Clustering Techniques

Berlin Chen 2005

References:

1. *Introduction to Machine Learning ,* Chapter 7
2. *Modern Information Retrieval*, Chapters 5, 7
3. *Foundations of Statistical Natural Language Processing*, Chapter 14
4. "*A Gentle Tutorial of the EM Algorithm and its Application to Parameter Estimation for Gaussian Mixture and Hidden Markov Models*," Jeff A. Bilmes, U.C. Berkeley TR-97-021

Clustering

- Place similar objects in the same group and assign dissimilar objects to different groups
  - **Word clustering**
    - Neighbor overlap: words occur with the similar left and right neighbors (such as *in* and *on*)
  - **Document clustering**
    - Documents with the similar topics or concepts are put together
- But clustering cannot give a comprehensive description of the object
  - How to label objects shown on the visual display
- Regarded as a kind of semiparametric learning approach
  - Allow a mixture of distributions to be used for estimating the input samples (a parametric model for each group of samples)

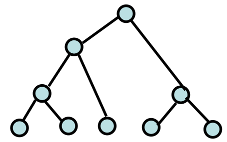
Clustering vs. Classification

- Classification is supervised and requires a set of labeled training instances for each group (class)

- Clustering is unsupervised and learns without a teacher to provide the labeling information of the training data set
  - Also called automatic or unsupervised classification

Types of Clustering Algorithms

- Two types of structures produced by clustering algorithms
  - Flat or non-hierarchical clustering
  - Hierarchical clustering
- **Flat clustering**
  - Simply consisting of a certain number of clusters and the relation between clusters is often undetermined
  - Measure: construction error minimization or probabilistic optimization
- **Hierarchical clustering**
  - A hierarchy with usual interpretation that each node stands for a subclass of its mother's node
    - The leaves of the tree are the single objects
    - Each node represents the cluster that contains all the objects of its descendants
  - Measure: similarities of instances



Hard Assignment vs. Soft Assignment

- Another important distinction between clustering algorithms is whether they perform soft or hard assignment

- **Hard Assignment**
  - Each object is assigned to one and only one cluster

- **Soft Assignment (probabilistic approach)**
  - Each object may be assigned to multiple clusters
  - An object x_i has a probability distribution $P(\cdot|x_i)$ over clusters c_j where $P(x_i|c_j)$ is the probability that x_i is a member of c_j
  - Is somewhat more appropriate in many tasks such as NLP, IR, …

Hard Assignment vs. Soft Assignment (cont.)

- Hierarchical clustering usually adopts hard assignment
- While in flat clustering both types of assignments are common

Summarized Attributes of Clustering Algorithms

- Hierarchical Clustering
  - Preferable for detailed data analysis
  - Provide more information than flat clustering
  - No single best algorithm (each of the algorithms only optimal for some applications)
  - Less efficient than flat clustering (minimally have to compute *n* x *n* matrix of similarity coefficients)

Summarized Attributes of Clustering Algorithms (cont.)

- Flat Clustering
  - Preferable if efficiency is a consideration or data sets are very large
  - *K*-means is the conceptually method and should probably be used on a new data because its results are often sufficient
  - *K*-means assumes a simple Euclidean representation space, and so cannot be used for many data sets, e.g., nominal data like colors (or samples with features of different scales)
  - The EM algorithm is the most choice. It can accommodate definition of clusters and allocation of objects based on complex probabilistic models
    - Its extensions can be used to handle topological/hierarchical orders of samples
      - Probabilistic Latent Semantic Analysis (PLSA), Topic Mixture Model (TMM), etc.

Hierarchical Clustering

Hierarchical Clustering

- Can be in either bottom-up or top-down manners
  - **Bottom-up** (agglomerative) 凝集的
    - Start with individual objects and grouping the most similar ones
      - E.g., with the minimum distance apart

$$sim(x, y) = \frac{1}{1 + d(x, y)}$$

distance measures will be discussed later on

    - The procedure terminates when one cluster containing all objects has been formed
  - **Top-down** (divisive) 分裂的
    - Start with all objects in a group and divide them into groups so as to maximize within-group similarity

Hierarchical Agglomerative Clustering (HAC)

- A bottom-up approach
- Assume a similarity measure for determining the similarity of two objects
- Start with all objects in a separate cluster and then repeatedly joins the two clusters that have the most similarity until there is one only cluster survived
- The history of merging/clustering forms a binary tree or hierarchy

HAC (cont.)

- Algorithm

Given: a set $\mathcal{X} = \{x_1, \ldots x_n\}$ of objects
a function sim: $\mathcal{P}(\mathcal{X}) \times \mathcal{P}(\mathcal{X}) \rightarrow \mathbb{R}$
for $i := 1$ **to** n **do**
$c_i := \{x_i\}$ **end**
$C := \{c_1, \ldots, c_n\}$
$j := n + 1$
while $|C| > 1$
$(c_{n_1}, c_{n_2}) := \arg\max_{(c_u, c_v) \in C \times C} \text{sim}(c_u, c_v)$
$c_j = c_{n_1} \cup c_{n_2}$
$C := C \backslash \{c_{n_1}, c_{n_2}\} \cup \{c_j\}$
$j := j + 1$

Initialization (for tree leaves): Each object is a cluster

cluster number

merged as a new cluster

The original two clusters are removed

Figure 14.2 Bottom-up hierarchical clustering.

Distance Metrics

- Euclidian Distance (L_2 norm)

$$L_2(\vec{x},\vec{y}) = \sum_{i=1}^{m}(x_i - y_i)^2$$

  - Make sure that all attributes/dimensions have the same scale (or the same variance)

- L_1 Norm (City-block distance)

$$L_1(\vec{x},\vec{y}) = \sum_{i=1}^{m}\left|x_i - y_i\right|$$

- Cosine Similarity (transform to a distance by subtracting from 1)

$$1 - \frac{\vec{x} \bullet \vec{y}}{|\vec{x}| \cdot |\vec{y}|}$$

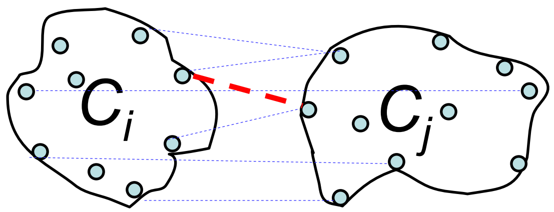
ranged between 0 and 1

Measures of Cluster Similarity

- Especially for the bottom-up approaches

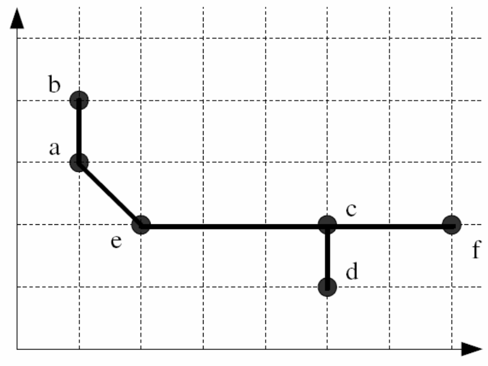
- **Single-link** clustering
  - The similarity between two clusters is the similarity of the two closest objects in the clusters
  - Search over all pairs of objects that are from the two different clusters and select the pair with the greatest similarity
  - Elongated clusters are achieved

$$sim(c_i, c_j) = \max_{\vec{x} \in c_i, \vec{y} \in c_j} sim(\vec{x}, \vec{y})$$



greatest similarity

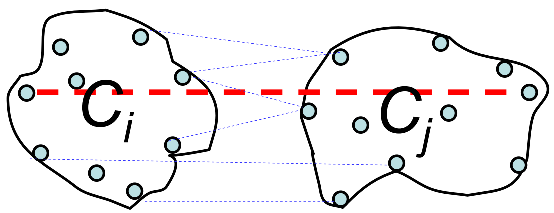
cf. the minimal spanning tree



Measures of Cluster Similarity (cont.)

