# Hidden Markov Model
# 隐马尔可夫模型

**杨猛，郑伟诗**

https://cse.sysu.edu.cn/content/2970

SUN YAT-SEN University

机器智能与先进计算教育部重点实验室

# 语言模型

- 我在中山大学读
- 我在中山大学读研
- 我在中山大学读研究
- 我在中山大学读研究生
- 我在中山大学读研究生......

# 下面模型存在什么问题

- 滑窗法：例如利用卷积神经网络处理，输入端的数据进行加窗后输入到网络

- n-gram模型：例如在自然语言中，利用统计方法统计n个连续文字出现的频次

# 时间序列Times Series

- 随机过程$\{X_1, X_2, \ldots\}$， $X_i \in \mathcal{X}$
  - $\mathcal{X}$称为状态空间，我们假设$\mathcal{X} = \{1,2, \ldots, N\}$
  - 假设对所有的$i$， $\mathcal{X}$都相同
  - 假设只处理时间序列，即$i$代表时间
  - 随机性
- 目的是希望“过去”对“现在”有帮助
  - 即如果有对$X_1, \ldots, X_{t-1}$的了解，能帮助确定$X_t$
  - Formally， $P(X_t | X_{1:t-1})$ vs. $P(X_t)$

# Markov Property

- Curse of dimensionality
  - $P(X_2|X_1)$需要多少存储空间才能指定？
  - $P(X_3|X_2,X_1)$需要多少存储空间才能指定？
  - $P(X_t|X_{1:t-1})$需要多少存储空间才能指定？
    - $N^t$!
- Markov Property马尔科夫性质
  - 限定：$P(X_t|X_{1:t-1}) = P(X_t|X_{t-1})$，含义是？
  - 无记忆性memoryless
  - 这个假设有效吗？
  - 好处是什么？

