

第3章：参数估计(续)

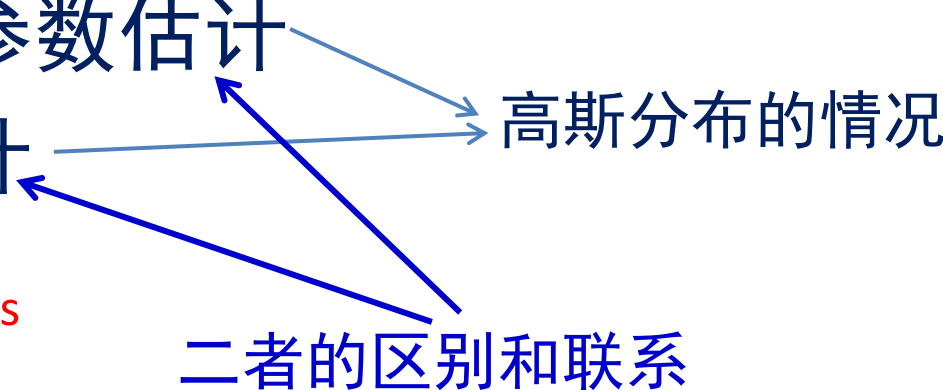
刘成林(liucl@nlpr.ia.ac.cn)

2015年10月14日

助教：杨学行(xhyang@nlpr.ia.ac.cn)

吴一超(yichao.wu@nlpr.ia.ac.cn)

上次课主要内容回顾

- 离散变量贝叶斯决策
 - 复合模式分类
 - 最大似然参数估计
 - 贝叶斯估计
- 高斯分布的情况
- Parameter space vs feature space
- 二者的区别和联系
- 



提 纲

- 第3章
 - 特征维数问题
 - 期望最大法
 - 隐马尔可夫模型

特征维数问题

- 统计模式分类
 - 特征空间划分
 - 贝叶斯决策：最小风险规则，MAP
- 增加特征有什么好处
 - 判别性：类别间有差异的特征有助于分类
- 带来什么问题
 - 计算
 - 存储
 - 泛化性能，Overfitting

分类错误率与特征的关系

- 二类高斯分布

- $p(\mathbf{x}|\omega_j) \sim N(\mu_j, \Sigma)$, $j=1,2$, 等协方差矩阵
- Bayes error rate

$$P(e) = \frac{1}{\sqrt{2\pi}} \int_{r/2}^{\infty} e^{-u^2/2} du \quad r^2 = (\mu_1 - \mu_2)^t \Sigma^{-1} (\mu_1 - \mu_2)$$

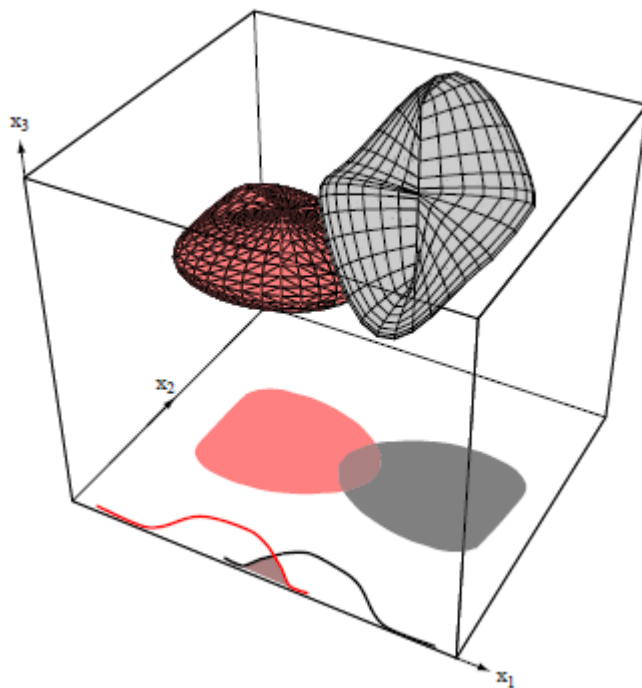
↑

$$\begin{aligned} P(\text{error}) &= P(\mathbf{x} \in \mathcal{R}_2, \omega_1) + P(\mathbf{x} \in \mathcal{R}_1, \omega_2) \\ &= P(\mathbf{x} \in \mathcal{R}_2 | \omega_1) P(\omega_1) + P(\mathbf{x} \in \mathcal{R}_1 | \omega_2) P(\omega_2) \\ &= \int_{\mathcal{R}_2} p(\mathbf{x} | \omega_1) P(\omega_1) d\mathbf{x} + \int_{\mathcal{R}_1} p(\mathbf{x} | \omega_2) P(\omega_2) d\mathbf{x}. \end{aligned}$$

- Conditionally independent case $\Sigma = \text{diag}(\sigma_1^2, \dots, \sigma_d^2)$
 - 每一维二类均值之间距离反映区分度，决定错误率
 - 特征增加有助于减小错误率(r^2 增大)

$$r^2 = \sum_{i=1}^d \left(\frac{\mu_{i1} - \mu_{i2}}{\sigma_i} \right)^2$$

- 特征维数决定可分性的例子
 - 3D空间完全可分
 - 2D和1D投影空间有重叠



然而，增加特征也可能导致分类性能更差，因为有模型估计误差(wrong model)

计算复杂度

- 最大似然估计

- 高斯分布， d 维特征， n 个样本
- 参数估计的复杂度，主要由 Σ 决定

$$g(\mathbf{x}) = -\frac{1}{2}(\mathbf{x} - \hat{\boldsymbol{\mu}})^t \overbrace{\hat{\boldsymbol{\Sigma}}^{-1}}^{O(nd^2)} (\mathbf{x} - \hat{\boldsymbol{\mu}}) - \underbrace{\frac{d}{2} \ln 2\pi}_{O(1)} - \underbrace{\frac{1}{2} \ln |\hat{\boldsymbol{\Sigma}}|}_{O(d^2n)} + \underbrace{\ln P(\omega)}_{O(n)}$$
$$\hat{\boldsymbol{\Sigma}} = \frac{1}{n} \sum_{k=1}^n (\mathbf{x}_k - \mathbf{m}_n)(\mathbf{x}_k - \mathbf{m}_n)^t$$

