# 视听觉信号处理

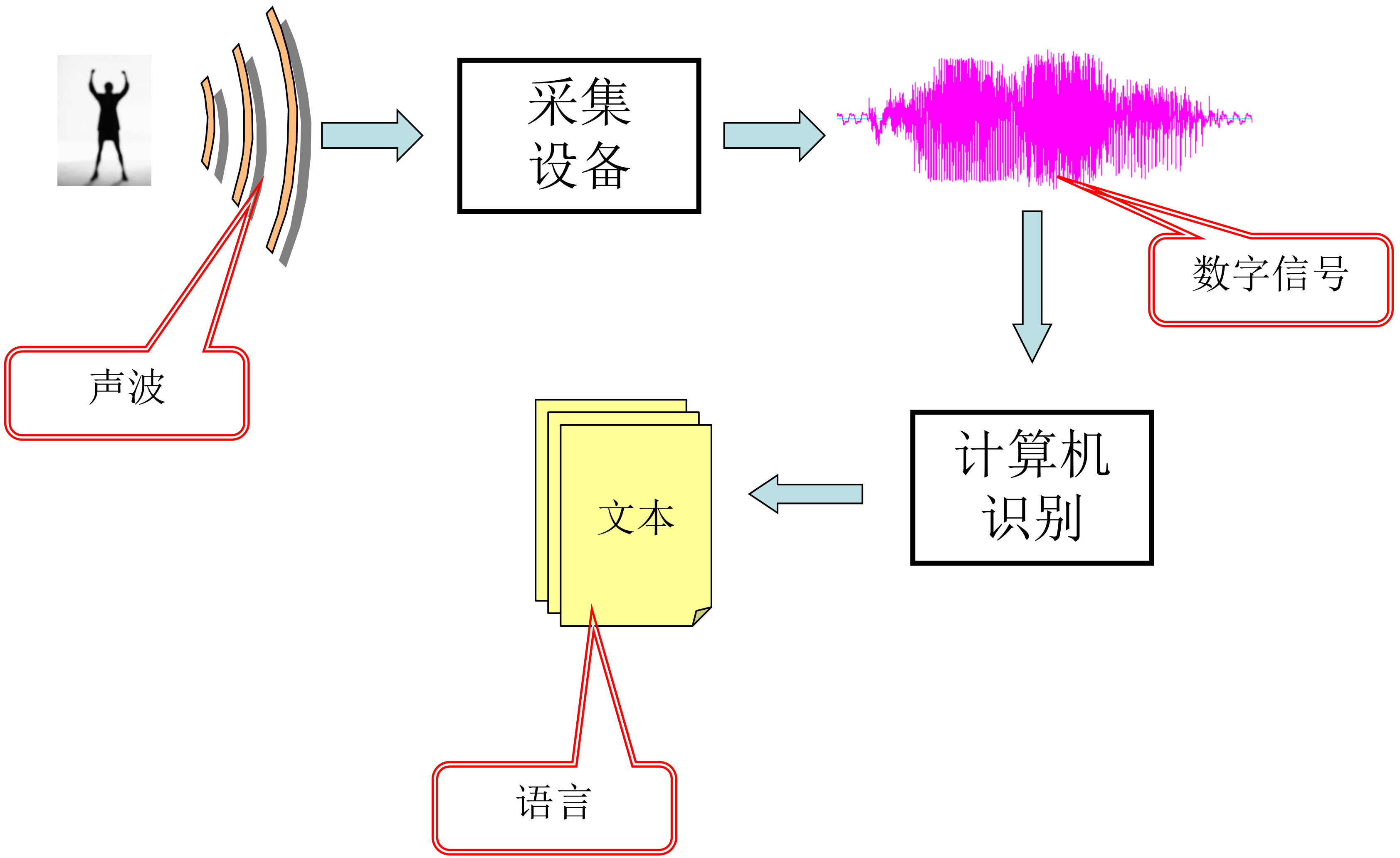
# Visual and Auditory Signal Processing

哈爾濱工業大學
Harbin Institute of Technology

# 语音识别技术概述

哈爾濱工業大學
Harbin Institute of Technology

# 语音识别示意图



# 语音识别算法

- 语音识别任务的分类

- 按词汇表（**Vocabulary**）的大小分
  小词汇表系统：包括10~100个词条
  中词汇表系统：包括100~1000个词条
  大词汇表系统：至少包含1000个以上的词条
- 按照发音方式分
  孤立词（**Isolated Word** ）识别
  连接词（**Connected Word**）识别
  连续语音（**Continuous Speech**）识别

# 语音识别算法

- 按说话人的限定范围分

  特定人（Speaker Dependent，SD）识别

  非特定人（Speaker-Independent，SI）识别

特定人小词表孤立词系统

动态时间归正方法（DTW）

非特定人大词表连续语音识别任务

隐马尔科夫模型方法 (HMM)

# 语音识别算法

## ➢动态时间归正

**DTW(Dynamic Time Warping )**是一种模板匹配技术，是基于相似度计算与匹配实现的识别方法。

- 计算两个标量 $x_1$和 $x_2$ 的相似度

$$d = | x_1 - x_2 |$$

- 计算两个矢量 $\vec{x}_1 = \{x_{11}, \ldots x_{1n}\}$和 $\vec{x}_2 = \{x_{21}, \ldots x_{2n}\}$ 的相似度

$$d(\vec{x}_1, \vec{x}_2) = \sum_{i=1}^{n} (x_{1i} - x_{2i})^2$$

欧式距

- 经过预处理和特征提取后的语音可以看作矢量的序列

$$X = (\vec{x}_1, \vec{x}_2, \ldots, \vec{x}_M)$$

# 语音识别算法

- 如何计算两个矢量序列$X_1$和$X_2$之间的相似度 ???

一个直接的想法

$$D(X_1, X_2) = \sum_{i=1}^{M} d(\vec{x}_{1i}, \vec{x}_{2i})$$

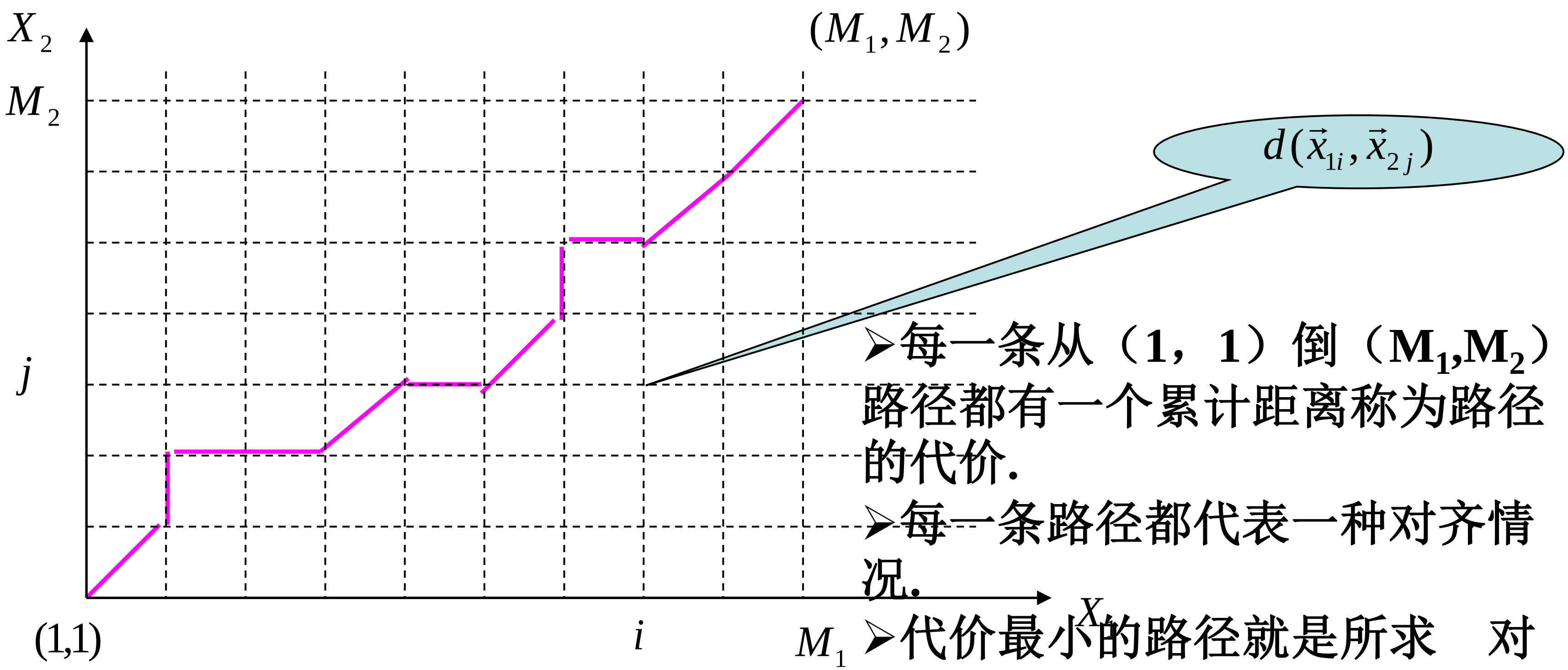
存在问题：

- 长度不同，$M_1 \neq M_2$
- 对不准

**DTW:**将表示两个语音段的矢量序列对准了再计算相似度。
或者说在时间上归正后再计算相似度。

# 语音识别算法

- 如何对准




➢每一条从（**1**，**1**）倒（$\mathbf{M_1,M_2}$）路径都有一个累计距离称为路径的代价.

➢每一条路径都代表一种对齐情况.

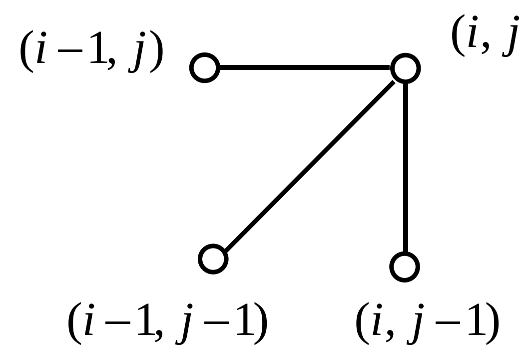
➢代价最小的路径就是所求　对准路径。

# 语音识别算法

- 将对准问题，或者说将求两个语音段的相似度问题，转化成了搜索代价最小的最优路径问题。
- 事实上，在搜索过程中，往往要进行路径的限制

（1）起点/终点的限制

（2）连续性限制




- 再此限制条件下，可以将全局最优化问题转化为许多局部最优化问题一步一步地来求解，这就动态规划(Dynamic Programming，简称DP )的思想。

# 语音识别算法

定义一个代价函数 $\Phi(i,j)$ 表示从起始点**(1,1)**出发，到达（***i***，***j***）点最小代价路径的累计距离。

有： $$\Phi(i,j)=\min_{(i',j')->(i,j)}\{\Phi(i',j')+d(\vec{x}_{1i},\vec{x}_{2j})w_n\}$$

则：$$\begin{aligned}\Phi(M_1,M_2)=\min\{&\Phi(M_1-1,M_2)+d(\vec{x}_{1M1},\vec{x}_{2M2})w_n,\\ &\Phi(M_1,M_2-1)+d(\vec{x}_{1M1},\vec{x}_{2M2})w_n,\\ &\Phi(M_1-1,M_2-1)+d(\vec{x}_{1M1},\vec{x}_{2M2})w_n\}\end{aligned}$$

