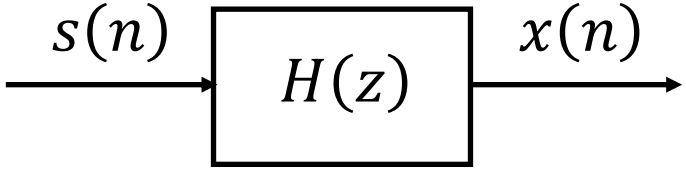
第三章　语音识别

倒谱分析与线性预测编码方法的回顾

- 倒谱分析与线性预测编码方法的主要区别？
  - 线性预测编码：直接利用了语音生成的源-滤波器模型，预测了自回归模型的线性预测系数。

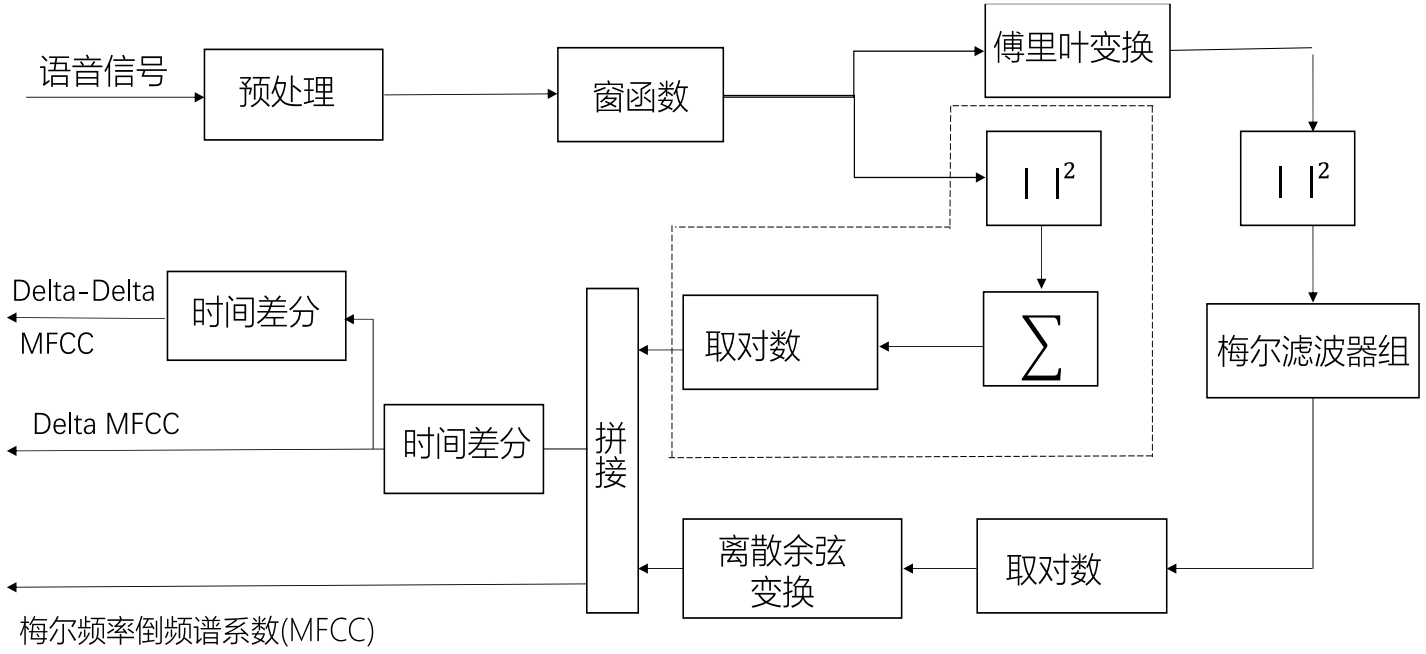


语音信号产生的模型化

自回归模型的传递函数：$H(z) = \dfrac{G}{1 - \sum_{i=1}^{p} a_i z^{-i}}$

  - 倒谱分析：间接利用了语音生成的源-滤波器模型，通过信号处理的时频分析方法分离声源信号与滤波器（声道）信号。

MFCC的计算流程图（回顾）



- C0：语音的平均强度，在语音识别中一般不直接使用。
- 随着阶数增加，MFCC表示频谱的更多细节。但是无论是MFCC还是LPC系数都不能直接展示共振峰的频谱包络的细节信息。

语音识别是语音信息处理重要的研究方向

- 基于模板匹配的语音识别技术是20世纪70年代的主流技术。

- 上世纪80年代以来，语音识别算法逐步从模板匹配算法进展到基于统计模型的算法。2010年之前，基于GMM-HMM是语音识别的经典声学模型。2010年前后，深度学习算法在语音识别领域取得巨大成功，DNN-HMM逐步替代了GMM-HMM。2015年前后，端到端模型受到越来越多的关注。

- 除了声学模型，本章也会介绍语音识别的另外两个主要组成部分：语言模型和解码算法。

第三章　语音识别

- 3.1 语音识别概述
- 3.2 声学模型
- 3.3 语言模型
- 3.4 语音识别解码算法
- 3.5 语音识别技术的展望

第三章　语音识别

- 3.1 语音识别概述
  - 3.1.1 语音识别的基本概念（基本概念）
  - 3.1.2 语音识别方法的回顾（基本概念）
  - 3.1.3 语音识别的典型应用（了解）
- 3.2 声学模型
- 3.3 语言模型
- 3.4 语音识别解码算法
- 3.5 语音识别技术的展望

什么是语音识别（ASR）？

- 定义：
  - 自动语音识别（Automatic Speech Recognition）：机器通过识别把人类的语音信号转变为相应的文本的技术。
  - 语音信息处理的最核心方法

语音 → ASR 系统 → “How are you?”

文本

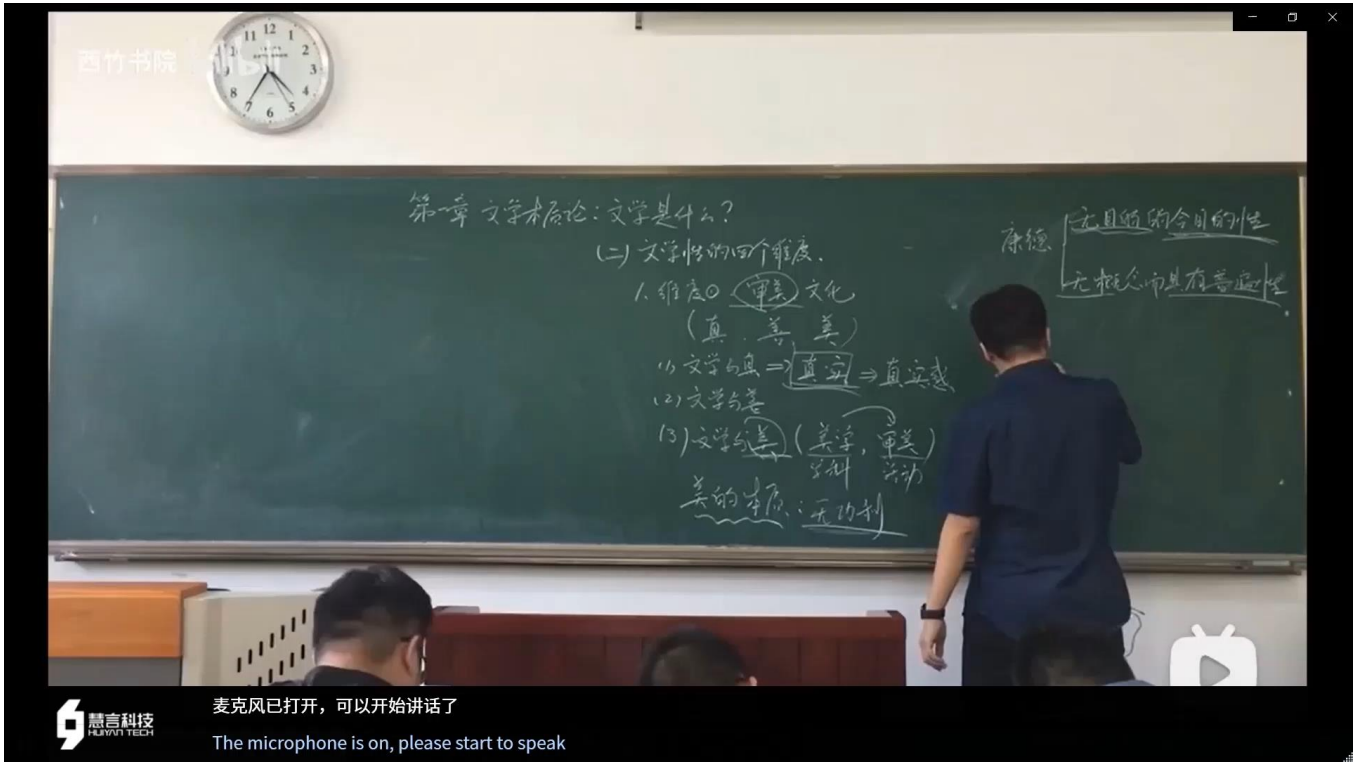
- 语音识别的结果作为
  - 最终输出（转写：Dictation）
  - 自然语言处理（对话系统、机器翻译）的输入

语音识别的用途

- 语音识别的结果作为
  - 最终输出（转写：Dictation）

语音识别的用途（续）

- 语音识别的结果作为
  - 最终输出（转写：Dictation）
  - 自然语言处理（对话系统、机器翻译）的输入



语音识别的分类

- 按发音方式分
  - 孤立词（isolated words）识别

    “打开空调”
  - 连接词（connected words）识别

    “现在 我 简单 介绍 语音 识别 的 基本 概念。”
    连续数字识别
  - 连续语音（continuous speech）识别

    “现在我简单介绍语音识别的基本概念。”

语音识别的分类

- 按词汇表（Vocabulary）的大小分
  - 小词汇表识别系统

    包括10~100个词条

  - 中词汇表识别系统

    包括100~1000个词条

  - 大词汇表识别系统

    包括1000个以上的词条

语音识别的分类

- 按说话人分
  - 特定说话人（speaker-dependent）语音识别

    只能识别固定某个人的语音
  - 非特定说话人（speaker-independent）语音识别

    能识别任意人的语音

语音识别评价指标

- 单词错误率（Word Error Rate）

$$\text{Word Error Rate} = \frac{100\ (\#\text{Insertions} + \#\text{Substitutions} + \#\text{Deletions})}{\text{Total Word in Correct Transcript}}$$

Insertion：插入；Substitution：置换；Deletion：删除

Alignment example:

REF: portable **** PHONE UPSTAIRS last night so

HYP: portable FORM OF STORES last night so

Eval I S S

WER = 100 (1+2+0)/6 = 50%

第三章　语音识别

- 3.1 语音识别概述
  - 3.1.1 语音识别的基本概念
  - 3.1.2 语音识别方法的回顾
  - 3.1.3 语音识别的典型应用
- 3.2 声学模型
- 3.3 语言模型
- 3.4 语音识别解码算法
- 3.5 语音识别技术的展望

语音识别主流方法的演化

2020年以后？

Before mid 70's	Mid 70's – mid 80's	After mid 80's
启发式 （Heuristic） Rule-based and declarative	模板匹配 （Template matching） Deterministic and data-driven	统计模型 （Mathematical） Probabilistic and data-driven

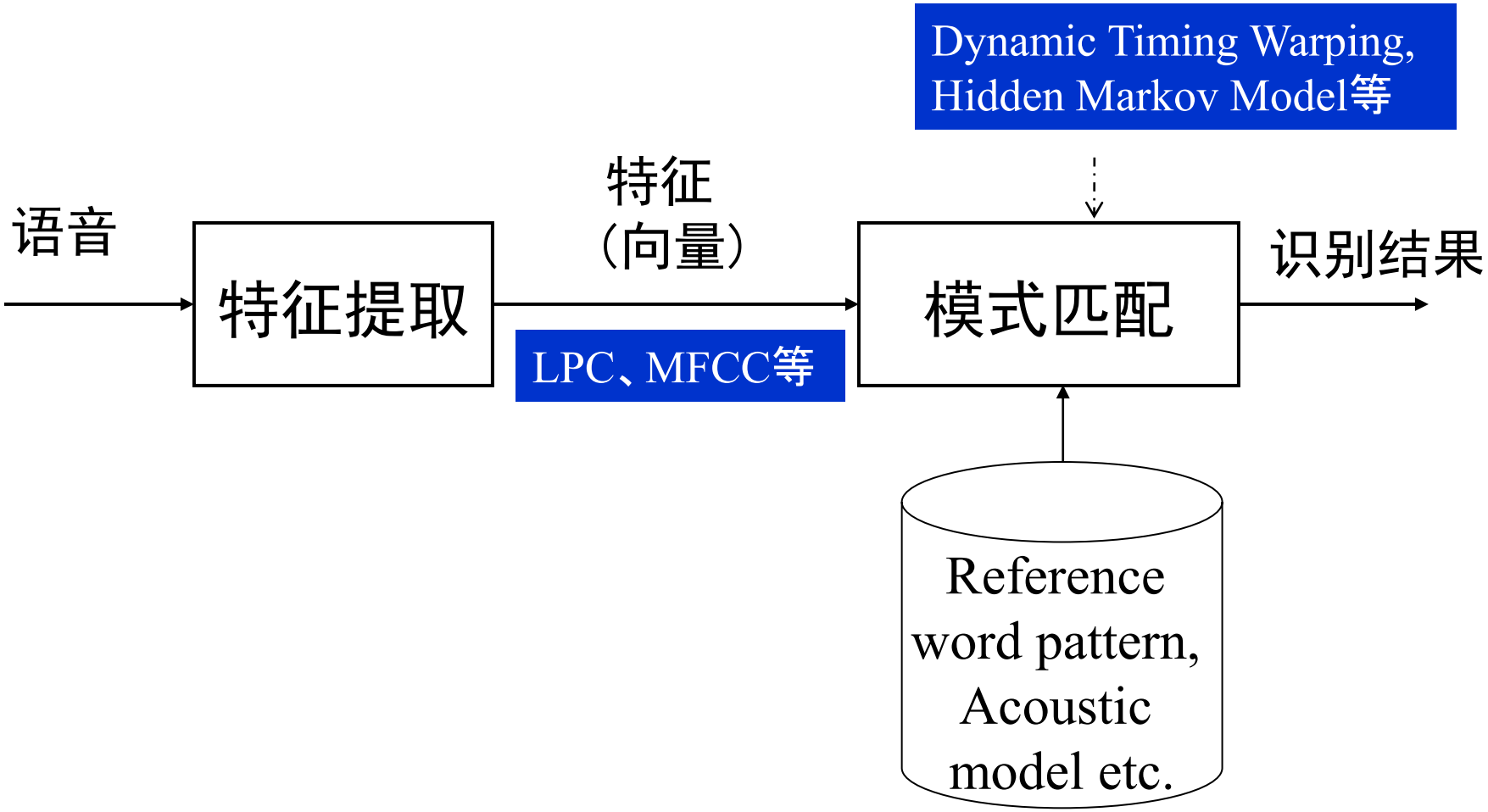
模式识别方法
（Pattern Recognition Approach）

基于规则的语音识别

- 利用规则来指导语音识别:
  - Phonetics（语音学），phonology（音韵学）
  - Syntax（语法）
  - Pragmatics（语用学）

- 缺点
  - 难以表达规则
  - 难以知道如何提升系统的性能

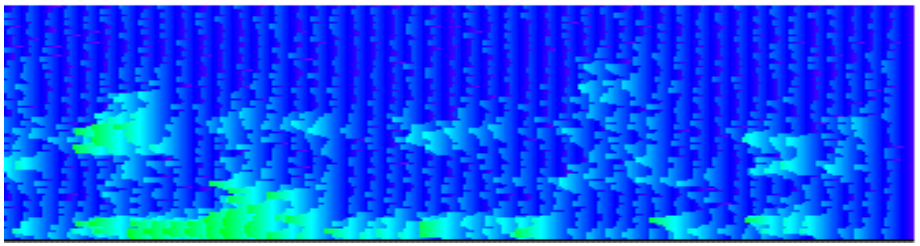
基于数据驱动的语音识别



基于模板匹配的语音识别

- 概念：将输入模板与参考模板比较，查找最相似的模板

输入模板



X x_1 x_2 x_3 … x_I

Different length

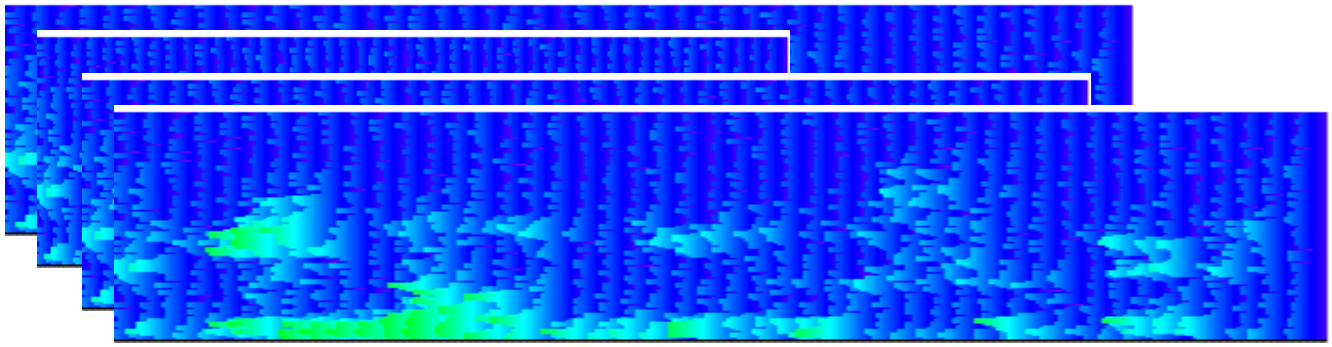
↓

How to compare?

How about linear alignment?

参考模板1

Y^1 y^1_1 y^1_2 y^1_3 … y^1_{J1}



参考模板N

Y^N y^N_1 y^N_2 y^N_3 … … y^N_{JN}

基于模板匹配的语音识别（续）

- 如何计算两个矢量序列X和Y之间的相似度?

$$X=\{x_1, x_2, x_3, \cdots, x_I\}$$

$$Y=\{y_1, y_2, y_3, \cdots, y_J\}$$

存在问题：

- 长度不同，$I \neq J$
- 对不准

- 经典解决方法：动态时间弯折/归正（DTW）

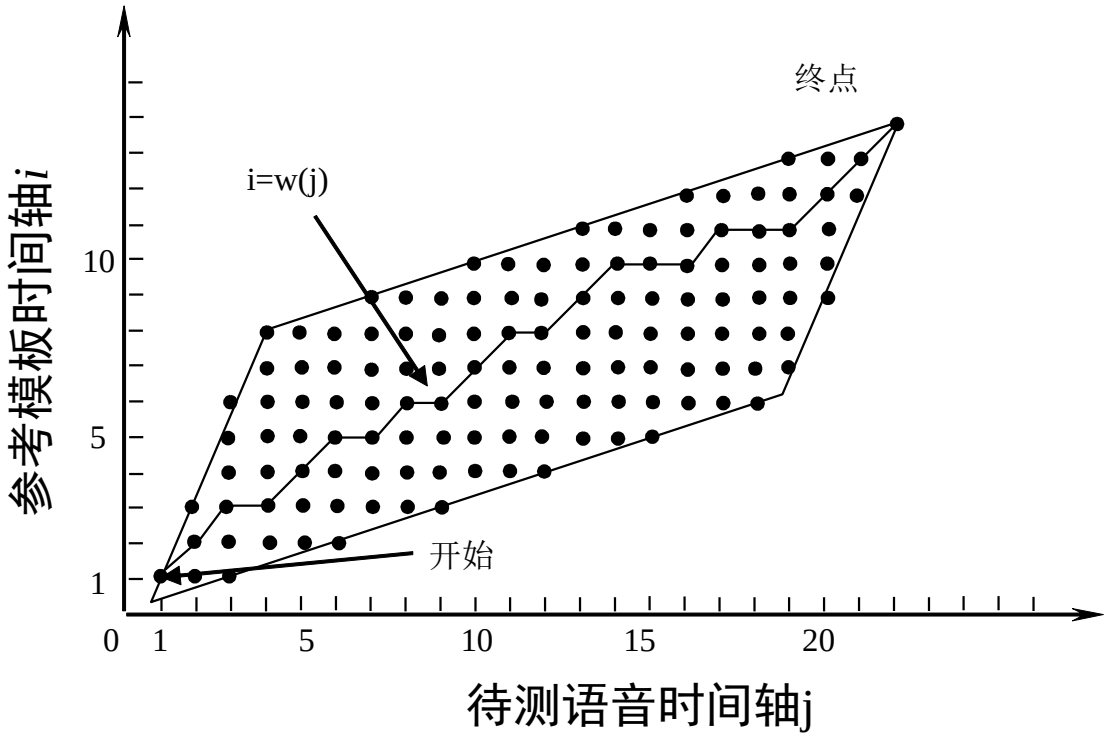
把时间归正和距离测度计算结合起来的非线性归正技术。

DTW是采用动态规划（Dynamic Programming，DP）技术，将一个复杂的全局问题转化为许多局部最优问题，逐步进行决策。

将表示两个语音段的矢量序列对准了再计算相似度。

动态时间归正的思路

- 如何对准？



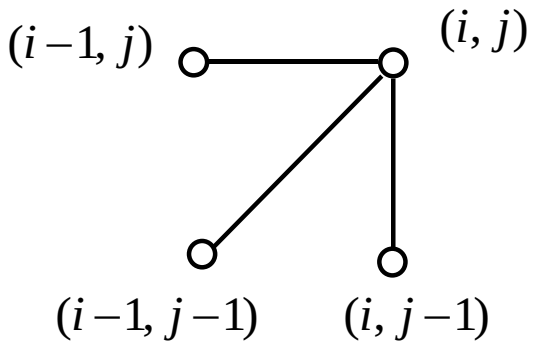
- 每一条从（1，1）到（*I*，*J*）路径都有一个累计距离称为路径的代价。
- 每一条路径都代表一种对齐情况。
- 代价最小的路径就是所求对准路径。