- 参数存储复杂度

$$c(d + d(d+1)/2)$$

- 分类复杂度？

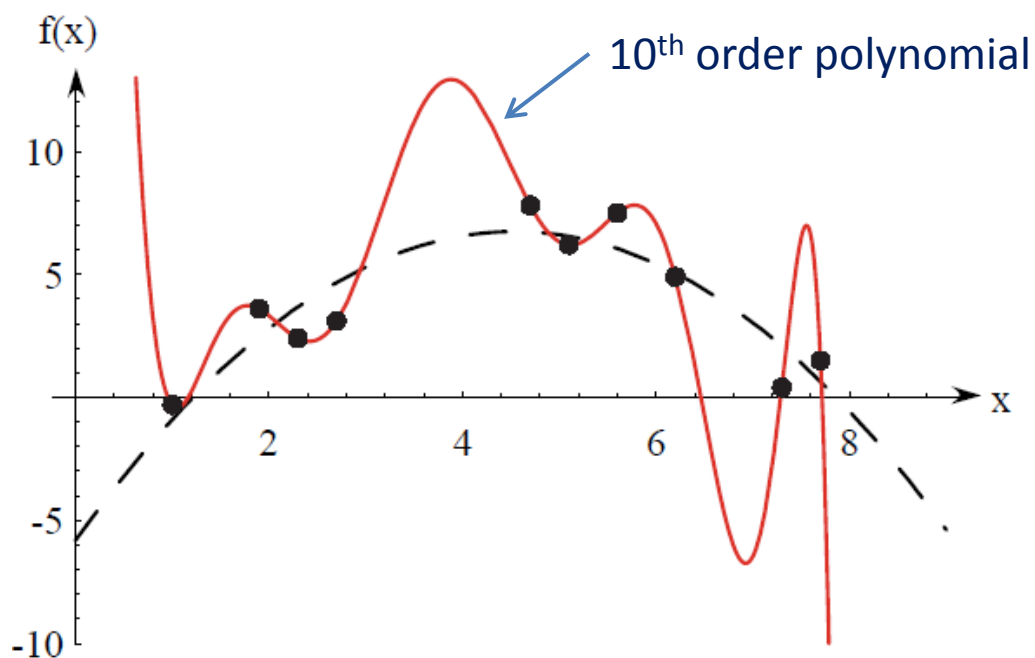
过拟合(Overfitting)

- Overfitting
 - 特征维数高、训练样本少导致模型参数估计不准确
 - 比如协方差矩阵需要样本数在 d 以上
- 克服办法
 - 特征降维：特征提取(变换)、特征选择
 - 参数共享/平滑
 - 共享协方差矩阵 Σ_0
 - Shrinkage (a.k.a. Regularized Discriminant Analysis)

$$\Sigma_i(\alpha) = \frac{(1 - \alpha)n_i \Sigma_i + \alpha n \Sigma}{(1 - \alpha)n_i + \alpha n}$$

$$\Sigma(\beta) = (1 - \beta)\Sigma + \beta\mathbf{I}$$

- 过拟合的例子



$$f(x) = ax^2 + bx + c + \epsilon \text{ where } p(\epsilon) \sim N(0, \sigma^2)$$

期望-最大法(EM)

- 数据缺失情况下的参数估计

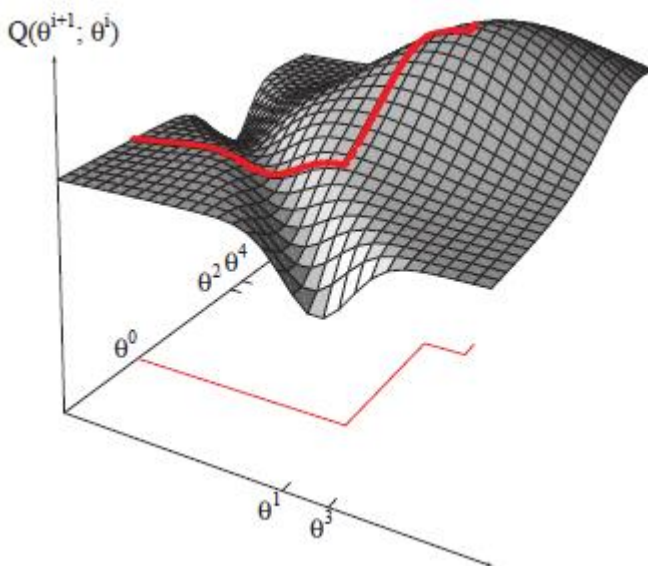
- Good features, missing/bad features $\mathbf{x}_k = \{\mathbf{x}_{kg}, \mathbf{x}_{kb}\}$

$$\mathcal{D} = \{\mathbf{x}_1, \dots, \mathbf{x}_n\} = \mathcal{D}_g \cup \mathcal{D}_b$$

- 已知参数 θ^i 情况下估计新参数 θ

- 对缺失数据求期望(marginalize)

$$\max Q(\theta; \theta^i) = \mathcal{E}_{\mathcal{D}_b} [\ln p(\mathcal{D}_g, \mathcal{D}_b; \theta) | \mathcal{D}_g; \theta^i]$$



- Expectation-Maximization (EM)

Algorithm 1 (Expectation-Maximization)

```
1 begin initialize  $\theta^0, T, i = 0$   
2       do  $i \leftarrow i + 1$   
3           E step : compute  $Q(\theta; \theta^i)$   
4           M step :  $\theta^{i+1} \leftarrow \arg \max_{\theta} Q(\theta; \theta^i)$   
5           until  $Q(\theta^{i+1}; \theta^i) - Q(\theta^i; \theta^{i-1}) \leq T$   
6       return  $\hat{\theta} \leftarrow \theta^{i+1}$   
7 end
```

The EM algorithm guarantees that the log-likelihood of good data increases monotonically.

