc} 1 & 2 & 3 \end{array} \\ \begin{pmatrix} 0.2 & 0.3 & 0.5 \\ 0.3 & 0.6 & 0.1 \\ 0.1 & 0.8 & 0.1 \end{pmatrix} \end{array}$$

- 定义状态集合 $S = \{1, 2, 3\}$，分别对应三个桶。
- 假设每次从桶 $i \in \{1, 2, 3\}$ 中取出不同颜色球的似然 $b_i(\cdot)$ 是不变的，且如上表所示。
- 定义三个状态之间的转移矩阵为 $\mathbf{P} = \{a_{ij}\}_{ij}$ $(i, j \in \{1, 2, 3\})$。
- 假设三个状态的初始似然相等，均为 $1/3$。
- 给定某次连续取球的结果序列为“红红青青青灰灰红”，求对应最可能的状态序列（即 8 次取球所属的桶组成的序列）。

使用 HMM 进行解码 — 例子

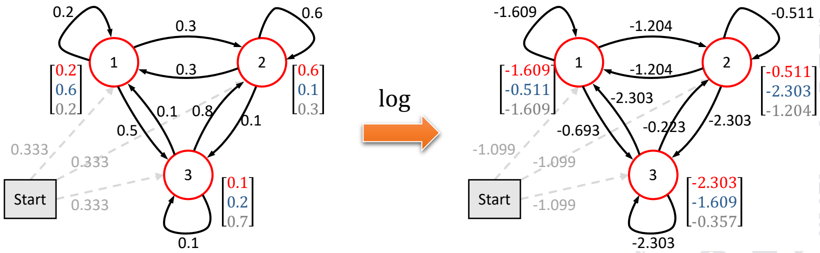
计算使得观测序列的（对数）似然最大的状态序列

计算每一步所取的球属于每个状态的对数似然$(\log \prod_t p_t = \sum_t \log p_t)$

(表中第 j 行第 t 列的值对应于第 t 次取球来自状态 j 的最大似然 $\phi_j(t)$)

$$\log \phi_j(t) = \log b_j(\text{第}t\text{次取出的球}) + \max_{1\le i\le 3}(\log a_{ij} + \log \phi_i(t-1))$$

状态	红球	红球	蓝球	蓝球	蓝球	灰球	灰球	红球
1								
2								
3								

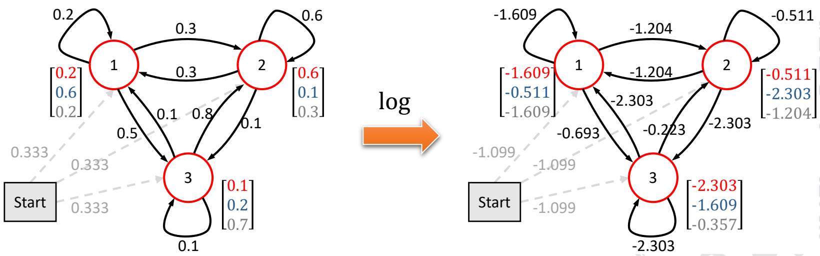


计算每一步所取的球属于每个状态的对数似然 $(\log \prod_t p_t = \sum_t \log p_t)$

（表中第 j 行第 t 列的值对应于第 t 次取球来自状态 j 的最大似然 $\phi_j(t)$）

$$\log \phi_j(t) = \log b_j(\text{第}t\text{次取出的球}) + \max_{1\le i\le 3}(\log a_{ij} + \log \phi_i(t-1))$$

状态	红球	红球	蓝球	蓝球	蓝球	灰球	灰球	红球
1	-2.708							
2	-1.609							
3	-3.401							



使用 HMM 进行解码 — 例子

计算使得观测序列的（对数）似然最大的状态序列

计算每一步所取的球属于每个状态的对数似然$(\log \prod_t p_t = \sum_t \log p_t)$
(表中第 j 行第 t 列的值对应于第 t 次取球来自状态 j 的最大似然 $\phi_j(t)$)

$$\log \phi_j(t) = \log b_j(\text{第}t\text{次取出的球}) + \max_{1\le i\le 3}(\log a_{ij} + \log \phi_i(t-1))$$

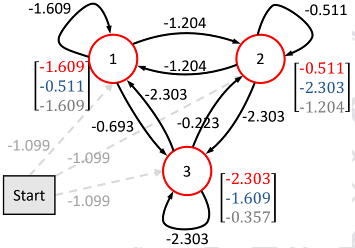
状态	红球	红球	蓝球	蓝球	蓝球	灰球	灰球	红球
1	-2.708	-4.422						
2	-1.609							
3	-3.401							

对于第二列 $t=2$，先考虑红球由状态 1产生的情况：

$$1 \to 1: \log b_1(\text{红球}) + \log a_{11} + \log \phi_1(1)$$
$$= -1.609 - 1.609 - 2.708 = -5.926$$

$$2 \to 1: \log b_1(\text{红球}) + \log a_{21} + \log \phi_2(1)$$
$$= -1.609 - 1.204 - 1.609 = -4.422$$

$$3 \to 1: \log b_1(\text{红球}) + \log a_{31} + \log \phi_3(1)$$
$$= -1.609 - 2.303 - 3.401 = -7.313$$