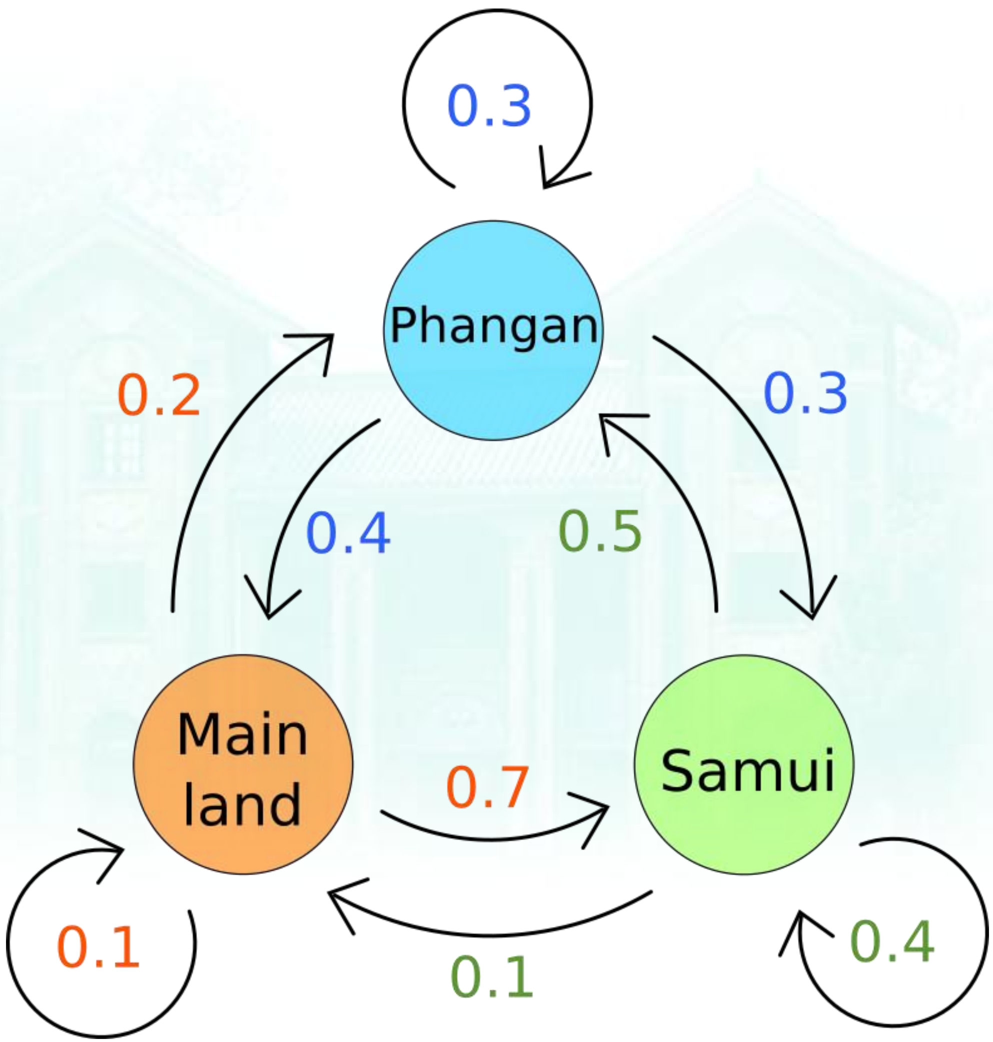
# 人物介绍

- Andrey Markov
- https://en.wikipedia.org/wiki/Markov_chain

| | | |
|---|---|---|
| Computational finance | Speech recognition | Speech synthesis |
| Speech synthesis | Part-of-speech tagging | Single-molecule kinetic analysis |
| Cryptanalysis | Handwriting recognition | Machine translation |

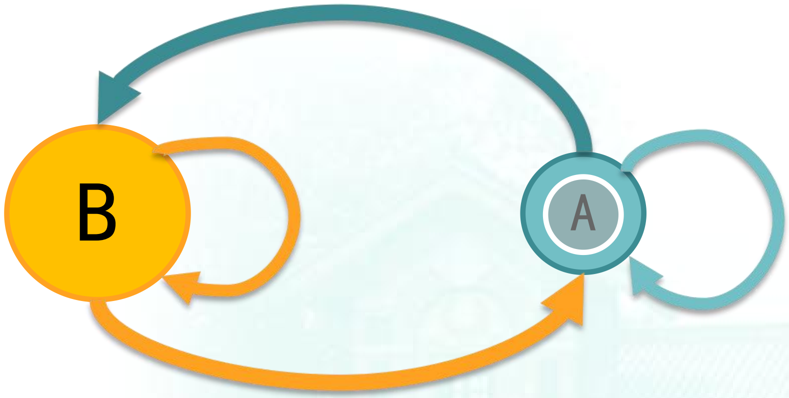
# Markov Chain马尔科夫链

- Markov chain (discrete-time Markov chain or DTMC)



# Markov Chain马尔科夫链

- Markov chain (discrete-time Markov chain or DTMC)



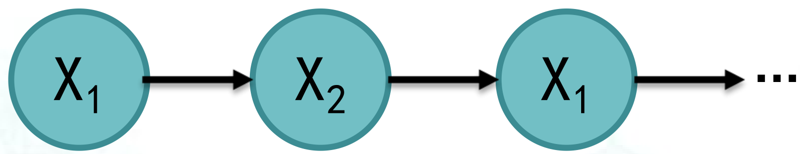

| | A | B |
|---|---|---|
| A | P(A\|A):0.50 | P(B\|A):0.50 |
| B | P(A\|B):0.50 | P(B\|B):0.50 |

状态空间中有A和B两种状态。共4种可能的转换。
1. 在A时，可以过渡到B或留在A。
2. 在B时，可以过渡到A或者留在B。

在图中，从**任意状态到任意状态**的转移概率是0.5。
**人们会通过使用“转移矩阵”来计算转移概率。**状态空间的每个状态都会出现在表格中的一列或者一行中。矩阵的每个单元格指明了从**行状态**转换到**列状态**的概率。
状态空间新增一个状态，矩阵将对应增加一行和一列，向现有的列和行中添加一个单元格。**这意味着当我们向马尔可夫链添加状态时，单元格的数量会呈二次方增长。**因此，**转换矩阵**就起到了很大的作用。