依次类推，$\Phi(M_1-1,M_2)$、$\Phi(M_1,M_2-1)$、$\Phi(M_1-1,M_2-2)$ 可由更低一层的代价函数计算得到。

# 语音识别算法

- 这样就可从

$\Phi(1,1)$ 逐步向上搜索。

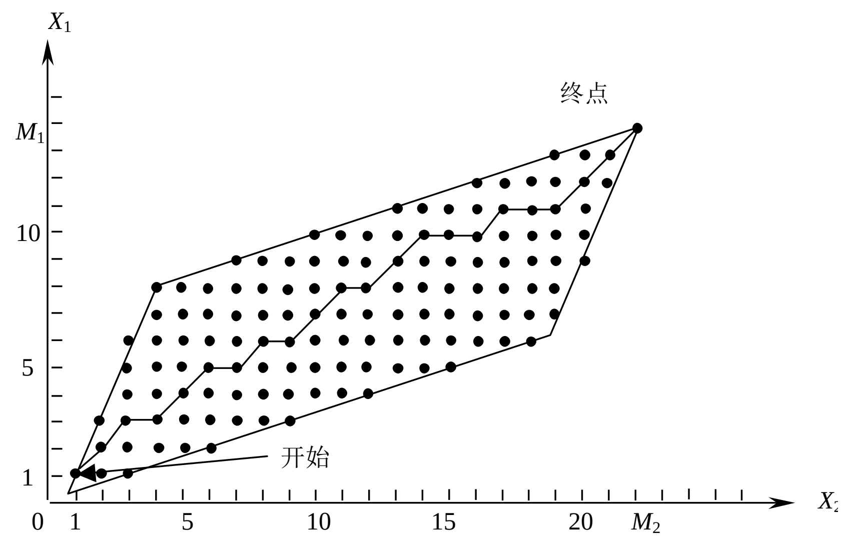
- 加权系数的取值与局部路径有关

$$w_n = \begin{cases} 2 & (i-1, j-1) \to (i, j) \\ 1 & \text{其它} \end{cases}$$

- 定义回溯函数

$P(i, j)$

- 平行四边形区域约束



# 语音识别算法

**DTW路径搜索算法**

（**1**）初始化： $i = j = 1,\ \Phi(1,1) = d(\vec{x}_{11}, \vec{x}_{21})$

$$\Phi(i,j) = \begin{cases} 0 & \text{当}(i,j) \in \mathrm{Reg} \\ huge & \text{当}(i,j) \notin \mathrm{Reg} \end{cases}$$

**其中约束区域Reg可以假定是这样一个平行四边形，它有两个顶点位于(1,1)和($\mathbf{M_1}$,$M_2$)，相邻两条边的斜率分别为2和1/2。**

# 语音识别算法

(2) 递推求累计距离 并记录回溯信息

$$\Phi(i,j)=\min\{\Phi(i-1,j)+d(\vec{x}_{1i},\vec{x}_{2j})\cdot W_n(1);\Phi(i-1,j-1)+d(\vec{x}_{1i},\vec{x}_{2j})\cdot W_n(2);$$

$$\Phi(i,j-1)+d(\vec{x}_{1i},\vec{x}_{2j})\cdot W_n(3)\}$$

$$i=2,3,\cdots,M_1;\ j=2,3,\cdots,M_2;(i,j)\in \mathrm{Reg}$$

一般取距离加权值为，$W_n(1)=W_n(3)=1$ $W_n(2)=2$

并将$(i,j)$点的回溯信息记录在$p(i,j)$中

# 语音识别算法

**（3）回溯求出所有的匹配点对：根据每步的上一步最佳局部路径*p*(*i*,*j*)，由匹配点($M_1$,$M_2$)对向前回溯一直到(1,1)。这个回溯过程对于求平均模板或聚类中心来讲是必不可少的，但在识别过程往往不必进行。**

- 对所求得的 $\Phi(M_1, M_2)$ 还需用 $\sum W_n$ 来归正

# 语音识别算法

- 模板的训练

**1 偶然训练法**

将每个词的每一遍语音形成一个模板。在识别时，待识别矢量序列用**DTW**算法分别求得与每个模板的累计失真，综合在一起来形成总失真。

这种方法具有很大的偶然性。

**2 顽健模板训练方法**

这种方法将每个词重复说多遍，直到得到一对一致性较好的特征矢量序列。最终得到的模板是在一致性较好的特征矢量序列对在沿**DTW**的路径上求平均。

# 语音识别算法

若 $D(X_1, X_2) < \sigma$ 且最优路径为

$$(i(1), j(1)), (i(2), j(2)), \ldots, (i_{T_y}(T_y), j_{T_y}(T_y))$$

则可得到新的模板 ***Y***，长度为$\boldsymbol{T}_{\mathbf{y}}$

$$y_k = \frac{1}{2}(x_{1i(k)} + x_{2j(k)}), \qquad k = 1, 2, \ldots T_y$$

比偶然训练法可靠，但不够充分。当识别任务是针对非特定人时，这种问题更为突出。

# 语音识别算法

- 非特定人识别的模板训练算法—聚类方法

令***Ω***为***L***个训练序列的集合，***Ω*** $=\{X_1, X_2, \cdots, X_L\}$，其中，每个元素为某特定语音的一次实现，即一次发音。对每两次发音的特征矢量序列进行匹配计算，得到的匹配距离 $\delta(X_i, X_j)$ ，则可构成一个 $L\times L$ 的距离矩阵。聚类的目的是将训练集***Ω***聚成***N***个不同的类 $\{\omega_i; i=1,2,\ldots,N\}$，使 $\Omega=\bigcup_{i=1}^{N}\omega_i$ ，在同一类中的语音模式比较相近。

**MKM聚类算法.doc**

# 语音识别算法

课堂练习：

要求：编制**DTW**匹配程序

输入：语音矢量序列$\mathbf{X_1}$，$\mathbf{X_2}$

输出：$\mathbf{X_1}$，$\mathbf{X_2}$的相似度得分

# 马尔可夫链

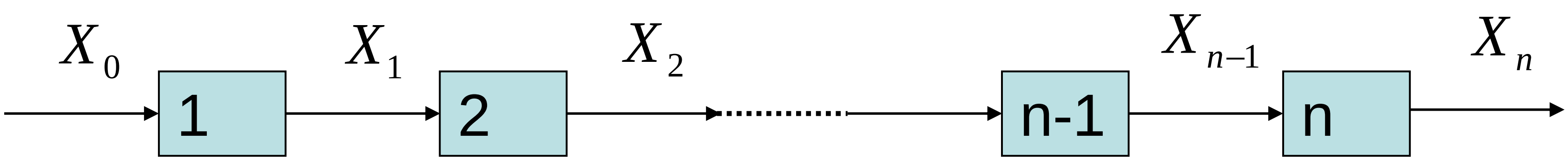
# 马尔可夫链

- 因俄国数学家安德烈·马尔可夫而得名，是他在1906年提出来的。
- 状态空间是有限的或可列；
- 是一种离散时间随机过程。即指标集 $T=(0,1,2,\ldots)$；
- 具有马尔可夫性质（无后效性）。即在给定当前知识或信息的情况下，过去（即当期以前的历史状态）对于预测将来（即当期以后的未来状态）是无关的。

# 马尔可夫性质示例

◆ **简单信号模型**

在某数字通讯系统中，只传输 0、1两种信号，且传输要经过很多级，且每级中由于噪声的存在会引起误差。假设每级输入0、1信号后，其输出不产生误差的概率为 $X_n$． 记$\{X_n, n \geq 0\}$为第 $n$ 级的输出信号． 则它是状态有限的马尔科夫链，．



# 马尔可夫性质示例

◆ 抛硬币输赢模型

假设甲乙两人以抛硬币的方式进行赌博，每次抛同一枚硬币；若出现正面，则甲付给乙一元钱，若出现反面，则乙付给甲一元钱.记 $X_n$ 为第 $n$ 局之后甲赢的总钱数.则 $\{X_n, n \geq 0\}$ 是马尔可夫链.

# 马尔可夫性质示例

- 有时是计算处理的需要；
- 计算符号串的概率

如拼音输入法

拼音串：wo zai deng ni

对应字串：我在等你

我在瞪你

我载邓妮

窝仔灯拟

。。。

决策时需要计算每个可能的字串的概率

# 定义----马尔可夫性质示例

◆ 计算字符串 $\{X_n, n \geq 0\}$ 的发生概率

$P(X_0 = i_0$

$= P(X_K =$ … $_{K-1})$

$= P(X_K$

$P(X_{K-}$

- 各条件概率理论上是可以估计的，因而可以事先得到并存储。
- 计算特定字符串的概率时，通过查表来得到。
- 然而，存储各条件概率的数据表需要多少存储空间呢？

•

$P(X_2 = i_2 \mid X_0 = i_0, X_1 = i_1) \cdot$

$P(X_1 = i_1 \mid X_0 = i_0) \cdot P(X_0 = i_0)$

# 马尔科夫链的定义

定义 设随机过程 $\{X_n, n \geq 0\}$ 的状态空间为：$S = \{0, 1, 2, 3, \cdots\}$

若对任意的 $n \geq 0$ ，及 $i_0, i_1, i_2, \cdots, i_{n-1}, i, j \in S$ 有

$$P\{X_{n+1} = j \mid X_n = i, X_{n-1} = i_{n-1}, \cdots, X_1 = i_1, \ X_0 = i_0\}$$
$$= P\{X_{n+1} = j \mid X_n = i\}$$

马氏性

则称 $\{X_n, n \geq 0\}$ 为离散时间、离散状态的马尔可夫过程，或简称为马尔可夫链。

# 马尔科夫链的定义

- 字符串的概率可计算为：

$$P(X_0 = i_0, X_1 = i_1, \ldots, X_{K-1} = i_{K-1}, X_K = i_K)$$

$$= P(X_K = i_K \mid X_{K-1} = i_{K-1}) \cdot P(X_{K-1} = i_{K-1} \mid X_{K-2} = i_{K-2}) \cdot$$

$$\cdots$$

$$P(X_2 = i_2 \mid X_1 = i_1) \cdot P(X_1 = i_1 \mid X_0 = i_0) \cdot P(X_0 = i_0)$$

- 即马尔可夫链$\{X_n, n \geq 0\}$的有限维分布完全由初始分布

$P\{X_0 = i\}$ 和 条件概率 $P\{X_n = j \mid X_{n-1} = i\}$ 确定.

# 马尔科夫链的定义

定义1 设$\{X_n, \ n \geq 0\}$是马尔可夫链，记

$$a_{ij}(n) = P\{X_{n+1} = j \mid X_n = i\}$$

称$a_{ij}(n)$为马尔可夫链$\{X_n, \ n \geq 0\}$在时刻$n$时的一步转移概率。有：

$$a_{ij}(n) \geq 0, \ \forall i, \ j \in S, \ n > 0;$$
$$\sum_{j \in S} a_{ij}(n) = 1, \ \forall i \in S, \ n > 0.$$

若其一步转移概率$a_{ij}(n)$与时间$n$无关，即：

$$a_{ij} = P\{X_{n+1} = j \mid X_n = i\} = P\{X_1 = j \mid X_0 = i\}$$

则称$\{X_n, \ n \geq 0\}$为齐次马尔可夫链

# 马尔科夫链的定义

齐次马尔科夫链的一步转移概率矩阵.

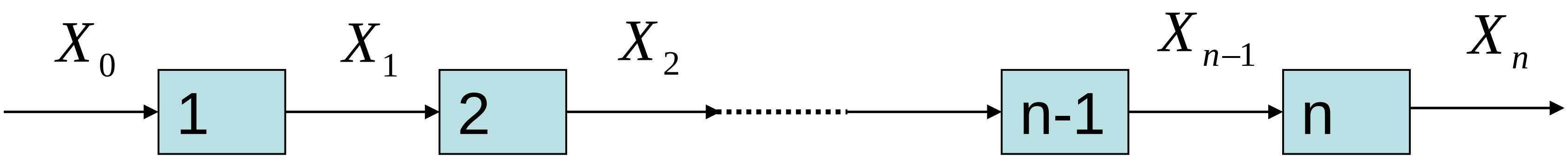
矩阵的每一行都是一条件分布律

$$A=(a_{ij})=\begin{pmatrix} a_{00} & a_{01} & a_{02} & \cdots & a_{0j} & \cdots \\ a_{10} & a_{11} & a_{12} & \cdots & a_{1j} & \cdots \\ a_{20} & a_{21} & a_{22} & \cdots & a_{2j} & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \cdots \\ a_{i0} & a_{i1} & a_{i2} & \cdots & a_{ij} & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \cdots \end{pmatrix}$$

记 $\pi=(\pi_0,\ \pi_1,\cdots)$，$(\pi_i=P\{X_0=i\},\ i\in S)$. 称 $\pi$ 为齐次马尔可夫链的初始分布.