使用 HMM 进行解码 — 例子

计算使得观测序列的（对数）似然最大的状态序列

计算每一步所取的球属于每个状态的对数似然$(\log \prod_t p_t = \sum_t \log p_t)$

（表中第 j 行第 t 列的值对应于第 t 次取球来自状态 j 的最大似然 $\phi_j(t)$）

$$\log \phi_j(t) = \log b_j(\text{第}t\text{次取出的球}) + \max_{1\leq i\leq 3}(\log a_{ij} + \log \phi_i(t-1))$$

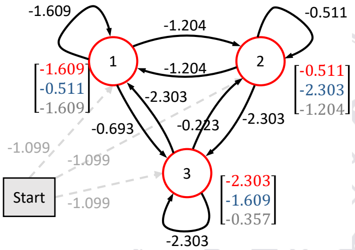
状态	红球	红球	蓝球	蓝球	蓝球	灰球	灰球	红球
1	-2.708	-4.422						
2	-1.609	-2.631						
3	-3.401							

对于第二列 $t=2$，再考虑红球由状态 2产生的情况：

$$1 \rightarrow 2 : \log b_2(\text{红球}) + \log a_{12} + \log \phi_1(1)$$
$$= -0.511 - 1.204 - 2.708 = -4.423$$

$$2 \rightarrow 2 : \log b_2(\text{红球}) + \log a_{22} + \log \phi_2(1)$$
$$= -0.511 - 0.511 - 1.609 = -2.631$$

$$3 \rightarrow 2 : \log b_2(\text{红球}) + \log a_{32} + \log \phi_3(1)$$
$$= -0.511 - 0.223 - 3.401 = -4.135$$



计算每一步所取的球属于每个状态的对数似然 $(\log \prod_t p_t = \sum_t \log p_t)$

(表中第 j 行第 t 列的值对应于第 t 次取球来自状态 j 的最大似然 $\phi_j(t)$)

$$\log \phi_j(t) = \log b_j(\text{第}t\text{次取出的球}) + \max_{1\le i\le 3}(\log a_{ij} + \log \phi_i(t-1))$$

状态	红球	红球	蓝球	蓝球	蓝球	灰球	灰球	红球
1	-2.708	-4.422						
2	-1.609	-2.631						
3	-3.401	-5.704						

对于第二列 $t=2$，再考虑红球由状态 3产生的情况：

$1 \to 3 : \log b_3(\text{红球}) + \log a_{13} + \log \phi_1(1)$

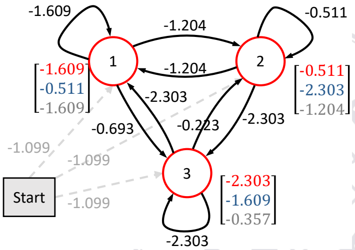
$= -2.303 - 0.693 - 2.708 = -5.704$

$2 \to 3 : \log b_3(\text{红球}) + \log a_{23} + \log \phi_2(1)$

$= -2.303 - 2.303 - 1.609 = -6.215$

$3 \to 3 : \log b_3(\text{红球}) + \log a_{33} + \log \phi_3(1)$

$= -2.303 - 2.303 - 3.401 = -8.007$



使用 HMM 进行解码 — 例子

计算使得观测序列的（对数）似然最大的状态序列

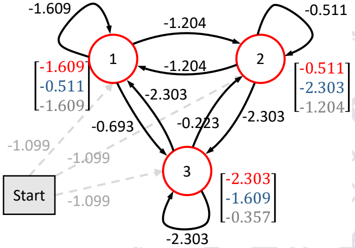
计算每一步所取的球属于每个状态的对数似然$(\log \prod_t p_t = \sum_t \log p_t)$

（表中第 j 行第 t 列的值对应于第 t 次取球来自状态 j 的最大似然 $\phi_j(t)$）

$$\log \phi_j(t) = \log b_j(\text{第}t\text{次取出的球}) + \max_{1 \le i \le 3} (\log a_{ij} + \log \phi_i(t-1))$$

状态	红球	红球	蓝球	蓝球	蓝球	灰球	灰球	红球
1	-2.708	-4.422						
2	-1.609	-2.631						
3	-3.401	-5.704						

得到 $t = 1 \to 2$ 时的三个潜在最优状态转移路径，但无法直接决定哪一条路径与后面路径相连后的（对数）似然最大，需要继续计算 $t = 1 \to 3$ 时的潜在最优状态转移路径



使用 HMM 进行解码 — 例子

计算使得观测序列的（对数）似然最大的状态序列

计算每一步所取的球属于每个状态的对数似然$(\log \prod_t p_t = \sum_t \log p_t)$

(表中第 j 行第 t 列的值对应于第 t 次取球来自状态 j 的最大似然 $\phi_j(t)$)

$$\log \phi_j(t) = \log b_j(\text{第}t\text{次取出的球}) + \max_{1\le i\le 3}(\log a_{ij} + \log \phi_i(t-1))$$

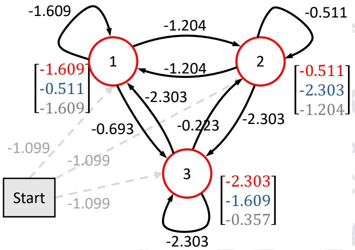
状态	红球	红球	蓝球	蓝球	蓝球	灰球	灰球	红球
1	-2.708	-4.422	-4.346					
2	-1.609	-2.631	-5.445					
3	-3.401	-5.704	-6.543					