# 可视化和形式化

- 可视化：



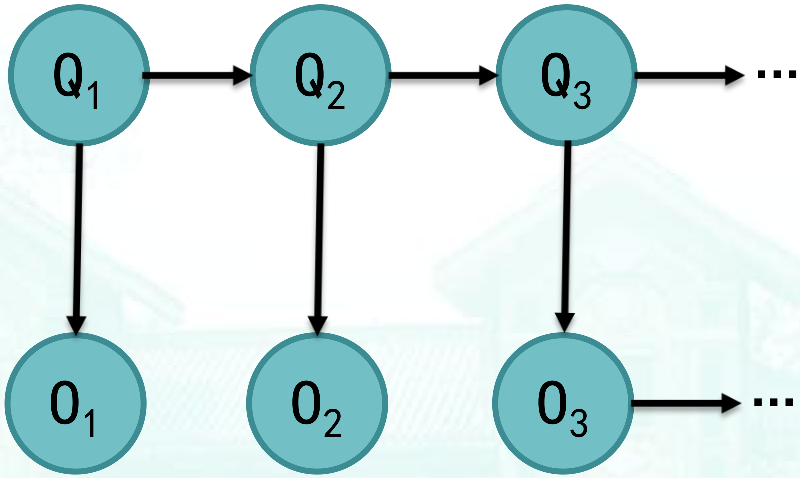

  - 注意填充的变量表示观察值（即随机变量值已知）
- 那么，如何形式化定义DTMC？需要哪些量？
  - 系统初始化Initialization: $P(X_1)$或者$X_1 = x_1$
  - Transition probability: $A = P(X_{t+1}|X_t)$
  - 还需要别的吗？
  - 两次运行结果会一样吗？

# 转移概率矩阵

- Transition probability matrix转移概率矩阵
  - $A$是一个$N \times N$的矩阵
  - $A_{ij} = P(X_t = j | X_{t-1} = i)$
  - 行和为1！

- 如果运行足够久（$t \to \infty$），那么$X_t$的分布在很多情况下将稳定下来，叫Stationary distribution，记为$\boldsymbol{\pi}$
  - $\boldsymbol{\pi} = A\boldsymbol{\pi}$

# 隐马尔可夫模型HMM



# 形式化

- $Q$：隐变量(hidden variable)，不可观测的状态
- $N$：number of states 状态数，$N$个可能的状态为$\{S_1, \ldots, S_N\}$
- $O(o)$：观察值(observations)，$M$个可能的观察值$\{V_1, V_2, \ldots, V_M\}$
- $T$：时间序列的长度
- $\boldsymbol{\pi}$：初始化，$\pi_j = P(Q_1 = S_j)$
- $A$：transition probability matrix，$A_{ij} = P(Q_{t+1} = S_j | Q_t = S_i)$
- $B$：emission probability 发出观察值的概率
  - $b_j(k) = \Pr(O_t = V_k | Q_t = S_j)$
  - 假设$B$不随时间变化，当未知状态为$j$时观察到为$k$的概率
  - 那么，$j$，$k$的取值范围是？$B$的行和是？

# HMM中要解决的问题

- 怎样设计状态？-- 自动学习？
- 怎样设计观察值？--根据问题的特点和实践反复设计
- 与具体问题无关的
  - 指定一个HMM需要的所有参数：$\lambda = (\boldsymbol{\pi}, A, B)$
  - 问题1：Evaluation估值
  - 问题2：Decoding解码
  - 问题3：Learning学习

# Problem 1. Evaluation

- **输入**
  - 一个完全指定的HMM模型，即$\lambda = (\boldsymbol{\pi}, A, B)$已知
  - 一个完全观测的输出序列$O_1 O_2 \cdots O_T$，或$\boldsymbol{O} = O_{1:T}$
- **输出**
  - $P(\boldsymbol{O}|\lambda)$ – 含义是？
  - 在这个模型$\lambda$中观察到特定输出$\boldsymbol{O}$的概率
- **作用是？**
  - 可以看出此模型对该观察序列的成绩score
  - 可以用来从多个模型中选择最适合的模型

# Problem 1. Evaluation

## 假设状态已知

- 已知$\lambda, o_{1:T}$，求$P(o_{1:T}|\lambda)$
- 若假设oracle已告知所有的隐变量的值$q_{1:T}$
  - $\Pr(o_{1:T}|\lambda, q_{1:T}) = \prod_{i=1}^{T} \Pr(o_t|q_t, \lambda) = \prod_{i=1}^{T} b_{q_i}(o_i)$
  - 证明？含义？
  - $\lambda$的存在只是表明概率的大小是基于该模型参数计算的，可以去除而不影响计算

# Problem 1. Evaluation

## 一种naïve的计算方法

- 那么隐变量序列$q_{1:T}$的可能性多大呢？
  - $\Pr(q_{1:T}|\lambda) = \pi_{q_1} A_{q_1 q_2} A_{q_2 q_3} \cdots A_{q_{T-1} q_T}$
  - 含义？
- 用全概率公式对所有可能的$q_{1:T}$求和可以得到$\Pr(o_{1:T}|\lambda)$
  - $\Pr(o_{1:T}|\lambda) = \sum_{all\ Q} \Pr(o_{1:T}|\lambda, q_{1:T}) \Pr(q_{1:T}|\lambda)$，复杂度？
  - $O(T \times N^T)$

# Problem 1. Evaluation

## 那么，如何快速计算？

**动态规划！**

只看最后一步（$t = T$），该如何计算?