# 转移概率和初始分布

例1 简单信号模型的转移概率矩阵




$$a_{ij} = P\{X_{n+1} = j \mid X_n = i\} = \begin{cases} p, & j = i \\ q, & j \neq i \end{cases} \quad i, j = 0,1$$

$$A = \begin{bmatrix} p & q \\ q & p \end{bmatrix}$$

# 转移概率和初始分布

例2（一个简单的疾病死亡模型）

考虑一个包含两个健康状态$S_1$和$S_2$以及两个死亡状态$S_3$和$S_4$（即由不同原因引起的死亡）的模型。若个体病愈，则认为它处于状态$S_1$，若患病，则认为它处于$S_2$，个体可以从$S_1$，$S_2$进入$S_3$和$S_4$，易见这是一个马氏链，转移矩阵为

$$A=\begin{pmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

# 转移概率和初始分布

例3 某计算机机房的一台计算机经常出故障，研究者每隔15分钟观察一次计算机的运行状态，收集了24个小时的数(共作97次观察)，用1表示正常状态，用0表示不正常状态，所得的数据序列如下：

111001001111111001111011111100111111111000110110

1111011011010111101110111101111110011011111100111

设$X_n$为第$n$($n$=1,2,…,97)个时段的计算机状态，可以认为它是一个齐次马氏链。求

(1)一步转移概率矩阵；

(2)已知计算机在某一时段(15分钟)的状态为0，问在此条件下，从此时段起，该计算机能连续正常工作45分钟(3个时段)的条件概率.

# 转移概率和初始分布

**解：**(1) 设$X_n$为第$n$($n$=1,2,…,97)个时段的计算机状态，可以认为它是一个齐次马氏链，状态空S={0,1}，

96次状态转移情况是： 0→0：8次； 0→1：18次；

1→0：18次； 1→1：52次；

因此一步转移概率可用频率近似地表示为：

$$a_{00}=P\left(X_{n+1}=0\,|\,X_n=0\right)\approx\frac{8}{8+18}=\frac{8}{26}$$

$$a_{01}=P\left(X_{n+1}=1\,|\,X_n=0\right)\approx\frac{18}{18+8}=\frac{18}{26}$$

$$a_{10}=P\left(X_{n+1}=0\,|\,X_n=1\right)\approx\frac{18}{18+52}=\frac{18}{70}$$

$$a_{11}=P\left(X_{n+1}=1\,|\,X_n=1\right)\approx\frac{52}{18+52}=\frac{52}{70}$$

$$\text{即}:A=\begin{bmatrix}\frac{8}{26} & \frac{18}{26}\\ \frac{18}{70} & \frac{52}{70}\end{bmatrix}$$

# 转移概率和初始分布

(2) 某一时段的状态为0, 定义其为初始状态, 即$X_0=0$, 所求概率为:

$$
\begin{aligned}
&P\left(X_1=1, X_2=1, X_3=1 \mid X_0=0\right)\\
&=P\left(X_1=1 \mid X_0=0\right) P\left(X_2=1 \mid X_0=0, X_1=1\right)\\
&\quad \times P\left(X_3=1 \mid X_0=0, X_1=1, X_2=1\right)\\
&=a_{01} a_{11} a_{11}\\
&=\frac{18}{26} \frac{52}{70} \frac{52}{70}=0.382
\end{aligned}
$$

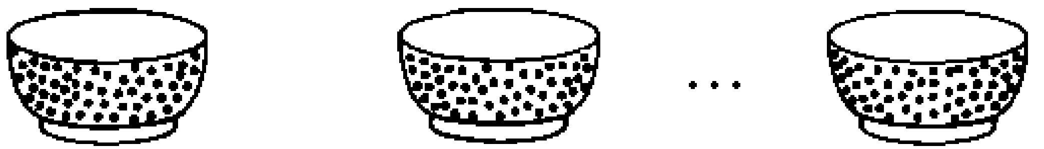
# 隐马尔可夫模型

# 语音识别算法

- 实际问题比Markov链模型所描述的更为复杂。观察到的事件并不是与状态一一对应，而是通过一组概率分布相联系 。
- 使用双重随机过程来描述模型，一个是Markov链，描述状态的转移；另一个随机过程描述状态和观察值之间的统计对应关系。
- 由于状态是不可见的，因此称之为“隐” Markov模型。

# 语音识别算法

- 一个HMM的例子： **Ball and Urn**



缸 1　　缸 2　　缸 $N$

$P(红)=b_1(1)$　$P(红)=b_2(1)$　$P(红)=b_N(1)$

$P(绿)=b_1(2)$　$P(绿)=b_2(2)$　$P(绿)=b_N(2)$

$P(蓝)=b_1(3)$　$P(蓝)=b_2(3)$　$P(蓝)=b_N(3)$

# 语音识别算法

◆一个HMM可以由下列参数描述

初始状态概率 $\pi = (\pi_1, \cdots, \pi_N)$

$$\pi_i = P(q_1 = \theta_i) \qquad 1 \le i \le N$$

状态转移概率矩阵A= $(a_{ij})_{N \times N}$

$$a_{ij} = P(q_{t+1} = \theta_j \mid q_t =$$

单高斯概率密度函数
或 混合高斯概率密度函数

观察概率序列B= $(b_1(o), b_2(o), \ldots, b_N(o))$

$$b_j(o) = \sum_{k=1}^{K} c_{jk} N(o, u_j, \Sigma_j)$$

◆一个HMM的参数组为:

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

# 语音识别算法

- **HMM的三个基本问题**

1 已知一个**HMM**参数组 $\lambda=(\pi,A,B)$，和给定一个观察序列 $O=o_1o_2....o_T$ 的条件下，如何计算在给定模型 $\lambda$ 条件下观察序列***O***的概率$P(O|\lambda)$。

2 如何确定最佳状态序列 $Q=q_1q_2...q_T$,以最好的解释观察序列***O***。

3 给定一组观察序列的集合$\{O_m\}$，如何调整参数 $\lambda$ ，以使$P(\{O_m\}|\lambda)$达到最大值。

# 语音识别算法

- 计算概率 $P(O|\lambda)$

先计算 $P(O,Q|\lambda)$，其中$Q$为一给定的状态序列

$$Q = q_1 q_2, \ldots, q_T$$

有：$$P(O,Q|\lambda) = P(O|Q,\lambda)P(Q|\lambda)$$

而 $$P(O|Q,\lambda) = \prod_{t=1}^{T} P(o_t|q_t,\lambda) = b_{q_1}(o_1)b_{q_2}(o_2)\cdots b_{q_T}(o_T)$$

$$P(Q|\lambda) = \pi_{q_1} a_{q_1 q_2} \cdots a_{q_{T-1} q_T}$$

所以 $$P(O,Q|\lambda) = \pi_{q_1} b_{q_1}(o_1) a_{q_1 q_2} b_{q_2}(o_2) \cdots a_{q_{T-1} q_T} b_{q_T}(o_T)$$

# 语音识别算法

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

$$=\sum_{\forall Q}\ \pi_{q_2}b_{q_1}(o_1)a_{q_1q_2}b_{q_2}(o_2)\cdots a_{q_{T-1}q_T}b_{q_T}(o_T)$$

计算量 ： $2TN^T$

当$N=5,\ \ T=100$ 时，计算量达$10^{72}$

- 前向—后向算法

定义前向变量为： $\alpha_t(i)=P(o_1o_2\cdots o_t,q_t=i|\lambda)$

# 语音识别算法

前向变量有如下性质：

（1）初值易求 $\alpha_1(i) = P(o_1, q_1 = i) = \pi_i b_i(o_1)$

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

（3）有递推关系 $\alpha_{t+1}(j) = \left[ \sum_{i=1}^{N} \alpha_t(i) a_{ij} \right] b_j(o_{t+1})$

因此可以利用递推关系，逐层递推，计算出全部 $\alpha_t(j)$ $1 \le t \le T-1$ $1 \le j \le N$。最后再由 $\alpha_T(i)$ 计算得到 $P(O|\lambda)$

# 语音识别算法

(a) 初始化：对$1 \le i \le N$

$$\alpha_1(i) = \pi_i b_i(o_1)$$

(b) 递推：对$1 \le t \le T-1$，$1 \le j \le N$

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

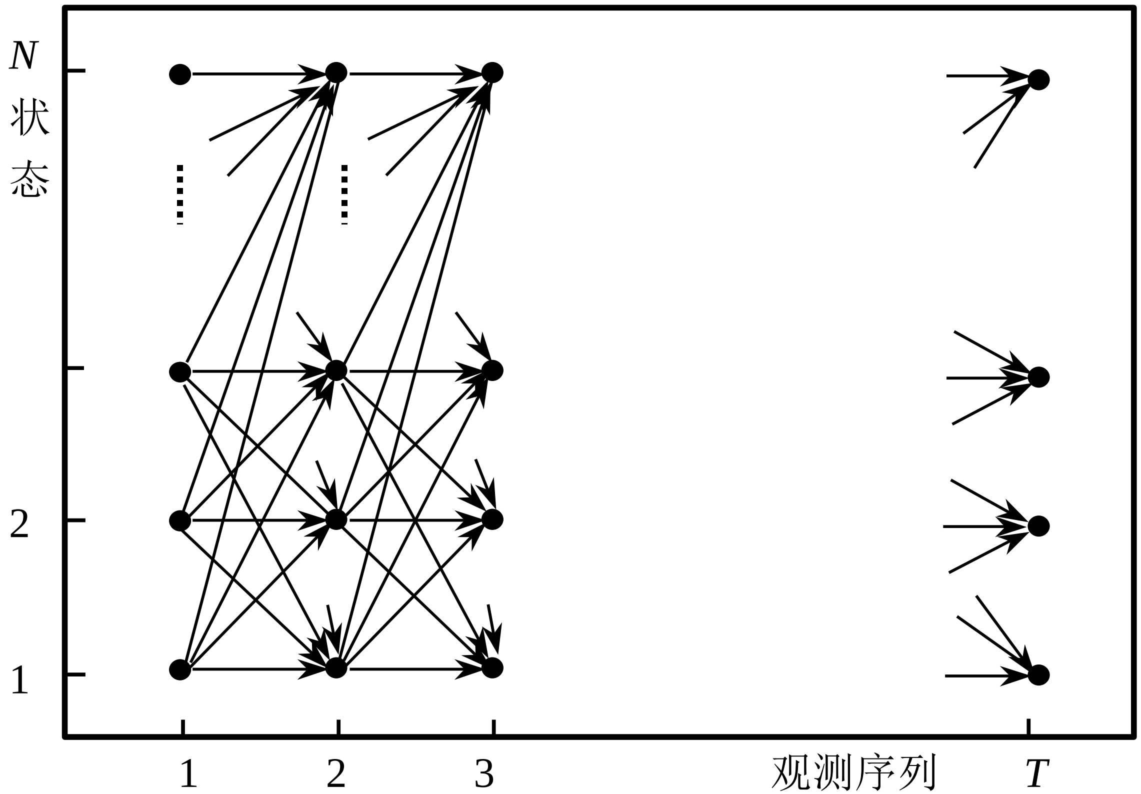
(c) 终止：

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

**计算量为$N^2T$，$N$=5  $T$=100时，只需2500次乘法运算**

# 语音识别算法

## 格型结构



# 语音识别算法

定义后向变量为

$$\beta_t(i) = P(o_{t+1}o_{t+2}\cdots o_T \mid q_t = i, \lambda)$$

有初值 $\beta_T(i) = b_i(O_T)$

有递推关系 $\beta_t(i) = \sum_{j=1}^{N} a_{ij} b_j(o_{t+1}) \beta_{t+1}(j)$

且易知 $P(O \mid \lambda) = \sum_{j=1}^{N} \alpha_t(j) \beta_t(j)$

# 语音识别算法

- 最佳状态链的确定

确定一个最佳状态序列 $Q^* = q_1^*, q_2^*, \cdots, q_T^*$ ，使 $P(O, Q^* | \lambda)$为最大。

$$Q^* = \arg\max_{Q} P(O, Q | \lambda)$$

**Viterbi算法**

定义 $\delta_t(i) = \max_{q_1, q_2, \cdots q_{t-1}} P(q_1 q_2 \cdots q_{t-1}, q_t = i, o_1 o_2 \cdots o_t | \lambda)$