动态时间归正的思路（续）

- 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



受一步局部路径约束

- 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。

动态时间归正的思路（续）

定义一个代价函数 $g(i,j)$ 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：
$$\mathrm{g}(i,j)=\min_{(i',j')\to(i,j)}\{g(i',j')+d(\boldsymbol{x_i},\boldsymbol{y_j})W\}$$

则：
$$\mathrm{g}(i,j)=\min\begin{Bmatrix} g(i-1,j)+d(\boldsymbol{x_i},\boldsymbol{y_j})W(1) \\ g(i-1,j-1)+d(\boldsymbol{x_i},\boldsymbol{y_j})W(2) \\ g(i,j-1)+d(\boldsymbol{x_i},\boldsymbol{y_j})W(3) \end{Bmatrix}$$

依次类推，$\mathrm{g}(i-1,j)$、$g(i,j-1)$、$g(i-1,j-1)$
可由更低一层的代价函数计算得到。

动态时间归正的思路（续）

- 这样就可从

$$g(1,1)$$ 逐步向上搜索。

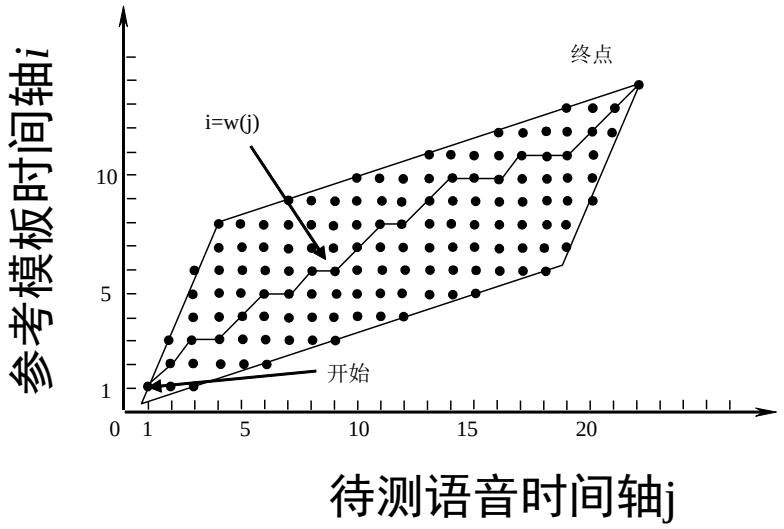
  - 加权系数的取值与局部路径有关

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

  - 定义回溯函数

$$p(i,j)$$

  - 平行四边形区域约束



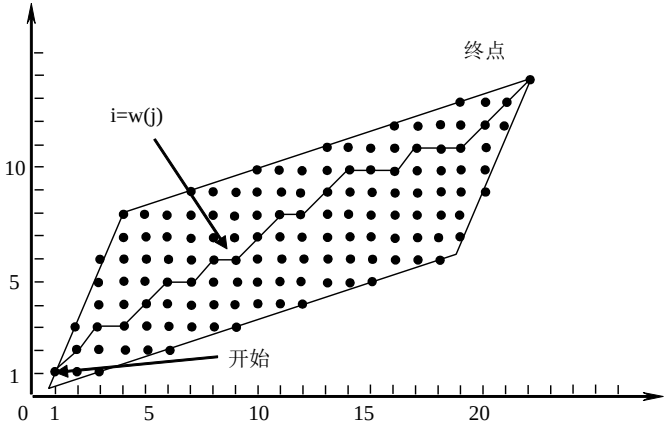
动态时间归正路径搜索算法

（1）初始化：

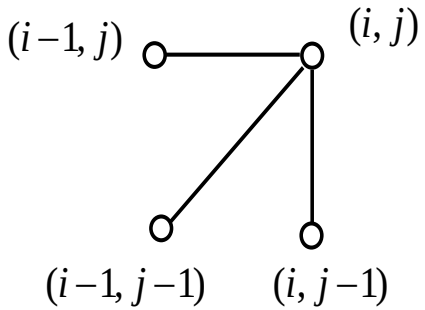
$$i = j = 1,\ g(1,\ 1) = 2d(\boldsymbol{x_i}, \boldsymbol{y_j})$$

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

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(*I*, *J*)，相邻两条边的斜率分别为2和1/2。



动态时间归正路径搜索算法（续）



受一步局部路径约束

(2) 递推求累计距离，并记录回溯信息：

$$g(i,j)=\min\begin{Bmatrix} g(i-1,j)+d(\boldsymbol{x_i},\boldsymbol{y_j})W(1) \\ g(i-1,j-1)+d(\boldsymbol{x_i},\boldsymbol{y_j})W(2) \\ g(i,j-1)+d(\boldsymbol{x_i},\boldsymbol{y_j})W(3) \end{Bmatrix}$$

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

一般取距离加权值为，$W(1)=W(3)=1$，　$W(2)=2$

并将(i,j)点的回溯信息记录在$p(i,j)$中。

动态时间归正路径搜索算法（续）

（3）回溯求出所有的匹配点对：

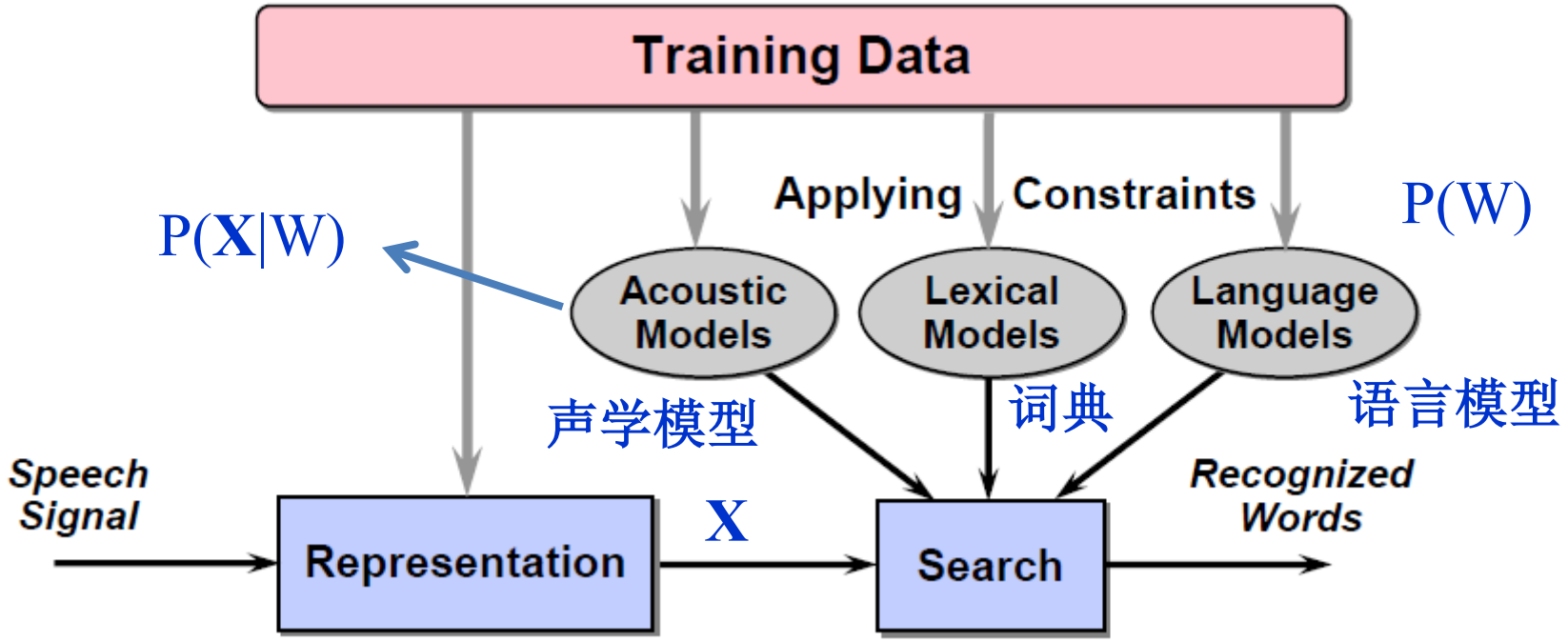
根据每步的上一步最佳局部路径$p(i,j)$，由匹配点(I,J)对向前回溯一直到$(1,1)$。这个回溯过程对于求平均模板或聚类中心来讲是必不可少的，但在识别过程往往不必进行。

- 对所求得的 $g(I,J)$ 还需用 $\sum W$ 作为分母来归正

动态时间归正算法缺点

- 很难区分非常类似的模板（硬匹配）
- 当输入和模板之间的声学条件不同时，语音识别的性能急剧下降（缺少多样性）
- 没有充分利用语音信号的时序动态信息
- 不利于进行大词汇表连续语音识别（LVCSR）
- 需要新的技术来消除以上问题
  - 更易于训练不同的模板
  - 更加鲁棒的匹配技术
  - 更适合于进行LVCSR
- 解决方案：基于统计模型的语音识别

基于统计模型的语音识别框架（重点掌握）



语音识别-噪音通道模型

- 噪音通道（Noisy Channel）模型
  - 源句子（source sentence）经过噪音通道（noisy channel）会生成噪音句子（noisy sentence）。我们通过对噪音句子的解码得到猜测的句子（guessed sentence）
  - 搜索所有可能的句子
  - 找出最优解

语音识别-噪音通道模型（掌握）

- 噪音通道（Noisy Channel）模型
  - 本质是求argmax P(W|**X**)：
    给出声学特征**X**，查找最匹配的单词序列W。
  - 应用贝叶斯法则（Bayes' rule）：

声学模型　语言模型

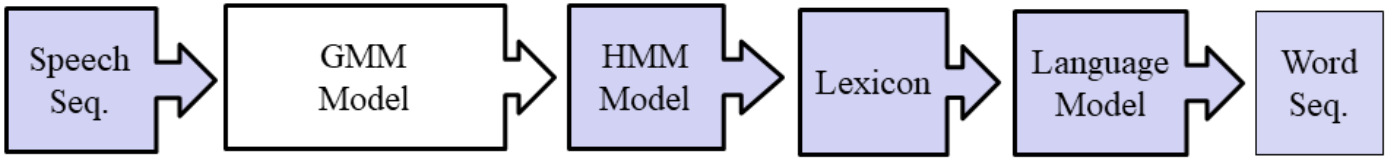
$$\underset{W}{\operatorname{argmax}} P(W|\boldsymbol{X}) = \underset{W}{\operatorname{argmax}} \frac{P(\boldsymbol{X}|W)P(W)}{P(\boldsymbol{X})}$$

因P(**X**)与W无关，可以去掉分母P(**X**)。

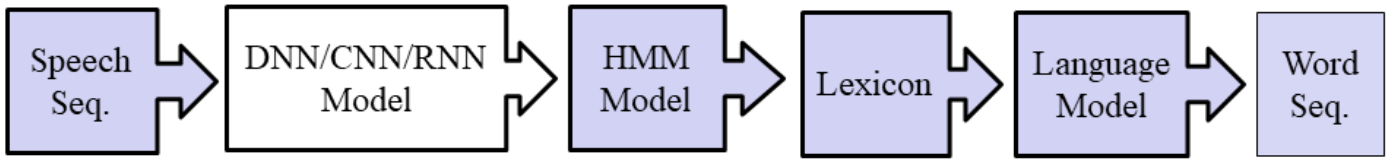
语音识别–声学模型

从GMM-HMM到端到端模型

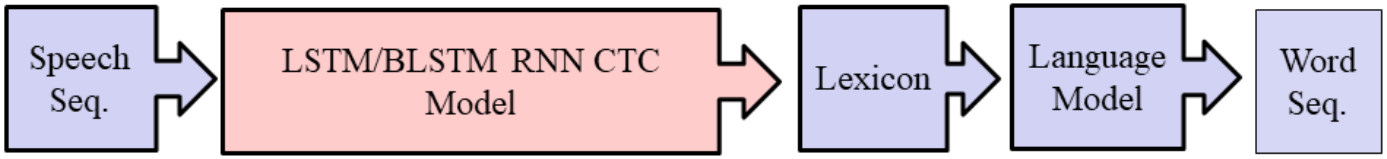
1990s-2009: The GMM-HMM hybrid system. (CU-HTK)



2009-Now: The DNN-HMM hybrid system. (JHU-Kaldi, MS-CNTK)

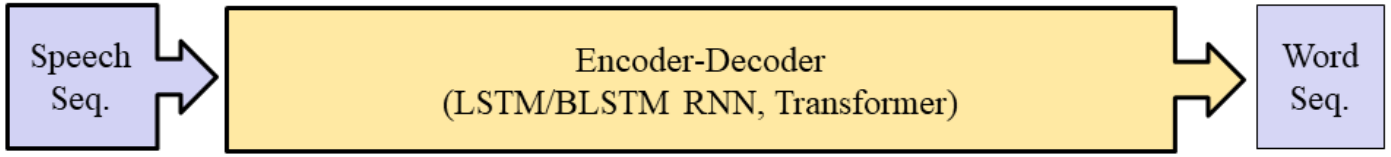


2014: The CTC End-to-End system. (CMU-EESEN, Baidu-WarpCTC)



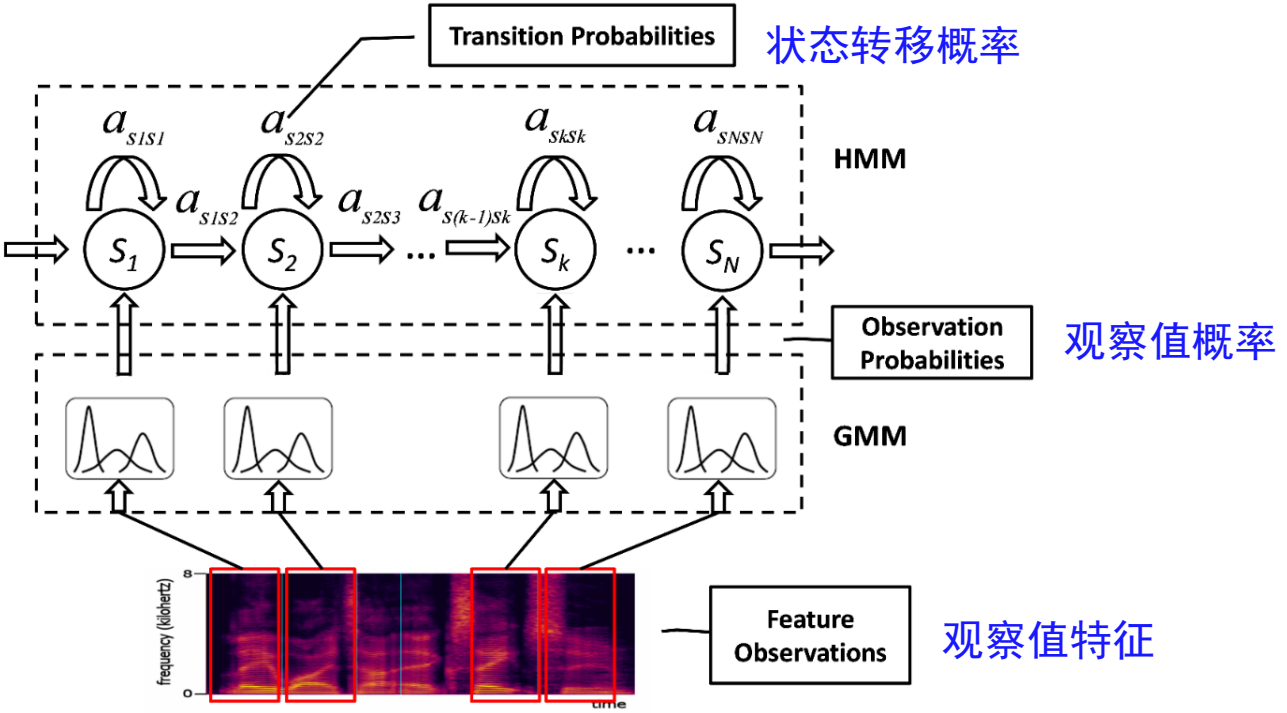
2016: The Encoder-Decoder End-to-End system.

(Google-LAS/Transformer, facebook-wav2letter, JHU-ESPNet)



语音识别-声学模型

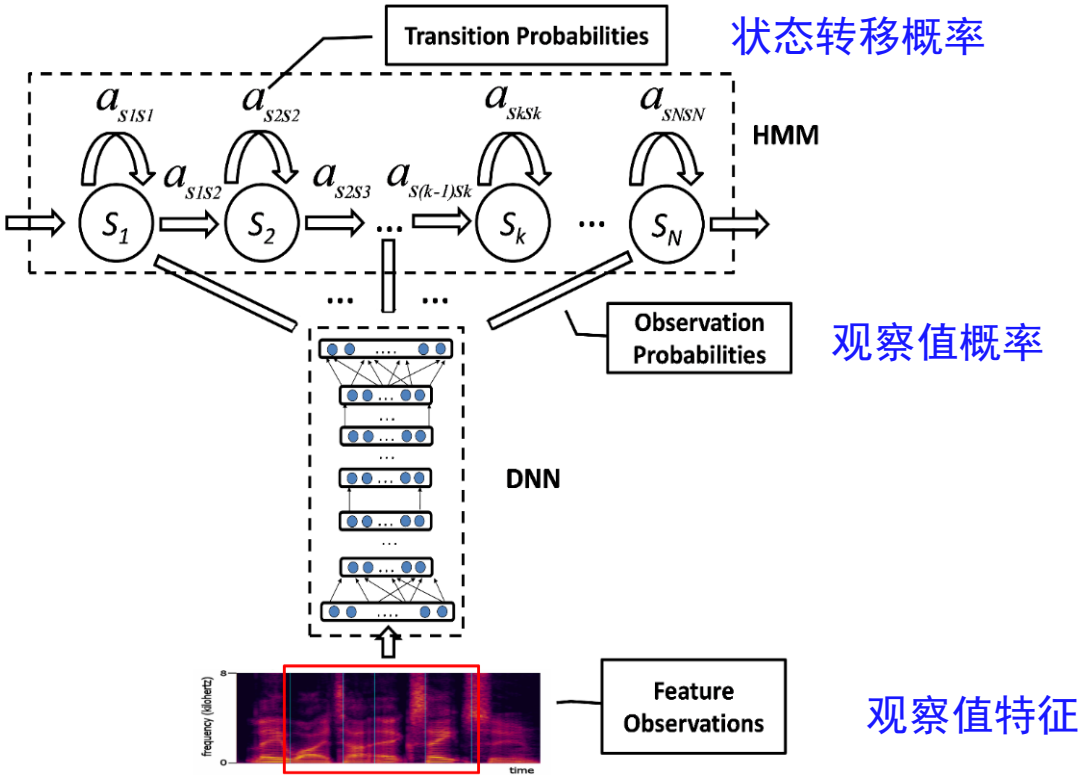
- GMM-HMM（高斯混合模型-隐马尔可夫模型）



- L. R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. Proc. IEEE, 77(2):257–286, 1989.