- Example: EM for a 2D Gaussian

$$\mathcal{D} = \{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \mathbf{x}_4\} = \left\{ \begin{pmatrix} 0 \\ 2 \end{pmatrix}, \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \begin{pmatrix} 2 \\ 2 \end{pmatrix}, \begin{pmatrix} * \\ 4 \end{pmatrix} \right\}$$

parameters $\theta = \begin{pmatrix} \mu_1 \\ \mu_2 \\ \sigma_1^2 \\ \sigma_2^2 \end{pmatrix}$ initially $\theta^0 = \begin{pmatrix} 0 \\ 0 \\ 1 \\ 1 \end{pmatrix}$

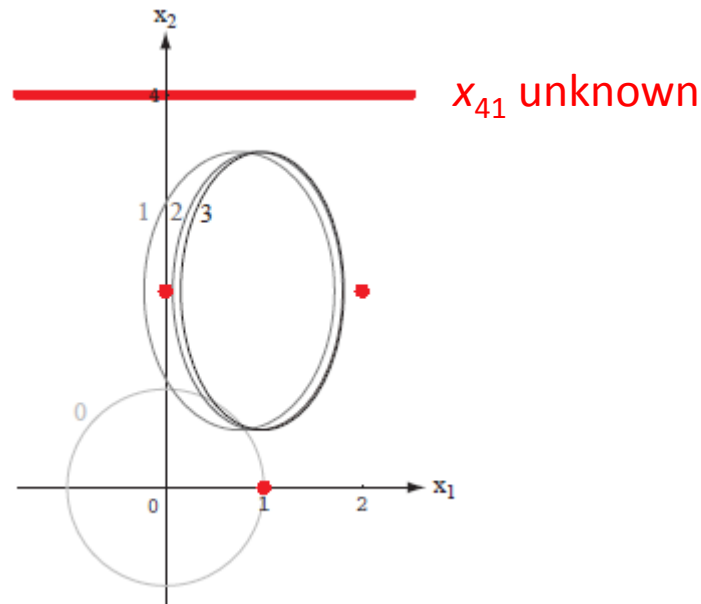
$$\begin{aligned} Q(\theta; \theta^0) &= \mathcal{E}_{x_{41}} [\ln p(\mathbf{x}_g, \mathbf{x}_b; \theta | \theta^0; \mathcal{D}_g)] \\ &= \int_{-\infty}^{\infty} \left[\sum_{k=1}^3 \ln p(\mathbf{x}_k | \theta) + \ln p(\mathbf{x}_4 | \theta) \right] p(x_{41} | \theta^0; x_{42} = 4) dx_{41} \\ &= \sum_{k=1}^3 [\ln p(\mathbf{x}_k | \theta)] + \int_{-\infty}^{\infty} \ln p \left(\begin{pmatrix} x_{41} \\ 4 \end{pmatrix} \middle| \theta \right) \underbrace{\frac{p \left(\begin{pmatrix} x_{41} \\ 4 \end{pmatrix} | \theta^0 \right)}{\left(\int_{-\infty}^{\infty} p \left(\begin{pmatrix} x'_{41} \\ 4 \end{pmatrix} | \theta^0 \right) dx'_{41} \right)}}_{\equiv K} dx_{41} \\ Q(\theta; \theta^0) &= \sum_{k=1}^3 [\ln p(\mathbf{x}_k | \theta)] + \frac{1}{K} \int_{-\infty}^{\infty} \ln p \left(\begin{pmatrix} x_{41} \\ 4 \end{pmatrix} \middle| \theta \right) \frac{1}{2\pi \begin{vmatrix} 1 & 0 \\ 0 & 1 \end{vmatrix}} \exp \left[-\frac{1}{2} (x_{41}^2 + 4^2) \right] dx_{41} \\ &= \sum_{k=1}^3 [\ln p(\mathbf{x}_k | \theta)] - \frac{1 + \mu_1^2}{2\sigma_1^2} - \frac{(4 - \mu_2)^2}{2\sigma_2^2} - \ln (2\pi\sigma_1\sigma_2). \end{aligned}$$



$$\max \sum_{k=1}^3 [\ln p(\mathbf{x}_k | \boldsymbol{\theta})] - \frac{1 + \mu_1^2}{2\sigma_1^2} - \frac{(4 - \mu_2)^2}{2\sigma_2^2} - \ln (2\pi\sigma_1\sigma_2)$$

$$\theta^1 = \begin{pmatrix} 0.75 \\ 2.0 \\ 0.938 \\ 2.0 \end{pmatrix}$$

After 3 iterations $\mu = \begin{pmatrix} 1.0 \\ 2.0 \end{pmatrix}$, and $\Sigma = \begin{pmatrix} 0.667 & 0 \\ 0 & 2.0 \end{pmatrix}$



- EM for Gaussian mixture

- 参数型概率密度函数，可以表示复杂的分布

$$p(\mathbf{x}) = \sum_{k=1}^K \pi_k p(\mathbf{x} | \theta_k)$$

$$\text{subject to } \sum_{k=1}^K \pi_k = 1$$

- Gaussian component

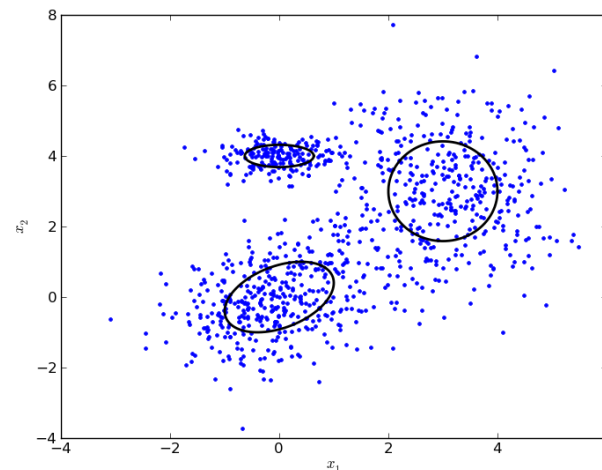
$$p(\mathbf{x} | \theta_k) = \mathcal{N}(\mathbf{x} | \mu_k, \Sigma_k)$$

- 参数估计：Maximum Likelihood (ML)

$$\max LL = \log \prod_{n=1}^N p(\mathbf{x}_n) = \sum_{n=1}^N \log \sum_{k=1}^K \pi_k p(\mathbf{x}_n | \theta_k)$$

$$\nabla_{\pi_k} LL = 0, \quad \nabla_{\mu_k} LL = 0, \quad \nabla_{\Sigma_k} LL = 0$$