为在时刻$t$，沿一条路径$q_1, q_2, \cdots, q_t$，且 $q_t = i$，产生出$o_1 o_2 \cdots o_t$的最大概率

# 语音识别算法

**是否满足DP算法的三个条件**

**1 初值易求：**

$$\delta_1(i) = P(q_1 = i, o_1 \mid \lambda) = \pi_i b_i(o_1)$$

**2 能够解决问题：**

$$P(O, Q^* \mid \lambda) = \max_i \max_{q_1..q_{T-1}} P(o_1 o_2 .. o_T, q_1 .. q_{T-1}, q_T = i \mid \lambda) = \max_i \delta_T(i)$$

$i$

**3 有递推关系：**

# 语音识别算法

若已知 $\delta_t(i)$ 对应的最佳状态链 $q_1 q_2 \cdots q_{t-1} q_t$，设 $q_{t-1} = j$

可以证明 $\delta_{t-1}(j)$ 对应的最佳状态链为 $q_1 q_2 \cdots q_{t-1}$

即若已知 $q_{t-1} = j$ 则 $\delta_t(i) = \delta_{t-1}(j) a_{ji} b_i(o_t)$

实际上不知道 $q_{t-1} = j$，可以遍历所有的 $q_{t-1}$ 求最值有：

$$\delta_t(i) = \max_{1 \le j \le N} [\delta_{t-1}(j) a_{ji}] b_j(o_t)$$

可以用回溯的方式求出$Q^*$

# 语音识别算法

那么，求取最佳状态序列 $Q^*$的过程为

**(a)** 初始化：对$1 \le i \le N$

$$\delta_1(i) = \pi_i b_i(o_1)$$

$$\varphi_1(i) = 0$$

**(b)** 递推：对$2 \le t \le T,\ 1 \le j \le N$

$$\delta_t(j) = \max_{1 \le i \le N}[\delta_{t-1}(i) a_{ij}] b_j(o_t)$$

$$\varphi_t(j) = \arg\max_{1 \le i \le N}[\delta_{t-1}(i) a_{ij}]$$

# 语音识别算法

**c)** 终止：

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

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

路径回溯，确定最佳状态序列：

$$q_t^* = \varphi_{t+1}(q_{t+1}^*)\text{，}\quad t = T-1, T-2, \cdots, 1$$

# 语音识别算法

$\max_{Q} P(O,Q|\lambda)$ 事实上是 $\sum_{Q} P(O,Q|\lambda)$ 中举足轻重的唯一成分，因此，常常等价地使用 $\max_{Q} P(O,Q|\lambda)$ 来近似 $\sum_{Q} P(O,Q|\lambda)$ 。即**Viterbi**算法也就能用来计算 $P(O|\lambda)$ 。

在连接词和连续语音识别中，更多地采用**Viterbi**算法来进行识别操作。因为它不仅能计算得分，还能通过最佳状态链获得词的边界信息。

# 语音识别算法

- **MLE和EM**
  - **MLE---一元高斯分布**

$$\hat{\mu}, \hat{\sigma} = \arg\max_{\mu,\sigma} p(\{x_i\}|\mu, \sigma) = \arg\max_{\mu,\sigma} \prod_{i=1}^{N} p(x_i|\mu, \sigma)$$

改变目标函数

$$\hat{\mu}, \hat{\sigma} = \arg\max_{\mu,\sigma} \ln\{\prod_{i=1}^{N} p(x_i|\mu, \sigma)\}$$

$$= \arg\max_{\mu,\sigma} \sum_{i=1}^{N} \ln p(x_i|\mu, \sigma)$$

# 语音识别算法

$$\begin{aligned}\hat{\mu},\hat{\sigma} &= \arg\max_{\mu,\sigma}\sum_{i=1}^{N}\left(-\frac{(x_i-\mu)^2}{2\sigma^2}-\ln\sigma-\ln\sqrt{2\pi}\right)\\ &= \arg\max_{\mu,\sigma}\sum_{i=1}^{N}\left(-\frac{(x_i-\mu)^2}{2\sigma^2}-\ln\sigma\right)\\ &= \arg\min_{\mu,\sigma}\sum_{i=1}^{N}\left(\frac{(x_i-\mu)^2}{2\sigma^2}+\ln\sigma\right)\end{aligned}$$

目标函数

$J(\mu,\sigma)$

求极值

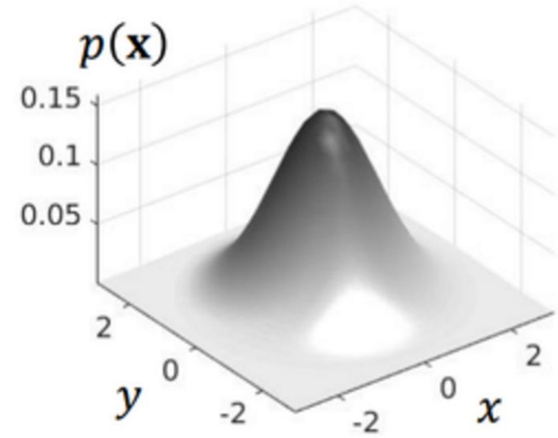
$$\begin{aligned}\frac{\partial J}{\partial \mu} &= \sum_{i=1}^{N}\frac{-2(x_i-\mu)}{2\sigma^2}\\ &= \frac{1}{\sigma^2}\left(N\mu-\sum_{i=1}^{N}x_i\right)\\ &= 0\end{aligned}$$

$$\hat{\mu}=\frac{1}{N}\sum_{i=1}^{N}x_i$$

$$\begin{aligned}\frac{dJ'(\sigma)}{d\sigma} &= \frac{N}{\sigma}-\sum_{i=1}^{N}\frac{(x_i-\mu)^2}{\sigma^3}\\ &= \frac{1}{\sigma}\left(N-\frac{1}{\sigma^2}\sum_{i=1}^{N}(x_i-\hat{\mu})^2\right)\\ &= 0\end{aligned}$$

$$\hat{\sigma}^2=\frac{1}{N}\sum_{i=1}^{N}(x_i-\hat{\mu})^2$$

# 语音识别算法




**MLE---多元高斯分布**

$$p(\boldsymbol{x}) = \frac{1}{(2\pi)^{D/2}|\Sigma|^{1/2}} \exp\{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^T\Sigma^{-1}(\boldsymbol{x}-\boldsymbol{\mu})\}$$

有

$$\begin{aligned}
\hat{\boldsymbol{\mu}}, \hat{\Sigma} &= \arg\max_{\boldsymbol{\mu},\Sigma} \ \ln\prod_{i=1}^{N} p(\boldsymbol{x_i}|\boldsymbol{\mu},\Sigma) \\
&= \arg\max_{\boldsymbol{\mu},\Sigma} \ \sum_{i=1}^{N} \ln p(\boldsymbol{x_i}|\boldsymbol{\mu},\Sigma) \\
&= \arg\max_{\boldsymbol{\mu},\Sigma} \ \sum_{i=1}^{N}(-\frac{1}{2}(\boldsymbol{x_i}-\boldsymbol{\mu})^T\Sigma^{-1}(\boldsymbol{x_i}-\boldsymbol{\mu}) - \frac{1}{2}\ln|\Sigma| - \frac{D}{2}\ln(2\pi)) \\
&= \arg\min_{\boldsymbol{\mu},\Sigma} \ \sum_{i=1}^{N}(\frac{1}{2}(\boldsymbol{x_i}-\boldsymbol{\mu})^T\Sigma^{-1}(\boldsymbol{x_i}-\boldsymbol{\mu}) + \frac{1}{2}\ln|\Sigma|)
\end{aligned}$$

# 语音识别算法

$$\frac{\partial J}{\partial \boldsymbol{\mu}} = \frac{\partial}{\partial \boldsymbol{\mu}} \sum_{i=1}^{N} \left(\frac{1}{2}(\boldsymbol{x}_i - \boldsymbol{\mu})^T \Sigma^{-1} (\boldsymbol{x}_i - \boldsymbol{\mu}) + \frac{1}{2}|\Sigma|\right)$$

$$= \frac{\partial}{\partial \boldsymbol{\mu}} \frac{1}{2} \sum_{i=1}^{N} (\boldsymbol{x}_i{}^T \Sigma^{-1} \boldsymbol{x}_i - \boldsymbol{x}_i{}^T \Sigma^{-1} \boldsymbol{\mu} - \boldsymbol{\mu}^T \Sigma^{-1} \boldsymbol{x}_i + \boldsymbol{\mu}^T \Sigma^{-1} \boldsymbol{\mu})$$

$$= \frac{\partial}{\partial \mu} \sum_{i=1}^{N} \left(\frac{1}{2} \boldsymbol{\mu}^T \Sigma^{-1} \boldsymbol{\mu} - \boldsymbol{\mu}^T \Sigma^{-1} \boldsymbol{x}_i\right)$$

$$= \Sigma^{-1} \sum_{i=1}^{N} (\boldsymbol{\mu} - \boldsymbol{x}_i)$$

$$\hat{\boldsymbol{\mu}} = \frac{1}{N} \sum_{i=1}^{N} \boldsymbol{x}_i$$

$$\frac{\partial J}{\partial \Sigma} = \frac{\partial}{\partial \Sigma} \sum_{i=1}^{N} \left(\frac{1}{2}(\boldsymbol{x}_i - \hat{\boldsymbol{\mu}})^T \Sigma^{-1} (\boldsymbol{x}_i - \hat{\boldsymbol{\mu}}) + \frac{1}{2} \ln|\Sigma|\right)$$

$$= \frac{1}{2} \sum^{N} (-\Sigma^{-1} (\boldsymbol{x}_i - \hat{\boldsymbol{\mu}})(\boldsymbol{x}_i - \hat{\boldsymbol{\mu}})^T \Sigma^{-1} + \Sigma^{-1})$$

$$\frac{\mathrm{d}\boldsymbol{x}^T A}{\mathrm{d}\boldsymbol{x}} = A \qquad \frac{\mathrm{d}A\boldsymbol{x}}{\mathrm{d}\boldsymbol{x}^T} = A$$

$$\frac{\mathrm{d}A\boldsymbol{x}}{\mathrm{d}\boldsymbol{x}} = A^T \qquad \frac{\mathrm{d}\boldsymbol{x}A}{\mathrm{d}\boldsymbol{x}} = A^T$$

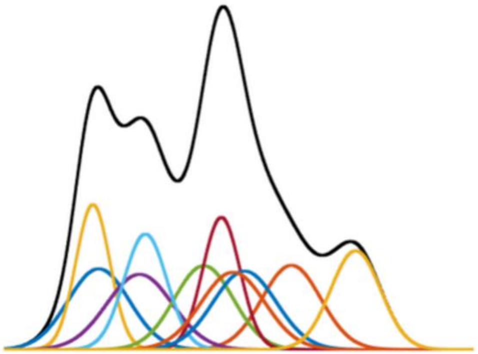
$$\frac{\partial \boldsymbol{u}^T \boldsymbol{v}}{\partial \boldsymbol{x}} = \frac{\partial \boldsymbol{u}^T}{\partial \boldsymbol{x}} \boldsymbol{v} + \frac{\partial \boldsymbol{v}^T}{\partial \boldsymbol{x}} \boldsymbol{u}^T$$

$$N \quad \sum_{i=1} \quad - \hat{\boldsymbol{\mu}})^T$$

# 语音识别算法

**EM---混合高斯分布**

$$p(\boldsymbol{x}) = \sum_{k=1}^{K} w_k g_k(\boldsymbol{x}|\boldsymbol{\mu}_k, \Sigma_k)$$



– 由于求和式出现对数函数中，因而求极值而得到的方程组不是线性方程组，无法求解
– 用迭代的方法来求解，先给个初值，然后认为均值、协方差矩阵已知，去估计权重，再根据权重，去估计均值和方差

# 语音识别算法

- **HMM模型的训练**

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 $P(O \mid \lambda)$ 最大。

实际上，不存在一种方法直接估计最佳的 $\lambda$。

替代的方法是：

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 $\bar{\lambda} = (\bar{\pi}, \bar{A}, \bar{B})$，保证有 $P(O \mid \bar{\lambda}) > P(O \mid \lambda)$。重复这个过程，逐步改进模型参数，直到 $P(O \mid \bar{\lambda})$ 收敛。