同样地，对于第三列 $t=3$，分别计算蓝球由状态 1，状态 2，状态 3产生时的最优状态转移路径：

$$1 \to 1 : \log b_1(\text{蓝球}) + \log a_{11} + \log \phi_1(2)$$
$$= -0.511 - 1.609 - 4.422 = -6.542$$

$$2 \to 1 : \log b_1(\text{蓝球}) + \log a_{21} + \log \phi_2(2)$$
$$= -0.511 - 1.204 - 2.631 = -4.346$$

$$3 \to 1 : \log b_1(\text{蓝球}) + \log a_{31} + \log \phi_3(2)$$
$$= -0.511 - 2.303 - 5.704 = -8.518$$



使用 HMM 进行解码 — 例子

计算使得观测序列的（对数）似然最大的状态序列

计算每一步所取的球属于每个状态的对数似然$(\log \prod_t p_t = \sum_t \log p_t)$

（表中第 j 行第 t 列的值对应于第 t 次取球来自状态 j 的最大似然 $\phi_j(t)$）

$$\log \phi_j(t) = \log b_j(\text{第}t\text{次取出的球}) + \max_{1\le i\le 3}(\log a_{ij} + \log \phi_i(t-1))$$

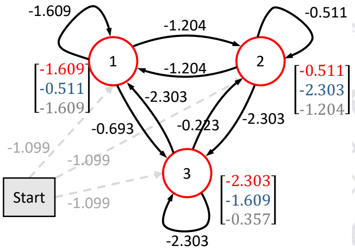
状态	红球	红球	蓝球	蓝球	蓝球	灰球	灰球	红球
1	-2.708	-4.422	-4.346					
2	-1.609	-2.631	-5.445					
3	-3.401	-5.704	-6.543					

同样地，对于第三列 $t=3$，分别计算蓝球由状态 1，状态 2，状态 3产生时的最优状态转移路径：

$$1 \to 2 : \log b_2(\text{蓝球}) + \log a_{12} + \log \phi_1(2)$$
$$= -2.303 - 1.204 - 4.422 = -7.929$$

$$2 \to 2 : \log b_2(\text{蓝球}) + \log a_{22} + \log \phi_2(2)$$
$$= -2.303 - 0.511 - 2.631 = -5.445$$

$$3 \to 2 : \log b_2(\text{蓝球}) + \log a_{32} + \log \phi_3(2)$$
$$= -2.303 - 0.223 - 5.704 = -8.23$$



计算每一步所取的球属于每个状态的对数似然$(\log \prod_t p_t = \sum_t \log p_t)$

(**表中第 j 行第 t 列的值对应于第 t 次取球来自状态 j 的最大似然 $\phi_j(t)$**)

$$\log \phi_j(t) = \log b_j(\text{第}t\text{次取出的球}) + \max_{1\le i\le 3}(\log a_{ij} + \log \phi_i(t-1))$$

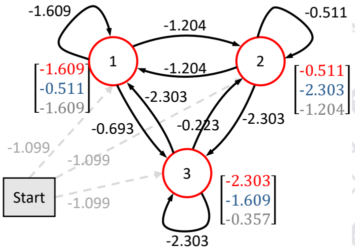
状态	红球	红球	蓝球	蓝球	蓝球	灰球	灰球	红球
1	-2.708	-4.422	-4.346					
2	-1.609	-2.631	-5.445					
3	-3.401	-5.704	-6.543					

同样地，对于第三列 $t=3$，分别计算蓝球由状态 1，状态 2，状态 3产生时的最优状态转移路径：

$$1 \to 3 : \log b_3(\text{蓝球}) + \log a_{13} + \log \phi_1(2)$$
$$= -1.609 - 0.693 - 4.422 = -6.724$$

$$2 \to 3 : \log b_3(\text{蓝球}) + \log a_{23} + \log \phi_2(2)$$
$$= -1.609 - 2.303 - 2.631 = -6.543$$

$$3 \to 3 : \log b_3(\text{蓝球}) + \log a_{33} + \log \phi_3(2)$$
$$= -1.609 - 2.303 - 5.704 = -9.616$$



计算每一步所取的球属于每个状态的对数似然$(\log \prod_t p_t = \sum_t \log p_t)$

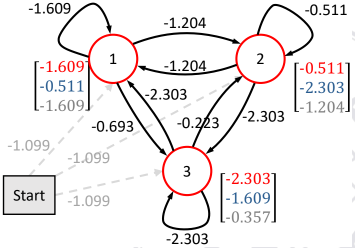
（表中第 j 行第 t 列的值对应于第 t 次取球来自状态 j 的最大似然 $\phi_j(t)$）

$$\log \phi_j(t) = \log b_j(\text{第}t\text{次取出的球}) + \max_{1\le i\le 3}(\log a_{ij} + \log \phi_i(t-1))$$

状态	红球	红球	蓝球	蓝球	蓝球	灰球	灰球	红球
1	-2.708	-4.422	-4.346	-6.467	-8.587	-11.805	-13.009	-13.877
2	-1.609	-2.631	-5.445	-7.854	-9.175	-10.196	-11.064	**-12.086**
3	-3.401	-5.704	-6.543	-6.649	-8.769	-9.637	-12.297	-15.67

依此类推，最终得到使得整个观测序列的（对数）似然最大的状态序列：

$$2 \rightarrow 2 \rightarrow 1 \rightarrow 1 \rightarrow 1 \rightarrow 3 \rightarrow 2 \rightarrow 2$$



$$\phi_j(t) = \max_{\mathbf{q}_1^t} p(\mathbf{O}_1^t, q_0 = 1, \mathbf{q}_1^{t-1}, q_t = j|\theta)$$