不能解析求解



- EM Algorithm for Gaussian mixture

- Incomplete data \mathbf{X} , complete data $\{\mathbf{X}, \mathbf{Z}\}$

$$z_{nk} \in \{0, 1\}, \quad k = 1, \dots, K$$

- **Expectation** of complete data log-likelihood

$$Q(\Theta, \Theta^{old}) = \sum_{\mathbf{Z}} \log p(\mathbf{X}, \mathbf{Z} | \Theta) p(\mathbf{Z} | \mathbf{X}, \Theta^{old})$$

1. Choose an initial set of parameters for Θ^{old}

2. E-step: Evaluate $p(\mathbf{Z} | \mathbf{X}, \Theta^{old})$

3. M-step: Update parameters

$$\Theta^{new} = \arg \max_{\Theta} Q(\Theta, \Theta^{old})$$

4. If convergence condition is not satisfied

$$\Theta^{old} \leftarrow \Theta^{new}$$

5. Return to E-step



- EM Algorithm for Gaussian mixture

E-step
$$p(\mathbf{X}, \mathbf{Z} | \Theta) = \prod_{n=1}^N \prod_{k=1}^K \pi_k^{z_{nk}} \mathcal{N}(\mathbf{x}_n | \mu_k, \Sigma_k)^{z_{nk}}$$

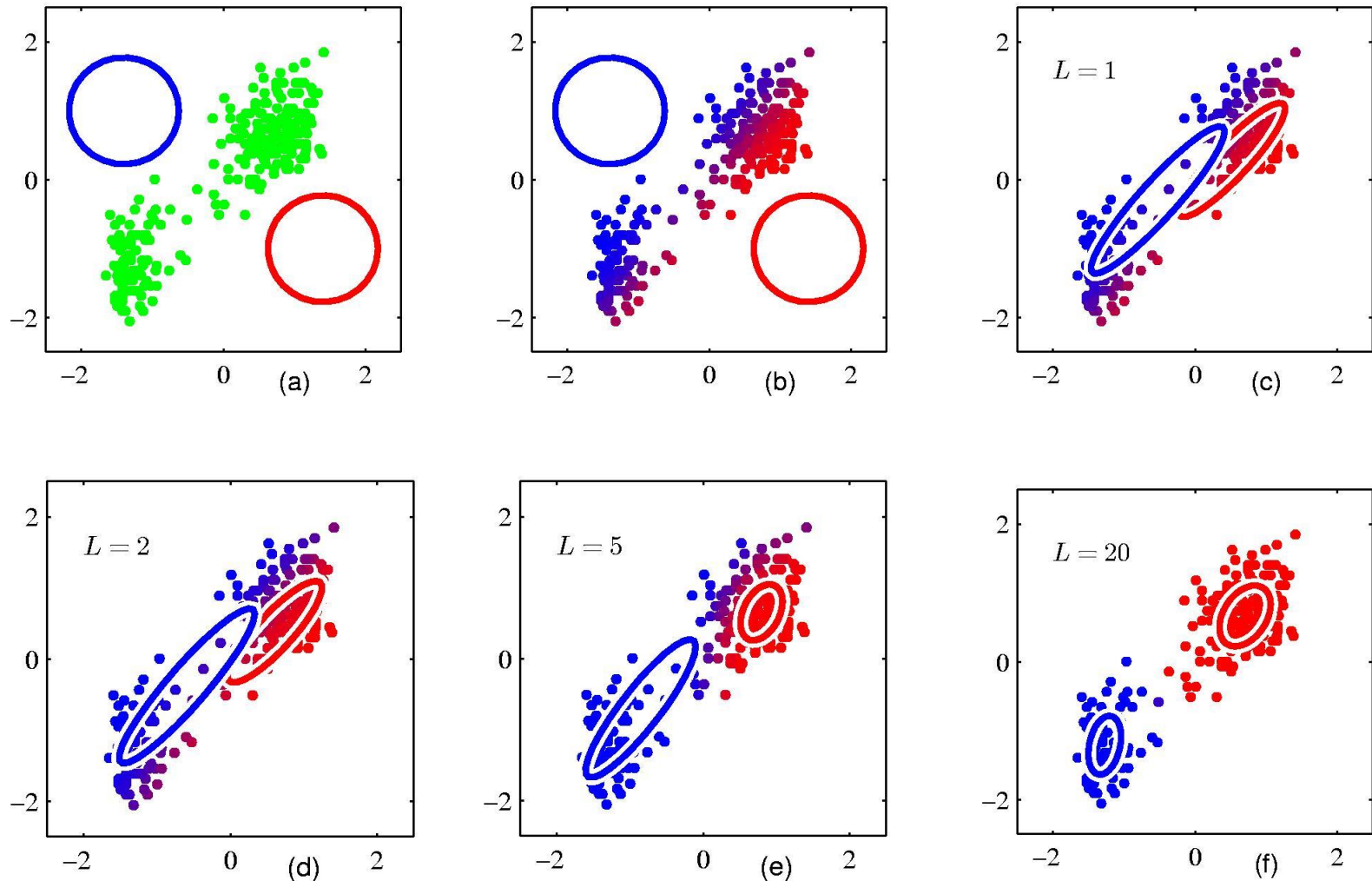
$$Q(\Theta, \Theta^{old}) = E_{\mathbf{Z}}[\log p(\mathbf{X}, \mathbf{Z} | \Theta)] = \sum_{n=1}^N \sum_{k=1}^K \gamma(z_{nk}) \{ \log \pi_k + \log \mathcal{N}(\mathbf{x}_n | \mu_k, \Sigma_k) \}$$

$$\gamma(z_{nk}) = P(z_{nk} = 1 | \mathbf{x}_n) = \frac{\pi_k N(\mathbf{x}_n | \mu_k, \Sigma_k)}{\sum_{j=1}^K \pi_j N(\mathbf{x}_n | \mu_j, \Sigma_j)}$$

M-step $\nabla_{\pi_k} Q = 0, \quad \nabla_{\mu_k} Q = 0, \quad \nabla_{\Sigma_k} Q = 0$

$$\left\{ \begin{array}{l} \mu_k^{new} = \frac{1}{N_k} \sum_{n=1}^N \gamma(z_{nk}) \mathbf{x}_n \\ \Sigma_k^{new} = \frac{1}{N_k} \sum_{n=1}^N \gamma(z_{nk}) (\mathbf{x}_n - \mu_k^{new})(\mathbf{x}_n - \mu_k^{new})^T \\ \pi_k^{new} = \frac{N_k}{N} \end{array} \right. \quad N_k = \sum_{n=1}^N \gamma(z_{nk})$$