语音识别-声学模型

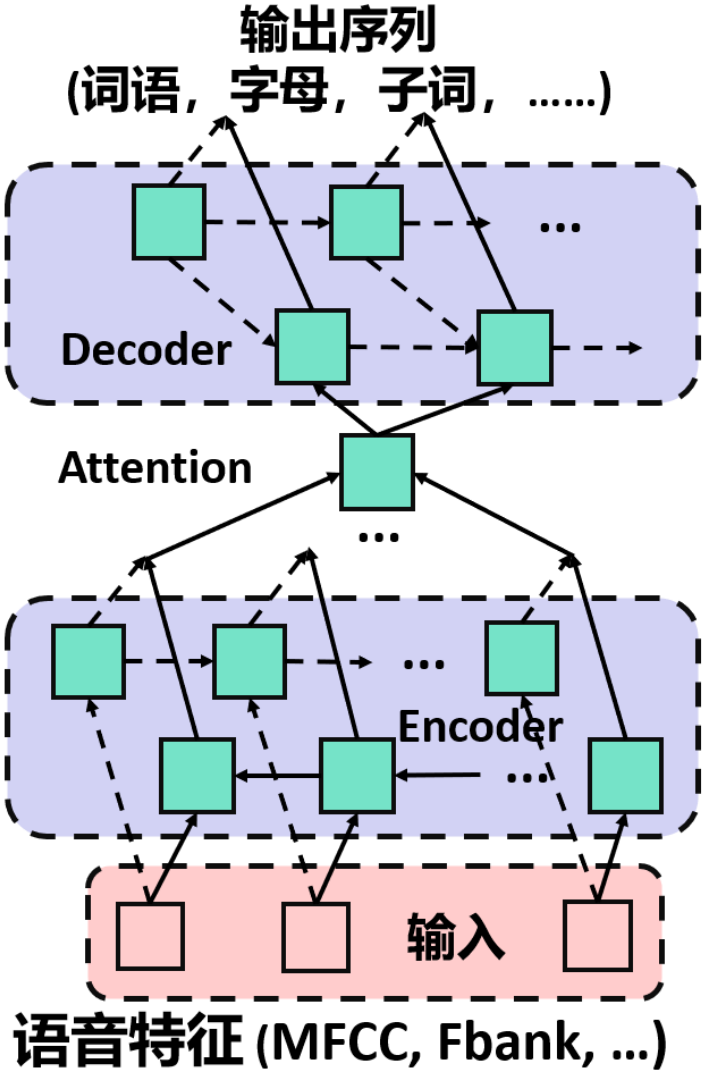
- DNN-HMM（深度神经网络-隐马尔可夫模型）



语音识别-声学模型

- End-to-end Model
  （端到端模型）

不需要语言模型和词典



语音识别-语言模型

- 语言模型：根据语言客观事实而进行的语言抽象数学建模，用以计算文字序列的概率

- 分类
  - 基于文法的语言模型
  - 基于统计的语言模型（N-gram语言模型，神经语言模型）

- 主要应用
  - 语音识别
  - 文本生成
  - 机器翻译
  - 阅读理解
  - … …

“语音识别、机器翻译” 和“语音识别及其翻译”相比，哪个更常见？

“语音识别与机器翻译” 比“语音识别与及其翻译”更像个正确句子。

语音识别-语言模型

- N-gram（N元文法）语言模型
  - 单词序列

$$w_1^n = w_1 \ldots w_n$$

  - 概率的链式法则

$$P(w_1^n) = P(w_1)P(w_2 \mid w_1)P(w_3 \mid w_1^2)\ldots P(w_n \mid w_1^{n-1}) = \prod_{k=1}^{n} P(w_k \mid w_1^{k-1})$$

  - N-gram近似　　假设词w_k出现的概率只与前N-1个单词有关

$$P(w_1^n) = \prod_{k=1}^{n} P(w_k \mid w_{k-N+1}^{k-1})$$

语音识别–解码算法

$$P(W|\boldsymbol{X}) = \underset{\{w_1^N\}\{t_1^N\}}{\operatorname{argmax}}\{\sum_{n=1}^{N} \log(P_{acoust}(\mathbf{x}_{t_{n-1}+1}^{t_n}|w_n)) + \lambda \cdot \sum_{n=1}^{N} \log(P_{lang}(w_n|w_{n-1}) + \delta)\}$$

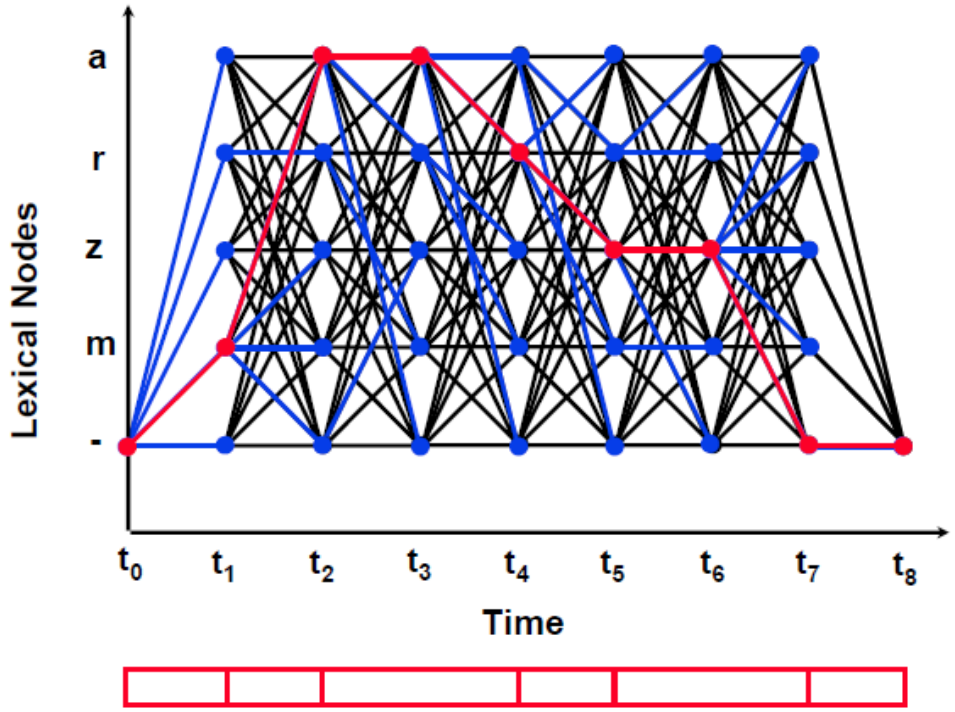
$P_{acoust}(\mathbf{x}_{t_{n-1}+1}^{t_n}|w_n)$：声学模型的似然

$P_{lang}(w_n|w_{n-1})$：语言模型的似然

λ：权重；　　δ：插入的惩罚

语音识别-解码算法

- Viterbi搜索算法



- WFST（加权有限状态转换器）搜索算法

第三章　语音识别

- 3.1 语音识别概述
  - 3.1.1 语音识别的基本概念
  - 3.1.2 语音识别方法的回顾
  - 3.1.3 语音识别的典型应用
- 3.2 声学模型
- 3.3 语言模型
- 3.4 语音识别解码算法
- 3.5 语音识别技术的展望

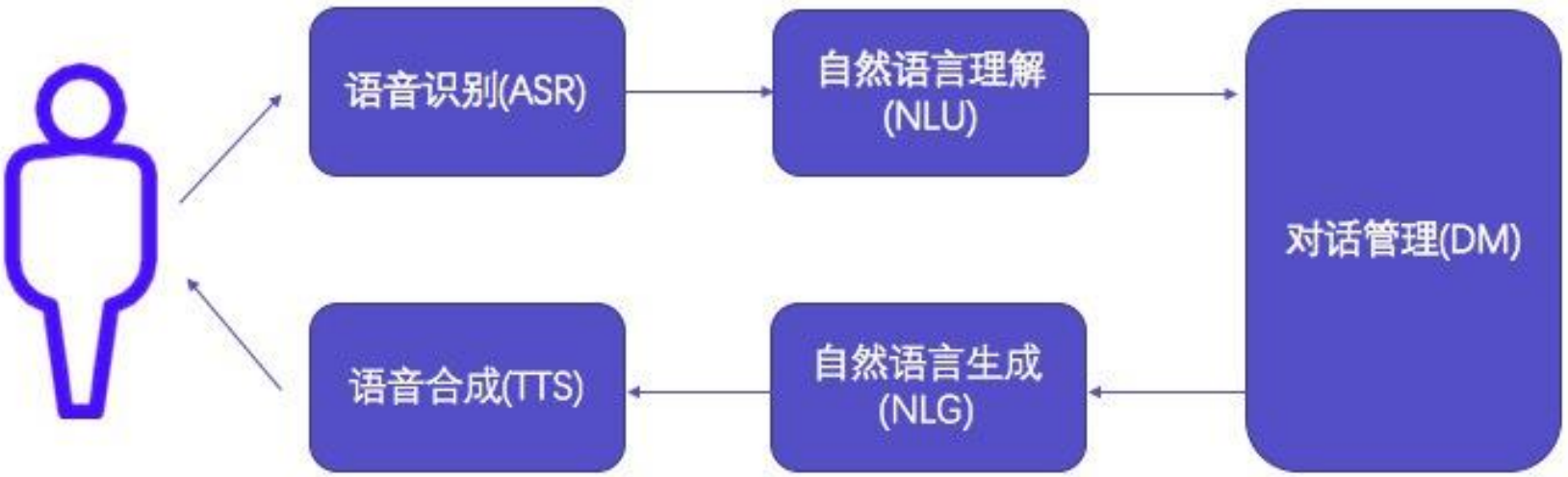
语音识别的典型应用

- 语音转写
- 语音对话
- 语音翻译
- ……

语音识别的典型应用–语音转写

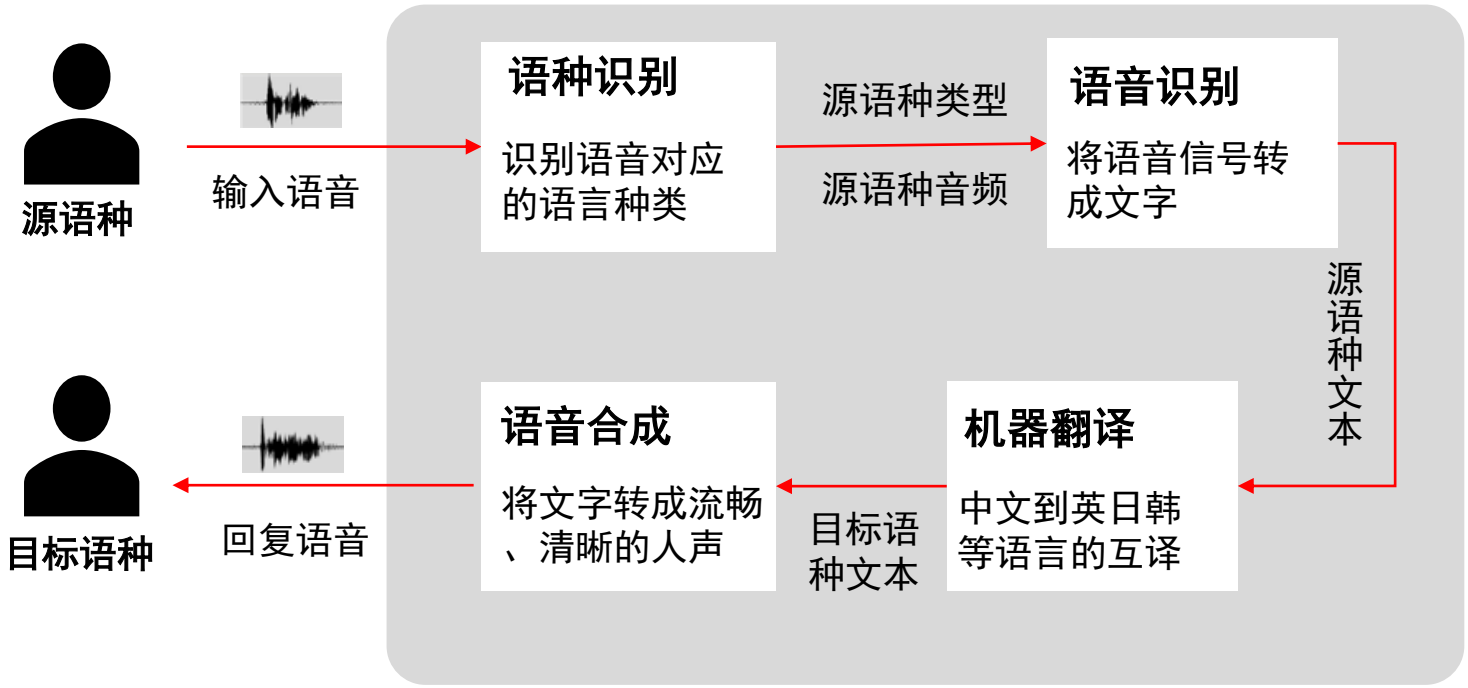
- 语音转写
  - 语音转写系统
  - 语音输入法

语音识别的典型应用-语音对话



- **语音识别：**完成语音到文本的转换，将用户说话的声音转化为语音。
- **自然语言理解：**完成对文本的语义解析，提取关键信息，进行意图识别与实体识别。
- **对话管理：**负责对话状态维护、数据库查询、上下文管理等。
- **自然语言生成：**生成相应的自然语言文本。
- **语音合成：**将生成的文本转换为语音。

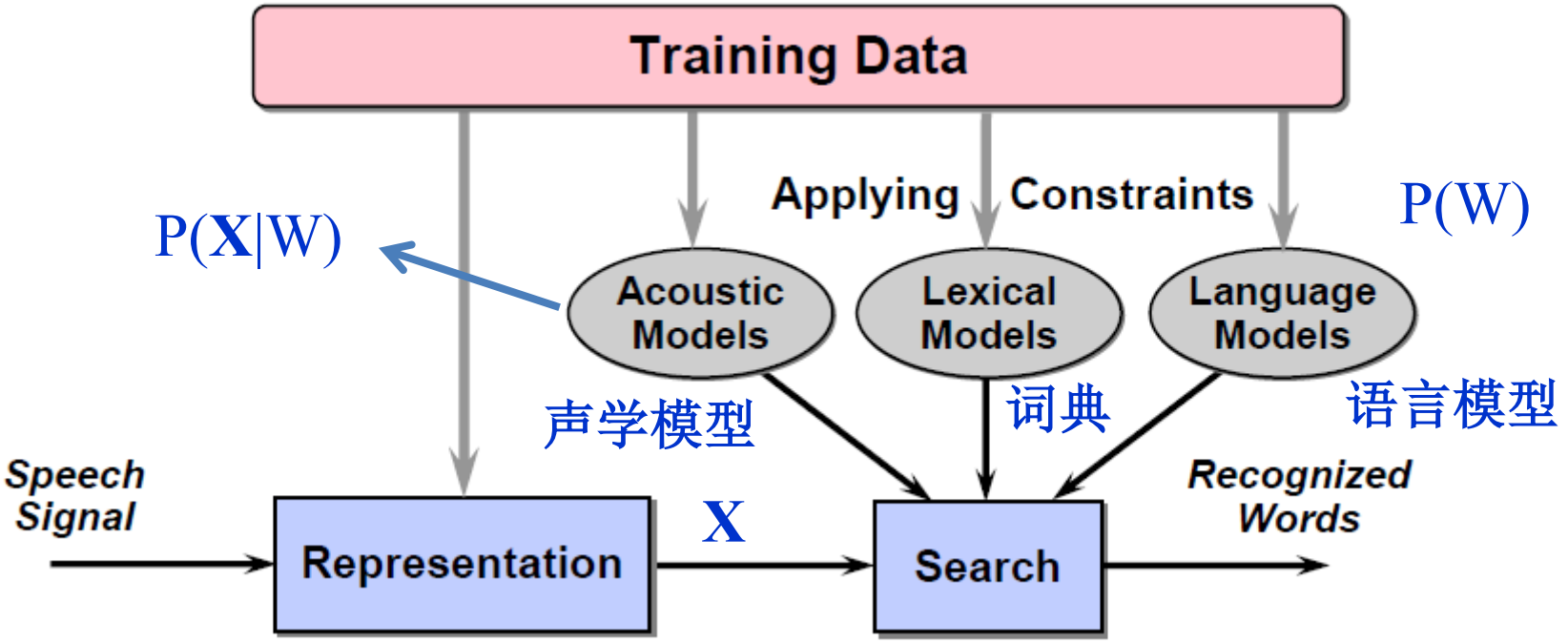
语音识别的典型应用–语音翻译



第三章　语音识别

- 3.1 语音识别概述
- 3.2 声学模型
  - 3.2.1 隐马尔可夫模型**（掌握、部分重点）**
  - 3.2.2 基于GMM-HMM的语音识别技术**（掌握、基本概念、了解）**
  - 3.2.3 DNN-HMM **（基本概念、了解）**
- 3.3 语言模型
- 3.4 语音识别解码算法
- 3.5 语音识别技术的展望

基于统计模型的语音识别框架回顾



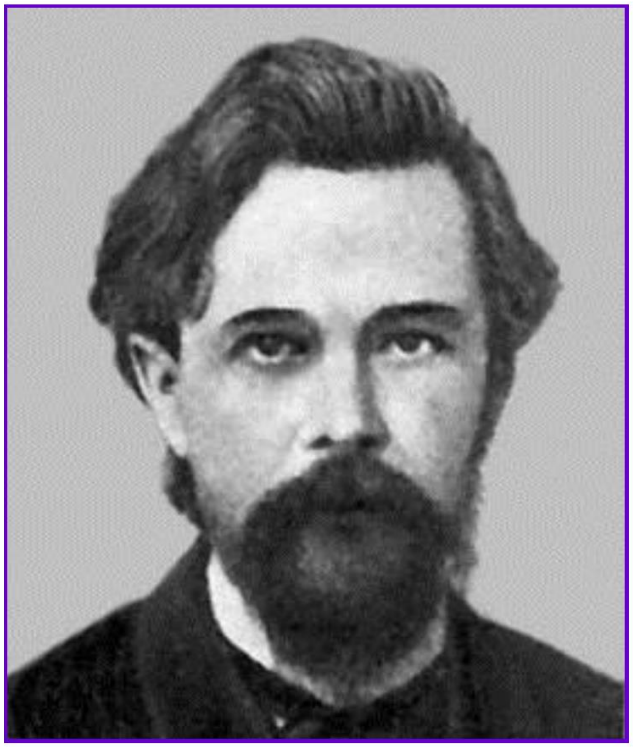
什么是声学模型？　　计算发音相似度的模型

第三章　语音识别

- 3.1 语音识别概述
- 3.2 声学模型
  - 3.2.1 隐马尔可夫模型
  - 3.2.2 基于GMM-HMM的语音识别技术
  - 3.2.3 DNN-HMM
- 3.3 语言模型
- 3.4 语音识别解码算法
- 3.5 语音识别技术的展望

马尔可夫（了解）

马尔可夫(Andrei Andreyevich Markov)(1856.6.14～1922.7.20)，前苏联数学家。切比雪夫(1821年5月16日～1894年12月8日)的学生。在概率论、数论、函数逼近论和微分方程等方面卓有成就。他提出了用数学分析方法研究自然过程的一般图式—马尔可夫链，并开创了随机过程(马尔可夫过程)的研究。



马尔可夫过程（了解）

- 马尔可夫过程为状态空间中经过从一个状态到另一个状态的转换的随机过程。该过程要求具备“无记忆”的性质，即下一状态的条件概率分布只能由当前状态决定，在时间序列中它前面的事件均与之无关。

- 马尔可夫过程具有马尔可夫性质（无后效性），当一个随机过程在给定现在状态及所有过去状态情况下，其未来状态的条件概率分布仅依赖于当前状态；换句话说，在给定现在状态时，它与过去状态（即该过程的历史路径）是条件独立的。

马尔可夫链（基本概念）

- 马尔可夫链：一种离散状态的马尔可夫过程（序列），为状态空间中经过从一个状态到另一个状态的转换的随机过程。具有如下性质：
  - 状态空间是有限的或可列；
  - 是一种离散时间随机过程；
  - 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。

马尔可夫链的定义（基本概念）

定义　设随机过程的状态空间为：$q_t \in \{s_j, j = 1,2,\dots,N\}$，

系统在时间t的状态只与其在时间$t-1$的状态相关，满足以下关系

$$P(q_t = s_j | q_{t-1} = s_i, q_{t-2} = s_k \dots)$$
$$= P(q_t = s_j | q_{t-1} = s_i), \qquad t = 1,2,\dots T$$

则称该随机过程为离散时间、离散状态的一阶马尔可夫过程，或简称为一阶马尔可夫链。

齐次马尔可夫链（了解）

定义 设 $\{\boldsymbol{Q} = q_1, q_2, \ldots, q_T\}$ 是马尔可夫链，记

$$P(q_t = s_j | q_{t-1} = s_i) = a_{ij}(t), \qquad i, j = 1,2, \ldots, N$$

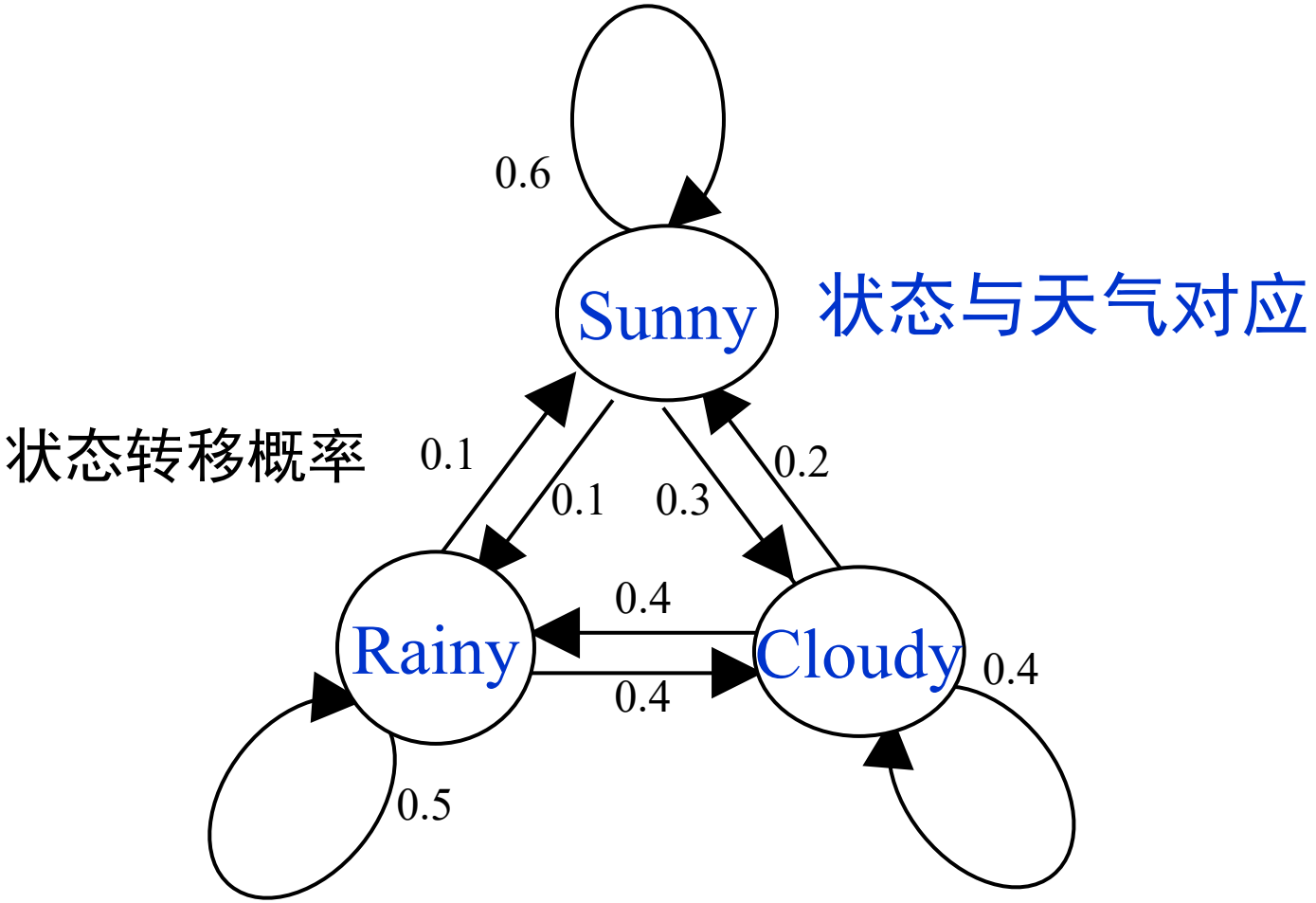
$a_{i,j}(t)$表示 t时刻从状态i转移到状态j的概率。如果这些转移概率与时间t无关，则得到齐次马尔可夫链。