- **Complete-link** clustering
  - The similarity between two clusters is the similarity of their two most dissimilar members
  - Sphere-shaped clusters are achieved
  - Preferable for most IR and NLP applications

$$sim\left(c_i, c_j\right) = \min_{\vec{x} \in c_i, \vec{y} \in c_j} sim\left(\vec{x}, \vec{y}\right)$$



least similarity

Measures of Cluster Similarity (cont.)

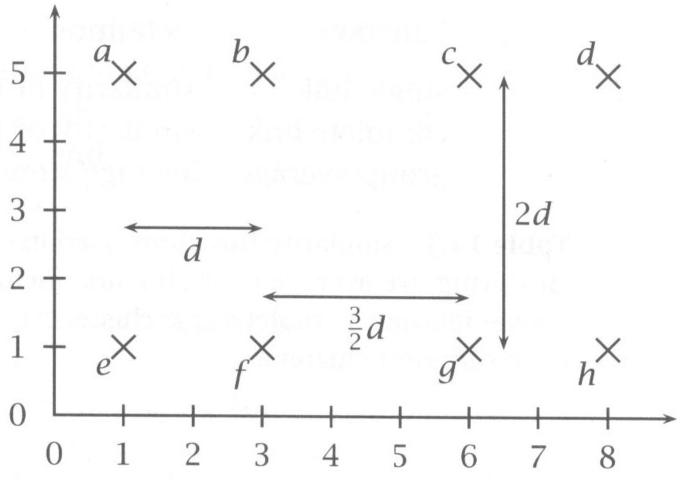


Figure 14.4 A cloud of points in a plane.

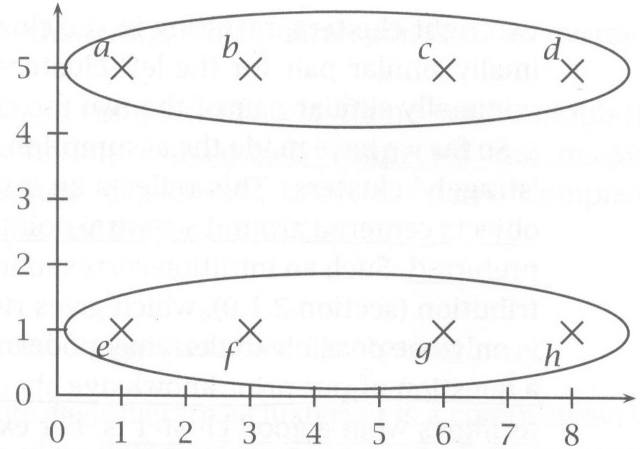


Figure 14.6 Single-link clustering of the points in figure 14.4.

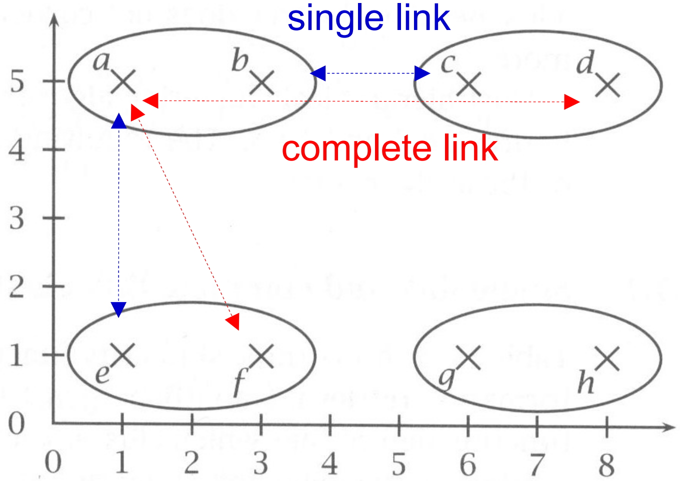


Figure 14.5 Intermediate clustering of the points in figure 14.4.

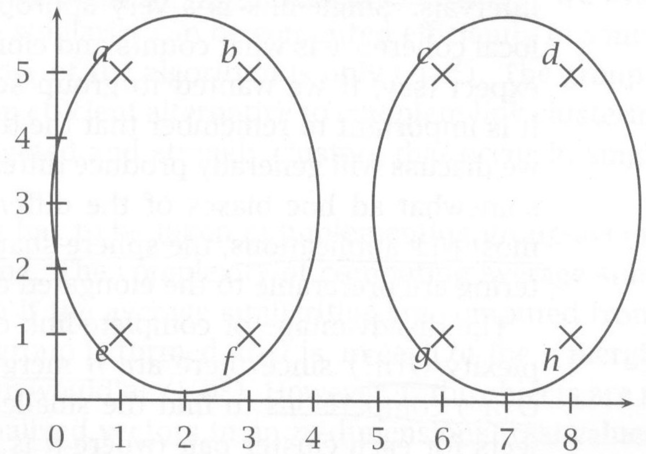


Figure 14.7 Complete-link clustering of the points in figure 14.4.

Measures of Cluster Similarity (cont.)

- **Group-average** agglomerative clustering
  - A compromise between single-link and complete-link clustering
  - The similarity between two clusters is the average similarity between members
  - If the objects are represented as length-normalized vectors and the similarity measure is the cosine
    - There exists an fast algorithm for computing the average similarity

$$sim(\vec{x}, \vec{y}) = \cos(\vec{x}, \vec{y}) = \frac{\vec{x} \cdot \vec{y}}{|\vec{x}||\vec{y}|} = \vec{x} \cdot \vec{y}$$

length-normalized vectors

Measures of Cluster Similarity (cont.)

- **Group-average** agglomerative clustering (cont.)

  – The average similarity *SIM* between vectors in a cluster c_j is defined as

$$SIM(c_j) = \frac{1}{|c_j|(|c_j|-1)} \sum_{\vec{x} \in c_j} \sum_{\substack{\vec{y} \in c_j \\ \vec{y} \neq \vec{x}}} sim(\vec{x}, \vec{y}) = \frac{1}{|c_j|(|c_j|-1)} \sum_{\vec{x} \in c_j} \sum_{\substack{\vec{y} \in c_j \\ \vec{y} \neq \vec{x}}} \vec{x} \cdot \vec{y}$$

  – The sum of members in a cluster c_j : $\vec{s}(c_j) = \sum_{\vec{x} \in c_j} \vec{x}$

  – Express $SIM(c_j)$ in terms of $\vec{s}(c_j)$

$$\vec{s}(c_j) \cdot \vec{s}(c_j) = \sum_{\vec{x} \in c_j} \vec{x} \cdot \vec{s}(c_j) = \sum_{\vec{x} \in c_j} \sum_{\vec{y} \in c_j} \vec{x} \cdot \vec{y}$$

length-normalized vector

$$= |c_j|(|c_j|-1) SIM(c_j) + \sum_{\vec{x} \in c_j} \vec{x} \cdot \vec{x}$$

=1

$$= |c_j|(|c_j|-1) SIM(c_j) + |c_j|$$

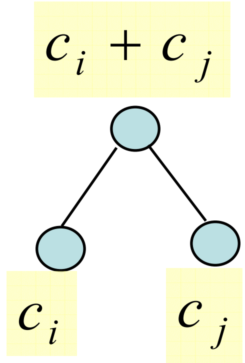
$$\therefore SIM(c_j) = \frac{\vec{s}(c_j) \cdot \vec{s}(c_j) - |c_j|}{|c_j|(|c_j|-1)}$$

Measures of Cluster Similarity (cont.)