1. 最后一步（$t = T$）时一共可能有$N$种状态 ：$q_T = S_1, \ldots, S_N$，其概率$\Pr(o_{1:T-1}, Q_T = S_i \mid \lambda) = ?$
2. 若最后一步状态为$S_i$，那么观察到输出$o_T$的概率是多少？
3. 所求的值是多少？（全概率公式）

$$\Pr(o_{1:T} \mid \lambda) = \sum_{i=1}^{N} \Pr(o_{1:T-1}, Q_T = S_i \mid \lambda) b_i(o_T)$$

只限于最后一步吗？

# Problem 1. Evaluation

## 如何计算$\Pr(o_{1:T-1}, Q_T = S_i|\lambda)$?

- 有$N$种可能，即$T-1$时刻状态为$q_{T-1} = S_j$，$j = 1,2,\ldots,N$，然后通过概率$A_{ji}$转移
- 全概率公式，again!

$$\Pr(o_{1:T-1}, Q_T = S_i|\lambda) = \sum_{j=1}^{N} \Pr(o_{1:T-1}, Q_{T-1} = S_j|\lambda) A_{ji}$$

# Problem 1. Evaluation

## 快速计算小结

- $\Pr(o_{1:T}|\lambda) = \sum_{i=1}^{N} \Pr(o_{1:T-1}, Q_T = S_i|\lambda\,)b_i(o_T) = \sum_{i=1}^{N}\left(b_i(o_T)\sum_{j=1}^{N} \Pr(o_{1:T-1}, Q_{T-1} = S_j|\lambda)A_{ji}\right)$
- 红色部分是什么？
  - 一个规模小一点的相同问题（$T-1$）
  - 但是需要对所有$j$的可能取值计算
  - 正如DTW中一样，可以通过动态规划解决，但是需要解决比原问题更多数目的小规模子问题
  - 但是，复杂的是，目前牵涉两个数值而不是一个：$\Pr(o_{1:T-1}, Q_T = S_i|\lambda\,)$和$P(o_{1:T}|\lambda)$
  - 计算的方向应该是什么？

# Problem 1. Evaluation

## 动态规划算法（前向forward算法）

- $P(o_{1:T}|\lambda) = \sum_{i=1}^{N} \Pr(o_{1:T-1}, Q_T = S_i|\lambda\,)b_i(o_T) = \sum_{i=1}^{N}\left(b_i(o_T)\sum_{j=1}^{N}\Pr(o_{1:T-1}, Q_{T-1} = S_j|\lambda)A_{ji}\right)$
- 定义
  - $\alpha_t(i) = P(o_{1:t}, Q_t = S_i|\lambda)$ －含义是？
  - Initialization: $\alpha_1(i) = \pi_i b_i(o_1), \quad 1 \le i \le N$
  - Induction: For $1 \le t \le T-1$

$$\alpha_{t+1}(i) = \left[\sum_{j=1}^{N}\alpha_t(j)A_{ji}\right]b_i(o_{t+1}), \quad 1 \le i \le N$$

  - Termination (output): $\Pr(o_{1:T}|\lambda) = \sum_{i=1}^{N}\alpha_T(i)$

# Problem 1. Evaluation

## 后向算法backward algorithm

- 定义$\beta_t(i) = \Pr(o_{t+1:T} | Q_t = S_i, \lambda)$
  - 若在时刻$t$状态为$S_i$，将来观测到$o_{t+1:T}$的概率
- 初始化：$\beta_T(i) = 1,\quad 1 \leq i \leq N$
- 反向更新：$t = T-1, T-2, \ldots, 2,1$

$$\beta_t(i) = \sum_{j=1}^{N} A_{ij} b_j(o_{t+1}) \beta_{t+1}(j),\quad 1 \leq i \leq N$$