状态转移概率$a_{i,j}$必须满足下列条件：

$$a_{ij} \geq 0 \ \forall i, j;$$

$$\sum_{j=1}^{N} a_{i,j} = 1 \ \forall i$$

马尔可夫链的例子（了解）



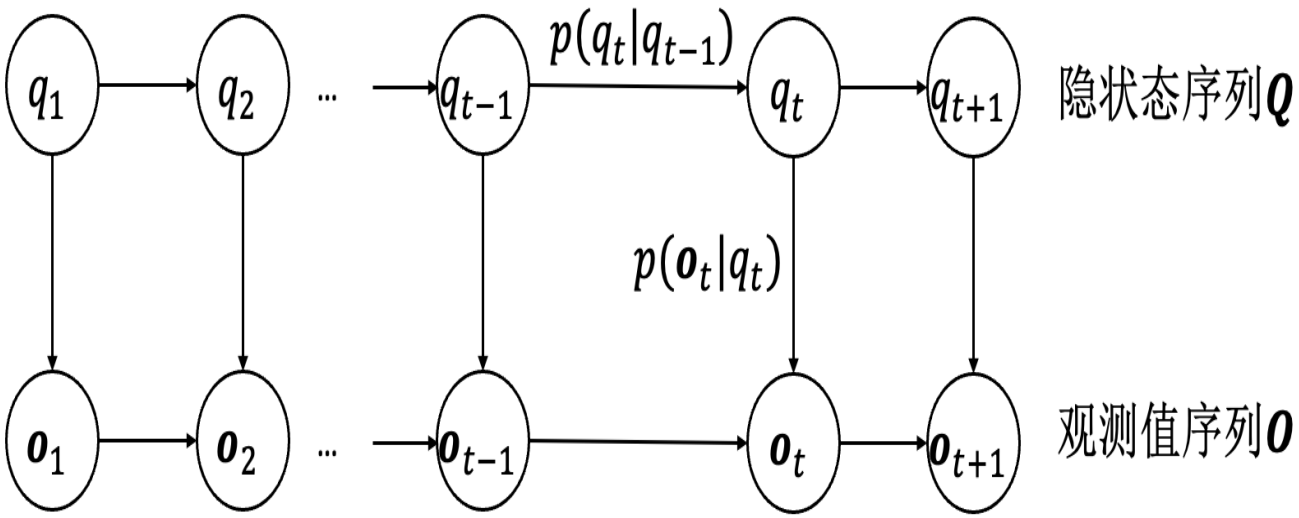
马尔可夫链的问题与隐马尔可夫模型（HMM）（基本概念）

- 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。

- HMM的定义：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（不可见）；另一个随机过程描述状态和观察值之间的统计对应关系（可见）。由于状态是不可见的，因此称之为“隐” Markov模型（HMM）。
- HMM创建于20世纪70年代，是美国数学家鲍姆(Leonard E. Baum)等人提出来的。

- 每个状态上有观察值概率分布。

HMM的示意图（基本概念）



用一个观测的概率分布与每一个状态对应，而不是一个确定的观测值或事件，这就在马尔可夫序列的状态中引入了随机性，使其状态并不能被直接观测。

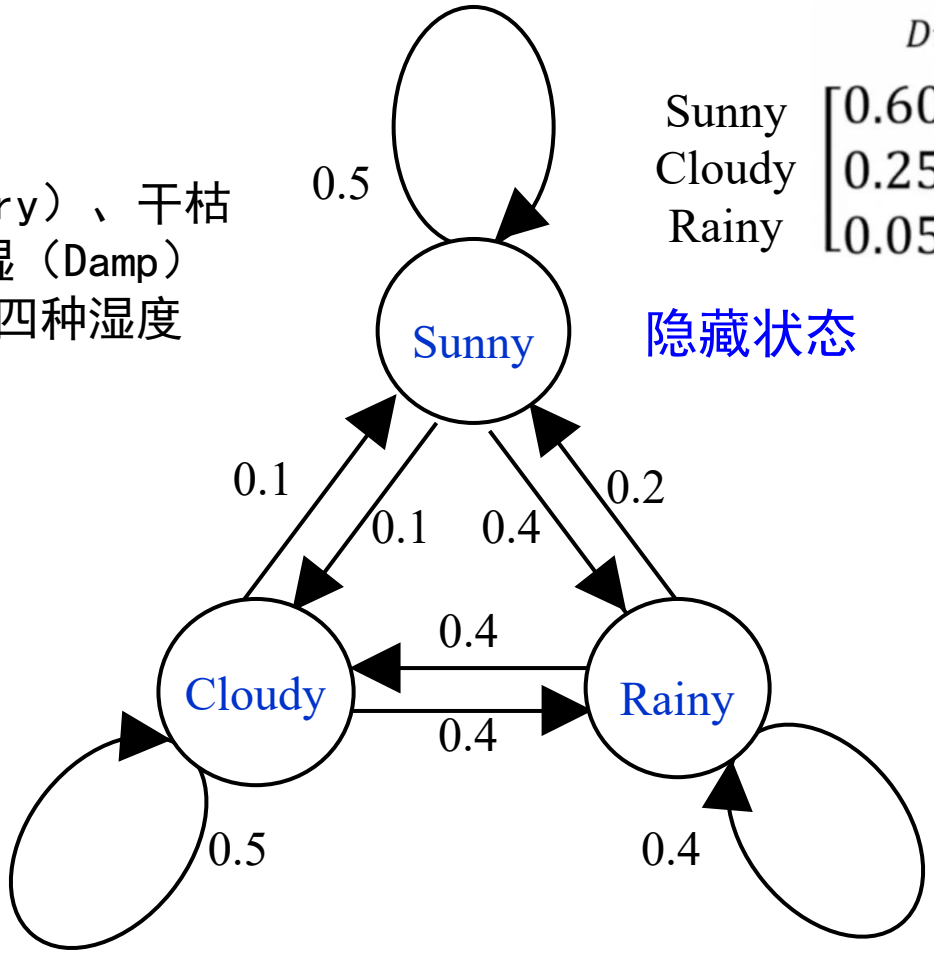
HMM的例子 --- 天气（了解）

状态不可见
观察值可见

观察值：干燥（Dry）、干枯（Dryish）、潮湿（Damp）、湿透（Soggy）四种湿度

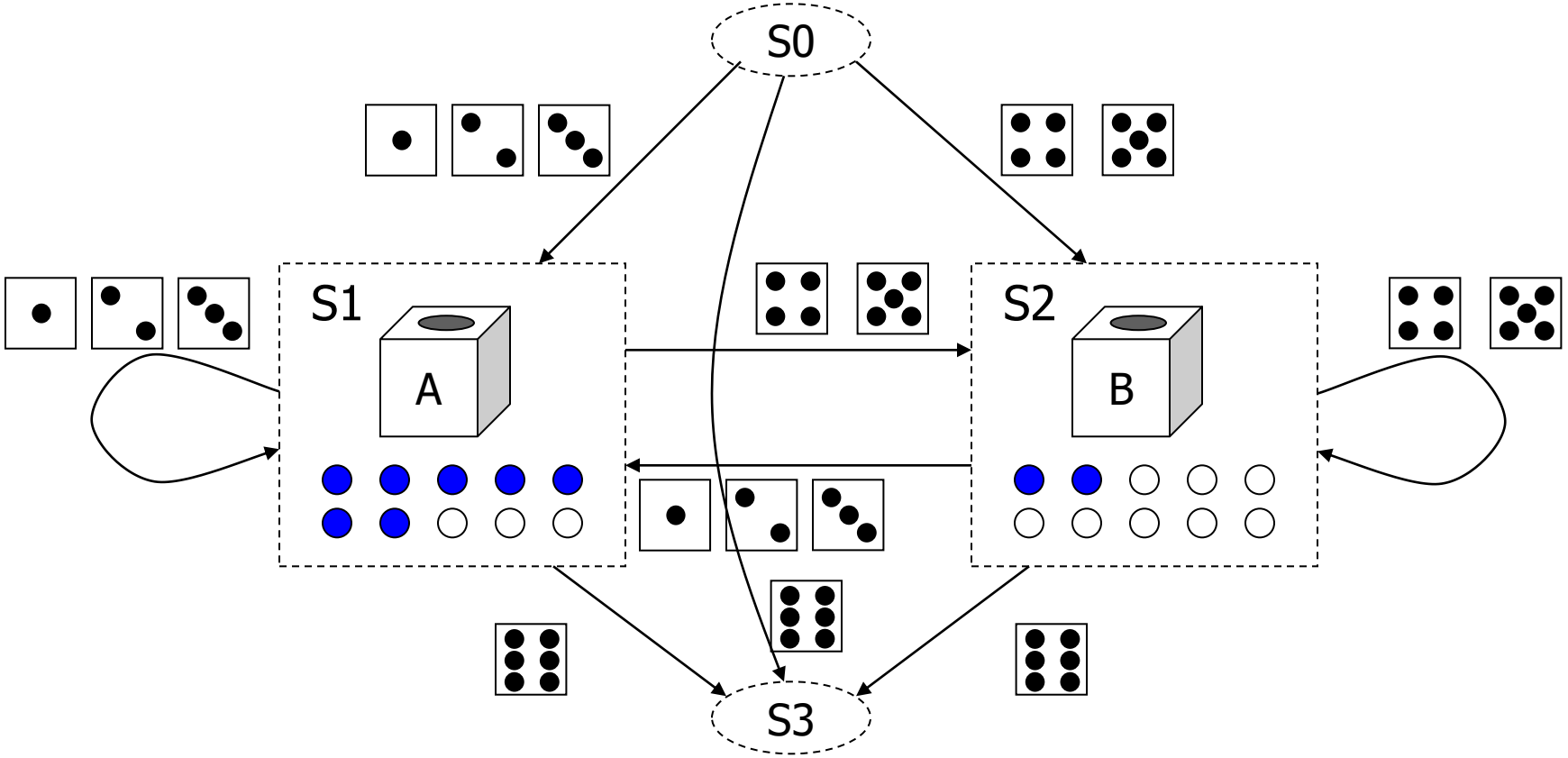
观测值概率矩阵：

	Dry	*Dryish*	*Damp*	*Soggy*
Sunny	0.60	0.20	0.15	0.05
Cloudy	0.25	0.25	0.25	0.25
Rainy	0.05	0.10	0.35	0.50

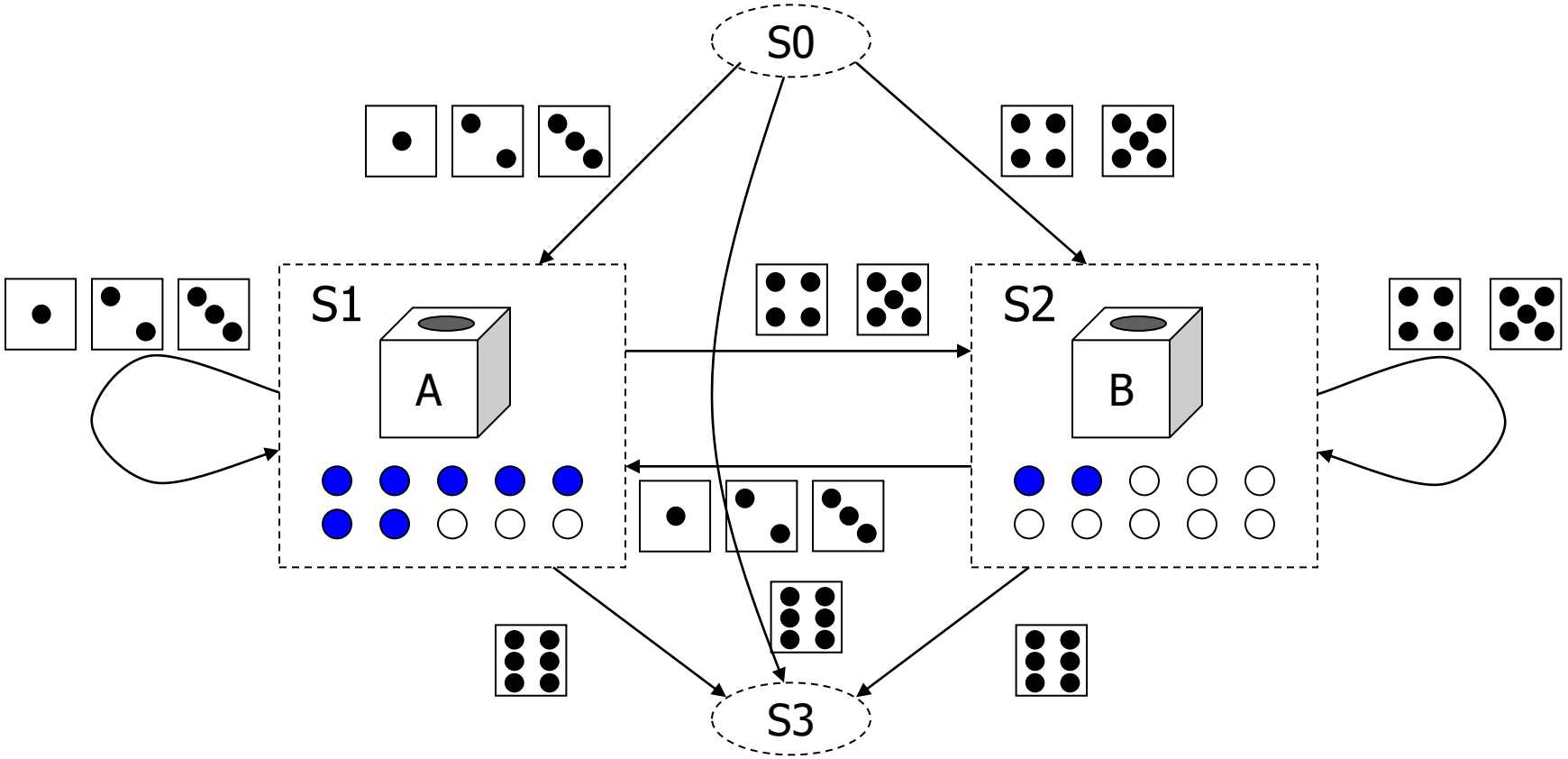


隐藏状态

HMM的例子 --- 掷骰子（了解）



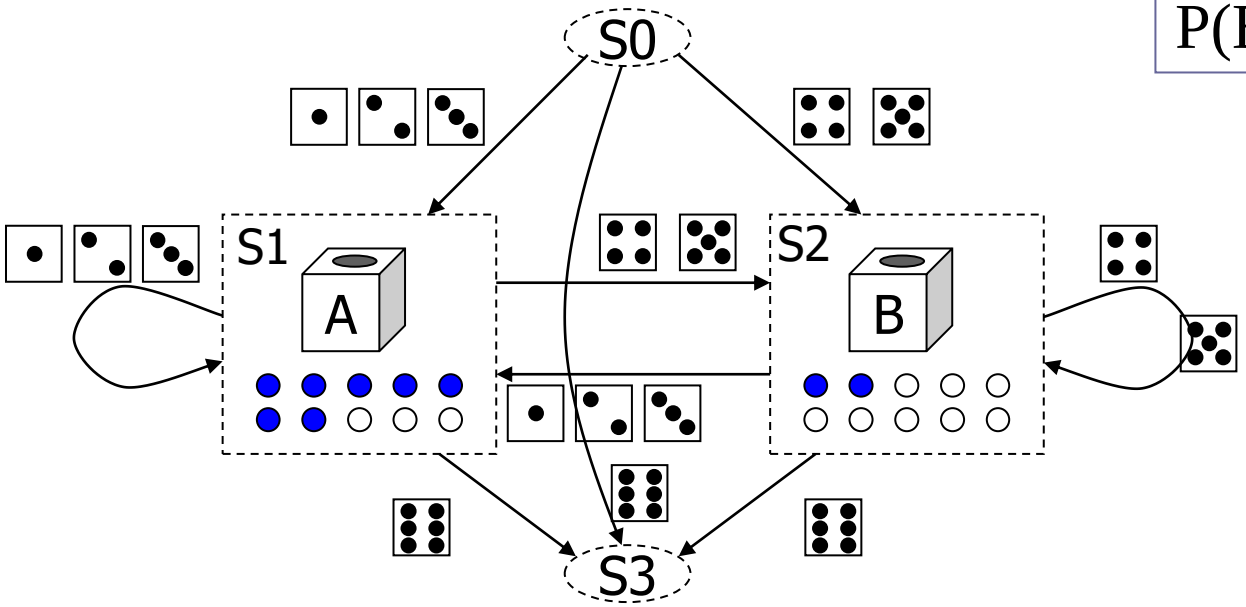
HMM的例子--- 掷骰子（续） （了解）



P(Blue · White|λ)?

HMM的例子 ---- 模型概率计算（了解）

P(Blue · White|λ)?



观察值概率

Observation Probability

$P(Blue \mid S_1) = 0.7$

$P(White \mid S_1) = 0.3$

$P(Blue \mid S_2) = 0.2$

$P(White \mid S_2) = 0.8$

状态转移概率

Transition Probability

$P(S_0S_1) = 0.5$

$P(S_0S_2) = 0.33$

$P(S_0S_3) = 0.17$

$P(S_1S_1) = 0.5$

$P(S_1S_2) = 0.33$

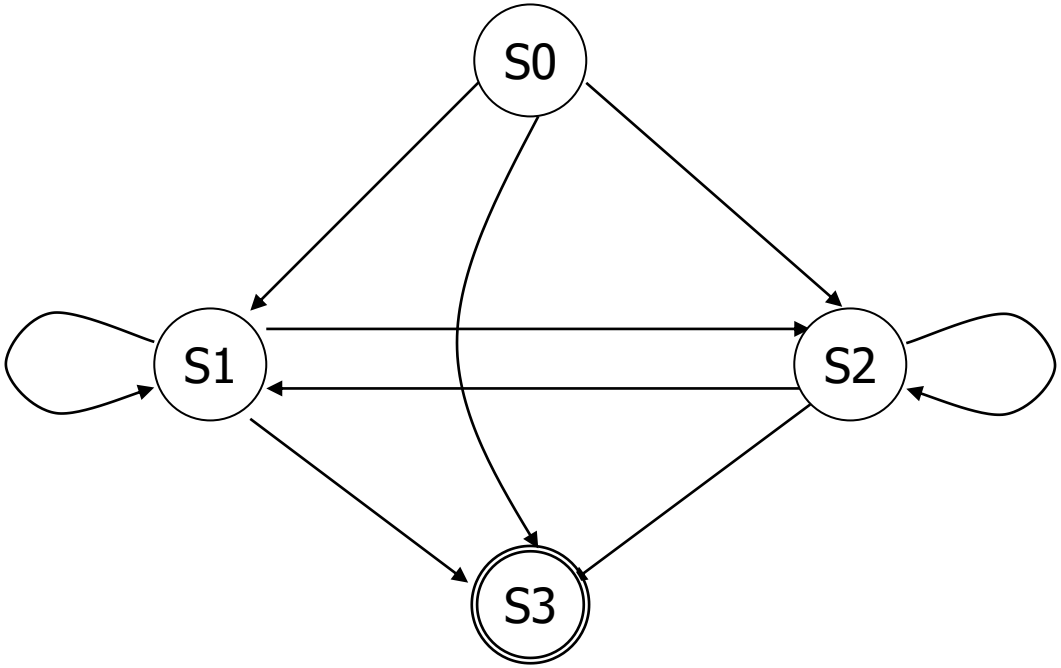
$P(S_1S_3) = 0.17$

$P(S_2S_1) = 0.33$

$P(S_2S_2) = 0.5$

$P(S_2S_3) = 0.17$

HMM的例子 --- 模型概率计算（续）（了解）



P(Blue · White|λ)?

观察值概率

Observation Probability

$P(Blue \mid S_1) = 0.7$

$P(White \mid S_1) = 0.3$

$P(Blue \mid S_2) = 0.2$

$P(White \mid S_2) = 0.8$

状态转移概率

Transition Probability

$P(S_0S_1) = 0.5$

$P(S_0S_2) = 0.33$

$P(S_0S_3) = 0.17$

$P(S_1S_1) = 0.5$

$P(S_1S_2) = 0.33$

$P(S_1S_3) = 0.17$

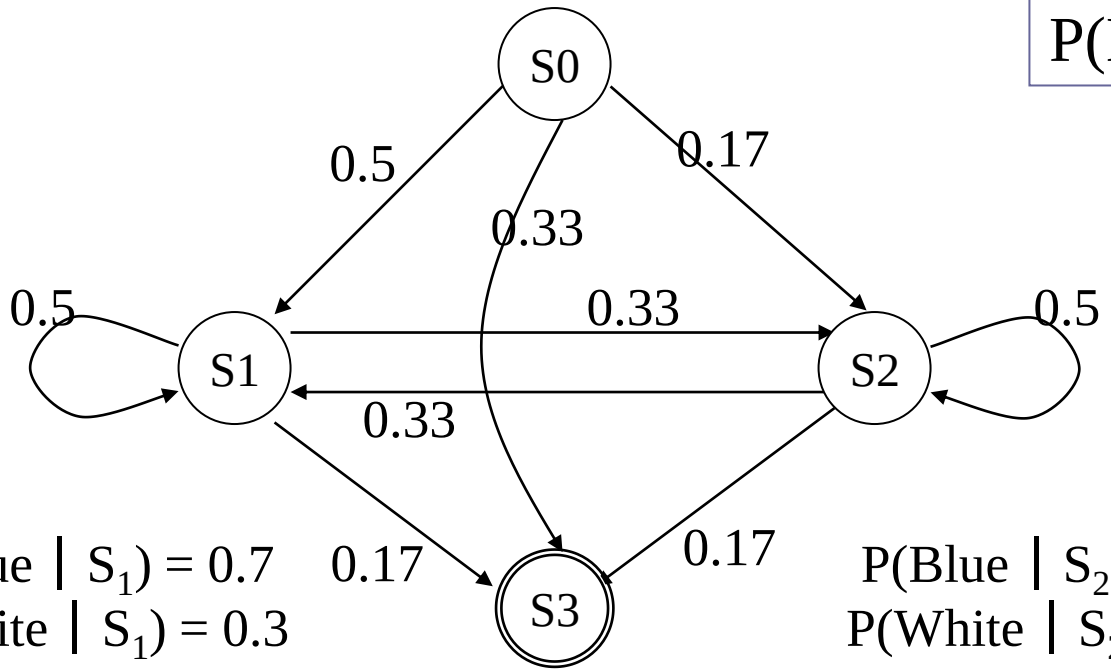
$P(S_2S_1) = 0.33$

$P(S_2S_2) = 0.5$

$P(S_2S_3) = 0.17$

HMM的例子 --- 模型概率计算（续）（了解）

P(Blue · White|λ)?