- **Group-average** agglomerative clustering (cont.)

-As merging two clusters c_i and c_j, the cluster sum vectors $\vec{s}(c_i)$ and $\vec{s}(c_j)$ are known in advance

$$\Rightarrow \quad \vec{s}(c_{New}) = \vec{s}(c_i) + \vec{s}(c_j), \quad |c_{New}| = |c_i| + |c_j|$$



– The average similarity for their union will be

$$SIM(c_i \cup c_j) = \frac{(\vec{s}(c_i) + \vec{s}(c_j)) \cdot (\vec{s}(c_i) + \vec{s}(c_j)) - (|c_i| + |c_j|)}{(|c_i| + |c_j|)(|c_i| + |c_j| - 1)}$$

Example: Word Clustering

- Words (objects) are described and clustered using a set of features and values
  - E.g., the left and right neighbors of tokens of words

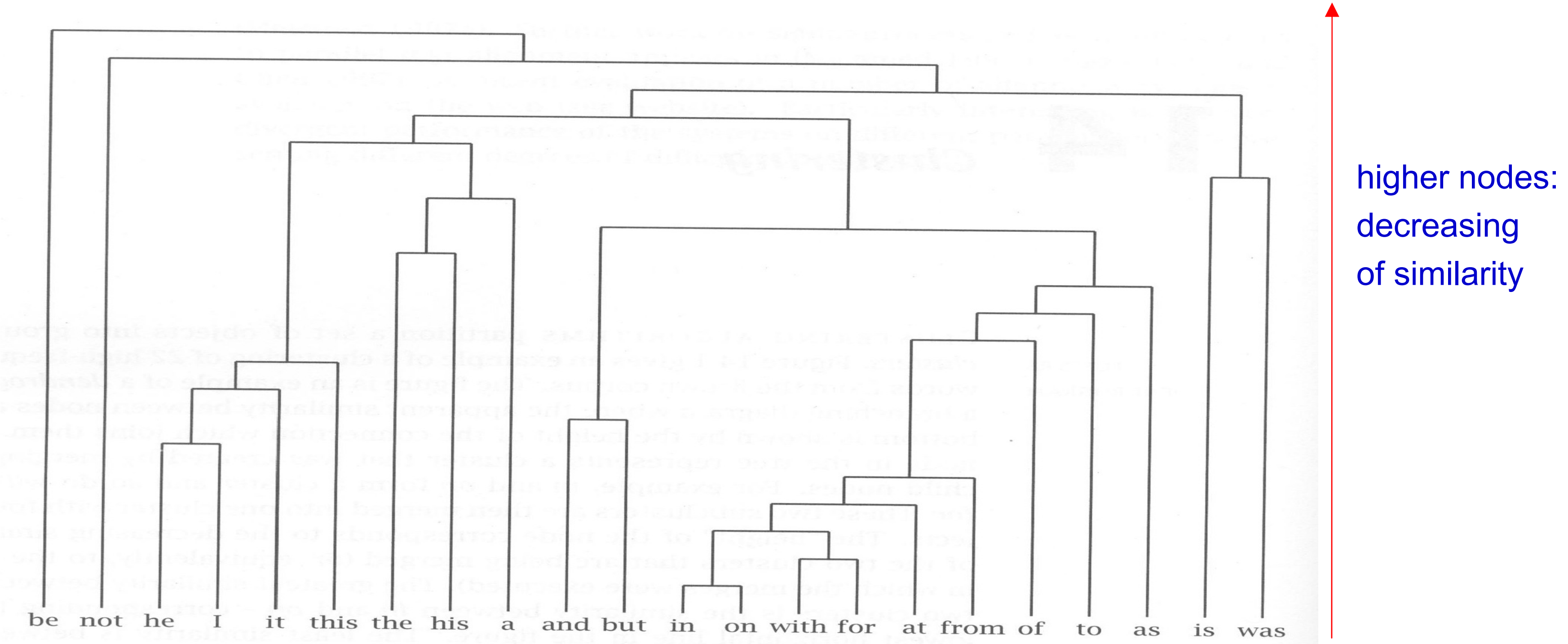


Figure 14.1 A single-link clustering of 22 frequent English words represented as a dendrogram.

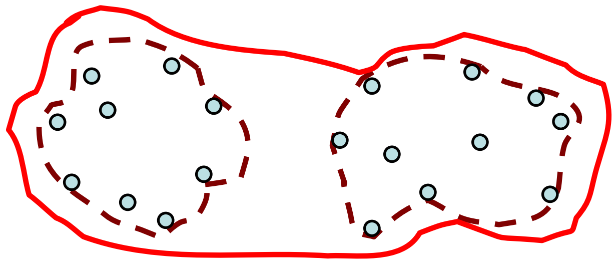
"be" has least similarity with the other 21 words !

Divisive Clustering

- A top-down approach
- Start with all objects in a single cluster
- At each iteration, select the least **coherent** cluster and **split** it
- Continue the iterations until a predefined criterion (e.g., the cluster number) is achieved
- The history of clustering forms a binary tree or hierarchy

Divisive Clustering (cont.)

- To select the least coherent cluster, the measures used in bottom-up clustering (e.g. HAC) can be used again here
  - Single link measure
  - Complete-link measure
  - Group-average measure



- How to split a cluster
  - Also is a clustering task (finding two sub-clusters)
  - Any clustering algorithm can be used for the splitting operation, e.g.,
    - Bottom-up (agglomerative) algorithms
    - Non-hierarchical clustering algorithms (e.g., *K*-means)

Divisive Clustering (cont.)

- Algorithm

Given: a set $\mathcal{X} = \{x_1, \ldots x_n\}$ of objects
a function coh: $\mathcal{P}(\mathcal{X}) \rightarrow \mathbb{R}$
a function split: $\mathcal{P}(\mathcal{X}) \rightarrow \mathcal{P}(\mathcal{X}) \times \mathcal{P}(\mathcal{X})$
$C := \{\mathcal{X}\} \quad (= \{c_1\})$
$j := 1$
while $\exists c_i \in C$ s.t. $|c_i| > 1$
$\quad c_u := \arg\min_{c_v \in C} \text{coh}(c_v)$
$\quad (c_{j+1}, c_{j+2}) := \text{split}(c_u)$
$\quad C := C \backslash \{c_u\} \cup \{c_{j+1}, c_{j+2}\}$
$\quad j := j + 2$

split the least coherent cluster

Generate two new clusters and remove the original one

Figure 14.3 Top-down hierarchical clustering.