1. **初始化**

$$\phi_j(0) = \begin{cases} 1.0 & j = 1 \\ 0 & 1 \le j \le N \end{cases} \qquad \phi_1(t) = 0,\ 1 \leqslant t \leqslant T$$

2. **递归**

   for $t = 1, 2, ..., T$

   for $j = 2, 3, .., N-1$

   Compute: $\phi_j(t) = b_j(\mathbf{o}_t) \max_{1 \leqslant i < N}(\phi_i(t-1)a_{ij})$

   Store: $q_j^{\max}(t) = \arg\max_{1 \leqslant i < N}(\phi_i(t-1)a_{ij})$

3. **终止**

   $p(\mathbf{O}, \hat{\mathbf{q}}|\theta) = \max_{1<k<N}(\phi_k(T)\ a_{kN})$

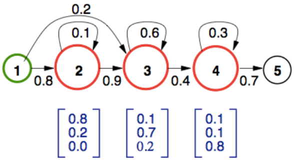
   最可能的路径可以利用存储在每个状态 $q_j^{\max}(t)$ 中的前继者的信息，通过回溯的方法得到。

维特比 (Viterbi) 算法

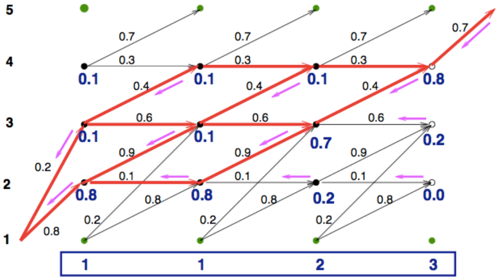
伪代码

```
float viterbi(float a[][], float b[][], float qbest[]) {
    int N = a.length, T= b.length;
    float v[][] = new float[N][T+1];
    int qmax[][] = new int[N][T+1];
    for (int j = 0; j < N; j++) alpha[j][0] = (j == 0) ? 1 : 0;
    for (int j = 0; j < N; j++)
        for (int t = 1; t <= T; t++) {
            v[j][t] = b[t][j] * FindMax(a, v, j, t);
            qmax[j][t] = FindMaxArg(a, v, j, t);
        }
    qbest[T] = FindMaxArg(a, v, N, T);
    for (int t = T-1; t >= 0; t++) qbest[t] = qmax[qbest[t+1]][t+1];
    return FindMax(a, v, N, T+1);
}
```

Viterbi — 示例



给定以上 HMM 模型和观测序列 $\mathbf{O} = [1, 1, 2, 3]$:



5	-	-	-	-	-	0.0072253
4	0.0	0.0	0.00080	0.002304	0.0103220	-
3	0.0	0.02	0.05760	0.032256	0.0038707	-
2	0.0	0.64	0.0512	0.001024	0.0	-
1	1	0	0	0	0	-
	-	1	1	2	3	-

维特比搜索中的实际问题

对数似然和剪枝

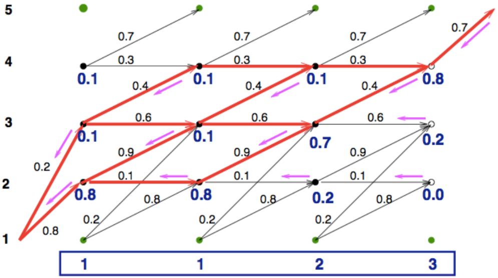
对数似然 常常会被使用，来避免数值溢出的问题：

$$\log \phi_j(t) = \max_i \{\log \phi_i(t-1) + \log a_{ij} + \log b_j(\mathbf{o}_t)\}$$

剪枝 的方法常常被用来移除那些相对似然度低的路径，从而可以保持较小的搜索空间

- **束剪枝（Beam pruning）：**
  1. 在时刻 t，找到最优路径的对数似然度 $\phi^*(t) = \max_j \phi_j(t)$
  2. 移除那些似然度低于门限的相关路径 $\phi_j(t) < \phi^*(t) - \tau$ 其中 τ 是预先定义好的常数，称为**束宽度 (beam width)**
- **直方图剪枝（Histogram pruning）：**
  1. 在时刻 t，得到所有路径对数似然的直方图分布
  2. 选择分数最高的 N 条路径，并且根据第 N^{th} 条路径的似然得到束宽度 τ
  3. 移除那些似然度低于门限的相关路径 $\phi_j(t) < \phi^*(t) - \tau$

Viterbi — 高效性



- Viterbi 算法仅仅做 **local** 决策 — 高效!
- 路径合并和分叉大体上相同比例 — 搜索空间没有增长
- 利用剪枝，搜索时间和观测向量的长度呈线性关系
- 相比 DTW，能力要强大很多

HMM 用于孤立词识别中的关键问题

将 HMM 用于语音识别中，需要回答如下一些问题:

- 似然分数计算: 如何计算一个给定观测序列的整体似然 $\mathbf{O} = [\mathbf{o}_1, \cdots, \mathbf{o}_T]$?