P(Blue | S_1) = 0.7
P(White | S_1) = 0.3

P(Blue | S_2) = 0.2
P(White | S_2) = 0.8

S0→S1→S1→S3
↓ ↓
Blue White

S0→S1→S2→S3
↓ ↓
Blue White

S0→S2→S1→S3
↓ ↓
Blue White

S0→S2→S2→S3
↓ ↓
Blue White

隐马尔可夫模型(HMM)介绍（了解）

- HMM是一个经典的机器学习模型，被广泛用于语音识别，语音合成，模式识别等领域。

- HMM的重要性随着深度学习算法的流行而下降。但是作为一个经典模型，学习并理解HMM建模对提高我们的问题解决能力，扩展算法思路，了解语音识别、语音合成技术的本质是十分有帮助的。

HMM的模型描述和参数（基本概念）

HMM可以用下列参数来描述，包括状态集合、观察值序列和3个概率矩阵：

1）状态集合：$\mathbf{S} = \{s_j,\ j = 1,2,\ldots,N\}$

N:可能的状态数目。这些状态之间满足马尔可夫性质，是马尔可夫模型中实际所隐含的状态。这些状态不可被直接观测而得到。

2）可观测值集合：$\mathbf{V} = \{\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots \boldsymbol{v}_K\}$

K：每个状态对应的可能的观察值数目。在模型中与隐含状态相关联，可通过直接观测而得到。时刻t观察到的观察值为$\mathbf{o}_t$，$\mathbf{o}_t \in \{\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots \boldsymbol{v}_K\}$。

HMM的模型描述和参数（续）（基本概念）

3）初始状态概率矩阵： $\pi=[\pi_1\ \pi_2\ \ldots\ \pi_N]$

表示隐含状态在初始时刻t=1的概率矩阵。其中

$\pi_i = P(q_1 = i)$。

4）状态转移矩阵： $\mathbf{A} = [a_{\mathrm{i,j}}],\ 1 \le i,j \le N$

描述了HMM中各个状态之间的转移概率，$a_{i,j}$表示在$t-1$时刻状态为s_i(即i)，且在t时刻状态为s_j(即j)的概率。

5）观测值概率矩阵： $\mathbf{B} = [b_j(k)],\ 1 \le j \le N, 1 \le k \le K$

表示在t时刻、隐含状态是s_j的条件下，观察值为$\boldsymbol{v}_k$的概率。

$$\mathrm{b_j}(\mathbf{o_t}) = \mathrm{P}(\mathbf{o_t} = \boldsymbol{v}_\mathrm{k} | \mathrm{q_t} = \mathrm{j}), \qquad k = 1,2,\ldots K;\quad j = 1,2,\ldots,N$$

一个HMM的参数组为： $\lambda = (\boldsymbol{S}, \boldsymbol{V}, \pi, \boldsymbol{A}, \boldsymbol{B})$； 简写为：$\lambda = (\pi, \boldsymbol{A}, \boldsymbol{B})$

HMM的三个基本问题（掌握）

问题1：模型概率计算问题

已知一个HMM参数组$\lambda = (\pi, \boldsymbol{A}, \boldsymbol{B})$，和给定一个观察值序列$\boldsymbol{O} = \boldsymbol{o}_1 \boldsymbol{o}_2 \ldots \boldsymbol{o}_T$的条件下，如何计算在给定模型$\lambda$的条件下观察值序列$\boldsymbol{O}$的概率$P(\boldsymbol{O}|\lambda)$。

问题2：模型解码问题

给定模型λ和观察值序列$\boldsymbol{O}$，如何确定一个最佳状态序列$\boldsymbol{Q}^* = \{q_1^*, q_2^*, \ldots, q_T^*\}$。

问题3：模型参数估计问题

给定观察值序列集合$\boldsymbol{O}$，如何调整参数λ，使得$P(\boldsymbol{O}|\lambda)$达到最大值。

模型概率计算 --- 直接计算法（了解）

先计算 $P(\boldsymbol{O},\boldsymbol{Q}|\lambda)$，其中$\boldsymbol{Q}$为一给定的状态序列$\boldsymbol{Q}=q_1q_2,\ldots,q_T$

有 $$P(\boldsymbol{O},\boldsymbol{Q}|\lambda)=P(\boldsymbol{O}|\boldsymbol{Q},\lambda)P(\boldsymbol{Q}|\lambda)$$

而 $$P(\boldsymbol{O}|\boldsymbol{Q},\lambda)=\prod_{t=1}^{T}P(\boldsymbol{o}_t|q_t,\lambda)=b_{q_1}(\boldsymbol{o}_1)b_{q_2}(\boldsymbol{o}_2)\cdots b_{q_T}(\boldsymbol{o}_T)$$

$$P(\boldsymbol{Q}|\lambda)=\pi_{q_1}a_{q_1q_2}\cdots a_{q_{T-1}q_T}$$

所以 $$P(\boldsymbol{O},\boldsymbol{Q}|\lambda)$$
$$=\pi_{q_1}b_{q_1}(\boldsymbol{o}_1)a_{q_1q_2}b_{q_2}(\boldsymbol{o}_2)\cdots a_{q_{T-1}q_T}b_{q_T}(\boldsymbol{o}_T)$$

模型概率计算 ——— 直接计算法（续）（了解）

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

$$= \sum_{\forall \boldsymbol{Q}} \pi_{q_1} b_{q_1}(\boldsymbol{o}_1) a_{q_1 q_2} b_{q_2}(\boldsymbol{o}_2) \cdots a_{q_{T-1} q_T} b_{q_T}(\boldsymbol{o}_T)$$

计算量： $2TN^T$

当 $N=5,\ T=100$ 时，计算量达 10^{72}

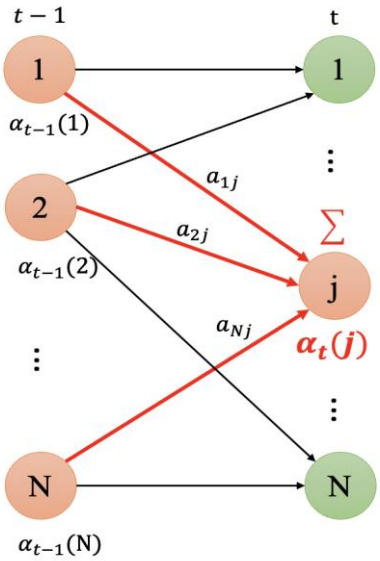
解决思路：前向、后向算法

（forward algorithm, backward algorithm）

模型概率计算 --- 前向算法(forward algorithm)（掌握）

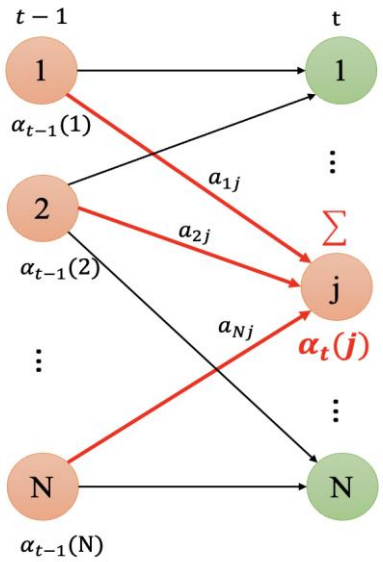
前向变量：$\alpha_t(i) = P(\boldsymbol{o}_1\boldsymbol{o}_2\cdots\boldsymbol{o}_t, q_t = i|\lambda)$

定义：到时刻t为止部分观测序列为$\mathbf{o}_1, \mathbf{o}_2, \ldots, \mathbf{o}_\mathrm{t}$，且状态为$s_i$（为方便表示，在后面的算法中状态$s_i$简记为$i$）的概率。



前向概率计算示意图

模型概率计算 --- 前向算法（续）（掌握）



前向概率计算示意图

前向变量有如下性质：

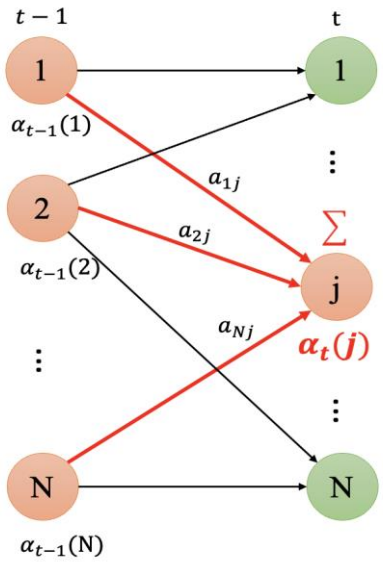
（1）初值易求 $\alpha_1(i) = P(\boldsymbol{o}_1, q_1 = i) = \pi_i b_i(\boldsymbol{o}_1)$

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

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

因此可以利用递推关系，逐层递推，计算出全部$\alpha_t(j)$。
最后再由$\alpha_T(j)$计算得到$P(\boldsymbol{O}|\lambda)$。

模型概率计算 --- 前向算法（续）（掌握）



前向概率计算示意图

前向算法：

（1）初始化：对$1 \le i \le N$,有

$$\alpha_1(i) = \pi_i b_i(\boldsymbol{o}_1)$$

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

$$\alpha_t(j) = \left[\sum_{i=1}^{N} \alpha_{t-1}(i) a_{ij}\right] b_j(\boldsymbol{o}_t)$$

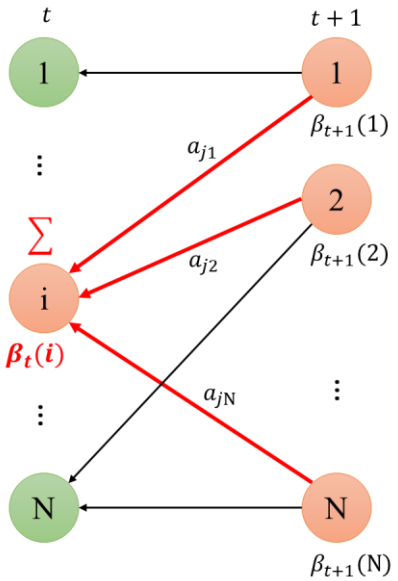
计算量为N^2T，
当N=5，T=100时，
只需2500次乘法运算

（3）终止：$P(\boldsymbol{O}|\lambda) = \sum_{i=1}^{N} \alpha_T(i)$

模型概率计算---后向算法(backward algorithm)（重点掌握）

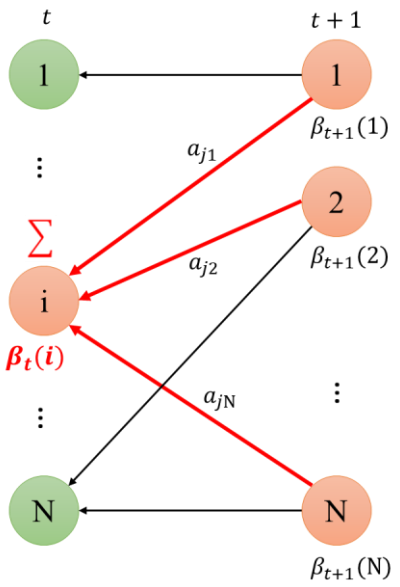
后向变量：$\beta_t(i) = P(\boldsymbol{o}_{t+1}\boldsymbol{o}_{t+2}\cdots\boldsymbol{o}_T|q_t = i, \lambda)$

定义：从时刻$t+1$开始到T 为止部分观测序列为$\mathbf{o}_{t+1}, \mathbf{o}_{t+2}, \ldots, \mathbf{o}_{\mathrm{T}}$，且状态为$i$的概率。



后向概率计算示意图

模型概率计算 --- 后向算法（续）（重点掌握）



后向概率计算示意图

后向算法：

（1）初始化：对$1 \le i \le N$,有

$$\beta_T(i) = 1$$

（2）递推：对$t = T-1, T-2, \ldots, 1, 1 \le i \le N$,有

$$\beta_t(i) = \left[\sum_{j=1}^{N} \beta_{t+1}(j) a_{ij}\right] b_j(\boldsymbol{o}_{t+1})$$

计算量为N^2T数量级

（3）终止：

$$P(\boldsymbol{O}|\lambda) = \sum_{i=1}^{N} \pi_i b_i(\boldsymbol{o}_1) \beta_1(i)$$

模型解码问题的描述（掌握）

- 最佳状态序列的确定

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

$$\boldsymbol{Q}^* = \underset{\boldsymbol{Q}}{\text{argmax}} P(\boldsymbol{O}, \boldsymbol{Q}|\lambda)$$

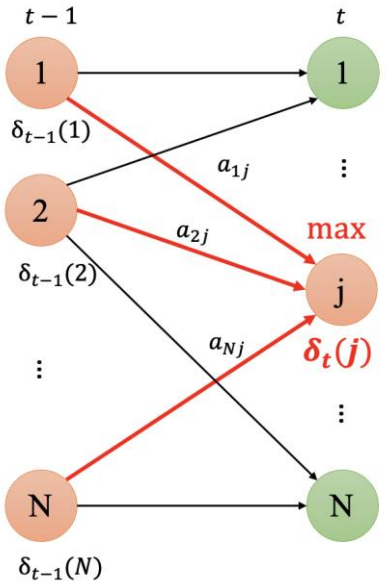
定义$\delta_t(i) = \max_{q_1, q_2, \cdots q_{t-1}} P(q_1 q_2 \cdots q_{t-1}, q_t = i, \boldsymbol{o}_1 \boldsymbol{o}_2 \cdots \boldsymbol{o}_t|\lambda)$为时刻t时沿着一条路径$q_1 q_2 \cdots q_{t-1}$，且$q_t = i$，产生$\boldsymbol{o}_1 \boldsymbol{o}_2 \cdots \boldsymbol{o}_t$的最大概率。

如何求解最佳状态序列Q?

解决思路：Viterbi算法

Viterbi算法（掌握）

求取最佳状态序列$\boldsymbol{Q}^*$的过程为



（1）初始化：对$1 \le i \le N$,有

$$\delta_1(i) = \pi_i b_i(\boldsymbol{o}_1)$$
$$\varphi_1(i) = 0$$

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

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

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

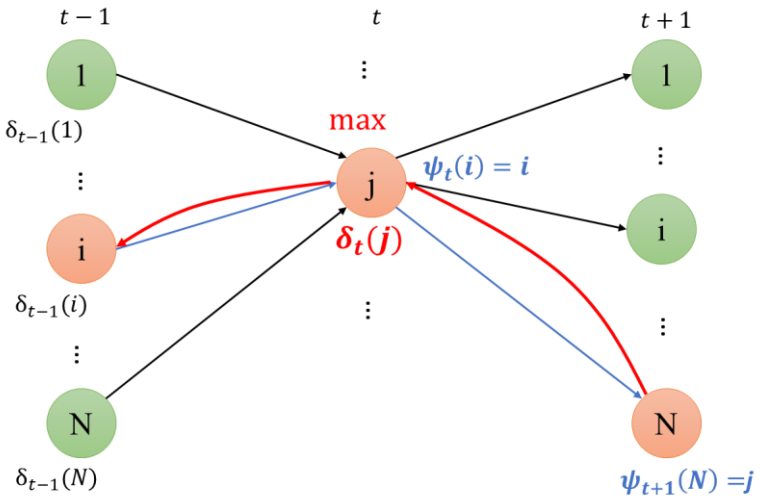
Viterbi算法（续） （掌握）

求取最佳状态序列$\boldsymbol{Q}^*$的过程为

（3）终止：

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

$$q_T^* = \operatorname{argmax}_{1 \le i \le N} [\delta_T(i)]$$



（4）路径回溯，确定最佳状态序列：

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

最终得到最佳状态序列$\mathbf{Q}^*$以及所对应的概率P^*

模型参数估计的思路（基本概念）

给定一个观察值序列$O = \boldsymbol{o}_1\boldsymbol{o}_2 \ldots \boldsymbol{o}_T$确定一个$\lambda = (\pi, \boldsymbol{A}, \boldsymbol{B})$使得$P(\boldsymbol{O}|\lambda)$最大。实际上，不存在一种方法直接估计最佳的$\lambda$。

思路：

根据观察值序列选取初始模型$\lambda = (\pi, \boldsymbol{A}, \boldsymbol{B})$，然后依据某种方法求得一组新参数$\hat{\lambda} = (\hat{\pi}, \hat{\boldsymbol{A}}, \hat{\boldsymbol{B}})$，保证有$P\left(\boldsymbol{O}\middle|\hat{\lambda}\right) > P(\boldsymbol{O}|\lambda)$。利用递归的思路重复这个过程，逐步改进模型参数，直到$P\left(\boldsymbol{O}\middle|\hat{\lambda}\right)$收敛。

- 经典的方法：Baum-Welch算法

Baum-Welch算法（基本概念）

- Baum-Welch算法是用来解决当观测序列和隐状态序列未知时的HMM参数的算法，该算法是期望最大化（EM）算法的一个特例。

- 设辅助函数$L(\lambda,\hat{\lambda})=\sum_{\boldsymbol{Q}} P(\boldsymbol{Q}|\boldsymbol{O},\hat{\lambda}) log\, P(\boldsymbol{O},\boldsymbol{Q}|\lambda)$。
  - 在步骤E(期望)求给定参数集λ上的$L(\lambda,\hat{\lambda})$；
  - 在步骤M(最大化)更新模型参数 $\lambda=(\boldsymbol{A},\boldsymbol{B},\pi)$。

$$\hat{\lambda}=arg\,\max_{\lambda}\sum_{\boldsymbol{Q}} P(\boldsymbol{Q}|\boldsymbol{O},\hat{\lambda}) log\, P(\boldsymbol{O},\boldsymbol{Q}|\lambda)$$

- Baum-Welch算法未必能求得全局最大值、而有可能得到一局部极值点。

Baum-Welch算法（续）（了解）

- 给定训练序列$\boldsymbol{O}$和模型λ时，
  定义$\xi_t(i,j)$为t时刻处于状态i，t+1时刻处于状态j的概率；
  定义$\gamma_t(i)$为t时刻位于状态i的概率。

- 使用初始化参数 $\pi_i, a_{ij}, b_i(k)$ 和前向概$\alpha_t(j)$,后向概率$\beta_t(j)$，在步骤E时计算期待值$\xi_t(i,j)$和$\gamma_t(i)$。

$$\xi_t(i,j) = P(q_t = i, q_{t+1} = j|\boldsymbol{O},\lambda) = \frac{\alpha_t(i)a_{ij}b_j(\boldsymbol{o}_{t+1})\beta_t(j)}{\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_t(i)a_{ij}b_j(\boldsymbol{o}_{t+1})\beta_t(j)};$$

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

$\xi_t(i,j)$：在t时刻处于状态i且在t+1时刻处于状态j的概率。
$\gamma_t(i)$：t时刻处于状态i的概率，即为状态占用概率。

Baum-Welch算法（续）（了解）

- 在步骤M时重新估计HMM的参数如下：

$$\hat{a}_{ij} = \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)} \quad ;$$

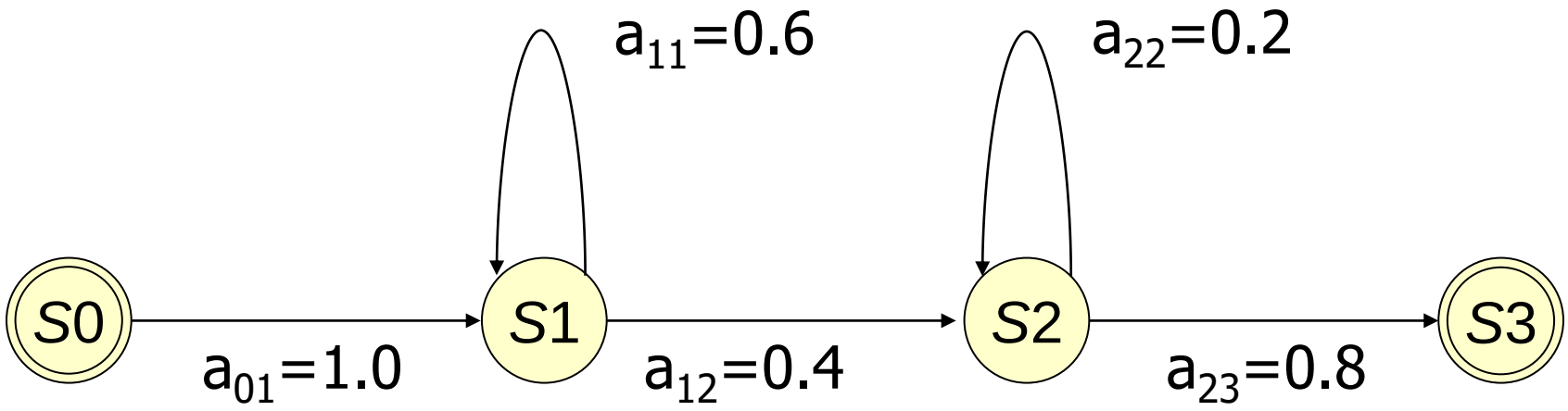
$$\hat{b}_j = \frac{\sum_{t=1,\mathbf{o}_t=v_k}^{T-1} \mathbf{o}_t=v_k \gamma_t(j)}{\sum_{t=1}^{T-1} \gamma_t(i)};$$