Non-Hierarchical Clustering

Non-hierarchical Clustering

- Start out with a partition based on randomly selected seeds (one seed per cluster) and then refine the initial partition
  - In a multi-pass manner (recursion/iterations)

- **Problems** associated with non-hierarchical clustering
  - When to stop   group average similarity, likelihood, mutual information
  - What is the right number of clusters $k\text{-}1 \rightarrow k \rightarrow k\text{+}1$

    Hierarchical clustering also has to face this problem

- **Algorithms** introduced here
  - The *K*-means algorithm
  - The EM algorithm

The *K*-means Algorithm

- Also called *Linde-Buzo-Gray* (LBG) in signal processing
  - A hard clustering algorithm
  - Define clusters by the **center of mass** of their members
- The K-means algorithm also can be regarded as
  - A kind of vector quantization
    - Map from a continuous space (high resolution) to a discrete space (low resolution)
  - E.g. color quantization
    - 24 bits/pixel (16 million colors) → 8 bits/pixel (256 colors)
    - A compression rate of 3

$$\boldsymbol{X} = \{\boldsymbol{x}^t\}_{t=1}^{n} \xrightarrow{\text{index } j} \boldsymbol{F} = \{\boldsymbol{m}_j\}_{j=1}^{k}$$

Dim($\boldsymbol{x}^t$)=24 → k=2^8

$\boldsymbol{m}_j$: cluster centriod or reference vector, code word, code vector

The *K*-means Algorithm (cont.)

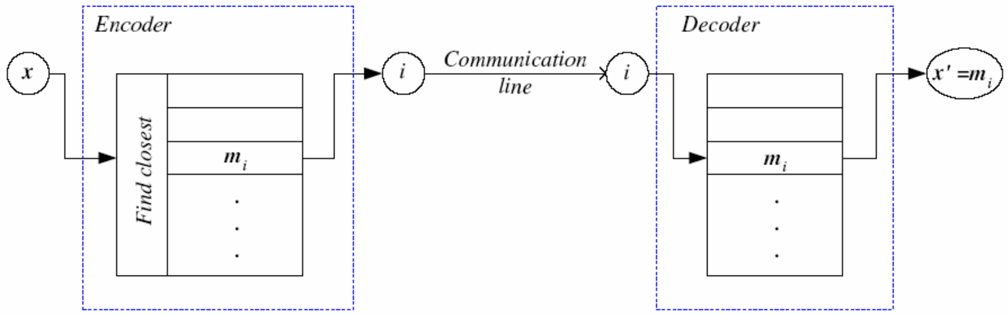


Figure 7.1: Given $\boldsymbol{x}$, the encoder sends the index of the closest code word and the decoder generates the code word with the received index as $\boldsymbol{x}'$. Error is $\|\boldsymbol{x}' - \boldsymbol{x}\|^2$.

Total Reconstruction error

label

$$E\left(\{\boldsymbol{m}_i\}_{i=1}^{k} \middle| \boldsymbol{X}\right) = \sum_{t=1}^{N}\sum_{i=1}^{k} b_i^t \left\|\boldsymbol{x}^t - \boldsymbol{m}_i\right\|^2, \text{ where } b_i^t = \begin{cases} 1 & \text{if } \left\|\boldsymbol{x}^t - \boldsymbol{m}_i\right\| = \min_j \left\|\boldsymbol{x}^t - \boldsymbol{m}_j\right\| \\ 0 & \text{otherwise} \end{cases}$$

- b_i^t and $\boldsymbol{m}_i$ are unknown
- b_i^t depends on $\boldsymbol{m}_i$ and this optimization problem can not be solved analytically

The *K*-means Algorithm (cont.)

- **Initialization**
  - A set of initial cluster centers is needed $\{\boldsymbol{m}_i\}_{i=1}^{k}$
- **Recursion**
  - Assign each object $\boldsymbol{x}^t$ to the cluster whose center is closest

$$b_i^t = \begin{cases} 1 & \text{if } \left\|\boldsymbol{x}^t - \boldsymbol{m}_i\right\| = \min_j \left\|\boldsymbol{x}^t - \boldsymbol{m}_j\right\| \\ 0 & \text{otherwise} \end{cases}$$

  - Then, re-compute the center of each cluster as the centroid or mean (average) of its members
    - Using the medoid as the cluster center ?

    (a medoid is one of the objects in the cluster)

$$\boldsymbol{m}_i = \frac{\sum_{t=1}^{N} b_i^t \cdot \boldsymbol{x}^t}{\sum_{t=1}^{N} b_i^t}$$

These two steps are repeated until $\boldsymbol{m}_i$ stabilizes

The *K*-means Algorithm (cont.)

- Algorithm

Initialize $\boldsymbol{m}_i, i = 1, \ldots, k$, for example, to k random $\boldsymbol{x}^t$

Repeat

For all $\boldsymbol{x}^t \in \mathcal{X}$

$$b_i^t \leftarrow \begin{cases} 1 & \text{if } \|\boldsymbol{x}^t - \boldsymbol{m}_i\| = \min_j \|\boldsymbol{x}^t - \boldsymbol{m}_j\| \\ 0 & \text{otherwise} \end{cases}$$

For all $\boldsymbol{m}_i, i = 1, \ldots, k$

$$\boldsymbol{m}_i \leftarrow \sum_t b_i^t \boldsymbol{x}^t / \sum_t b_i^t$$

Until $\boldsymbol{m}_i$ converge

The *K*-means Algorithm (cont.)

- Example 1

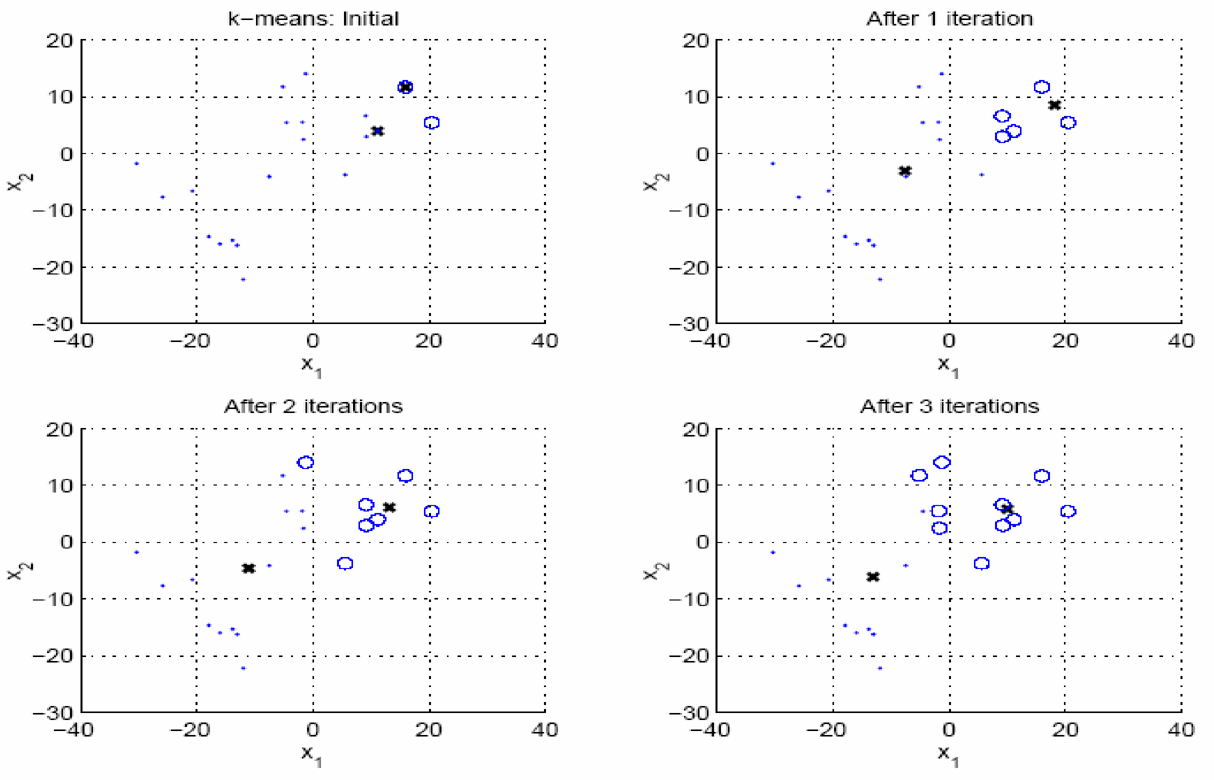


Figure 7.2: Evolution of k-means. Crosses indicate center positions. Data points are marked depending on the closest center.