- 输出：$\beta_1(i) = \Pr(o_{2:T} | Q_1 = S_i, \lambda)$

$$P(o_{1:T}|\lambda) = \sum_{i=1}^{N} \pi_i b_i(o_1) \beta_1(i)$$

# Problem 2：Decoding

- 输入
  - 一个完全指定的HMM模型，即$\lambda = (\boldsymbol{\pi}, A, B)$已知
  - 一个完全观测的输出序列$O_1 O_2 \cdots O_T$，或$\boldsymbol{O} = O_{1:T}$
  - 某个标准criterion
- 输出
  - 一个完全指定的隐变量序列$X_{1:T}$的值
- 作用是?
  - 如，语音识别中状态可能有实际意义（各音节）
    - 唯一吗?
  - 可以用来观察模型结构，优化模型

# Problem 2: Decoding

## 发现“最好”的隐变量值

- 标准1：对于每个时刻，发现其后验概率最大的状态
  - 定义$\gamma_t(i) = \Pr(Q_t = S_i | o_{1:T}, \lambda)$，当观测到输出为$o_{1:T}$时，时刻$t$时隐变量为第$i$个状态的后验概率
  - 那么，对于一个输出序列$o_{1:T}$，选择
$$q_t = \underset{1\le i\le N}{\operatorname{argmax}} \gamma_t(i), \quad t = 1,2, \dots, T$$
  - 可能出现什么问题？
  - 不存在这样的路径$q_{1:T}$

# Problem 2: Decoding

## 怎样计算$\gamma$

- $\alpha_t(i)\beta_t(i) = P(o_{1:T}, Q_t = S_i|\lambda)$
  - 为什么？

- 贝叶斯定理

$$\gamma_t(i) = \Pr(Q_t = S_i|o_{1:T}, \lambda) = \frac{\Pr(o_{1:T}, Q_t = S_i|\lambda)}{\Pr(o_{1:T}|\lambda)} = \frac{\alpha_t(i)\beta_t(i)}{\Pr(o_{1:T}|\lambda)}$$

  - $\Pr(o_{1:T}|\lambda) = \sum_{i=1}^{N} \alpha_t(i)\beta_t(i)$ for any $t$!
  - 三种计算方法计算$P(o_{1:T}|\lambda)$了

- 或者 1）$\gamma_i = \alpha_t(i)\beta_t(i)$ 2） L1 normalize: $\gamma_i \leftarrow \frac{\gamma_i}{\sum_i \gamma_i}$

# Problem 2: Decoding

## 寻找最大概率的路径

- 一共有$N^T$种可能的路径，有些的概率可能为0
  - 比如通过准则1得到的路径
  - 那么，如果寻找所有可能路径里面概率最大的那个呢？

$$q_{1:T} = \underset{Q_{1:T}}{\operatorname{argmax}} \Pr(Q_{1:T} | o_{1:T}, \lambda) = \underset{Q_{1:T}}{\operatorname{argmax}} \Pr(Q_{1:T}, o_{1:T} | \lambda)$$

- Naïve的方法复杂性是$N^T$，有没有更好的方法？
  - Viterbi方法

# Problem 2: Decoding

## Viterbi decoding

- $q_{1:T} = \underset{Q_{1:T}}{\operatorname{argmax}} \Pr(Q_{1:T}, o_{1:T} | \lambda)$
- 定义更多的子问题

$$\delta_t(i) = \max_{Q_{1:t-1}} \Pr(Q_{1:t-1}, Q_t = S_i, o_{1:t} | \lambda)$$

  - 含义：当限定两个条件1）前$t$个时刻的输出为$o_{1:t}$，2）第$t$个时刻的隐状态为第$i$个状态的时候，最佳路径所能取得的最大概率
  - 怎么取得$q_t$?
    - 用另外一个变量$\psi_t(i)$做记录
  - 怎么从$t$进展到$t+1$?