$$p(\mathbf{O}|\theta) = ?$$

- 解码: 如何找到给定观测序列的最可能的状态序列？

$$\hat{Q} = \max_{Q} p(\mathbf{O}, Q|\theta)$$

- **参数估计: 如何找到最优的模型参数？**

$$\hat{\theta} = \max_{\theta} \mathcal{L}(\theta)$$

其中 $\mathcal{L}(\theta)$ 是一个特定的准则函数。

HMM 参数估计

数据:

$$\mathbf{O}^{(r)} = [\mathbf{o}_1^{(r)}, \cdots, \mathbf{o}_T^{(r)}] \quad H^{(r)} = w^{(r)}$$

模型: 孤立词 w 确定了所有可能的状态序列和待估计的参数

$$p(\mathbf{O}|\theta) = \sum_{\mathbf{q}} p(\mathbf{O}|\mathbf{q}, \theta)P(\mathbf{q}|\theta) = \sum_{\mathbf{q}} \left(\prod_{t=1}^{T} p(\mathbf{o}_t|q_t, \theta)P(q_t|q_{t-1}, \theta) \right)$$

准则:

- 最大似然

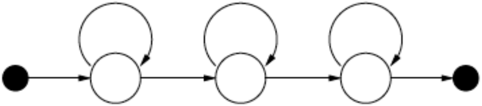
$$\hat{\theta}_{\mathtt{ML}} = \max_{\theta} \prod_{r} p(\mathbf{O}^{(r)}|\theta)$$

- 最大后验准则

$$\hat{\theta}_{\mathtt{MAP}} = \max_{\theta} \prod_{r} p(\theta|\mathbf{O}^{(r)})$$

维特比训练

在状态和观测量之间的硬对齐



	State 1	State 2	State 3
Seq 1	a a a a a	b b b b b	c c c c c
Seq 2	a a c b a	b b a b b	c c b c c
Seq 3	a a a a c	b b c a b	c c b b c
Seq 4	c c c c c	b b b b b	a a a a a
PDF	$\left[\frac{12}{20} \quad \frac{1}{20} \quad \frac{7}{20}\right]$	$\left[\frac{2}{20} \quad \frac{17}{20} \quad \frac{1}{20}\right]$	$\left[\frac{5}{20} \quad \frac{3}{20} \quad \frac{12}{20}\right]$

转移概率: self-transition$=\frac{4}{5}$, forward-transition$\frac{1}{5}$

	Probability	State 1	State 2	State 3
Alignment 1	0.6	a a c b a [0.60 0.20 0.20]	b b a b b [0.20 0.80 0.00]	c b c c c [0.00 0.20 0.80]
Alignment 2	0.3	a a c [0.71 0.0 0.29]	b a b b a b b [0.29 0.71 0.00]	c b c c c [0.00 0.20 0.80]
Alignment 3	0.1	a a c b a [0.60 0.20 0.20]	b b a b b c b [0.14 0.72 0.14]	c c c [0.00 0.00 1.00]
Total	1.0	[0.63 0.14 0.23]	[0.22 0.77 0.01]	[0.00 0.18 0.82]

最终的状态输出分布是一个在每一种对齐序列情况下的**加权平均** PDF。这种估计更加鲁棒和稳定，因为它考虑了对齐所带来的不确定性和不准确性。

Baum-Welch 训练

软对齐和转移概率

	Probability	State 1	State 2	State 3
Alignment 1	0.6	a a c b a [0.80 0.20]	b b a b b [0.80 0.20]	c b c c c [0.80 0.20]
Alignment 2	0.3	a a c [0.67 0.33]	b a b b a b b [0.86 0.14]	c b c c c [0.80 0.20]
Alignment 3	0.1	a a c b a [0.80 0.20]	b b a b b c b [0.86 0.14]	c c c [0.67 0.33]
Total	1.0	[0.76 0.24]	[0.82 0.18]	[0.79 0.21]

类似于状态输出分布，得到的转移概率是在每一种对齐序列情况下的**加权平均** PDF。

HMM 的参数集合:

- 转移概率 A: $a_{ij} = P(q_t = j | q_{t-1} = i), \quad 1 \le j \le N$
- 状态输出分布 B: $b_j(\mathbf{o}), \quad 1 < j < N$

离散	概率向量	$[c_{j1}, \cdots, c_{jK}], \ \sum_k c_{jk} = 1$
连续	单高斯	$\mathcal{N}(\mathbf{o} \vert \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)$
	高斯混合模型	$\sum_{m=1}^{M} c_{jm} \mathcal{N}(\mathbf{o} \vert \boldsymbol{\mu}_{jm}, \boldsymbol{\Sigma}_{jm})$

最大似然准则:

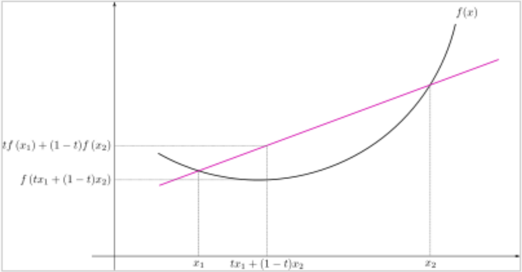
$$\mathcal{L}(\theta) = \sum_{r=1}^{R} \log p(\mathbf{O}^{(r)} | \theta) = \sum_{r=1}^{R} \log \left(\sum_{\mathbf{q}} p(\mathbf{O}^{(r)}, \mathbf{q} | \theta) \right)$$

其中 r 是句子的索引值。

Jensen's Inequality

Recap

假如 X 是一个随机变量，并且 φ 是一个凸函数。Then $\varphi(\mathbb{E}[X]) \leq \mathbb{E}[\varphi(X)]$.