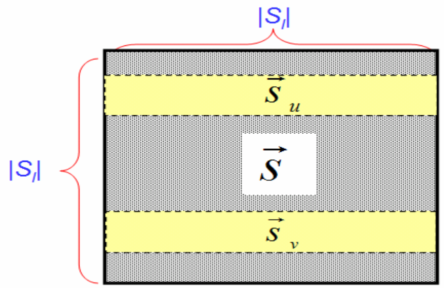
The *K*-means Algorithm (cont.)

- Example 2

Cluster	Members	
1	*ballot* (0.28), *polls* (0.28), *Gov* (0.30), *seats* (0.32)	government
2	*profit* (0.21), *finance* (0.21), *payments* (0.22)	finance
3	*NFL* (0.36), *Reds* (0.28), *Sox* (0.31), *inning* (0.33), *quarterback* (0.30), *scored* (0.30), *score* (0.33)	sports
4	*researchers* (0.23), *science* (0.23)	research
5	*Scott* (0.28), *Mary* (0.27), *Barbara* (0.27), *Edward* (0.29)	name

Table 14.4 An example of K-means clustering. Twenty words represented as vectors of co-occurrence counts were clustered into 5 clusters using K-means. The distance from the cluster centroid is given after each word.

$$s_{u,v} = \frac{\vec{s}_u \cdot \vec{s}_v}{|\vec{s}_u| \times |\vec{s}_u|}$$



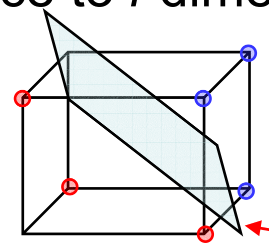
The *K*-means Algorithm (cont.)

- Choice of initial cluster centers (seeds) is important
  - Pick at random
  - Calculate the mean of all data and generate *k* initial centers $\boldsymbol{m}_i$ by adding small random vector to the mean $\boldsymbol{m}_i \pm \boldsymbol{\delta}$
  - Project data onto the principal component (first eigenvector), divide it range into *k* equal interval, and take the mean of data in each group as the initial center $\boldsymbol{m}_i$
  - Or use another method such as hierarchical clustering algorithm on a subset of the objects
    - E.g., buckshot algorithm uses the group-average agglomerative clustering to randomly sample of the data that has size square root of the complete set
- Poor seeds will result in **sub-optimal** clustering

The *K*-means Algorithm (cont.)

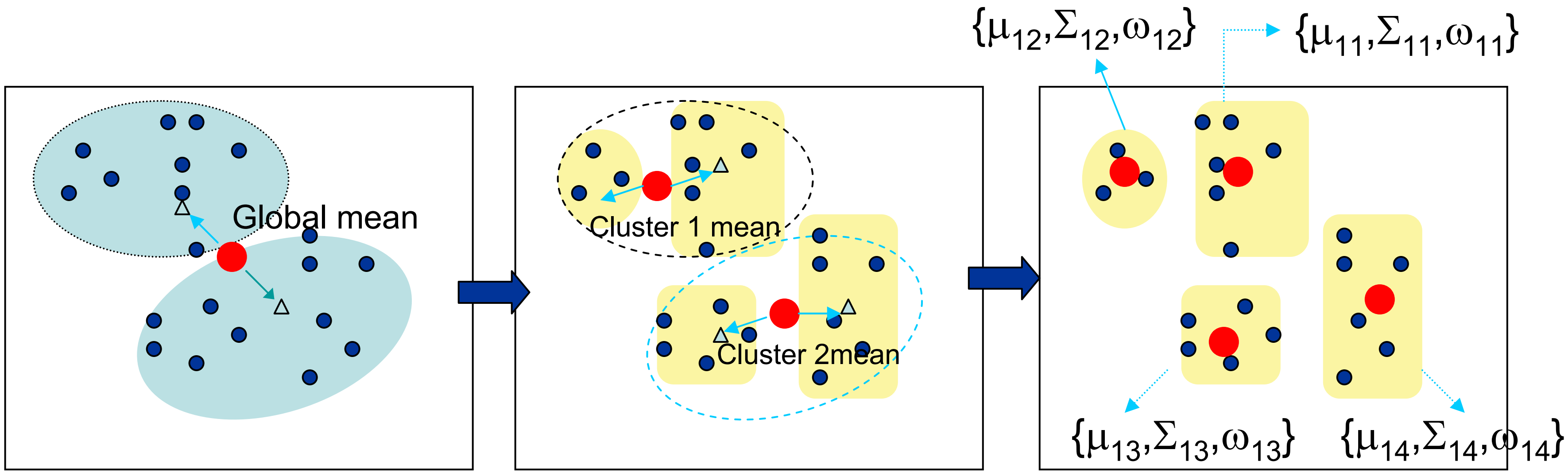
- How to break ties when in case there are several centers with the same distance from an object
  - Randomly assign the object to one of the candidate clusters
  - Or, perturb objects slightly
- Applications of the *K*-means Algorithm
  - Clustering
  - Vector quantization
  - A preprocessing stage before classification or regression
    - Map from the original space to *l*-dimensional space/hypercube

$l=\log_2 k$ (k clusters)



The *K*-means Algorithm (cont.)

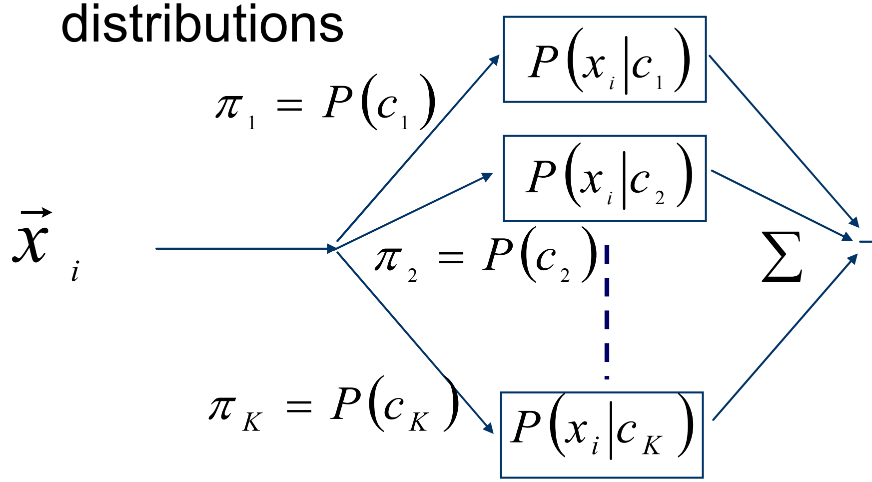
- E.g., the LBG algorithm
  - By Linde, Buzo, and Gray



M→2M at each iteration

The EM Algorithm

- A **soft version** of the *K*-mean algorithm
  - Each object could be the member of multiple clusters
  - Clustering as estimating a mixture of (continuous) probability distributions