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

第三章　语音识别

- 3.1 语音识别概述
- 3.2 声学模型
  - 3.2.1 隐马尔可夫模型
  - 3.2.2 基于GMM-HMM的语音识别技术
  - 3.2.3 DNN-HMM
- 3.3 语言模型
- 3.4 语音识别解码算法
- 3.5 语音识别技术的展望

语音识别领域离散HMM的典型结构（基本概念）



$b_1(v_1) = 0.7$　　$b_2(v_1) = 0.6$

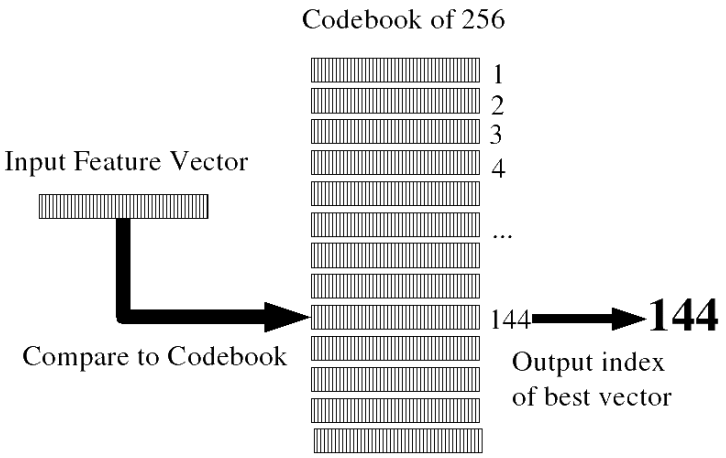
$b_1(v_2) = 0.3$　　$b_2(v_2) = 0.4$

离散随机变量的概率分布（Discrete Distribution）

从左向右HMM
（Left-to-right HMM）

如何计算连续随机变量的概率？（了解）

- 语音的特征向量为连续数值！
- 如何计算连续数值的特征值所对应的概率？
  - 矢量量化（Vector Quantization）：把特征向量聚类到有限（离散）的类



  - 用概率密度函数
    - 经典模型：混合高斯模型（GMM）用来计算观测值概率

单高斯模型 ——— 单元高斯模型（了解）

- 高斯分布即正态分布，是最常见概率分布模型，它经常被用来刻画一些随机量的变化情况。当样本数据x是一维数据时，高斯分布（称为单元高斯模型）遵从下方概率密度函数：

$$\mathrm{P}(x|\theta) = \frac{1}{\sqrt{2\pi\sigma^2}}\exp\left\{-\frac{(x-\mu)^2}{2\sigma^2}\right\}$$

则称x服从均值(期望)为μ，方差为σ^2的正态分布，记为$x \sim N(\mu,\ \sigma^2)$。

单高斯模型 ——— 多元高斯模型（了解）

- 当样本数据 $\mathbf{x} = (x_1, x_2, \ldots, x_D)^T$为多维数据时，遵从下方概率密度函数，称为多元高斯模型：

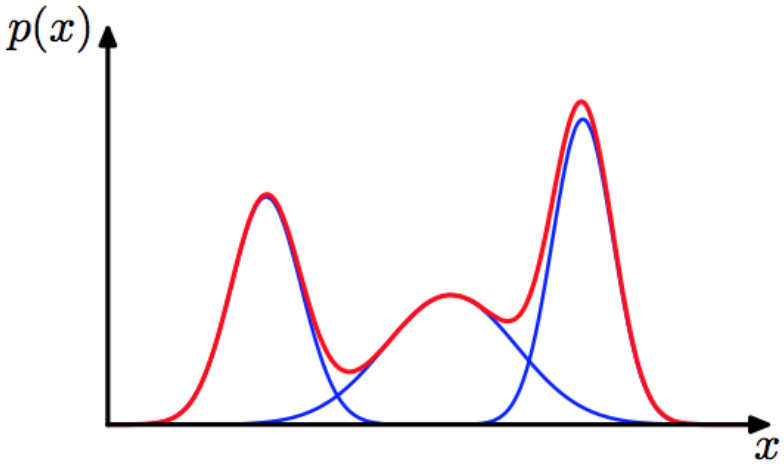
$$P(\mathbf{x}|\theta) = \frac{1}{(2\pi)^{D/2}(\boldsymbol{\Sigma})^{1/2}} \exp\left\{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^T \Sigma^{-1}(\mathbf{x}-\boldsymbol{\mu})\right\}$$

其中，$\boldsymbol{\mu} = (\mu_1, \mu_2, \ldots, \mu_D)^T = E(\mathbf{x})$，
$\boldsymbol{\Sigma} = E[(\mathbf{x}-\bar{\mathbf{x}})(\mathbf{x}-\bar{\mathbf{x}})^T]$分别为$\mathbf{x}$的均值和协方差矩阵。

在语音识别中，多元高斯模型的输入为t时刻观测值的向量$\boldsymbol{o}_t$（比如：实际系统中可为某一帧的特征向量MFCC等）。

高斯混合模型（Gaussian Mixture Models）（基本概念）

- 高斯混合模型是用高斯概率密度函数精确地量化事物，将一个事物分解为若干的基于高斯概率密度函数（正态分布曲线）形成的模型。它可以看作是由M个单高斯模型组合而成的模型来描述复杂的数据（比如语音数据）分布。



使用高斯混合模型是因为高斯分布具备很好的数学性质以及良好的计算性能。

高斯混合模型（续）（掌握）

- 对于M个高斯模型的混合模型，c_m是观测数据属于第m个子模型的先验概率（混合权重），则高斯混合模型的概率分布为：

$$P(\mathbf{x}|\theta) = \sum_{m=1}^{M} c_m N(\mathbf{x}|\boldsymbol{\mu}_m, \boldsymbol{\Sigma}_m)$$

$$N(\mathbf{x}|\boldsymbol{\mu}_m, \boldsymbol{\Sigma}_m) = \frac{1}{(2\pi)^{D/2}(\boldsymbol{\Sigma}_m)^{1/2}} \exp\left\{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_m)^T \boldsymbol{\Sigma}_m^{-1}(\mathbf{x}-\boldsymbol{\mu}_m)\right\}$$

其中，$\sum_{m=1}^{M} c_m = 1$，$N(\mathbf{x}|\boldsymbol{\mu}_m, \boldsymbol{\Sigma}_m)$是第m个子模型的高斯分布密度函数。

高斯混合模型（续）（掌握）

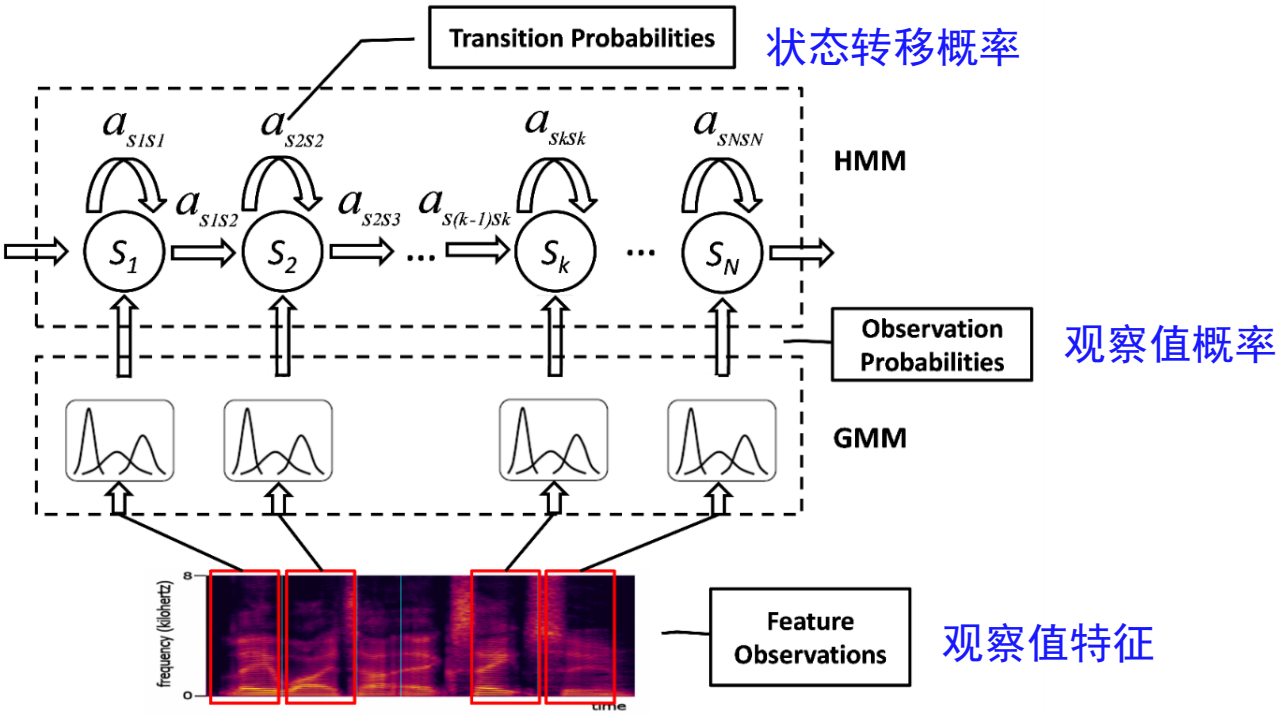
- 在语音处理中，我们通常用连续的概率密度函数来描述连续的观察值向量（$\mathbf{o}_t$）的概率分布。混合高斯模型是其中应用最成功的、最广泛的概率密度函数，若用混合高斯模型（GMM）来表示$\mathbf{o}_t$的概率分布

$$b_i(\mathbf{o}_t) = \sum_{m=1}^{M} \frac{c_{i,m}}{(2\pi)^{D/2}\left|\boldsymbol{\Sigma}_{i,m}\right|^{1/2}} \exp\left\{-\frac{1}{2}\left(\mathbf{o}_t - \boldsymbol{\mu}_{i,m}\right)^T \boldsymbol{\Sigma}_{i,m}^{-1}\left(\mathbf{o}_t - \boldsymbol{\mu}_{i,m}\right)\right\}$$

其中，$c_{i,m}$是状态i时第m个高斯子模型的权重，$\boldsymbol{\mu}_{i,m}$和$\boldsymbol{\Sigma}_{i,m}$为状态i时第m个高斯分布的均值向量和协方差矩阵。

基于GMM-HMM的语音识别框架（基本概念）

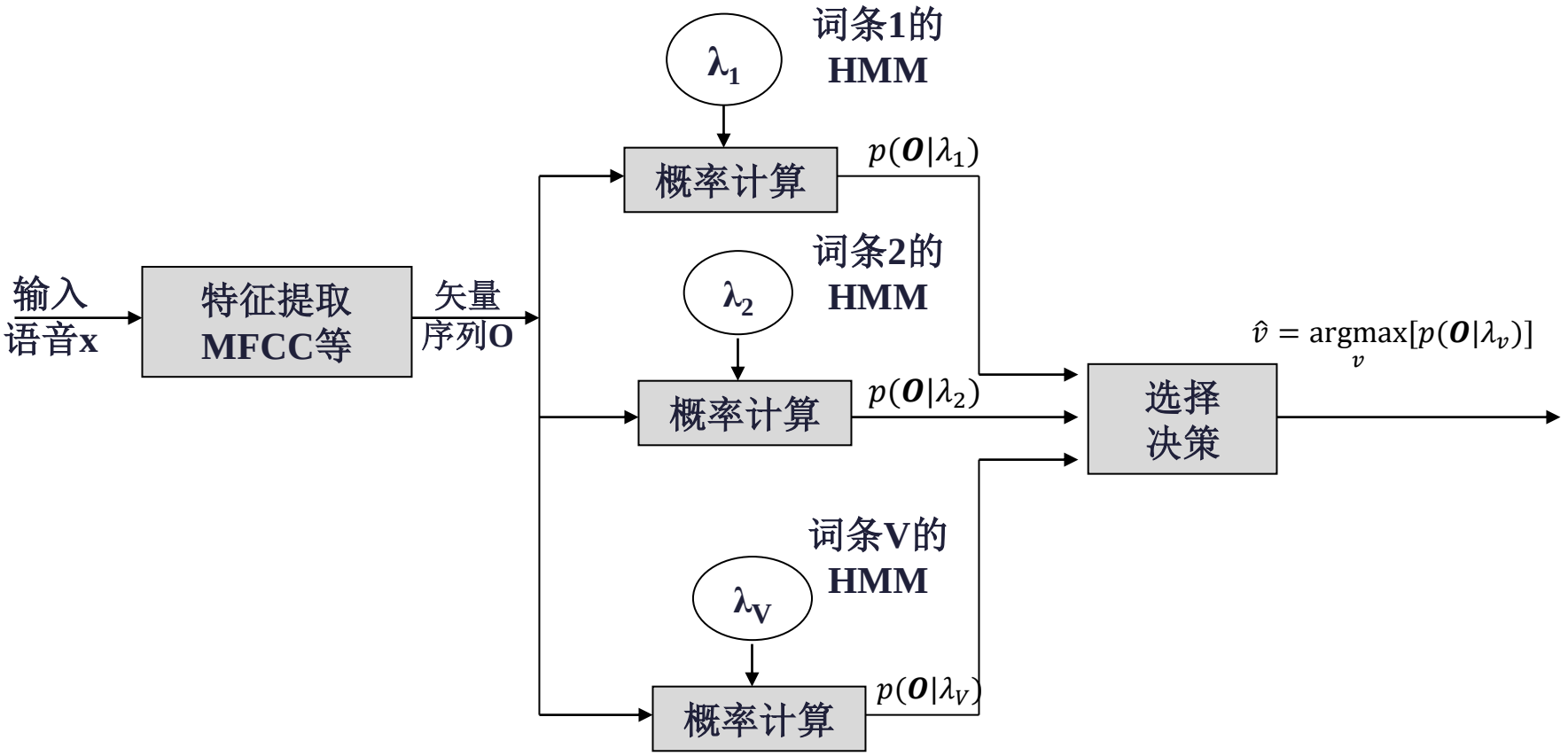
- GMM-HMM（高斯混合模型-隐马尔可夫模型）



- **L. R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. Proc. IEEE, 77(2):257–286, 1989.**

基于HMM的孤立词识别技术（基本概念）

- 孤立词识别原理图

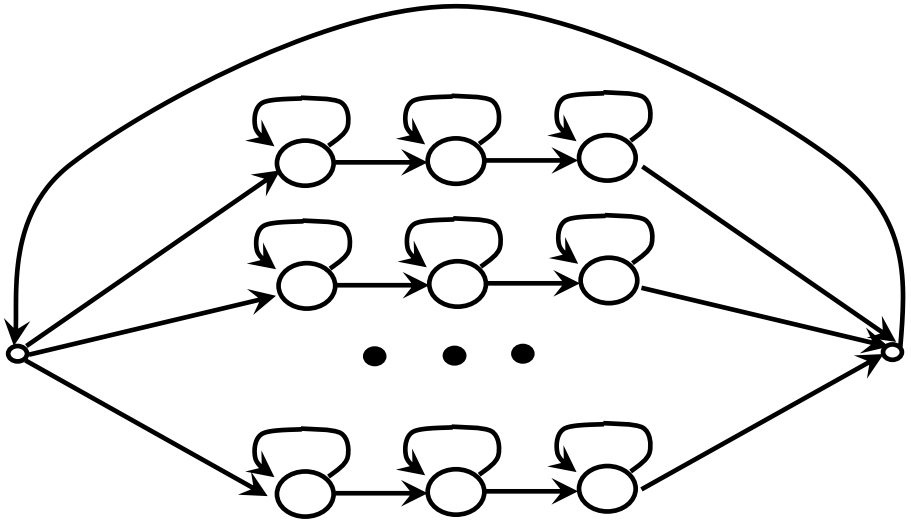


建模单元：孤立词或者子词（音素等）

基于HMM的连接词识别技术（了解）

- 基于HMM的连接词识别
  - 典型例子：数字串识别

- 0-9共10个数字，采用从左至右无跨越HMM
- 形成一个新的HMM
- 通过Viterbi算法解码



基于HMM的连续语音识别技术（基本概念）

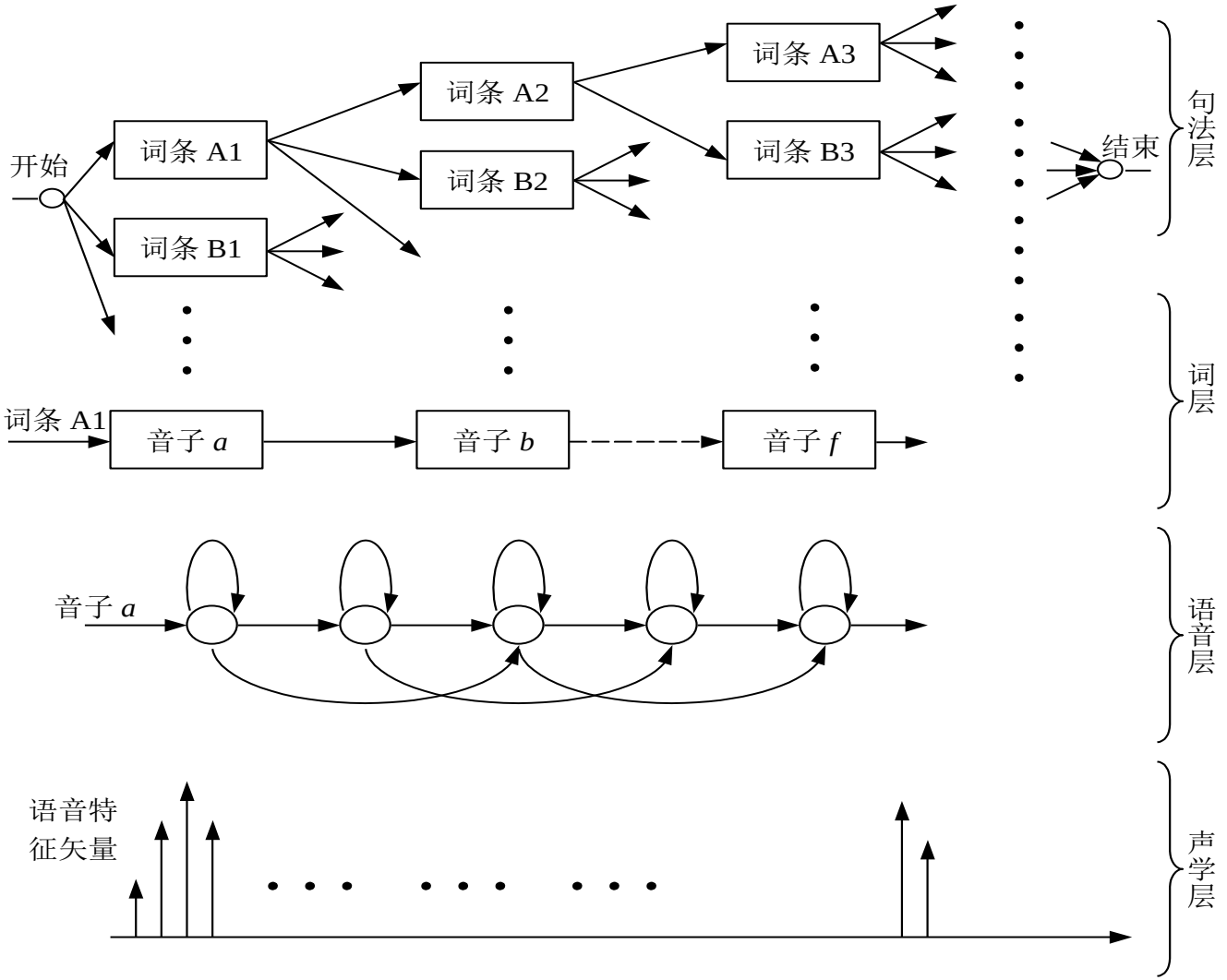
- 连续语音识别（特别是大词表连续语音识别：LVCSR）是语音识别研究中意义最重大、应用成果最丰富，同时最具有挑战性的研究课题。

- 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；
  - 句子中的单词数不确定；
  - 单词边界（起始和结束时间）难以确定；
  - 发音变化（受协同发音的影响严重）。