# 语音识别算法

- 这一方法，未必能求得全局最大值、而有可能得到一局部极值点
- 经典的方法：**Baum-Welch**算法。
- **Baum-Welch**算法的理论基础是EM算法。

定义辅助函数

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

# 语音识别算法

可以证明：若 $Q(\bar{\lambda},\lambda) \geq Q(\lambda,\lambda)$ 则有 $P(O\mid\bar{\lambda}) \geq P(O\mid\lambda)$

$$
\begin{aligned}
Q(\bar{\lambda},\lambda)-Q(\lambda,\lambda) &= \sum_{Q} P(Q,O|\lambda)\log\frac{P(Q,O|\bar{\lambda})}{P(Q,O|\lambda)} \\
&\leq \sum_{Q} P(Q,O|\lambda)\left(\frac{P(Q,O|\bar{\lambda})}{P(Q,O|\lambda)}-1\right) \\
&= P(O|\bar{\lambda})-P(O|\lambda).
\end{aligned}
$$

# 语音识别算法

计算 $\frac{\partial Q(\lambda,\bar{\lambda})}{\partial\bar{\lambda}}=0$ ，得到一组求取 $\bar{\lambda}$ 的公式，这一组公式就称为重估（**Re-Estimation** ）公式，它们是**Baum-Welch**算法 的核心内容。

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

$$= \sum_{Q} \log(\bar{\pi}_{q_1}\bar{b}_{q_1}(o_1)\prod_{t=2}^{T}\bar{a}_{q_{t-1}q_t}\bar{b}_{q_t}(o_t))p(O,Q\mid\lambda)$$

$$= \sum_{Q}\log\bar{\pi}_{q_1}p(O,Q\mid\lambda) + \sum_{Q}(\sum_{t=2}^{T}\log\bar{a}_{q_{t-1}q_t})P(O,Q\mid\lambda) + \sum_{Q}(\sum_{t=1}^{T}\log\bar{b}_{q_t}(O_t))p(O,Q\mid\lambda)$$

# 语音识别算法

$\pi_i$ 的重估

$$Q(\lambda,\bar{\lambda})=\sum_{Q}\log\bar{\pi}_{q_1}p(O,Q\mid\lambda)+\varphi_1=\sum_{i=1}^{N}\log\bar{\pi}_i p(O,q_1=s_i\mid\lambda)+\varphi_1$$

以及约束条件 $\sum_{i=1}^{N}\pi_i=1$

根据拉格朗日乘子法

$$\frac{\partial}{\partial\bar{\pi}_i}\left(\sum_{i=1}^{N}\log\bar{\pi}_i p(O,q_1=s_i\mid\lambda)+\varphi_1+r\ (1-\sum_{i=1}^{N}\bar{\pi}_i)\right)=0$$

$$\bar{\pi}_i=\frac{p(O,q_1=s_i\mid\lambda)}{p(O\mid\lambda)}=\frac{\alpha_1(i)\beta_1(i)}{\sum_{i=1}^{N}\alpha_1(i)\beta_1(i)}$$

# 语音识别算法

$a_{ij}$ 的重估

$$\varphi(\lambda,\bar{\lambda})=\sum_{i=1}^{N}\sum_{t=2}^{T}\log\bar{a}_{ij}\,p(O,q_{t-1}=s_i,q_t=s_j\mid\lambda)+\varphi_2$$

以及约束条件 $$\sum_{j=1}^{N}a_{ij}=1$$

以此类推，去重估观察概率。

# 语音识别算法

**Baum-Welch算法**

➢定义 $\xi_t(i, j)$ 为给定训练序列 $O$ 和模型 $\lambda$ 时，**HMM**模型在 $t$ 时刻处于状态 $i$，$t$+1时刻处于状态 $j$ 的概率。

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

易证

$$\xi_t(i, j) = [\alpha_t(i) a_{ij} b_j(o_{t+1}) \beta_{t+1}(j)] / P(O \mid \lambda)$$

➢定义**HMM**模型在$t$时刻处于状态$i$的概率为 $\gamma_t(i)$。

$$\gamma_t(i) = P(q_t = i \mid O, \lambda) = \sum_{j=1}^{N} \xi_t(i, j) = \alpha_t(i) \beta_t(i) / P(O \mid \lambda)$$

# 语音识别算法

重估公式可写成如下形式

$$\bar{\pi}_i = \gamma_1(i)$$

$$\bar{a}_{ij} = \sum_{t=1}^{T-1} \xi_t(i, j) \Big/ \sum_{t=1}^{T-1} \gamma_t(i)$$

若观察概率采用离散值

$$\bar{b}_{ik} = \sum_{\substack{t=1 \\ o_t = V_k}}^{T} \gamma_t(i) \Big/ \sum_{t=1}^{T} \gamma_t(i)$$

# 语音识别算法

若观察概率为多维连续高斯概率密度函数形式，即

$$b_i(o) = N(\mathbf{o},\boldsymbol{\mu}_i,\boldsymbol{\Sigma}_i) = \frac{1}{(2\pi)^{K/2}\left|\boldsymbol{\Sigma}_i\right|^{1/2}}\exp\left\{-\frac{(\mathbf{o}-\boldsymbol{\mu}_i)^T\boldsymbol{\Sigma}_i^{-1}(\mathbf{o}-\boldsymbol{\mu}_i)}{2}\right\}$$

则

$$\overline{u}_i = \frac{\sum_{t=1}^{T}\gamma_t(i)o_t}{\sum_{t=1}^{T}\gamma_t(i)} \qquad \overline{\Sigma}_i = \frac{\sum_{t=1}^{T}\gamma_t(i)(o_t-\overline{u}_i)(o_t-\overline{u}_i)^t}{\sum_{t=1}^{T}\gamma_t(i)}$$

# 语音识别算法

若观察概率为混合高斯分布形式，即

$$b_j(o_t)=\sum_{k=1}^{K}c_{jk}N(o_t,u_{jk},\Sigma_{jk})$$

则重估公式写为

$$\bar{c}_{jk}=\frac{\sum_{t=1}^{T}\upsilon_t(j,k)}{\sum_{t=1}^{T}\gamma_t(j)}\qquad \bar{u}_{jk}=\frac{\sum_{t=1}^{T}\upsilon_t(j,k)o_t}{\sum_{t=1}^{T}\upsilon_t(j,k)}$$

# 语音识别算法

$$\overline{\Sigma}_{jk} = \frac{\sum_{t=1}^{T} \upsilon_t(j,k)(o_t - \overline{u}_{jk})(o_t - \overline{u}_{jk})^t}{\sum_{t=1}^{T} \upsilon_t(j,k)}$$

式中

$$\upsilon_t(j,k) = \frac{\sum_{i=1}^{N} \alpha_{t-1}(i) a_{ij} c_{jk} b_{jk}(o_t) \beta_t(j)}{P(O \mid \lambda)}$$

# 语音识别算法

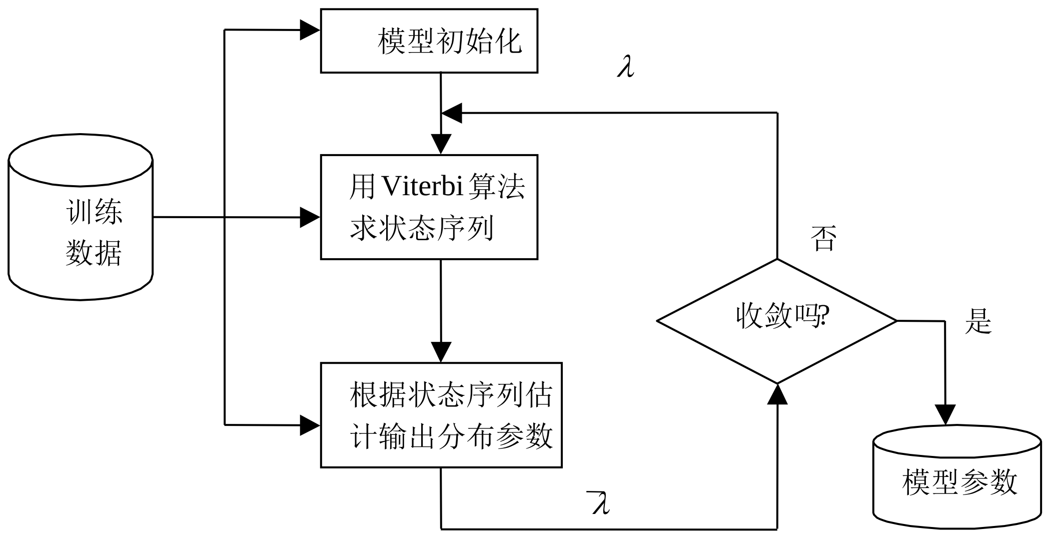
**HMM**算法实现中的问题

- 初始模型选取

初始模型的选取对**Baum-Welch**算法的结果有巨大影响。只有选取好的初始模型，才能使最后求出的局部极大与全局最大相接近。

最常采用的是一种基于Viterbi算法的初始模型选取方法。

# 语音识别算法



模型初始化
$\lambda$
训练
数据
用Viterbi算法
求状态序列
否
收敛吗?
是
根据状态序列估
计输出分布参数
$\bar{\lambda}$
模型参数

# 语音识别算法

- 根据状态序列重估

$$\bar{\pi}_i = \text{以状态}i\text{起始的语音段数量} / \text{语音段总数量}$$

$$\bar{a}_{ij} = \text{状态}i\text{后出现状态j的数量} / \text{状态}i\text{的数量}$$

若观察概率为单高斯概率密度函数形式

$$b_i(o) = N(\mathbf{o}, \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i) = \frac{1}{(2\pi)^{K/2} |\boldsymbol{\Sigma}_i|^{1/2}} \exp\left\{ -\frac{(\mathbf{o}-\boldsymbol{\mu}_i)^T \boldsymbol{\Sigma}_i^{-1} (\mathbf{o}-\boldsymbol{\mu}_i)}{2} \right\}$$

一般 $\bar{\Sigma}_i$ 采用对角阵形式

$$\bar{\Sigma}_i = \begin{bmatrix} \sigma_{i1}^2 & & 0 \\ & .. & \\ 0 & & \sigma_{iD}^2 \end{bmatrix}$$

# 语音识别算法

将 $\bar{u}_i$ 估计为所有状态标号为*i*的特征矢量的样本均值

将$\sigma_{id}^2$ 估计为所有状态标号为*i*的特征矢量第*d*维的方差

若观察概率为混合高斯概率密度函数形式

$$b_i(o_t)=\sum_{k=1}^{K}c_{ik}N(o_t,u_{ik},\Sigma_{ik})$$

需要将状态标号为*i*的特征矢量进行聚类,聚成*K*类,在每一类的样本中估计 $\bar{\mu}_{ik}$ $\bar{\Sigma}_{ik}$

$$\bar{c}_{ik}=\frac{\text{状态标号为}i\text{的特征矢量中属于第}k\text{类的数量}}{\text{状态标号为}i\text{的特征矢量的数量}}$$

# 语音识别算法

- 多个观察值序列训练

  用*L*个观察序列训练HMM时，要对**Baum-Welch**算法的重估公式加以修正。

$$\bar{\pi}_i = \sum_{l=1}^{L} \alpha_1^{(l)}(i)\beta_1^{(l)}(i) / P(O^{(l)} \mid \lambda)$$

$$\bar{a}_{ij} = \frac{\sum_{l=1}^{L}\sum_{t=1}^{T_l-1} \alpha_t^{(l)}(i) a_{ij} b_j(o_{t+1}^{(l)}) \beta_{t+1}^{(l)}(j) / P(O^{(l)} \mid \lambda)}{\sum_{l=1}^{L}\sum_{t=1}^{T_l-1} \alpha_t^{(l)}(i)\beta_t^{(l)}(j) / P(O^{(l)} \mid \lambda)}$$