A Mixture Gaussian HMM
(or A Mixture of Gaussians)

$$P(\vec{x}_i|\Theta)=\sum_{k=1}^{K} P(\vec{x}_i|c_k;\Theta)P(c_k|\Theta)$$

classification:

$$\max_k P(c_k|\vec{x}_i,\Theta)=\max_k \frac{(\vec{x}_i|c_k;\Theta)P(c_k|\Theta)}{P(\vec{x}_i|\Theta)}$$
$$=\max_k (\vec{x}_i|c_k;\Theta)P(c_k|\Theta)$$

Likelihood function for data samples: $\boldsymbol{X} = \vec{x}_1, \vec{x}_2, \ldots, \vec{x}_n$

$$P(\boldsymbol{X}|\Theta)=\prod_{i=1}^{n} P(\vec{x}_i|\Theta)$$
$$=\prod_{i=1}^{n}\sum_{k_i=1}^{K} P(\vec{x}_i|c_{k_i};\Theta)P(c_{k_i}|\Theta)$$

Continuous case:

$$P(\vec{x}_i|c_k;\Theta)=\frac{1}{\sqrt{(2\pi)^m|\Sigma_k|}}\exp\left(-\frac{1}{2}(\vec{x}_i-\vec{\mu}_k)^T\Sigma_k^{-1}(\vec{x}_i-\vec{\mu}_k)\right)$$

$\boldsymbol{X} = \{\vec{x}_1, \vec{x}_2, \cdots, \vec{x}_n\}$

$\vec{x}_i$'s are independent identically distributed (i.i.d.)

The EM Algorithm (cont.)

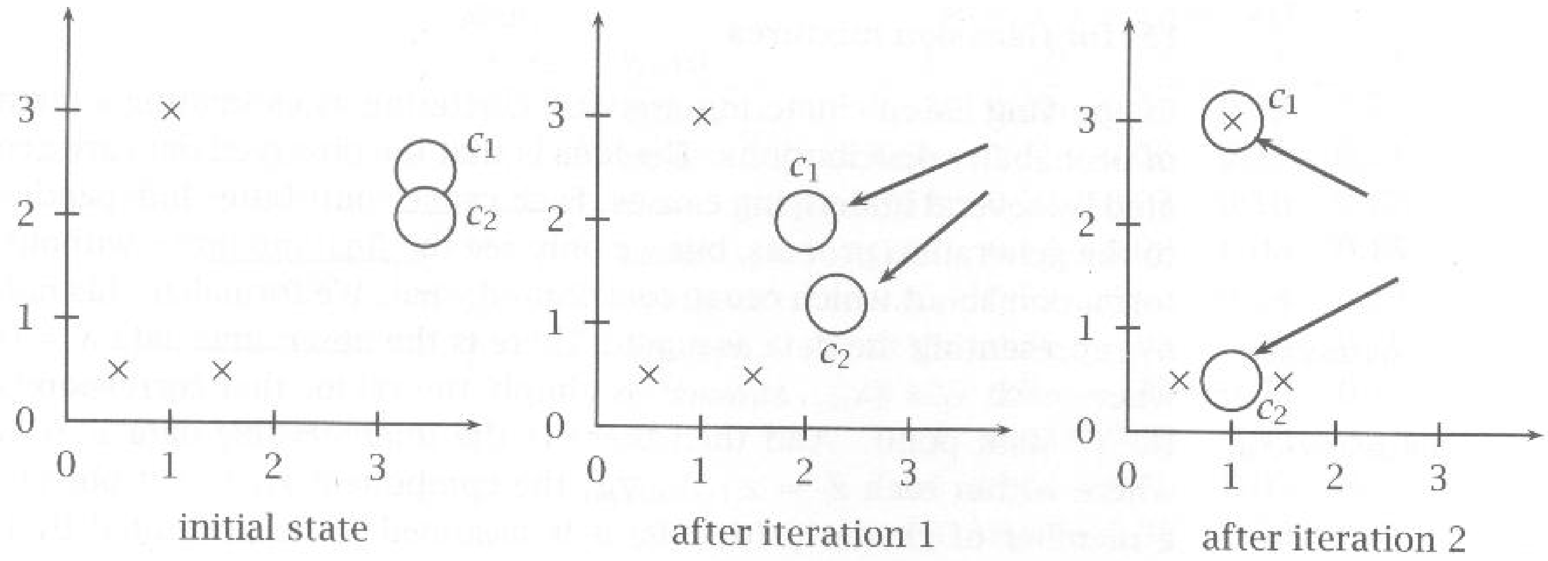
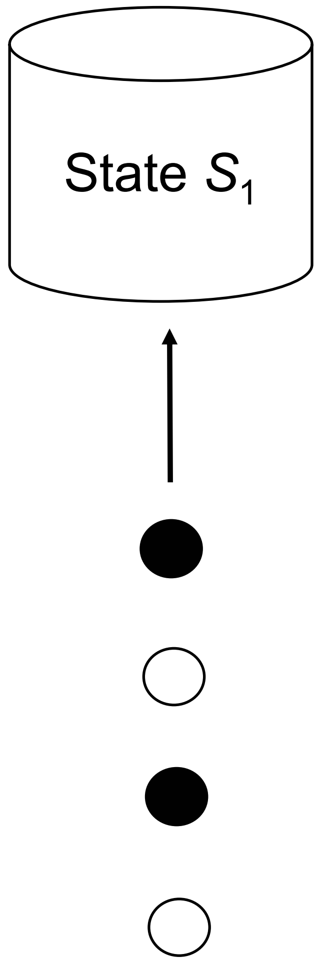


Figure 14.10 An example of using the EM algorithm for soft clustering.