Jensen's inequality generalizes the statement that a secant line of a convex function lies above the graph.

凹函数正好相反，它可以归纳为向量:

$$\log \mathbb{E}_{\mathbf{z}}[f(\mathbf{z})] \geq \mathbb{E}_{\mathbf{z}}[\log f(\mathbf{z})]$$

假定存在初始参数 $\hat{\theta}$, **应用 Jensen's inequality** w.r.t. **隐含状态序列** $\mathbf{q}$

$$
\begin{aligned}
\mathcal{L}(\theta) &= \sum_{r=1}^{R} \log\left(\sum_{\mathbf{q}} p(\mathbf{O}^{(r)}, \mathbf{q}|\theta)\right) \\
&= \sum_{r=1}^{R} \log\left(\sum_{\mathbf{q}} P(\mathbf{q}|\mathbf{O}^{(r)}, \hat{\theta}) \frac{p(\mathbf{O}^{(r)}, \mathbf{q}|\theta)}{P(\mathbf{q}|\mathbf{O}^{(r)}, \hat{\theta})}\right) \\
&\geq \sum_{r=1}^{R} \sum_{\mathbf{q}} P(\mathbf{q}|\mathbf{O}^{(r)}, \hat{\theta}) \log \frac{p(\mathbf{O}^{(r)}, \mathbf{q}|\theta)}{P(\mathbf{q}|\mathbf{O}^{(r)}, \hat{\theta})} \\
&= \sum_{r=1}^{R} \mathtt{H}\left(P(\mathbf{q}|\mathbf{O}^{(r)}, \hat{\theta})\right) + \mathcal{Q}(\theta, \hat{\theta})
\end{aligned}
$$

$\mathcal{L}(\theta)$ 的下界定义为**辅助函数:**

$$\mathcal{Q}(\theta,\hat{\theta}) = \sum_{r=1}^{R}\sum_{\mathbf{q}} P(\mathbf{q}|\mathbf{O}^{(r)},\hat{\theta}) \log p(\mathbf{O}^{(r)},\mathbf{q}|\theta)$$

这可以通过重新整理，分为两个部分

$$\mathcal{Q}_{\mathtt{A}}(\theta,\hat{\theta}) = K + \sum_{r=1}^{R}\sum_{t=1}^{T^{(r)}}\sum_{i=1}^{N}\sum_{j=1}^{N} \gamma_{(i,j)}(t) \log P(q_t|q_{t-1},\theta)$$

$$\mathcal{Q}_{\mathtt{B}}(\theta,\hat{\theta}) = K + \sum_{r=1}^{R}\sum_{t=1}^{T^{(r)}}\sum_{j=1}^{N} \gamma_{j}(t) \log p(\mathbf{o}_t|q_t=j,\theta)$$

其中，软分配的占用率 (occupancy) 为

$$\gamma_{(i,j)}(t) = P(q_{t-1}=i, q_t=j|\mathbf{O}^{(r)},\hat{\theta}),\ \gamma_j(t) = P(q_t=j|\mathbf{O}^{(r)},\hat{\theta})$$

Forward probability: 前向似然在之前的 HMM 似然度计算中已经定义过

$$\alpha_j(t) = p(\mathbf{O}_1^t, q_t = j) = b_j(\mathbf{o}_t)\sum_{i=1}^{N-1} a_{ij}\alpha_i(t-1) \quad 1 \le t \le T,\ 1 < j < N$$

Backward probability: 后向似然可以用和前向过程中类似的递归方式来定义:

$$\begin{aligned}\beta_j(t) &= p(\mathbf{O}_{t+1}^T | q_t = j) = \sum_{i=1}^{N-1} p(\mathbf{o}_{t+1}, \mathbf{O}_{t+2}^T, q_{t+1} = i | q_t = j) \\ &= \sum_{i=1}^{N-1} p(\mathbf{o}_{t+1}|q_{t+1} = i)P(q_{t+1} = i|q_t = j)p(\mathbf{O}_{t+2}^T|q_{t+1} = i) \\ &= \sum_{i=1}^{N-1} b_i(\mathbf{o}_{t+1})a_{ji}\beta_i(t+1) \qquad 1 \le t \le T,\ 1 < j < N\end{aligned}$$

非对称的前向和后向似然

- $\alpha_j(t)$: 在状态 $q_t = j$ 的部分观测序列 $\mathbf{O}_1^t$ 的似然度
- $\beta_j(t)$: 给定状态 $q_t = j$ 的部分观测序列 $\mathbf{O}_{t+1}^T$ 的似然度

边界条件

$$\beta_j(T) = a_{jN} \quad \beta_N(T+1) = 1$$

整个序列的似然

$$p(\mathbf{O}_1^T|\theta) = \alpha_N(T+1) = \beta_1(0) = \sum_{i=1}^{N} \alpha_i(t)\beta_i(t)$$

转移概率可以更新为: $a_{ij} = \log P(q_t|q_{t-1}, \theta)$

$$\mathcal{Q}_{\mathtt{A}}(\theta, \hat{\theta}) = K + \sum_{r=1}^{R} \sum_{t=1}^{T^{(r)}} \sum_{i=1}^{N} \sum_{j=1}^{N} \gamma_{(i,j)}(t) \log P(q_t|q_{t-1}, \theta)$$