# 语音识别算法

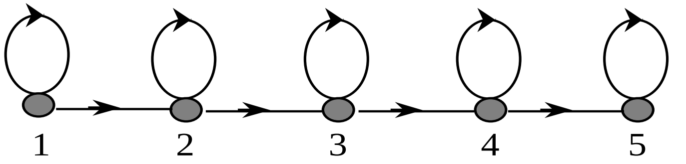
$$\bar{b}_{jk} = \frac{\sum_{l=1}^{L} \sum_{\substack{t=1\text{且} \\ o_t = V_k}}^{T_l} \alpha_t^{(l)}(i)\beta_t^{(l)}(j) / P(O^{(l)} | \lambda)}{\sum_{l=1}^{L} \sum_{t=1}^{T_l} \alpha_t^{(l)}(i)\beta_t^{(l)}(j) / P(O^{(l)} | \lambda)}$$

- 数据下溢问题

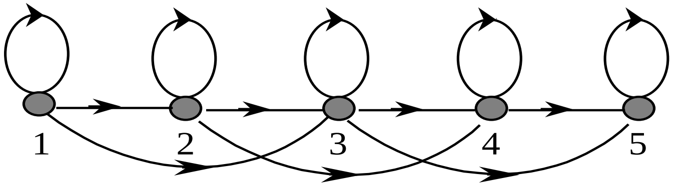
用对数似然度，取代概率值

# 语音识别算法

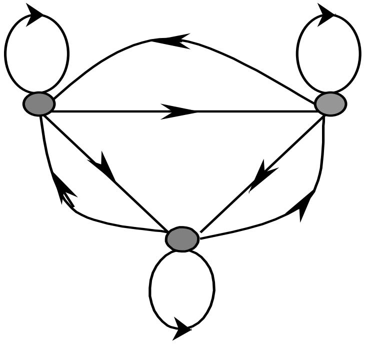
- **Markov链的形状和HMM的类型**




(a) 无跨越从左向右模型




(b) 有跨越从左向右模型



(c) 全连接模型

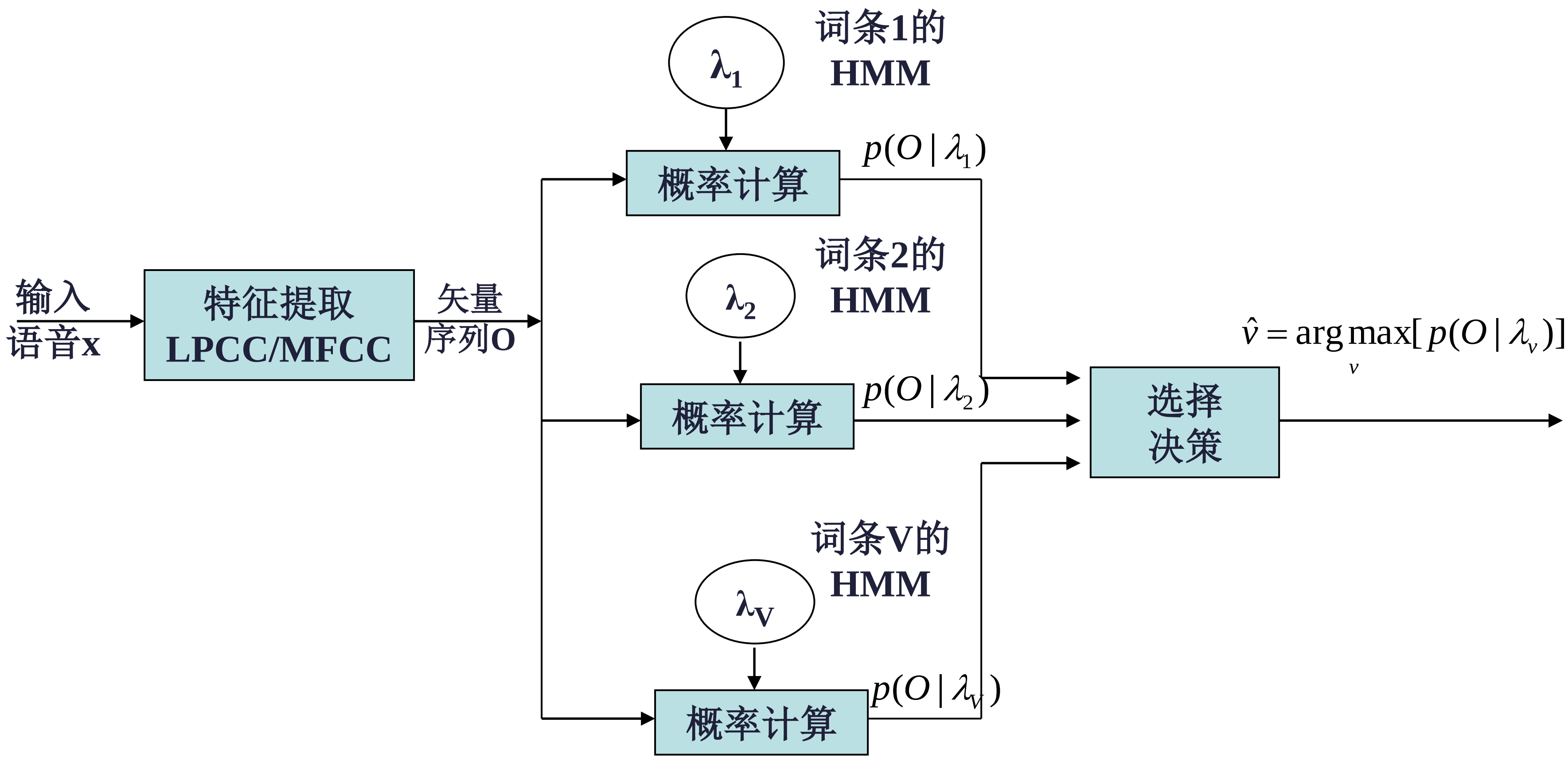
# 语音识别算法

**HMM的类型：**

- **离散HMM**
- **连续HMM**

# 语音识别算法

- 孤立词识别原理图



# 语音识别算法

- 连接词语音识别技术

- 连接词

（1） 连续发音，不知道语音中词的个数和词的边界信息。

（2） 词表有限，可以象孤立词识别一样，以词为单位建模。

- 连接词识别

连接词识别，就是指系统存储的模板或模型是针对孤立词的，但是识别的语音却是由这些词构成的词串。

# 语音识别算法

- 连接词识别问题的一般描述 (从DTW的角度）

设 给 定 测 试 发 音 的 特 征 矢 量 序 列 为 $O=\{o(1),o(2),...,o(M)\}$， 词表中 **V** 个词的模板分别为 $R_1,R_2,...,R_V$。某一个参考模板 $R_i$ 具有如下的形式：

$$R_i=\{r_i(1),r_2(2),...,r_i(N_i)\} \qquad 1\le i\le V$$

其中 $N_i$ 是第 ***i*** 个词参考模板的帧数。

连接词识别的问题变为，寻找与 ***O*** 序列最优匹配的参考模板序列 $R^*$， $R^*$ 是 ***L*** 个参考模板的连接：

$$R^*=\{R_{q^*(1)}\oplus R_{q^*(2)}\oplus R_{q^*(3)}\oplus....\oplus R_{q^*(L)}\}$$

其中每个 $q^*(l)$ 可能是[1， ***V***]中任意一个模板。

# 语音识别算法

一个超模板 $R^s$

$$R^s = R_{q(1)} \oplus R_{q(2)} \oplus R_{q(3)} \oplus .... \oplus R_{q(L)} = \{r^s(n)\}_{n=1}^{N^s}$$

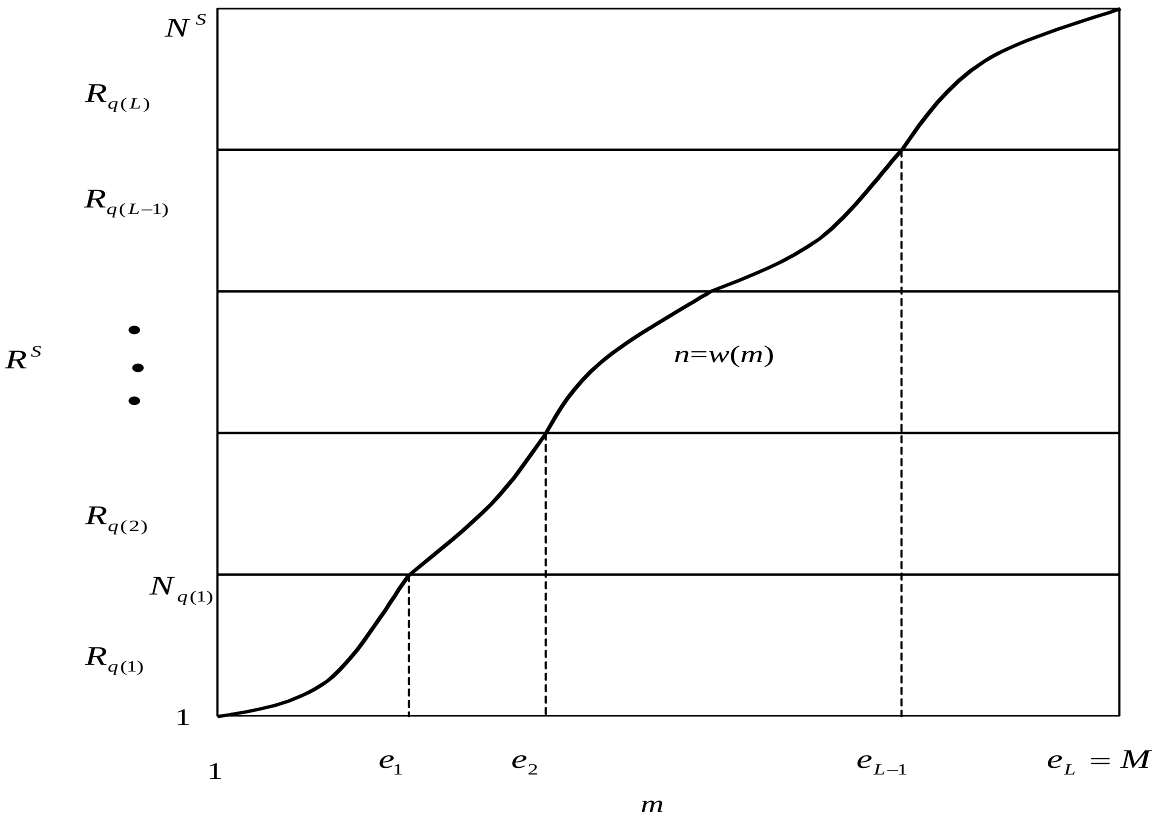
超模板与测试发音$O$之间的相似度可以用DTW来计算

$$D(R^s, O) = \min_{w(m)} \sum_{m=1}^{M} d(o(m), r^s(w(m)))$$

$d(\cdot\ ,\ \cdot)$ 为局部特征匹配距离，$w(\cdot)$ 是时间弯折函数

# 语音识别算法

## 超模板与测试发音之间的最优对齐路径



# 语音识别算法

$R^*$ 可以如下计算

$$D^* = \min_{R^s} D(R^s, O)$$

$$= \min_{L_{\min} \le L \le L_{\max}} \min_{\substack{q(1),q(2),\dots,q(L) \\ 1 \le q(i) \le V}} \min_{w(m)} \sum_{m=1}^{M} d(o(m), r^s(W(m)))$$

$$R^* = \arg\min_{R^s} D(R^s, O)$$

计算量太大，因此必须寻找快速算法：

- 二阶动态规划算法
- 分层构筑算法

# 语音识别算法

- 二阶动态规划算法

先定义两个函数

$$\tilde{D}(b,e)=\min_{1\le v<V}[\hat{D}(v,b,e)]$$

$$\tilde{N}(b,e)=\arg\min_{1\le v<V}[\hat{D}(v,b,e)]$$

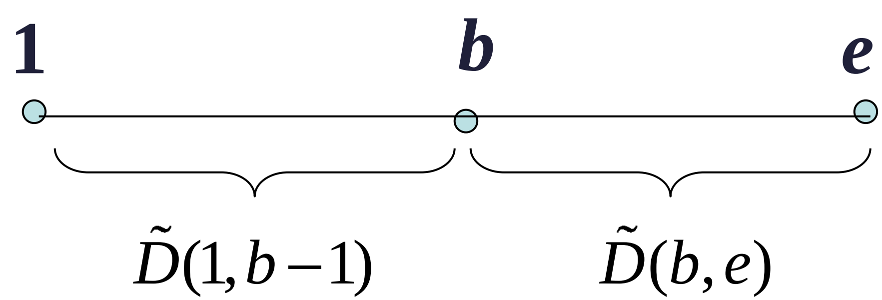
$\hat{D}(v,b,e)$表示起始点为***b***，结束点为***e***的语音段与模板***v***之间的DTW距离