Maximum Likelihood Estimation

- Hard Assignment

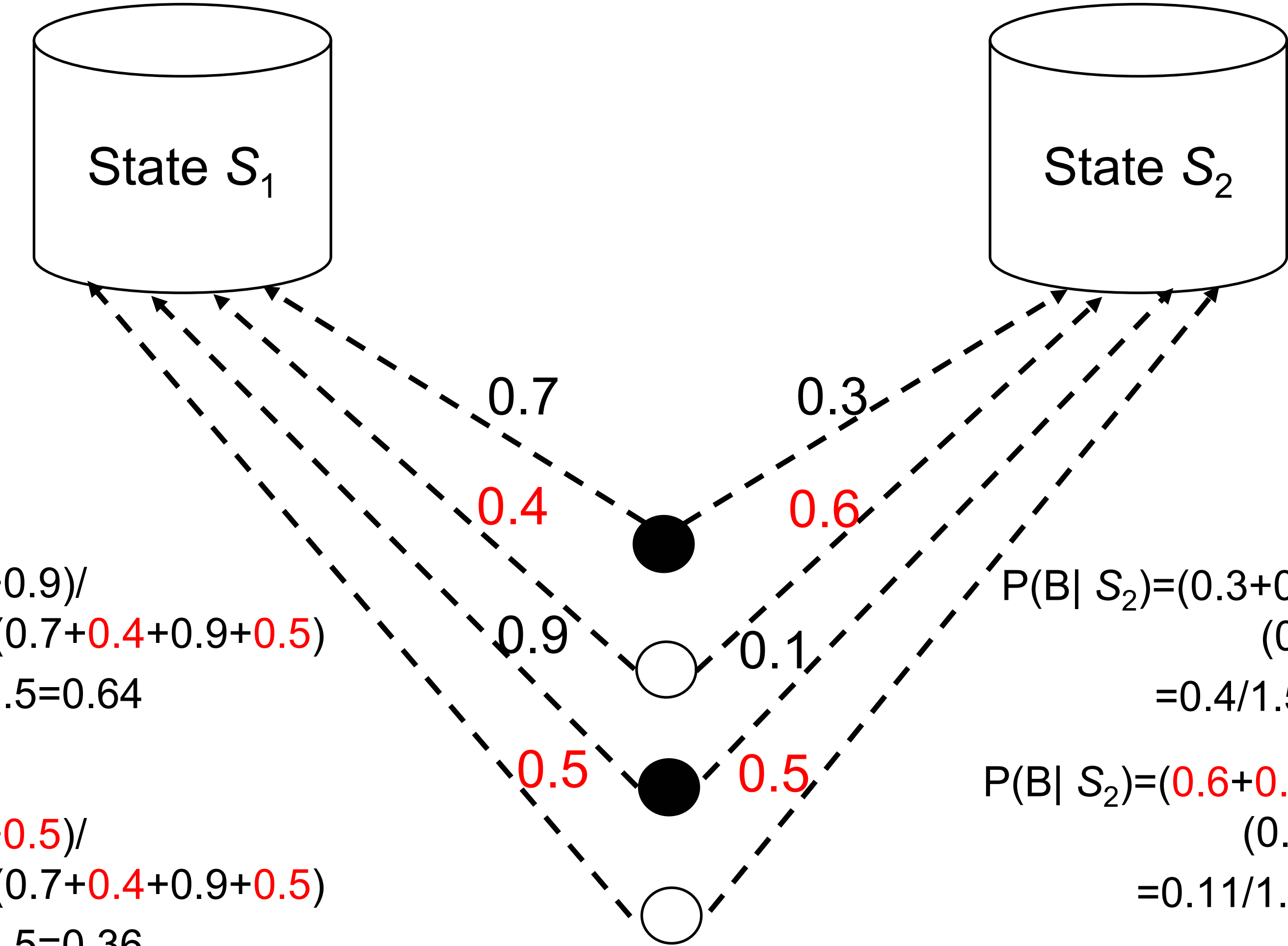


$P(B|\ S_1)=2/4=0.5$

$P(W|\ S_1)=2/4=0.5$

Maximum Likelihood Estimation

- Soft Assignment



$P(B|\ S_1)=(0.7+0.9)/(0.7+0.4+0.9+0.5)$
$=1.6/2.5=0.64$

$P(B|\ S_1)=(0.4+0.5)/(0.7+0.4+0.9+0.5)$
$=0.9/2.5=0.36$

$P(B|\ S_2)=(0.3+0.1)/(0.3+0.6+0.1+0.5)$
$=0.4/1.5=0.27$

$P(B|\ S_2)=(0.6+0.5)/(0.3+0.6+0.1+0.5)$
$=0.11/1.5=0.73$

The EM Algorithm (cont.)

Note :

$\prod_{t=1}^{T}\left(\sum_{k_t=1}^{M} a_{tk_t}\right)$

$=(a_{11}+a_{12}+...+a_{1M})(a_{21}+a_{22}+...+a_{2M})...(a_{T1}+a_{T2}+...+a_{TM})$

$=\sum_{k_1=1}^{M}\sum_{k_2=1}^{M}...\sum_{k_T=1}^{M}\prod_{t=1}^{T} a_{tk_t}$

- E–step (Expectation)
  - Derive the complete data likelihood function

likelihood function

$$P\left(\mathbf{X}\middle|\hat{\Theta}\right)=\prod_{i=1}^{n} P\left(\vec{x}_i\middle|\hat{\Theta}\right)=\prod_{i=1}^{n}\sum_{k_i=1}^{K} P\left(\vec{x}_i\middle|c_{k_i};\hat{\Theta}\right)P\left(c_{k_i}\middle|\hat{\Theta}\right)$$

$$=\left(P\left(\vec{x}_1\middle|c_1;\hat{\Theta}\right)P\left(c_1\middle|\hat{\Theta}\right)+..+P\left(\vec{x}_1\middle|c_K;\hat{\Theta}\right)P\left(c_K\middle|\hat{\Theta}\right)\right)\times\cdots$$

$$\times\left(P\left(\vec{x}_n\middle|c_1;\hat{\Theta}\right)P\left(c_1,\hat{\Theta}\right)+..+P\left(\vec{x}_n\middle|c_K;\hat{\Theta}\right)P\left(c_K\middle|\hat{\Theta}\right)\right)$$

$$=\sum_{k_i=1}^{K}\sum_{k_2=1}^{K}\cdots\sum_{k_n=1}^{K}\prod_{i=1}^{n}\left[P\left(\vec{x}_i\middle|c_{k_i};\hat{\Theta}\right)P\left(c_{k_i}\middle|\hat{\Theta}\right)\right]$$

$$=\sum_{k_i=1}^{K}\sum_{k_2=1}^{K}\cdots\sum_{k_n=1}^{K}\prod_{i=1}^{n}\left[P\left(\vec{x}_i,c_{k_i}\middle|\hat{\Theta}\right)\right]$$

$$=\sum_{k_i=1}^{K}\sum_{k_2=1}^{K}\cdots\sum_{k_n=1}^{K}\left[P\left(\vec{x}_1,c_{k_1},\vec{x}_2,c_{k_2},\cdots,\vec{x}_n,c_{k_n}\middle|\hat{\Theta}\right)\right]$$

$\mathbf{X}=\vec{x}_1\vec{x}_2\cdots\vec{x}_{n-1}\vec{x}_n$

$\mathbf{C}=c_{k_1}c_{k_2}\cdots c_{k_{n-1}}c_{k_n}$

How many kinds of $\mathbf{C}$?(K^n kinds)

$$=\sum_{k_i=1}^{K}\sum_{k_2=1}^{K}\cdots\sum_{k_n=1}^{K}\left[P\left(\vec{x}_1,c_{k_1},\vec{x}_2,c_{k_2},\cdots,\vec{x}_n,c_{k_n}\middle|\hat{\Theta}\right)\right]$$

$$=\sum_{\mathbf{C}}\left[P\left(\mathbf{X},\mathbf{C}\middle|\hat{\Theta}\right)\right]$$

the complete data likelihood function

The EM Algorithm (cont.)

- E–step (Expectation) known unknown
  - Define the auxiliary function $\Phi(\Theta, \hat{\Theta})$ as the expectation of the log complete likelihood function L^{CM} with respect to the hidden/latent variable C conditioned on known data (X, Θ)

$$\Phi(\Theta, \hat{\Theta}) = E\left[\log L^{CM}\right]_{C|X,\Theta} = E\left[\log P\left(X, C \middle| \hat{\Theta}\right)\right]_{C|X,\Theta}$$

$$= \sum_{C} P(C|X,\Theta) \log P\left(X, C \middle| \hat{\Theta}\right)$$

$$= \sum_{C} \frac{P(X, C|\Theta)}{P(X|\Theta)} \log P\left(X, C \middle| \hat{\Theta}\right)$$

  - Maximize the log likelihood function $\log P\left(X \middle| \hat{\Theta}\right)$ by maximizing the expectation of the log complete likelihood function $\Phi(\Theta, \hat{\Theta})$
    - We have shown this property when deriving the HMM-based retrieval model $\Phi(\Theta, \hat{\Theta}) \nearrow \Longrightarrow \log P\left(X \middle| \hat{\Theta}\right) \nearrow$

The EM Algorithm (cont.)