An example, from (C.M. Bishop, *Pattern Recognition and Machine Learning*, 2006. Figure 9.8)



Break

隐马尔可夫模型

- Sequential (Temporal) Pattern

- Variable length
- Distortion
- Ambiguous boundary between primitives (symbols)



Independence

- Bayesian Classification

- Sequence of patterns (observations) $\mathbf{O} = O_1 O_2 \cdots O_T$
- Sequence class (states) $\mathbf{q} = q_1 q_2 \cdots q_T$
- Posterior probability

$$P(\mathbf{q} | \mathbf{O}) = \frac{p(\mathbf{O} | \mathbf{q})P(\mathbf{q})}{p(\mathbf{O})}$$

- Hidden Markov Model (HMM)

- Model $p(\mathbf{O} | \mathbf{q}), p(\mathbf{O}, \mathbf{q})$

Markov Chain

- Sequence of States (classes)

$$P(q_1 q_2 \cdots q_T) = P(q_1) P(q_2 | q_1) P(q_3 | q_1 q_2) \cdots P(q_T | q_1 \cdots q_{T-1})$$

$$q_t \in \{S_1, \dots, S_N\}$$

- First-Order Markov

$$P(q_t = S_j | q_{t-1} = S_i, q_{t-2} = S_k, \dots) = P(q_t = S_j | q_{t-1} = S_i)$$

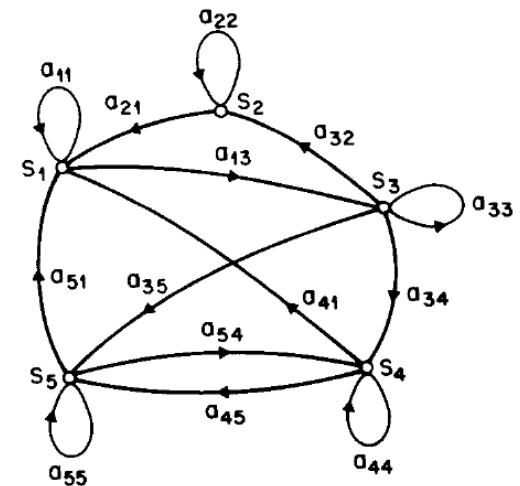
$$P(q_1 q_2 \cdots q_T) =$$

$$P(q_1) P(q_2 | q_1) P(q_3 | q_2) \cdots P(q_T | q_{T-1})$$

State transition probabilities

$$a_{ij} = P(q_t = S_j | q_{t-1} = S_i), \quad 1 \leq i, j \leq N$$

$$\sum_{j=1}^N a_{ij} = 1$$



– State duration (self-transition)

$$O = \{ \underset{1}{S_i}, \underset{2}{S_i}, \underset{3}{S_i}, \dots, \underset{d}{S_i}, \underset{d+1}{S_j} \neq S_i \}$$

$$P(O | \text{Model}, q_1 = S_i) = (a_{ii})^{d-1} (1 - a_{ii}) = p_i(d)$$

– Expected duration of specific state

$$\begin{aligned} \bar{d}_i &= \sum_{d=1}^{\infty} d p_i(d) \\ &= \sum_{d=1}^{\infty} d (a_{ii})^{d-1} (1 - a_{ii}) = \frac{1}{1 - a_{ii}} \end{aligned}$$

• Example: Transition of Weather

State 1: rain or (snow)	$\mathbf{A} = \{a_{ij}\} = \begin{bmatrix} 0.4 & 0.3 & 0.3 \\ 0.2 & 0.6 & 0.2 \\ 0.1 & 0.1 & 0.8 \end{bmatrix}$
State 2: cloudy	
State 3: sunny	

Expected number of days for sunny and cloudy?

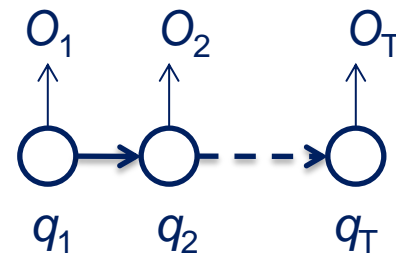
Hidden Markov Model (HMM)

- Markov Chain: States are Observable
- Hidden States: An Example
 - Imagine you are in a un-windowed room, cannot see the weather outside. Instead, you can guess the weather from the temperature and humidity in room
 - Observations: temperature, humidity
 - Hidden states: weather
 - Hidden Markov Model (HMM): Doubly embedded stochastic process

$$P(O_1, O_2, \dots, O_T)$$

$$P(q_1, q_2, \dots, q_T)$$

- Infer states from observations



$$q_i \in \{S_1, S_2, \dots, S_N\}$$

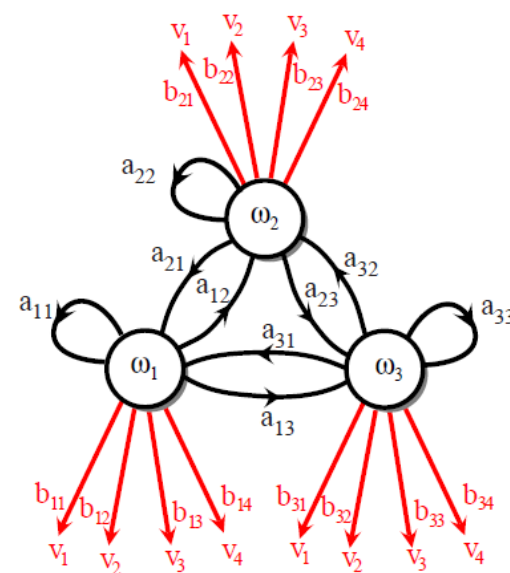
- Elements of an HMM $\lambda = (\mathbf{A}, \mathbf{B}, \boldsymbol{\pi})$