# 语音识别算法

设 $\bar{D}_l(e)$ 为有***l***个词时，以***e***为终点的 $D^*$，有初值

$$\bar{D}_1(e) = \tilde{D}(1,e) \quad 1 \le e \le M$$

- 看一看当词的数目确定为2时




$$D^* = \min_{1<b<e}[\tilde{D}(1,b-1) + \tilde{D}(b,e)]$$

$$\bar{D}_2(e) = \min_{1\le b\le e}\{\bar{D}_1(b-1) + \tilde{D}(b,e)\} \quad 1 \le e \le M$$

# 语音识别算法

词的数目为*l*个时的递推式

$$\bar{D}_l(e) = \min_{1 \le b < e}[\bar{D}_{l-1}(b-1) + \tilde{D}(b,e)] \quad 1 \le e \le M$$

而***D****

$$D^* = \min_{1 \le l \le L_{\max}}[\bar{D}_l(M)]$$

# 语音识别算法

算法描述：

（1）初始化

$$\overline{D}_l(e)=\infty,\quad 1\le l\le L_{\max}\quad 0\le e\le M$$

（2）对 ***l*=1**

$$\overline{D}_1(e)=\tilde{D}(1,e),\quad 2\le e\le M$$

（3）递推，对 ***e*** 从 ***l*=2** 到 $L_{\max}$ 进行循环

$$\overline{D}_2(e)=\min_{1\le b<e}[\tilde{D}(b,e)+\overline{D}_1(b-1)],\quad 3\le e\le M$$

$$\overline{D}_3(e)=\min_{1\le b<e}[\tilde{D}(b,e)+\overline{D}_2(b-1)],\quad 4\le e\le M$$

$$\overline{D}_l(e)=\min_{1\le b<e}[\tilde{D}(b,e)+\overline{D}_{l-1}(b-1)],\quad l+1\le e\le M$$

# 语音识别算法

（**4**）最优解

$$D^* = \min_{1\le l\le L_{\max}}[\overline{D}_l(M)]$$

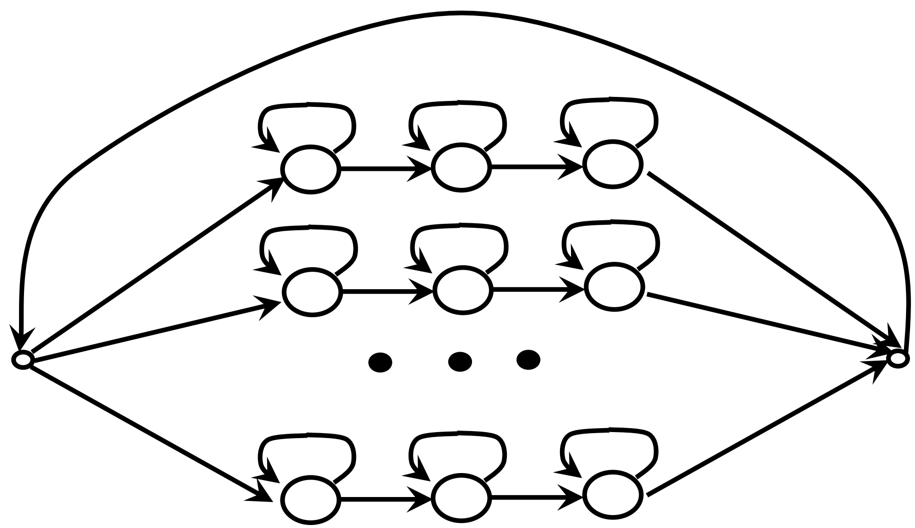
（**5**）回溯：利用$D^*$所对应的$\tilde{D}(b,e)$，可以找到其对应标号$\tilde{N}(b,e)$，以及最优路径上第 **l** 个模板的起始位置 **b**，**b–1** 即为第 **l–1** 个模板的结束位置 **e**，通过$\overline{D}_{l-1}(e)$可以找到第 **l–1** 个模板的起始位置，以及它的前一个模板的结束位置，以此类推，就可以逐步找出所有的最优模板。

# 语音识别算法

- 基于**HMM**的连接词识别

举例 ： 数字串识别

- **0-9**共**10**个数字，采用**3**状态从左至右无跨越**HMM**
- 形成一个新的**HMM(**识别网络**)**
- **Viterbi**解码



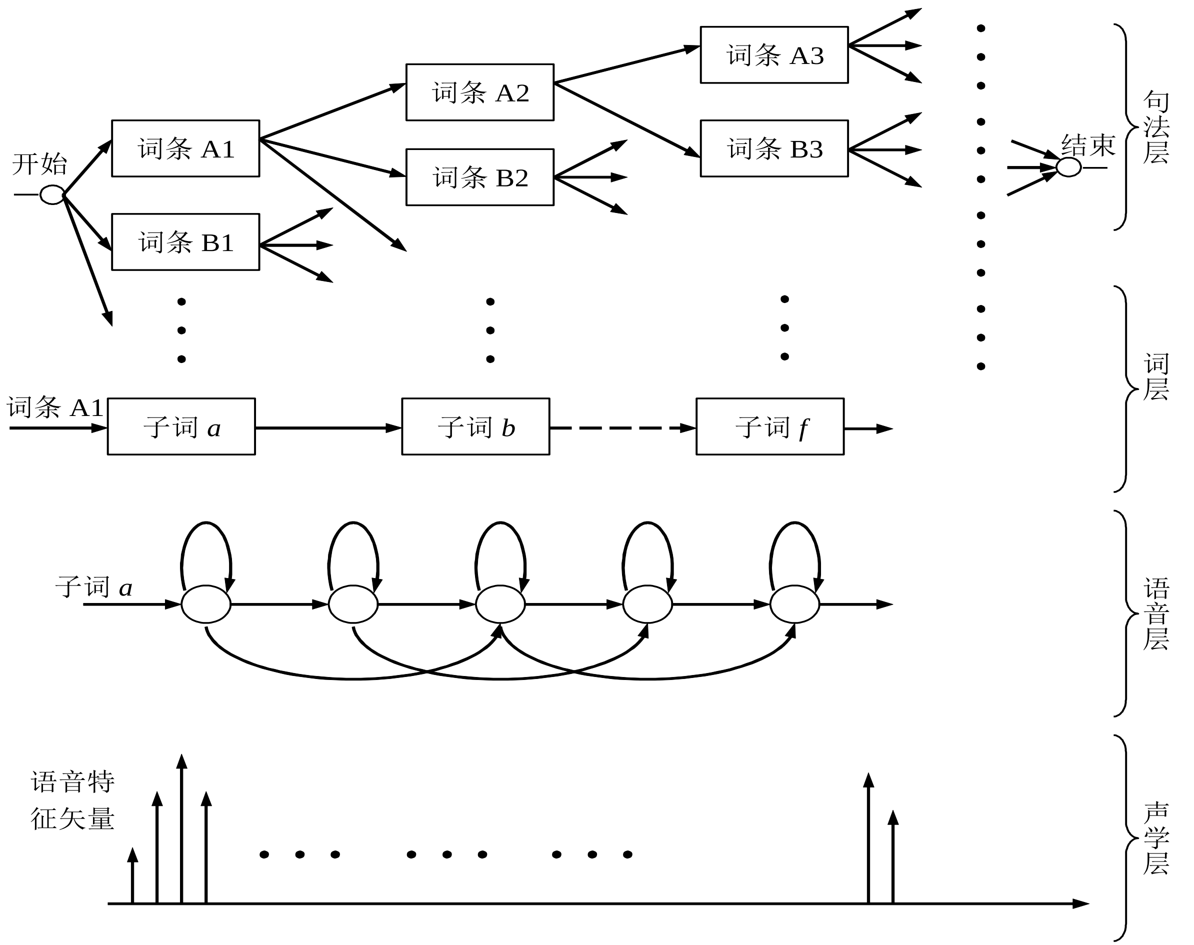
# 语音识别算法

- 大词汇量连续语音识别技术（LVCSR）
- 语音识别研究中意义最重大、应用成果最丰富，同时最具有挑战性的研究课题。
- 大词汇量非特定人的连续语音识别系统的词误识率大体为小词汇量、特定人的孤立词识别系统词误识率的50倍左右。
- 特有的问题：
  - ❖词（模式）的数量太多，语料不够。
  - ❖发音相近的内容多，误识严重。

# 语音识别算法

- 在上个世纪90年代初期，取得了里程碑式的成果（李开复和他的Sphinx）
- 基于HMM的LVCSR系统的统一框架，将整个识别系统分为三层：声学—语音层、词层和句法层。
  - ❖ 声学—语音层是识别系统的底层，它接受输入语音，并以一种“子词（Subword）”单位作为其识别输出，每个子词单位对应一套HMM结构和参数。
  - ❖ 词层规定词汇表中每个词是由什么音素—音子串接而成的
  - ❖ 句法层中规定词按照什么规则组合成句子。

# 语音识别算法


开始
词条 A1
词条 B1
词条 A2
词条 B2
词条 A3
词条 B3
结束
句法层
词条 A1
子词 a
子词 b
子词 f
词层
子词 a
语音层
语音特征矢量
声学层

# 语音识别算法

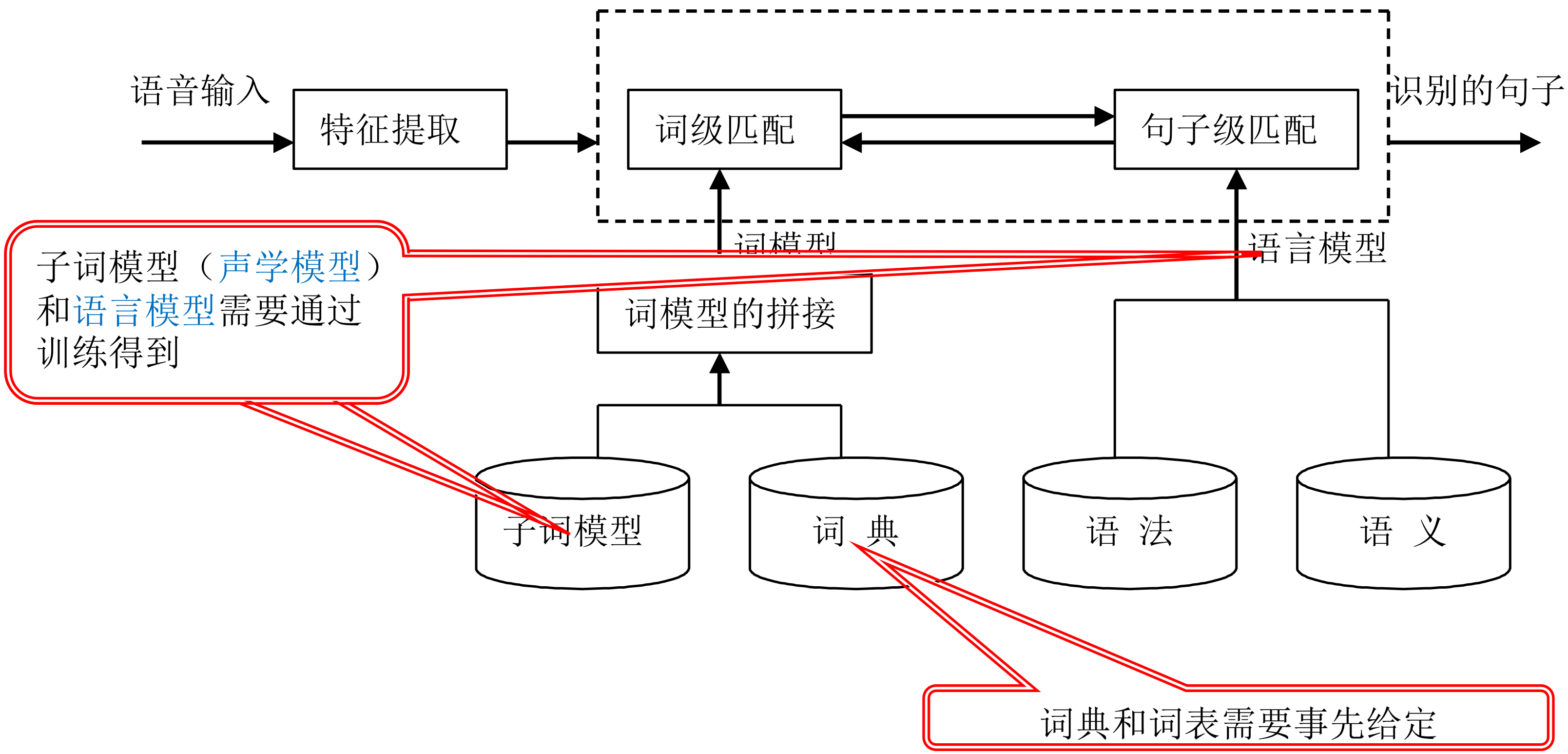
在句法层，每个句子由若干词条组成，需要通过语言模型评价所有可能的句子候选的合理性。