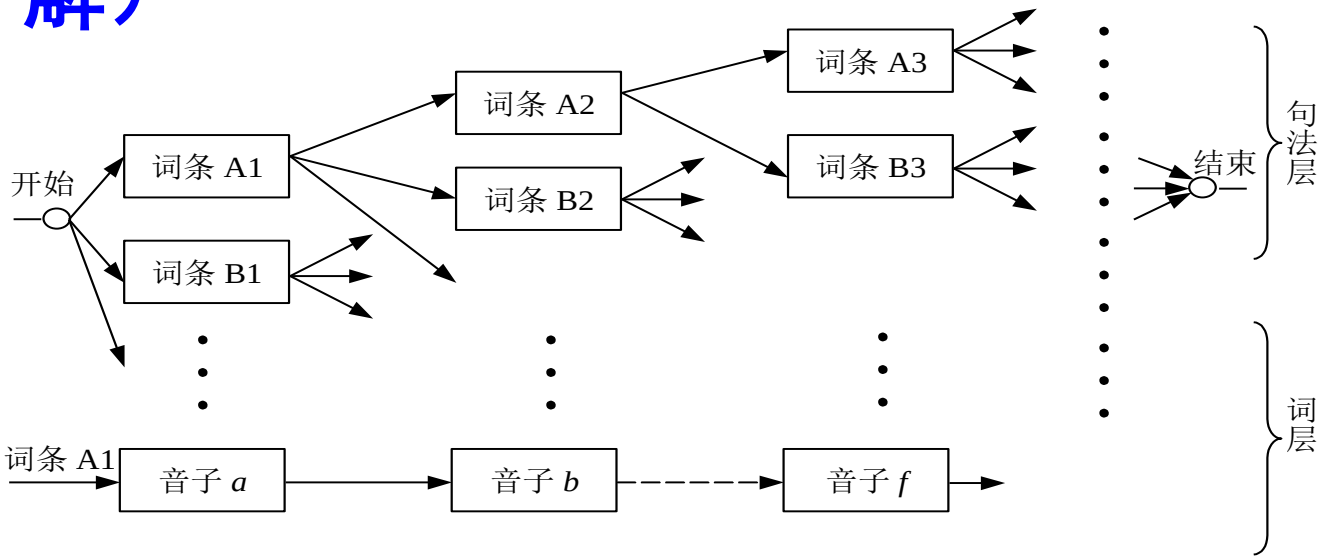
基于HMM的LVCSR系统的统一框架（基本概念）

- 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。

基于HMM的LVCSR系统的统一框架（续）（了解）



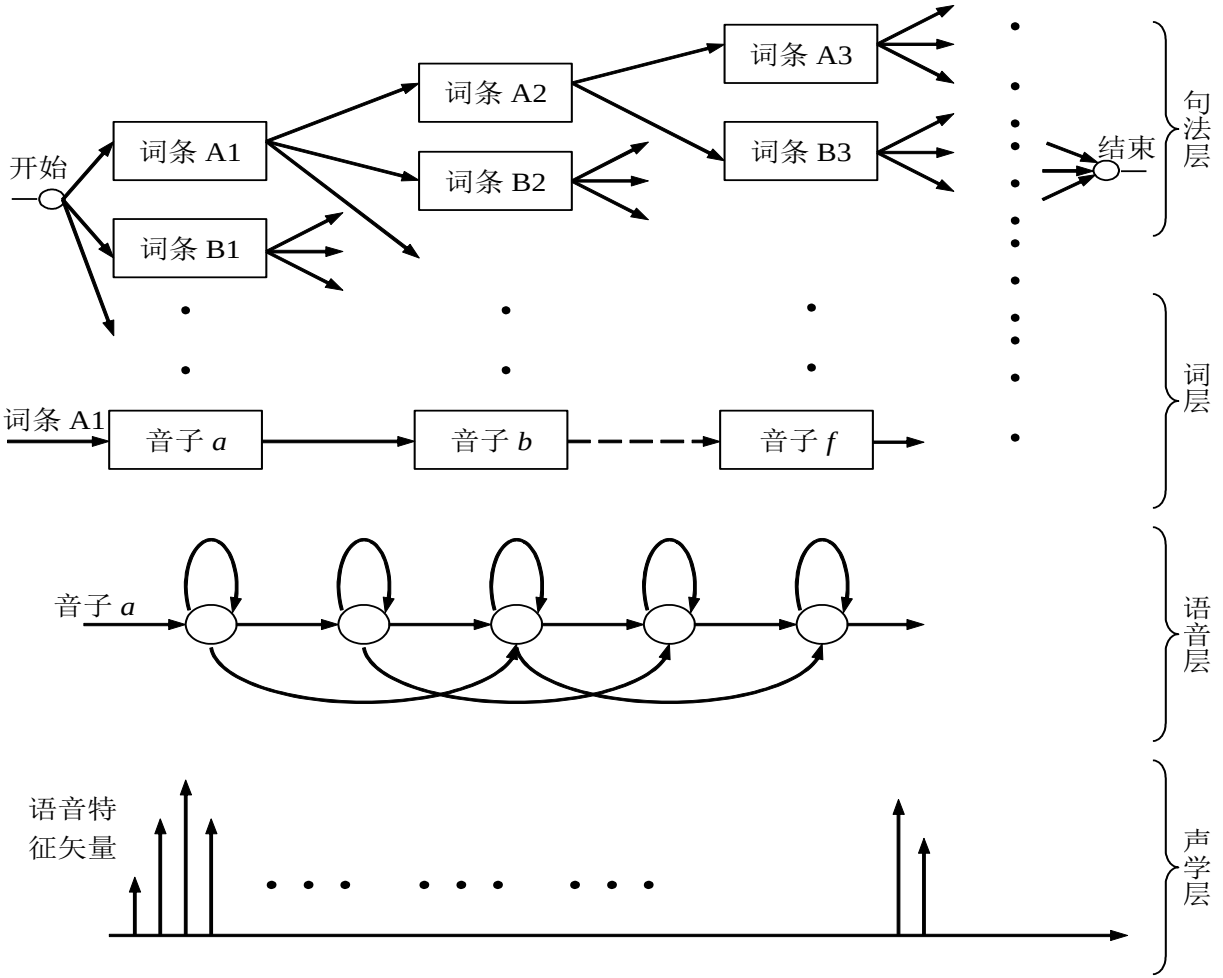
基于HMM的LVCSR系统的统一框架（续）（了解）



在句法层，每个句子由若干词条组成，每一个词条都选自词汇表。句中的一个要选择的词条以一定的概率出现，而第二个词条选择的概率与前一个词条有关(语言模型)，依此类推，直到句子的结束。

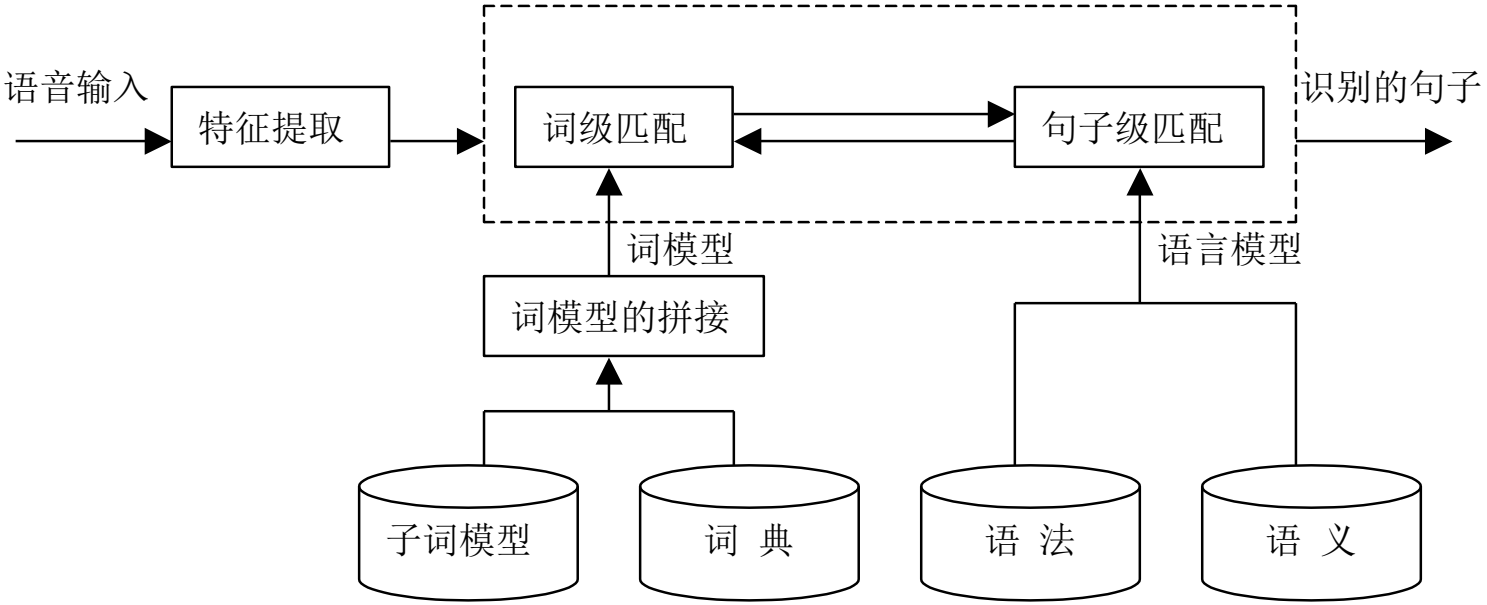
在词层，每一个词条由若干音子串接而成，为此需要一部字典来描述这种串接关系。

基于HMM的LVCSR系统的统一框架（续）（了解）



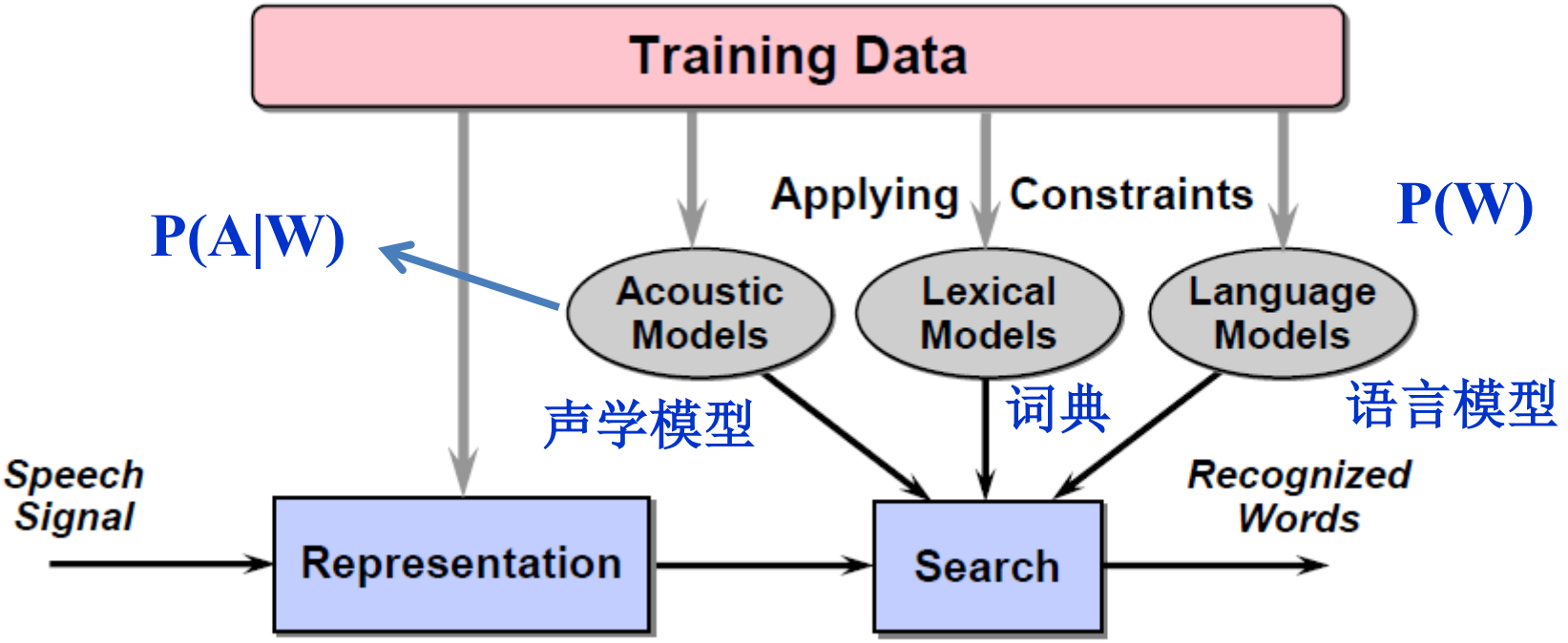
在语音层，每一个音子用一个HMM模型及一套参数来表示。

基于HMM的LVCSR系统的统一框架（续）（了解）



基于子词单元的连续语音识别系统总体框图

基于统计模型的语音识别框架（重点掌握）



连续语音识别系统需要解决的几个主要问题（了解）

1.	声学建模单元（子词）的选择	声学模型
2.	如何由子词构成词？	字典
3.	如何训练子词模型？	声学模型
4.	如何有效利用语言学知识？	语言模型
5.	如何识别得到最优单词序列？	解码算法

建模单元的选择 --- 原则（基本概念）

- 以词作为基本单元建立模型不合理 ，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识别基元，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。

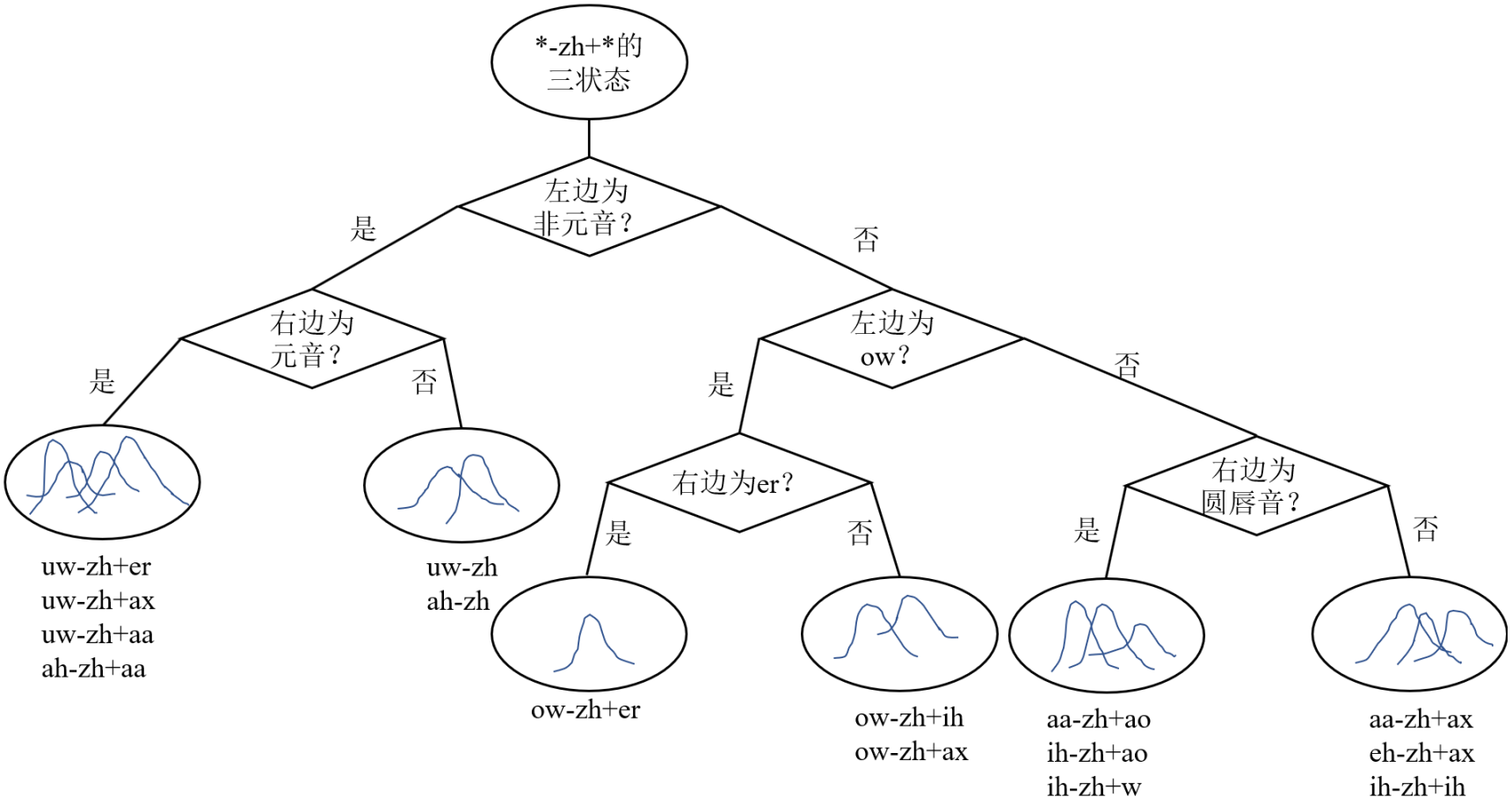
建模单元的选择 --- 三音素HMM

- 单音素建模没有考虑协同发音效应。协同发音(coarticulation)是指在语流中，音段并非是静止的、分离的声音，音段会对相邻的音段产生影响。

- 协同发音的一个解决方法是用三音素建模，考虑音素的上下文，表示为“l-c+r”，其中，c表示中心单元，l为左相关信息，r为右相关信息。

- 假设共有N个音素，那么就需要N^3个三音素，当N增大时，参数量也是非常大的。因此，在语音建模中，使用决策树来进行三音素建模。这一过程依赖于单音素建模后的对齐。

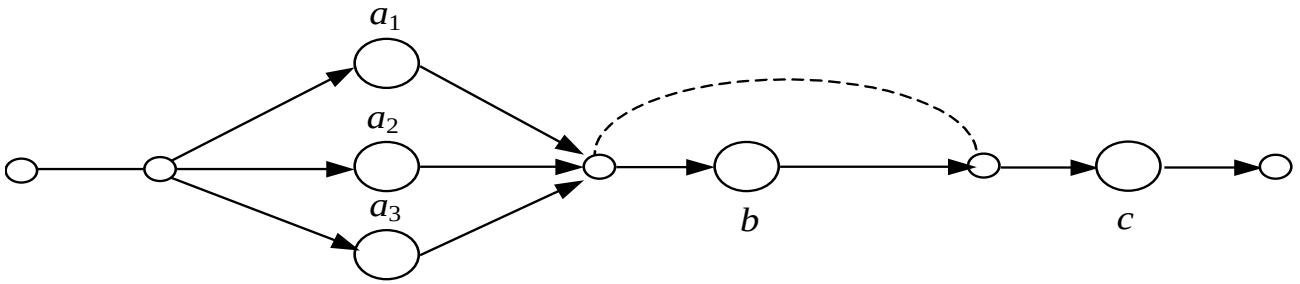
建模单元的选择 --- 决策树（了解）



如何由子词构成词？（了解）

- 在声学—语音学层与句法层之间有一个词层，在词层中应有一部字典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。

- 每个词的发音可能有多种变化方式，在子词串接时，必须有所体现。
  - 替换：即词中的某个音子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 。



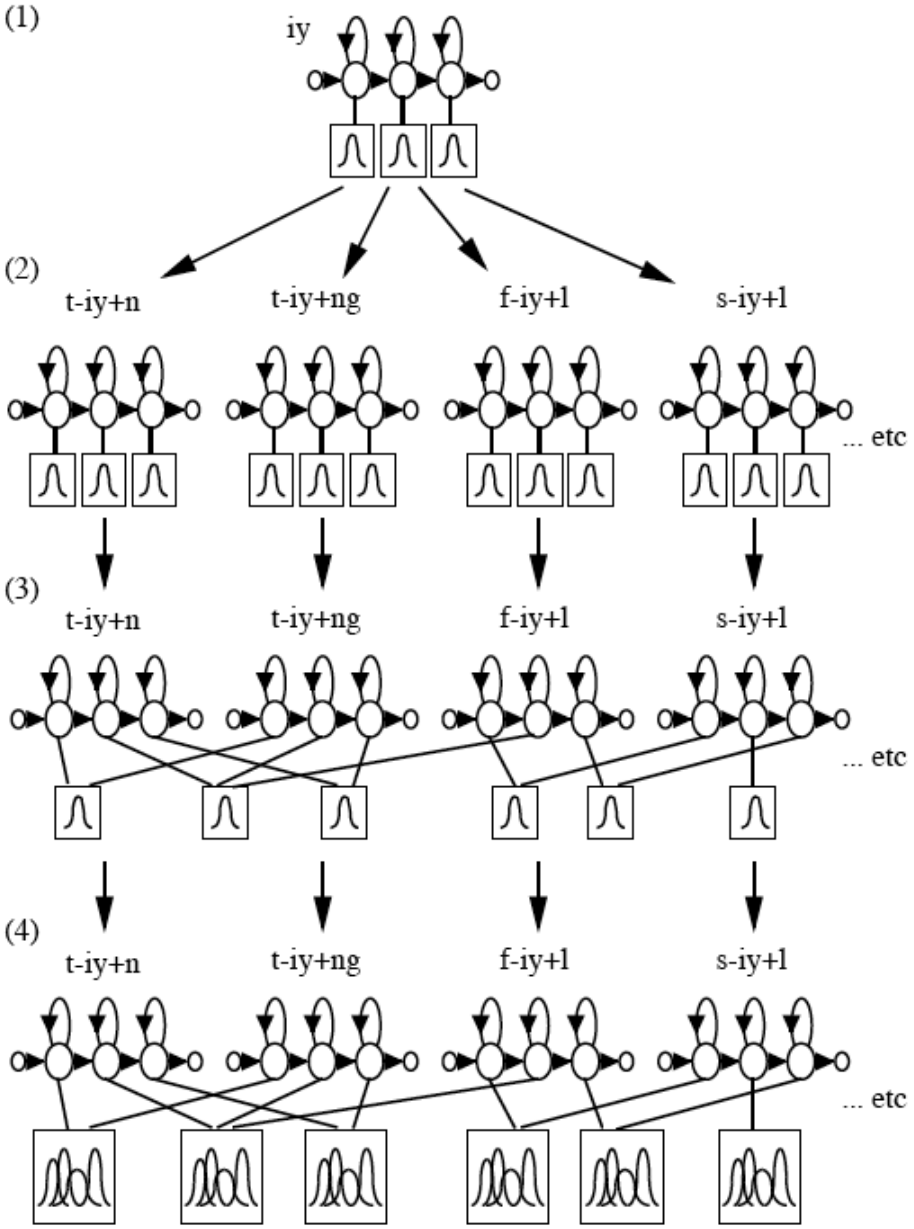
如何训练子词模型？（了解）

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。

从单音素到三音素GMM（基本概念）

- 始于单音素，进行EM训练
- 将高斯分布（Gaussians）复制到三音素中
- 创建决策树并对高斯分布进行聚类
- 复制并混合训练(GMMs)