- N : number of states in the model, $S = \{S_1, S_2, \dots, S_N\}$
- M : number of observation symbols, $V = \{v_1, v_2, \dots, v_M\}$
- State transition probability distribution $A = \{a_{ij}\}$

$$a_{ij} = P(q_{t+1} = S_j \mid q_t = S_i), \quad 1 \leq i, j \leq N$$

- Observation symbol (emission) probability distribution $B = \{b_j(k)\}$ $b_j(k) = P(v_k \text{ at } t \mid q_t = S_j), \quad 1 \leq j \leq N, 1 \leq k \leq M$
- Initial state distribution $\boldsymbol{\pi} = \{\pi_i\}$

$$\pi_i = P(q_1 = S_i), \quad 1 \leq i \leq N$$



- Three Basic Problems of HMM

- Problem 1 (Evaluation):

- How to efficiently compute the probability of observation sequence $P(O|\lambda)$

- Problem 2 (Decoding):

- How to choose the best state sequence responding to an observation sequence

- Problem 3 (Training):

- How to estimate the model parameters

Evaluation Problem

- Given model $\lambda = (\mathbf{A}, \mathbf{B}, \pi)$ and observation sequence $O = O_1 O_2 \dots O_T$, compute $P(O | \lambda)$
 - Direct computation

$$\begin{aligned} P(O | \lambda) &= \sum_{\text{all } Q} P(O | Q, \lambda) P(Q | \lambda) \\ &= \sum_{q_1, q_2, \dots, q_T} \pi_{q_1} b_{q_1}(O_1) a_{q_1 q_2} b_{q_2}(O_2) \cdots a_{q_{T-1} q_T} b_{q_T}(O_T) \end{aligned}$$

Conditional independence

$$\begin{aligned} P(O | Q, \lambda) &= \prod_{t=1}^T P(O_t | q_t, \lambda) \\ &= b_{q_1}(O_1) b_{q_2}(O_2) \cdots b_{q_T}(O_T) \end{aligned}$$

$$P(Q | \lambda) = \pi_{q_1} a_{q_1 q_2} a_{q_2 q_3} \cdots a_{q_{T-1} q_T}$$

Markov chain of states

- Complexity: $O(2TN^T)$!

- Evaluation: Forward Procedure

- Define **forward variable**

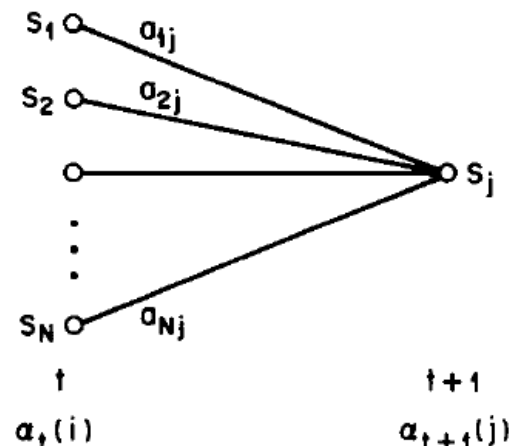
$$\alpha_t(i) = P(O_1 O_2 \cdots O_t, q_t = S_i | \lambda)$$

- Initialization $\alpha_1(i) = \pi_i b_i(O_1), 1 \leq i \leq N$

- Induction

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

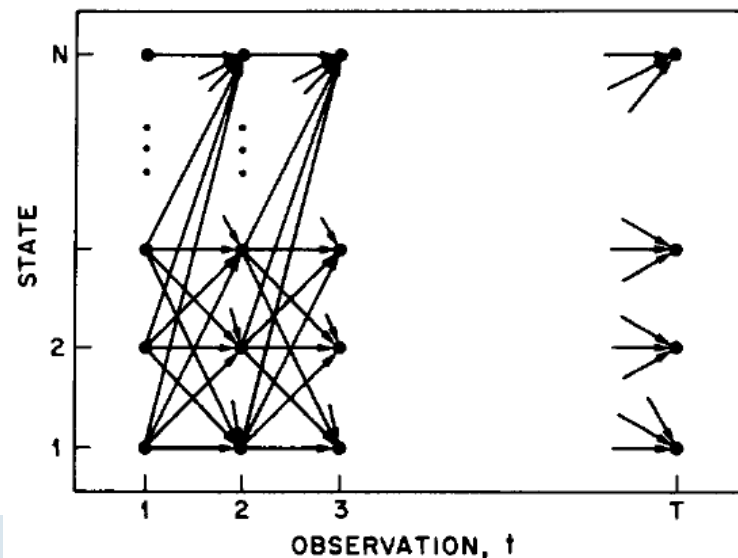
$$1 \leq t \leq T-1, \quad 1 \leq j \leq N.$$



- Termination

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

- Complexity: $O(TN^2)$



- Evaluation: Backward Procedure

- Define **backward variable**

$$\beta_t(i) = P(O_{t+1}, \dots, O_T \mid q_t = S_i, \lambda)$$

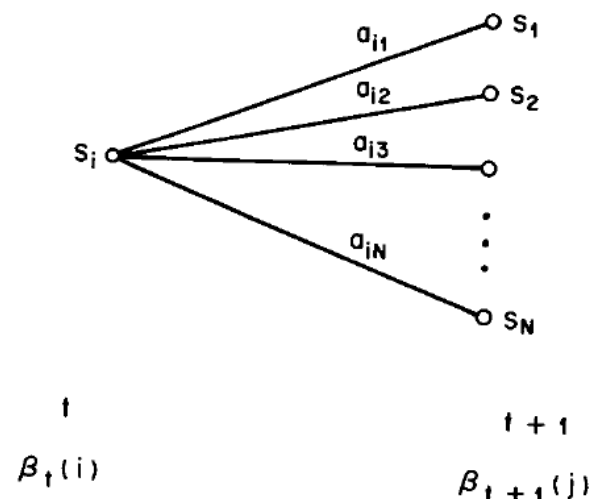
- Initialization

$$\beta_T(i) = 1, \quad 1 \leq i \leq N$$

- Induction

$$\beta_t(i) = \sum_{j=1}^N a_{ij} b_j(O_{t+1}) \beta_{t+1}(j),$$

$$1 \leq t \leq T-1, \quad 1 \leq i \leq N$$



- Termination

$$P(O \mid \lambda) = \sum_{i=1}^N \pi_i b_i(O_1) \beta_1(i) = \sum_{i=1}^N \alpha_1(i) \beta_1(i)$$

- Complexity?