在词层，每一个词条由若干子词串接而成，为此需要一部词典来描述这种串接关系。

在语音层，每一个子词用一个HMM模型及一套参数来表示。

# 语音识别算法

## 基于子词单元的连续语音识别系统总体框图



# 语音识别算法

$$W^* = \arg\max_W P(W \mid O)$$

$$= \arg\max_W \frac{P(O \mid W)P(W)}{P(O)}$$

$$= \arg\max_W \boxed{P(O \mid W)}\boxed{P(W)}$$

语言学得分

声学得分

# 语音识别算法

**几个问题：**

**1）基本声学单元（子词）的选择?**

**2）如何得到词模型?**

**3）如何训练子词模型?**

**4）如何利用语言学知识?**

**5）如何识别?**

# 语音识别算法

- 声学模型

(1) 基本声学单元的选择

- 以词作为基本单元建立模型会造成大量不必要的冗余存储和计算。因此一般采用比词小的子词基元，如音节、半音节、音素等 。
- 声学单元越小，其数量也就越少，训练模型的工作量也就越小；
- 但是，单元越小，对上下文的敏感性越大，越容易受到前后相邻的影响而产生变异，因此其类型设计和训练样本的采集更困难。

# 语音识别算法

- 子词的数量应该是固定不变的。因而，英语只能选择音素作为建模基元。
- 单音素模型（**Monophone**）：每个音素建立一个**HMM**模型。
- 三音素模型（**Triphone**）：考虑协同发音效应，上下文不同则建立不同的**HMM**模型。例如：

下文

**t-iy+n**

上文

# 语音识别算法

- **Triphone数量太多（48×48×48），建模时需要太多数据。**
- 解决方法：状态绑定。
- **Young, S. . "Tree-based state tying for high accuracy acoustic modeling." Proc. ARPA Human Language Technology Workshop 1994.**

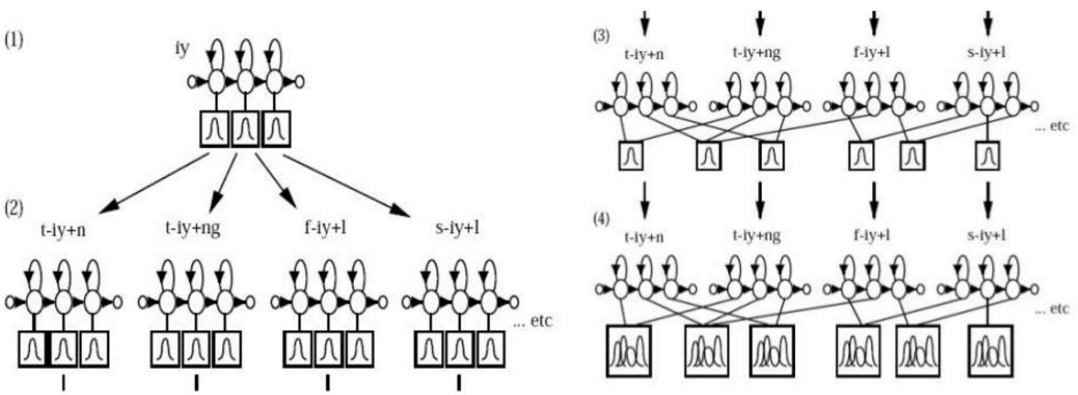



Figure: The tied-state HMM system build-procedure

**1）用BW算法训练Monophone**
**2）用Monophone初始化Triphone，用BW算法训练。**
**3）中心音素相同的三音素被聚类，一个典型的状态被选择出，同类的状态绑定该状态。**
**4） 增加GMM混合分量数。**

# 语音识别算法

- 对中文而言，音节、半音节、音素的数量都是固定不变的。
- 半音节（**Subsyllable**）是主要建模基元
- 汉字是单音节的，是声韵结构的，这种独特而规则的结构，使对音节、以及词条的表示变得比较规则和统一
- 使用半音节作为基元，其上下文有特殊的约束规则，**Triphone**的数目比较少，不是基元数的立方。

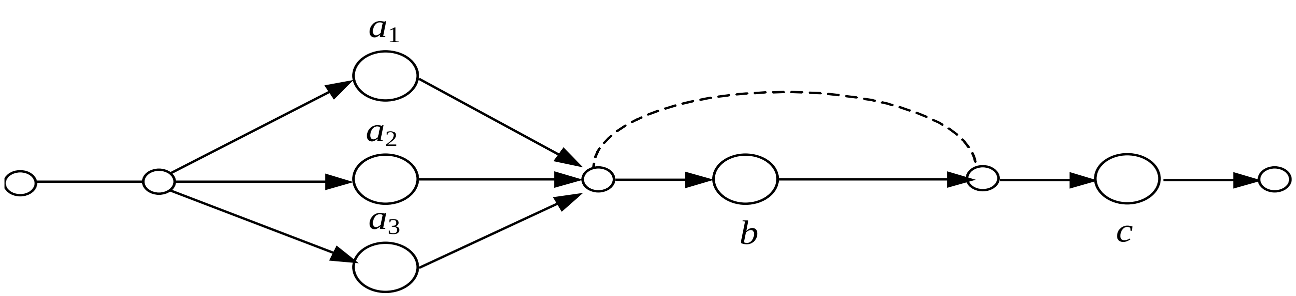
# 语音识别算法

（2）如何得到词模型

- 在词层用一部词典（**Dictionary**）来规定词表中每一个词是用哪些子词单元以何种方式构筑而成的。
- 最简单实用的方案是每个词用若干子词单元串接而成。
- 然而每个词的发音可能有多种变化方式
  - ❖ 替换：即词中的某个子词可能被用其它相似而略有差异的子词单元所替换 。
  - ❖ 插入和删除：词中有时增加了一个不是本词成分的子词单元，有时又将本词成分中的某个子词删除 。

# 语音识别算法

- 解决方案
  - ❖方案1：每一个词建立多套子词单元串接规则 。
  - ❖方案2：将子词单元构成词的规则用一个网络图来描述 。



# 语音识别算法

**(3) 基于子词单元的HMM训练**

- 子词单元的HMM一般采用从左到右的结构，状态数固定为2到4个。
- 在语音段中，子词太短，无法精确标出语音的边界。
- 已知句子内容，因此可以将子词模型串接成句子。
- 用分段*K*均值算法进行多次迭代，对各子词模型进行重估。最终它会自动收敛于一个最佳模型估计，同时达到合理的子词分段

# 语音识别算法

分段*K*均值算法

- 初始化：将每个训练语句线性分割成子词单元，将每个子词单元线性分割成状态，即假定在一个语句中，子词单元及其内部的状态驻留时间是均匀的；
- 聚类：对每个给定子词单元的每一个状态，其在所有训练语句段中特征矢量用*K*均值算法聚类；
- 参数估计：根据聚类的结果计算均值、各维方差和混合权值系数；
- 分段：根据上一步得到的新的子词单元模型，通过Viterbi算法对所有训练语句再分成子词单元和状态，重新迭代聚类和参数估计，直到收敛。

# 语音识别算法

- 这一过程也被称之为“强制对齐”**(Force Allignment)**
- 对齐过程合理分割了语音段，并得到子词边界
- 也初步估计出了每个子词的**HMM**参数。
- 以此参数为初值，采用**BW**算法迭代若干次即完成子词训练。

# 语音识别算法

- 语言模型

- 从一个词表中任意选择若干词所构成的序列不一定能构成自然语言中的句子，只有合乎句法者才能算是句子。
- 语言模型分为基于文法的语言模型和基于统计的语言模型。
- 在大词汇量的语音识别系统中，统计语言模型被广泛的应用。

# 语音识别算法

- 统计语言模型的基本原理是，采用大量的文本资料，统计各个词的出现概率以及其相互关联的条件概率。
  - 理想情况：对词串 $W = w_1, w_2, ..., w_Q$ ，

$$P(W) = P(w_1, w_2, ..., w_Q)$$

$$= P(w_1)P(w_2 | w_1)P(w_3 | w_1 w_2)...P(w_Q | w_1 w_2 ... w_{Q-1})$$

  - 一般采用简化模型

# 语音识别算法

(1) N元文法模型 :条件概率计算时，只考虑与前 $N-1$个词相关 。

$$P_N(W)=\prod_{i=1}^{Q}P(w_i\,|\,w_{i-1}w_{i-2}\ldots w_{i-N+1})$$

❖通常系统中采用的也只是二元和三元文法。

❖*N*元文法统计语言模型的建立，一般是通过相对频率计数得到:

$$\hat{P}(w_i\,|\,w_{i-1}w_{i-2}\ldots w_{i-N+1})=\frac{F(w_iw_{i-1}w_{i-2}\ldots w_{i-N+1})}{F(w_{i-1}w_{i-2}\ldots w_{i-N+1})}$$

*F*(*W*)是指词串*W*在训练数据中出现的次数

# 语音识别算法

❖训练数据稀疏时的解决方法：为了避免出现 $F(W)=0$ 或接近于零的情况，可以用三元、二元和一元相对频率做插值 。

$$\hat{P}(w_3 | w_1 w_2) = p_1 \frac{F(w_1 w_2 w_3)}{F(w_1 w_2)} + p_2 \frac{F(w_2 w_3)}{F(w_1)} + p_3 \frac{F(w_1)}{\sum_i F(w_i)}$$

其中 $\sum_{i=1}^{3} p_i = 1$ ，$\sum_i F(w_i)$是训练语料的总词数。

# 语音识别算法

（**4**）**$N$元词类文法模型**：每个词$w_t$只与其所在类$c_t$有关，而与前一时间的词所在类$c_{t\text{-}1}$中的成员无关。

$$P(W)=\sum_{C\in C^m}\prod_{t=1}^{Q}P(c_t\,|\,c_1c_2\ldots c_{t-N+1})P(w_t\,|\,c_t)$$

# 语音识别算法

- 如何识别
- 用viterbi算法做最优路径搜索，找出概率最大的状态序列：
  - HMM转移概率控制子词内状态间的转移；
  - 词典控制词内子词间的状态转移；
  - 语言模型控制词间的状态转移。
- 状态空间太大，例如：词表中有十万个词，平均每个词有10状态，则计算复杂度为：

$$O(100万^2 \times T)$$

# 语音识别算法

- 在此状态空间上，计算量非常大，无法保证识别算法的实时性
- 解决方案：**Viterbi Beam 搜索算法**
- 核心思想是剪枝，每个时刻仅保留少量状态可以向下个时刻扩展路径。例如，若仅保留**100**个状态，时间复杂度为：

$$O(100\text{万} \times 100 \times T)$$

- 剪枝的依据是当前局部路径的概率
- 贪心算法

# 语音识别算法

## Viterbi Beam搜索算法

- 初始化

  初始化活动路径（最高层）
- 递推

  For $m$=1 到$M$

  For 每一层次（指各个层次的语言和声学模型）

  For HMM的每个活动状态

  把每个活动路径向后扩展一帧至所有可以到达的状态

  执行Viterbi计算

  裁剪路径

  End｛活动状态｝

  End｛每一层次｝

  End｛观察矢量序列｝
- 终止：选择最可能的路径