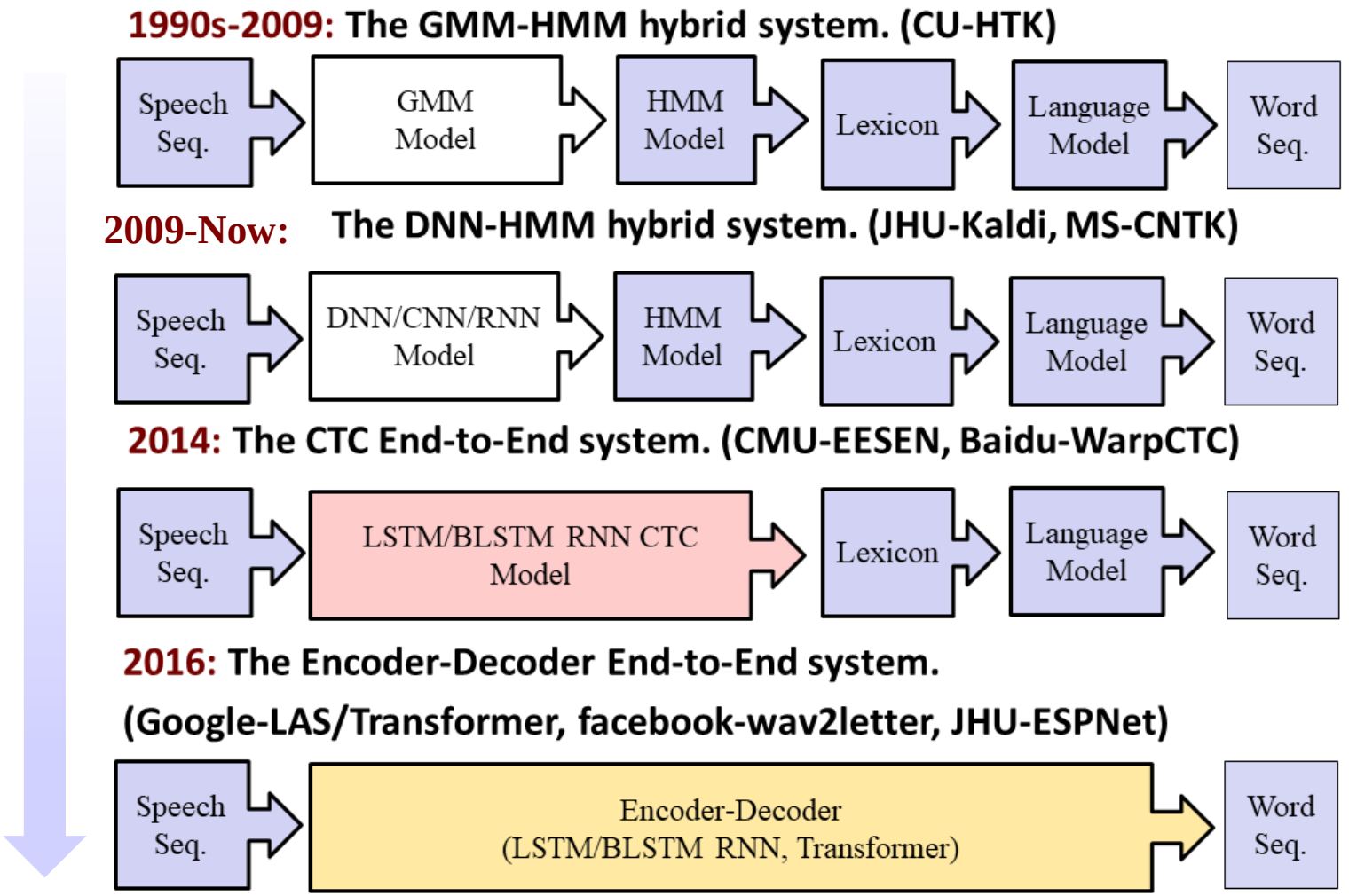


第三章　语音识别

- 3.1 语音识别概述
- 3.2 声学模型
  - 3.2.1 隐马尔可夫模型
  - 3.2.2 基于GMM-HMM的语音识别技术
  - 3.2.3 DNN-HMM
- 3.3 语言模型
- 3.4 语音识别解码算法
- 3.5 语音识别技术的展望

语音识别–声学模型（基本概念）

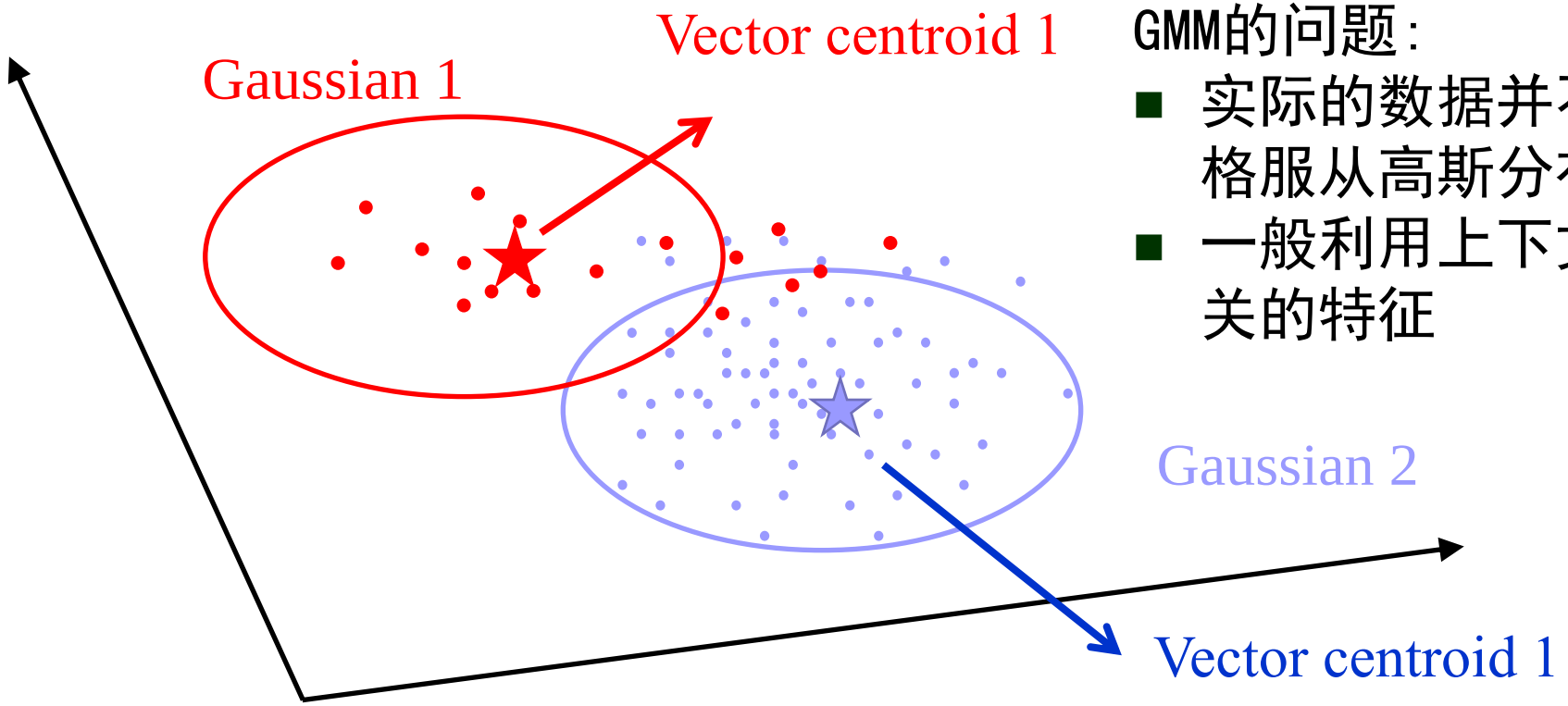
从GMM-HMM到端到端模型



HMM的观测值概率分布（基本概念）

- 离散分布（Discrete Distribution）HMM
  - 矢量量化（Vector Quantization：VQ）
- 连续分布（Continues Distribution）HMM （CD-HMM）
  - 高斯混合模型（Gaussian Mixture Model：GMM）
  - 深度神经网络（Deep Neural Network：DNN）

VQ和GMM的问题（了解）



GMM的问题：

- 实际的数据并不严格服从高斯分布
- 一般利用上下文无关的特征

- 二维特征
- 单高斯模型

基于DNN的语音识别算法（基本概念）

- 20世纪80年代就开始进行结合ANN的尝试
  - 用ANN替换GMM
  - 用BP算法进行训练
- 但由于当时机器运算能力的限制及多层网络训练的复杂性，其效果并不理想（过拟合）。
- 2010年，深度学习开始对语音识别领域产生重要的影响，其识别错误率显著下降。
- 基于DNN的语音识别技术也是深度学习方法在工业界的第一个成功应用，具有里程碑式的意义。

基于DNN的语音识别算法（续） （了解）

- 深度学习是机器学习的子领域，它是对多层表示和抽象的学习，通过多层表示来对数据之间的复杂关系进行建模。
- 通过多层非线性表示来对数据间的复杂关系进行建模。
- 引入深度学习的语音识别技术：
  - 基于DNN-HMM的语音识别方法（本课程的内容）
  - 基于RNN-HMM的语音识别方法
  - 端到端语音识别方法（展望）

深度神经网络（DNN）的介绍（基本概念）

- 深度神经网络（Deep Neural Network，DNN）可以简单的定义为在输入层和输出层之间有多层隐含层（超过一层）的人工神经网络。

- 经典的做法是先基于深度信念网络（Deep Belief Networks，DBN）做预训练，之后用反向传播（Backpropagation，BP）算法微调。

深度神经网络(DNN)的介绍（续）（基本概念）

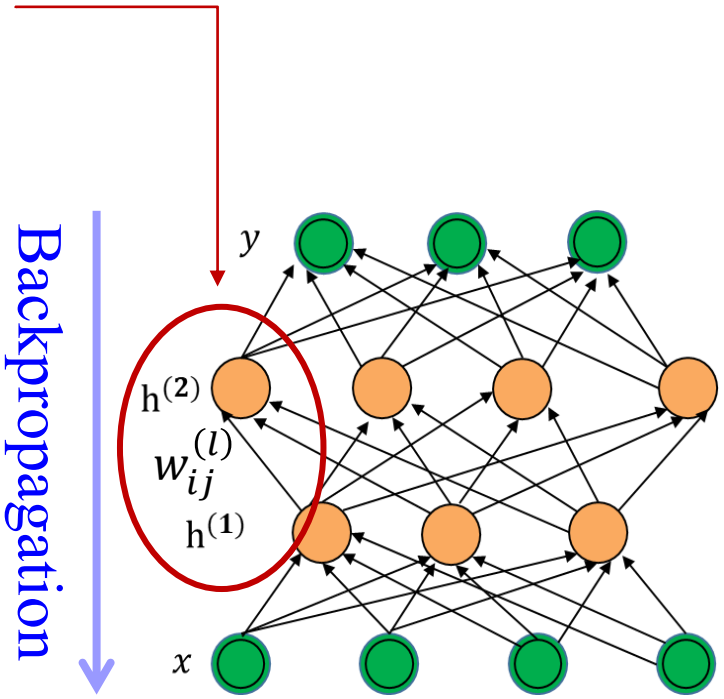
每一个神经元把经过乘法线性变换的上一层的输出求和得到$\mathbf{a}^l$，然后通过非线性激活函数输出给下一层。对于包含L个隐层的DNN，假设其输入为$\mathbf{h}^0 = \boldsymbol{o}_t$，第$l$层隐层的输出$\mathbf{h}^l$可以由如下公式表示：

$$\mathbf{a}^l = \mathbf{W}^l\mathbf{h}^{l-1} + \mathbf{b}^l \quad 1 \le l \le L+1$$

$$\mathbf{h}^l = f(\mathbf{a}^l) \quad 1 \le l \le L$$

$$f(a) = \frac{1}{1+e^{-a}}$$

其中 $f(\cdot)$ 是非线性激活函数，$\mathbf{W}^l$和$\boldsymbol{b}^l$是第l层隐层的权重和偏置向量。



深度神经网络(DNN)的介绍（续）（基本概念）

分类任务（比如：语音识别）的损失函数多采用最小交叉熵（cross-entropy，CE）准则，可以表示如下：

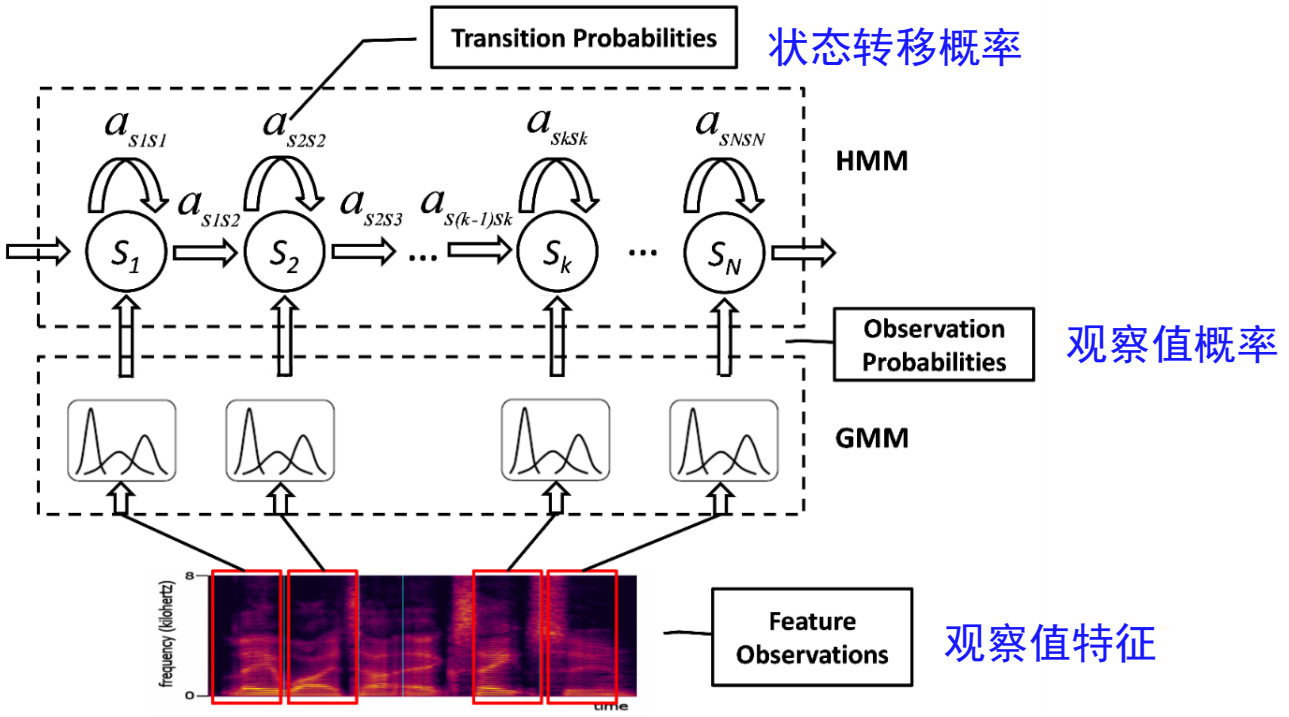
$$J_{CE} = -\sum_{i=1}^{C} y_i \log h_i^{L+1}$$

其中y_i是观测值$\boldsymbol{o}_t$属于第i类的经验概率分布（从训练数据的标注中来），h_i^{L+1}是采用DNN估计的观测值$\boldsymbol{o}_t$属于第i类的概率，表示为：

$$h_i^{L+1} = P(i|\mathbf{o}_t) = \frac{e^{-a_i^{L+1}}}{\sum_j e^{-a_j^{L+1}}}$$

利用反向传播（BP）算法最小化J_{CE}得到更新网络的参数。

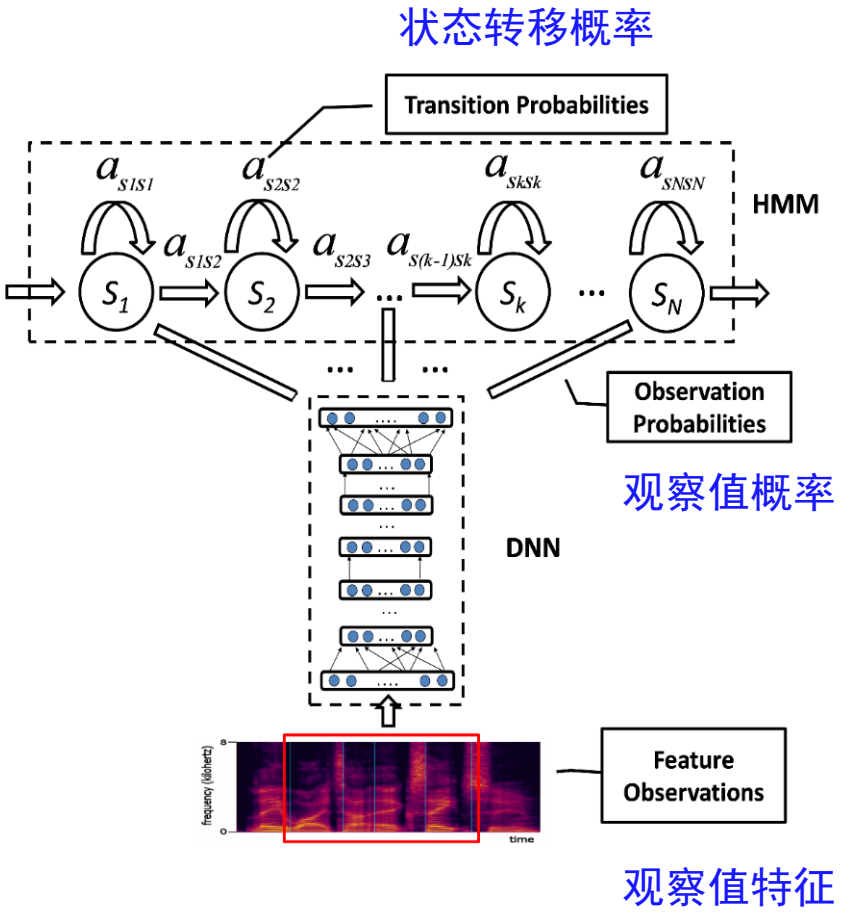
GMM-HMM（高斯混合模型-隐马尔可夫模型）回顾



- L. R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. Proc. IEEE, 77(2):257–286, 1989.

DNN-HMM语音识别算法框架（基本概念）

- 用DNN替换GMM
- 用DNN计算观察值概率
- 用GMM-HMM算法分割语音内容
- 在分割好的语音内容上训练DNN
- 识别性能大幅度改善



DNN-HMM的建模能力（基本概念）

- DNN不需要对声学特征所服从的分布进行假设；
- DNN的输入可以采用连续的拼接帧，更好的利用了上下文信息；
- DNN的训练过程可以采用随机优化算法来实现，因此当训练数据规模较大时也能进行非常高效的训练；
- 在发音模式分类上，DNN这种鉴别式模型也要比GMM这种生成式模型更加适合；
- 无监督预训练。

DNN的输入（基本概念）

- 更底层的滤波器组（Filter Bank，FBK）声学特征。它利用Mel滤波器组在功率谱上进行滤波并计算对数能量，然后采用其规整值作为特征表示；

- 目前，FBK特征比MFCC有效；证明了原始语音频谱对基于DNN的语音识别技术的重要性；

- 不同于传统的GMM采用单帧特征作为输入，DNN会进行拼接帧的操作。会将相邻的若干帧进行拼接得到一个包含更多信息的输入向量，例如每7帧构成一个输入。

DNN的输出（基本概念）

- DNN输出向量的维度对应HMM中状态的个数，通常每一维输出对应一个绑定的triphone状态。
- 训练时，为了得到每一帧语音在DNN上的目标输出值（标注值），需要通过事先训练好的GMM-HMM识别系统在训练语料上进行强制对齐（Force alignment）。
- 基于GMM-HMM通过基于Viterbi算法的强制对齐方法，给每个语音帧打上一个HMM状态标签，然后依据此状态标签，基于DNN训练算法训练一个DNN模型。
- 用DNN替换GMM

DNN的训练（基本概念）

- 用BP算法直接进行训练往往效果不佳，这也是早期基于ANN的混合声学模型未能成功应用的主要原因

- 多层神经网络参数优化是个高维非凸优化问题，常收敛较差的局部解。

- 梯度消失问题：BP算法计算出的误差会从输出层开始向下呈指数衰减，这样计算出各层的梯度也会随着深度的变化而显著下降，导致靠近输出层的隐层训练的比较好，而靠近输入层的隐层几乎不能得到有效训练。

- 解决方法：通过预训练找一个合适的初值。

DNN的训练(续) （基本概念）

- 分为无监督的预训练(Pre-training)和有监督的鉴别性微调(Fine-tuning)。

- Pre-training

  用受限波尔兹曼机(Restricted Boltzmann Machines, RBM)算法逐层训练构建深层置信网络(Deep Belief Networks, DBN)。

- Fine-tuning

  将DBN 作为 DNN 的初始化参数,, 采用标准误差反向传播(Standard Error Back Propagation, BP)算法对DNN进行调整。

DNN的训练(续)（了解）

- 使用海量训练数据进行训练能有效避免过拟合问题；
- Dropout等随机优化算法的提出也极大提高了DNN模型的泛化能力；
- 一些措施的采用也成功地减小了梯度消失问题的影响
  - 整流线性单元（Rectified Linear Units, ReLU）作为激活函数
  - 采用卷积神经网（Convolutional Neural Networks, CNN）这种深度网络结构。