# Problem 2: Decoding

## 两个步骤

- 从$t$进展到$t+1$
  - o $\delta_{t+1}(i) = \max_j \left(\left[\delta_t(j)A_{ji}\right]b_i(o_{t+1})\right)$
  - o $\delta_{t+1}(i)$是概率，如果只需要发现概率最大那个状态，$b_i(o_{t+1})$?
- 所以在时刻$t+1$，需要用另外一个变量$\psi_t(i)$记录最大概率的路径在时刻$t$是哪一个状态
  - o $\psi_{t+1}(i) = \operatorname{argmax}_{1\le j\le N}\left(\left[\delta_t(j)A_{ji}\right]\right)$

# Problem 2: Decoding

## Viterbi算法

- 初始化：$\delta_1(i) = \pi_i b_i(o_1)$， $\psi_1(i) = 0$， $1 \le i \le N$
- 递归：$2 \le t \le T$， $1 \le i \le N$

$$\delta_t(i) = \max_{1 \le j \le N} \left( \left[ \delta_{t-1}(j) A_{ji} \right] b_i(o_t) \right)$$

$$\psi_t(i) = \operatorname*{argmax}_{1 \le j \le N} \left( \left[ \delta_{t-1}(j) A_{ji} \right] \right)$$

- 输出：
  - 最大概率：$P^* = \max_{1 \le i \le n} \delta_T(i)$
  - 时刻$T$的最佳路径变量：$q_T^* = \operatorname*{argmax}_{1 \le i \le N} (\delta_T(i))$
  - 时刻$T-1, T-2, \ldots, 2{,}1$的最佳路径变量：$q_t^* = \psi_{t+1}(q_{t+1}^*)$

# Problem 2：Decoding

## 分析

- 问题1的动态规划$\alpha_{t+1}(i) = \sum_{j=1}^{N} \alpha_t(j) A_{ji}$
- 问题2的动态规划$\delta_t(i) = \max_j \left( \left[ \delta_{t-1}(j) A_{ji} \right] b_i(o_t) \right)$
- 最重要的操作分别是sum-product和max-product
  - 其复杂性均为$N^2 T$
  - 和naïve方法的$TN^T$比较，极其巨大的速度提高

# Problem 3：Learning

## 学习系统的参数

- 发现$\lambda=(A,B,\boldsymbol{\pi})$，使得对于固定的$N$，$T$，和观察值$\boldsymbol{O}$，似然（likelihood） $P(\boldsymbol{O}|\lambda)$最大
  - 目前没有方法能发现全局最优的解
  - 常用的方法是Baum-Welch算法，发现一个局部最优的解

# Problem 3: Learning

- 输入
  - 网络结构，状态数、输出数
  - 若干观测序列$\{\boldsymbol{O}\}$
- 输出
  - 最优的参数$\lambda = (\boldsymbol{\pi}, A, B)$使得$P(\{\boldsymbol{O}\}|\lambda)$最大
- 作用
  - 显而易见
  - 最重要的问题
  - 有时候一个足够长的观测序列就够了

$$\xi_t(i,j) = Pr(Qt = Si, Qt + 1 = Sj|o1{:}T, \lambda) = \frac{\alpha_t(i)A_{ij}b_j(o_{t+1})\beta_{t+1}(j)}{\mathrm{Pr(o1{:}T}|\lambda)}$$

# Baum-Welch算法

- Baum-Welch算法
- 1：初始化参数$\lambda^{(1)}$（例如随机地）
- 2：$r \leftarrow 1$
- 3：**while** 似然尚未收敛 **do**
- 4：　　对所有 $t(1 \leq t \leq T)$ 和所有 $i(1 \leq i \leq N)$，使用前向过程基于 $\lambda^{(r)}$计算 $\alpha_t(i)$
- 5：　　对所有 $t(1 \leq t \leq T)$ 和所有 $i(1 \leq i \leq N)$，使用后向过程基于$\lambda^{(r)}$计算 $\beta_t(i)$
- 6：　　对所有 $t(1 \leq t \leq T)$ 和所有 $i(1 \leq i \leq N)$，根据公式计算 $\gamma_t(i)$
- 7：　　对所有 $t(1 \leq t \leq T-1)$ 和所有$i,j\ (1 \leq i,j \leq N)$，根据表 12.1中的公式计算$\xi_t(i,j)$
- 8：　　更新参数为$\lambda^{(r+1)}$
- $$\pi_i^{(r+1)} = \gamma_1(i) \qquad 1 \leq i \leq N$$
- $$A_{ij}^{(r+1)} = \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)} \qquad 1 \leq i,j \leq N$$
- 
- $$b_j^{(r+1)}(k) = \frac{\sum_{t=1}^{T} [[o_t = k]] \gamma_t(j)}{\sum_{t=1}^{T} \gamma_t(j)} \qquad 1 \leq j \leq N \quad 1 \leq k \leq M$$
- 9：　　$r \leftarrow r + 1$
- 10：**end while**