Decoding Problem

- This is **Pattern Recognition**
- Optimal Sequence of States

$$\max_{q_1 q_2 \cdots q_T} P(q_1 q_2 \cdots q_T | O, \lambda) = \max_{q_1 q_2 \cdots q_T} P(q_1 q_2 \cdots q_T, O | \lambda)$$

- **Viterbi Algorithm (DP)**

- Define variable

$$\delta_t(i) = \max_{q_1, q_2, \dots, q_{t-1}} P(q_1 q_2 \cdots q_t = i, O_1 O_2 \cdots O_t / \lambda)$$

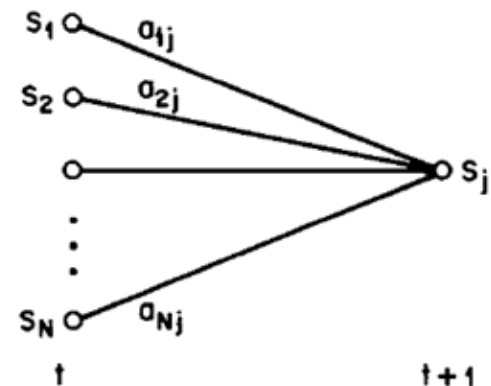
- DP

$$\delta_{t+1}(j) = \left[\max_i \delta_t(i) a_{ij} \right] \cdot b_j(O_{t+1})$$

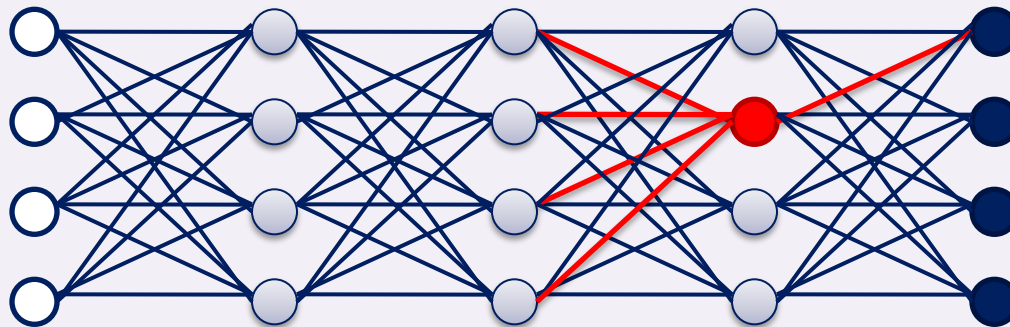
- Initialization

$$\delta_1(i) = \pi_i b_i(O_1), \quad 1 \leq i \leq N$$

$$\psi_1(i) = 0.$$



- Appendix: Dynamic Programming (DP) Principle (Bellman Principle of Optimality)
 - The best path through a particular, intermediate place is the best way from start to it, followed by the best way from it to the goal.
 - Implication: from multiple ways reaching an intermediate place, only retain the best one
 - Often used in sequence matching and HMMs



$$\delta_t(i) = \max_{q_1, q_2, \dots, q_{t-1}} P(q_1 q_2 \cdots q_t = i, O_1 O_2 \cdots O_t | \lambda)$$

- Viterbi Algorithm (Cont.)

- Recursion

$$\delta_t(j) = \left[\max_{1 \leq i \leq N} \delta_{t-1}(i) a_{ij} \right] b_j(O_t),$$

$$\psi_t(j) = \arg \max_{1 \leq i \leq N} \delta_{t-1}(i) a_{ij},$$

$$2 \leq t \leq T, \quad 1 \leq j \leq N$$

- Termination

$$P^* = \max_{1 \leq i \leq N} \delta_T(i)$$

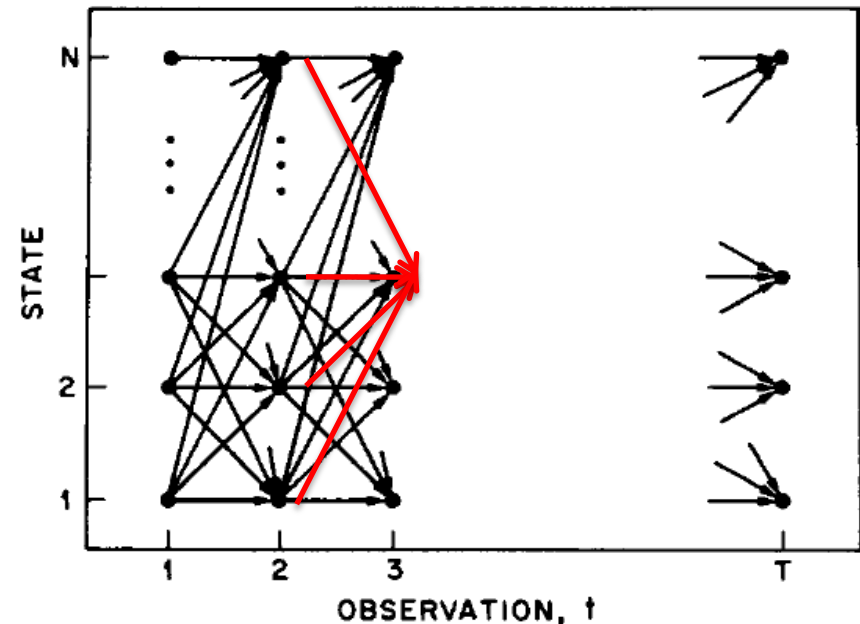
$$q_T^* = \arg \max_{1 \leq i \leq N} \delta_T(i)$$

- Backtracking

$$q_t^* = \psi_{t+1}(q_{t+1}^*),$$

$$1 \leq t \leq T-1$$

- Complexity: $O(TN^2)$



Training Problem

- Maximum Likelihood (ML) $\max_{A,B,\pi} P(O | \lambda)$
- Baum-Welch Algorithm (EM)