在时刻 t 给定 $\hat{\theta}$ 和整个观测序列情况下，从状态 i 跳转到状态 j 的转移的后验似然为

$$\begin{aligned}
\gamma_{(i,j)}(t) &= P(q_{t-1} = i, q_t = j|\mathbf{O}_1^T, \hat{\theta}) = \frac{p(q_{t-1} = i, q_t = j, \mathbf{O}_1^T|\hat{\theta})}{p(\mathbf{O}_1^T|\hat{\theta})} \\
&= \frac{p(q_{t-1} = i, \mathbf{O}_1^{t-1}) P(q_t = j|q_{t-1} = i) p(\mathbf{o}_t|q_t = j) p(\mathbf{O}_{t+1}^T|q_t = j)}{p(\mathbf{O}_1^T)} \\
&= \frac{\alpha_i(t-1)\hat{a}_{ij} b_j(\mathbf{o}_t)\beta_j(t)}{\alpha_N(T+1)}
\end{aligned}$$

$$\hat{a}_{ij} = \arg\max_{a_{ij}} \sum_{r=1}^{R} \sum_{t=1}^{T^{(r)}} \sum_{i=1}^{N} \sum_{j=1}^{N} \gamma_{(i,j)}(t) \log a_{ij} \quad s.t. \sum_{j=1}^{N} a_{ij} = 1$$

带约束的最优化问题可以通过使用拉格朗日乘子来解决

$$\hat{a}_{ij} = \arg\max_{a_{ij}} \left\{ \sum_{r=1}^{R} \sum_{t=1}^{T^{(r)}} \gamma_{(i,j)}(t) \log a_{ij} + \lambda \left(\sum_{j=1}^{N} a_{ij} - 1 \right) \right\}$$

答案是

$$\hat{a}_{ij} = \frac{1}{\lambda} \sum_{r=1}^{R} \sum_{t=1}^{T} \gamma_{(i,j)}(t) \quad \lambda = \sum_{r=1}^{R} \sum_{t=1}^{T^{(r)}} \sum_{j=1}^{N} \gamma_{(i,j)}(t)$$

辅助函数最优化 - 状态输出分布

Expectation Step

状态输出分布：估计： $b_j(\mathbf{o}_t) = p(\mathbf{o}_t | q_t = j, \theta)$

$$\mathcal{Q}_{\mathrm{B}}(\theta, \hat{\theta}) = K + \sum_{r=1}^{R} \sum_{t=1}^{T^{(r)}} \sum_{j=1}^{N} \gamma_j(t) \log b_j(\mathbf{o}_t)$$

在时刻 t 给定整个观测序列和 $\hat{\theta}$ 情况下，$q_t = j$ 的后验似然是

$$\begin{aligned}\gamma_j(t) &= P(q_t = j | \mathbf{O}_1^T, \hat{\theta}) \\ &= \cdots \text{（过程略去，请自行推导）} \\ &= \frac{\alpha_j(t)\beta_j(t)}{\alpha_N(T+1)}\end{aligned}$$

连续观测 (Gaussian): $b_j(\mathbf{o}_t) = \mathcal{N}(\mathbf{o}_t | \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)$

$$(\hat{\boldsymbol{\mu}}_j, \hat{\boldsymbol{\Sigma}}_j) = \arg\max_{\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j} -\frac{1}{2} \sum_{r=1}^{R} \sum_{t=1}^{T^{(r)}} \gamma_j(t) \left(\log |\boldsymbol{\Sigma}_j| + (\mathbf{o}_t - \boldsymbol{\mu}_j)^\top \boldsymbol{\Sigma}_j^{-1} (\mathbf{o}_t - \boldsymbol{\mu}_j) \right)$$

针对 $\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j$ 取微分，并且使得等于零，可以得到

$$\begin{aligned} \hat{\boldsymbol{\mu}}_j &= \frac{\sum_{r=1}^{R} \sum_{t=1}^{T^{(r)}} \gamma_j(t) \mathbf{o}_t}{\gamma_j} = \frac{1}{\gamma_j} \sum_{r=1}^{R} \hat{\boldsymbol{\mu}}_j^{\mathsf{acc}(r)} \\ \hat{\boldsymbol{\Sigma}}_j &= \frac{\sum_{r=1}^{R} \sum_{t=1}^{T^{(r)}} \gamma_j(t) (\mathbf{o}_t - \boldsymbol{\mu}_j)(\mathbf{o}_t - \boldsymbol{\mu}_j)^\top}{\gamma_j} = \frac{1}{\gamma_j} \sum_{r=1}^{R} \hat{\boldsymbol{\Sigma}}_j^{\mathsf{acc}(r)} \end{aligned}$$

其中 $\gamma_j = \sum_{r=1}^{R} \sum_{t=1}^{T^{(r)}} \gamma_j(t)$ 被称为状态 j 的 occupancy count。

辅助函数最优化 - 状态输出分布

Expectation Step for GMM

状态输出分布函数: $b_j(\mathbf{o}_t) = \sum_{m=1}^{M} c_{jm} b_{jm}(\mathbf{o}_t) = \sum_{m=1}^{M} c_{jm} \mathcal{N}(\mathbf{o}_t | \boldsymbol{\mu}_{jm}, \boldsymbol{\Sigma}_{jm})$

$$\mathcal{Q}_{\mathrm{B}}(\theta, \hat{\theta}) = K + \sum_{r=1}^{R} \sum_{t=1}^{T^{(r)}} \sum_{j=1}^{N} \gamma_j(t) \log b_j(\mathbf{o}_t)$$

$$\xrightarrow{\text{Jensen 不等式}} \quad \mathcal{Q}'_{\mathrm{B}}(\theta, \hat{\theta}) = K + \sum_{r=1}^{R} \sum_{t=1}^{T^{(r)}} \sum_{j=1}^{N} \sum_{m=1}^{M} \gamma_{jm}(t) \log c_{jm} b_{jm}(\mathbf{o}_t)$$