# 怎样在模式识别中发挥更大作用

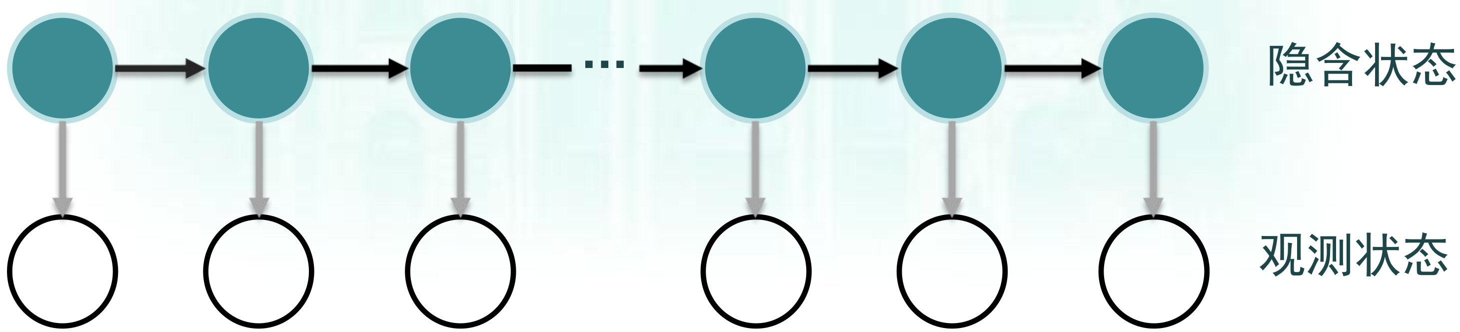
- 语音识别

语音识别的目的是将声音信号映射为文字信息，如何实现这种映射？

**分帧：**声音实际上是一种波，要对声音进行分析，需要对声音分帧，也就是把声音切开成一小段一小段，每小段称为一帧。分帧操作一般不是简单的切开，而是使用移动窗函数来实现，帧与帧之间一般有交叠。

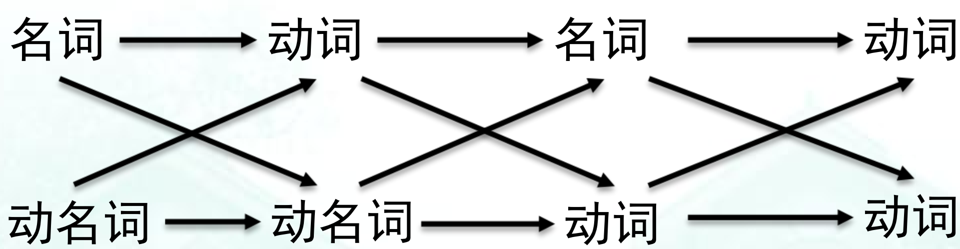
# 怎样在模式识别中发挥更大作用

- HMM用于NLP词性标注
- 对句子“教授喜欢画画”进行词性标注，分词之后的结果可能是“教授/喜欢/画/画”，“教授”词性可以是名词和动名词，“喜欢”词性可以是动词和动名词，“画”词性可以是名词和动词，画成图可以表示为：



# 怎样在模式识别中发挥更大作用

## “教授喜欢画画”




- 隐马是个生成模型，生成的过程是先生成状态节点，根据状态节点再生成观测节点。
- 首先生成“**教授**”词性是“**名词**”，然后生成词“**教授**”；
- 根据“**教授**”的词性节点“**名词**”生成“**喜欢**”的词性节点“**动词**”，然后生成词“**喜欢**”；
- 根据“**喜欢**”的词性“**动词**”生成“**画**”的词性“**动词**”，然后生成词“**画**”。

# 基于HMM的中文词性标注

- 数据处理：收集带有词性标注的中文语料（如1998人民日报词性标注语料库）
- 模型训练：根据数据估计HMM的模型参数：全部的词性集合、全部的词集合、初始概率向量、词性到词性的转移矩阵、词性到词的转移矩阵。可直接采用频率估计概率的方法，对于模型参数中大量的0，可采用拉普拉斯平滑处理。
- 模型预测：基于维特比算法进行解码，获得中文句子的词性标注。