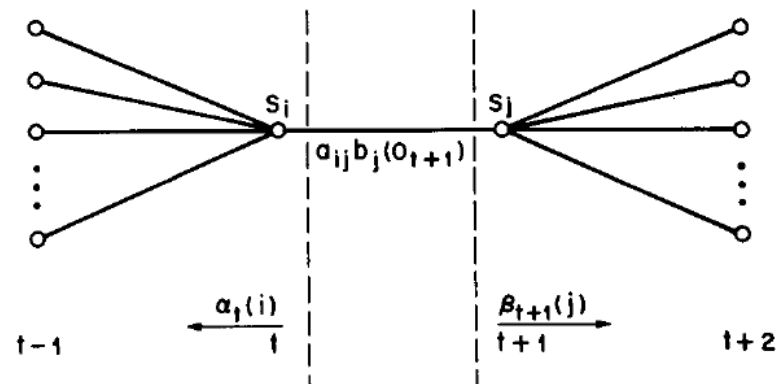
$$\max_{\bar{\lambda}} Q(\lambda, \bar{\lambda}) = \sum_Q \log P(Q, O | \bar{\lambda}) P(Q, O | \lambda)$$

– Define variable

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

$$\xi_t(i, j) = \frac{\alpha_t(i) a_{ij} b_j(O_{t+1}) \beta_{t+1}(j)}{P(O | \lambda)} \leftarrow P(O, q_t = S_i, q_{t+1} = S_j | \lambda)$$

$$= \frac{\alpha_t(i) a_{ij} b_j(O_{t+1}) \beta_{t+1}(j)}{\sum_{i=1}^N \sum_{j=1}^N \alpha_t(i) a_{ij} b_j(O_{t+1}) \beta_{t+1}(j)}$$



– Define probability

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

- Baum-Welch Algorithm (Cont.)

- Reestimation formulas

$\bar{\pi}_i$ = expected frequency (number of times) in state S_i at time $(t=1)=\gamma_1(i)$

$$\bar{a}_{ij} = \frac{\text{expected number of transitions from state } S_i \text{ to state } S_j}{\text{expected number of transitions from state } S_i}$$

$$= \frac{\sum_{t=1}^{T-1} \xi_t(i, j)}{\sum_{t=1}^{T-1} \gamma_t(i)} = \frac{\sum_{t=1}^{T-1} \alpha_t(i) a_{ij} b_j(O_{t+1}) \beta_{t+1}(j)}{\sum_{t=1}^{T-1} \alpha_t(i) \beta_t(i)}$$

$$\bar{b}_j(k) = \frac{\text{expected number of times in state } j \text{ and observing symbol } v_k}{\text{expected number of times in state } j}$$

$$= \frac{\sum_{t=1, \text{ s.t. } O_t=v_k}^{T-1} \gamma_t(j)}{\sum_{t=1}^{T-1} \gamma_t(j)} = \frac{\sum_{t=1, \text{ s.t. } O_t=v_k}^T \alpha_t(j) \beta_t(j)}{\sum_{t=1}^T \alpha_t(j) \beta_t(j)}$$

Continuous Density HMM

- Handling Continuous Observations
 - Continuous features: vector \mathbf{O}_T
 - Discretization: vector quantization (VQ)
 - Each vector replaced with its closest codevector, which is viewed as a symbol
 - Small codebook: distortion
 - Large codebook: large data required in emission probability estimation
 - Continuous emission density: Gaussian mixture (GM)

$$b_j(O) = \sum_{m=1}^M c_{jm} \mathcal{N}(O; \mu_{jm}, U_{jm}), \quad 1 \leq j \leq N$$

- Parameter Estimation of Continuous HMM (omitted)

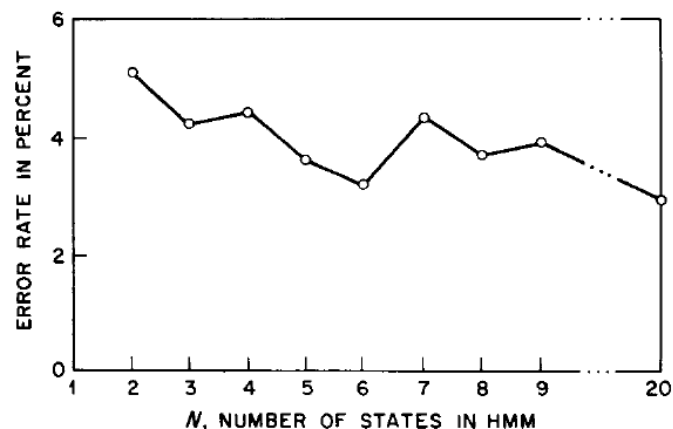
Application to Speech Recognition

- Isolated Word Recognition

- Given HMM λ^v for each word in the vocabulary
- Input observation sequence \mathbf{O} , Bayes decision (assuming equal prior probabilities)

$$v^* = \arg \max_{1 \leq v \leq V} P(\lambda^v | \mathbf{O}) = \arg \max_{1 \leq v \leq V} P(\mathbf{O} | \lambda^v)$$

- Acoustic features (\mathbf{O}_t) (details omitted)
- Vector quantization (discrete observation symbols)
- Choice of model parameters
 - Number of states: empirical, correspond roughly to the number of sounds (phonemes) in a word, maybe equal for all word models
 - Number of components in GM



Extensions of HMM

- Hybrid HMM/Neural
 - HMM: parametric $b_j(O_t)=p(O_t|q_t=S_j)$, conditional independence
 - Neural: discriminative emission probability $p(q_t=S_j|O_t)$
 - Neural network outputs approximate posterior probabilities
 - Replace $p(\mathbf{x}_t|q_t)$ with $p(q_t|\mathbf{x}_t)/p(q_t)$
$$\frac{p(\mathbf{x}_t | q_t)}{p(\mathbf{x}_t)} = \frac{p(q_t | \mathbf{x}_t)}{P(q_t)}$$
 - ANN may input multiple frames to learn the correlation
- Latest: deep neural networks

讨论

- 特征维数与过拟合
 - 克服过拟合的方法?
- 期望最大法(EM)
 - 对数似然度对缺失数据的期望
 - EM for Gaussian mixture
- 隐马尔可夫模型(HMM)
 - Three basic problems
 - Viterbi Algorithm
 - Extensions

下次课内容

- 第4章 非参数法
 - 密度估计
 - Parzen窗方法
 - K近邻估计
 - 最近邻规则
 - 距离度量
 - Reduced Coulomb Energy Network
 - Approximation by Series Expansion