在时刻 t，给定整个观测序列和 $\hat{\theta}$ 情况下，$q_t = j$ 并且 $g_t = m$ 的后验似然为

$$\begin{aligned} \gamma_{jm}(t) &= P(q_t = j, g_t = m | \mathbf{O}_1^T, \hat{\theta}) \\ &= \cdots \text{（过程略去，请自行推导）} \\ &= \gamma_j(t) \gamma_m(t) \end{aligned}$$

$g_t = m$ 指在 t 时刻状态输出概率由第 m 个高斯成分贡献的部分。

对函数 $\mathcal{Q}'_{\mathrm{B}}(\theta, \hat{\theta})$ 针对 $\boldsymbol{\mu}_{jm}, \boldsymbol{\Sigma}_{jm}$ 取微分，并且使得它等于零，可以求解得到 c_{jm}，$\boldsymbol{\mu}_{jm}$ 和 $\boldsymbol{\Sigma}_{jm}$。

请大家自行推导 c_{jm}，$\boldsymbol{\mu}_{jm}$ 和 $\boldsymbol{\Sigma}_{jm}$ 的更新公式。

在给定充分的统计量条件下，估计的各个结果为

- **转移概率:**

$$\gamma_{(i,j)} = \sum_{r=1}^{R} \sum_{t=1}^{T^{(r)}} \gamma_{(i,j)}(t)$$

- **观测似然:**

$$\begin{aligned}
\gamma_{jm} &= \sum_{r=1}^{R} \sum_{t=1}^{T^{(r)}} \gamma_{jm}(t) = \sum_{r=1}^{R} \gamma_{jm}^{(r)} \\
\boldsymbol{\mu}_{jm}^{\mathsf{acc}} &= \sum_{r=1}^{R} \sum_{t=1}^{T^{(r)}} \gamma_{jm}(t)\mathbf{o}_t = \sum_{r=1}^{R} \boldsymbol{\mu}_{jm}^{\mathsf{acc}(r)} \\
\boldsymbol{\Sigma}_{jm}^{\mathsf{acc}} &= \sum_{r=1}^{R} \sum_{t=1}^{T^{(r)}} \gamma_{jm}(t)\mathbf{o}_t\mathbf{o}_t^{\top} = \sum_{r=1}^{R} \boldsymbol{\Sigma}_{jm}^{\mathsf{acc}(r)}
\end{aligned}$$

$$a_{ij} = \frac{\gamma(i,j)}{\sum_{j=1}^{N} \gamma(i,j)} \qquad c_{jm} = \frac{\gamma_{jm}}{\sum_{m=1}^{M} \gamma_{jm}}$$

$$\boldsymbol{\mu}_{jm} = \frac{\boldsymbol{\mu}_{jm}^{\mathsf{acc}}}{\gamma_{jm}} \qquad \boldsymbol{\Sigma}_{jm} = \frac{\boldsymbol{\Sigma}_{jm}^{\mathsf{acc}}}{\gamma_{jm}} - \frac{\boldsymbol{\mu}_{jm}^{\mathsf{acc}} \boldsymbol{\mu}_{jm}^{\mathsf{acc}\top}}{\gamma_{jm}^2}$$

并行化：大训练数据集合下的并行化训练：

- 将大训练数据集分为若干个小的子集
- 对每一个子集**独立地**累积统计量
- 在完成所有子集的统计量累积之后，**合并各子集的统计量**，并且完成参数估计

Baum-Welsh 算法是一个迭代的 EM 算法:

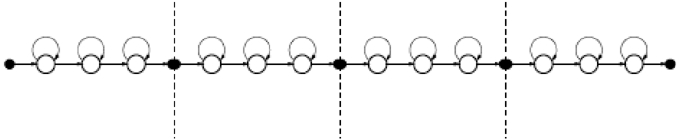
1. 初始化模型参数
2. 执行 E-step: 在给定当前模型参数条件下，估计后验 occupancy counts
3. 执行 M-step: 在得到后验 occupancy counts 条件下，估计新的模型参数
4. 跳到第 2 步，直到收敛

剪枝

- 由于大量可能的状态序列，前后向似然的计算代价很昂贵
- 计算得到了有效的部分路径的似然分数
- 具有低分数的路径 (由最大分数的束/beam 定义) 被移除
- Viterbi 训练是剪枝的一个极端情况

训练 HMM 集合

从孤立词到连续语音



- 声学模型通常情况下会定义在子词层面，比如“音素” (phones)
- 标注边界通常是在句子层面
- 很多子词的 HMM 模型拼接在一起，从而为每个句子组成一个更大的复合 HMM
- 然后，在这个复合 HMM 上，可以应用前-后向算法。

Homework

GMM-HMM 的 EM 算法的参数更新公式

请根据本节课所学内容，推导 P57, P58, P59, P61中 GMM-HMM 的参数更新公式，并总结最终的 EM 算法步骤。