- E–step (Expectation)
  - The auxiliary function $\Phi\left(\Theta, \hat{\Theta}\right)$

$$\Phi\left(\Theta, \hat{\Theta}\right) = \sum_{\boldsymbol{C}} \frac{P\left(\boldsymbol{X}, \boldsymbol{C} \middle| \Theta\right)}{P\left(\boldsymbol{X} \middle| \Theta\right)} \log P\left(\boldsymbol{X}, \boldsymbol{C} \middle| \hat{\Theta}\right)$$

$$= \sum_{\boldsymbol{C} = c_{k_1} c_{k_2} \ldots c_{k_n}} \left[\prod_{j=1}^{n} \frac{P\left(\vec{x}_j, c_{k_j} \middle| \Theta\right)}{P\left(\vec{x}_j \middle| \Theta\right)}\right] \left[\log \prod_{i=1}^{n} P\left(\vec{x}_i, c_{k_i} \middle| \hat{\Theta}\right)\right]$$

$$= \sum_{\boldsymbol{C} = c_{k_1} c_{k_2} \ldots c_{k_n}} \left[\prod_{j=1}^{n} P\left(c_{k_j} \middle| \vec{x}_j, \Theta\right)\right] \left[\sum_{i=1}^{n} \log P\left(\vec{x}_i, c_{k_i} \middle| \hat{\Theta}\right)\right]$$

$$\delta_{k,k_i} = \begin{cases} 1 & \text{if } k_i = k \\ 0 & \text{otherwise} \end{cases}$$

$$= \sum_{\boldsymbol{C} = c_{k_1} c_{k_2} \ldots c_{k_n}} \sum_{k=1}^{m} \left\{ \delta_{k,k_i} \left[\sum_{i=1}^{n} \log P\left(\vec{x}_i, c_k \middle| \hat{\Theta}\right)\right] \left[\prod_{j=1}^{n} P\left(c_{k_j} \middle| \vec{x}_j, \Theta\right)\right] \right\}$$

$$= \sum_{k=1}^{m} \sum_{i=1}^{n} \left\{ \left[\log P\left(\vec{x}_i, c_k \middle| \hat{\Theta}\right)\right] \sum_{\boldsymbol{C} = c_{k_1} c_{k_2} \ldots c_{k_n}} \delta_{k,k_i} \left[\prod_{j=1}^{n} P\left(c_{k_j} \middle| \vec{x}_j, \Theta\right)\right] \right\}$$

?

See Next Slide

$$= \sum_{k=1}^{m} \sum_{i=1}^{n} \left\{ \left[\log P\left(\vec{x}_i, c_k \middle| \hat{\Theta}\right)\right] P\left(c_k \middle| \vec{x}_i, \Theta\right) \right\}$$

$$= \sum_{k=1}^{m} \sum_{i=1}^{n} \left\{ P\left(c_{k_j} \middle| \vec{x}_j, \Theta\right) \log P\left(\vec{x}_i, c_k \middle| \hat{\Theta}\right) \right\}$$

$$= \sum_{k=1}^{m} \sum_{i=1}^{n} \left\{ P\left(c_{k_j} \middle| \vec{x}_j, \Theta\right) \log \left[P\left(\vec{x}_i \middle| c_k, \hat{\Theta}\right) P\left(c_k \middle| \hat{\Theta}\right)\right] \right\}$$

$$= \sum_{k=1}^{m} \sum_{i=1}^{n} \left\{ P\left(c_{k_j} \middle| \vec{x}_j, \Theta\right) \log P\left(c_k \middle| \hat{\Theta}\right) \right\} + \sum_{k=1}^{m} \sum_{i=1}^{n} \left\{ P\left(c_{k_j} \middle| \vec{x}_j, \Theta\right) \log P\left(\vec{x}_i \middle| c_k, \hat{\Theta}\right) \right\}$$

The EM Algorithm (cont.)

Note :

$$\prod_{t=1}^{T}\left(\sum_{k_t=1}^{M} a_{tk_t}\right)$$
$$=\left(a_{11}+a_{12}+\ldots+a_{1M}\right)\left(a_{21}+a_{22}+\ldots+a_{2M}\right)\ldots\left(a_{T1}+a_{T2}+\ldots+a_{TM}\right)$$
$$=\sum_{k_1=1}^{M}\sum_{k_2=1}^{M}\ldots\sum_{k_T=1}^{M}\prod_{t=1}^{T} a_{tk_t}$$

– Note that

$$\sum_{C=c_{k_1}c_{k_2}\ldots c_{k_n}} \delta_{k,k_i}\left[\prod_{j=1}^{n} P\left(c_{k_j}\middle|\vec{x}_j,\Theta\right)\right]$$

$$=\sum_{c_{k_1}=1}^{m}\sum_{c_{k_2}=1}^{m}\cdots\sum_{c_{k_n}=1}^{m}\prod_{j=1}^{n}\left[\delta_{k,k_i} P\left(c_{k_j}\middle|\vec{x}_j,\Theta\right)\right]$$

$$=\sum_{c_{k_1}=1}^{m}\sum_{c_{k_2}=1}^{m}\cdots\sum_{c_{k_n}=1}^{m}\prod_{j=1}^{n}\left[\delta_{k,k_i} P\left(c_{k_j}\middle|\vec{x}_j,\Theta\right)\right]$$

$$=\left[\prod_{j=1,j\neq i}^{n}\left[\sum_{k_j=1}^{m} P\left(c_{k_j}\middle|\vec{x}_j,\Theta\right)\right]\right]\left[\sum_{c_{k_i}=1}^{m}\delta_{k,k_i} P\left(c_{k_i}\middle|\vec{x}_i,\Theta\right)\right]$$

$$=\left[\prod_{j=1,j\neq i}^{n} 1\right] P\left(c_k\middle|\vec{x}_i,\Theta\right)$$

$\vec{x}_i$ can only be aligned to c_k

$$=P\left(c_k\middle|\vec{x}_i,\Theta\right)$$

The EM Algorithm (cont.)

- E–step (Expectation)
  – The auxiliary function can also be divided into two:

$$\Phi\left(\Theta, \hat{\Theta}\right) = \Phi_a\left(\Theta, \hat{\Theta}\right) + \Phi_b\left(\Theta, \hat{\Theta}\right)$$

where

$$\Phi_a\left(\Theta, \hat{\Theta}\right) = \sum_{i=1}^{n}\sum_{k=1}^{K} P\left(c_k \middle| \vec{x}_i, \Theta\right)\log P\left(c_k \middle| \hat{\Theta}\right)$$

$$= \sum_{i=1}^{n}\sum_{k=1}^{K} \frac{P\left(\vec{x}_i \middle| c_k, \Theta\right)P\left(c_k \middle| \Theta\right)}{P\left(\vec{x}_i \middle| \Theta\right)}\log P\left(c_k \middle| \hat{\Theta}\right)$$

auxiliary function for mixture weights

$$= \sum_{i=1}^{n}\sum_{k=1}^{K} \frac{P\left(\vec{x}_i \middle| c_k, \Theta\right)P\left(c_k \middle| \Theta\right)}{\sum_{l=1}^{K} P\left(\vec{x}_i \middle| c_l, \Theta\right)P\left(c_l \middle| \Theta\right)}\log P\left(c_k \middle| \hat{\Theta}\right)$$

$$\Phi_b\left(\Theta, \hat{\Theta}\right) = \sum_{i=1}^{n}\sum_{k=1}^{K} P\left(c_k \middle| \vec{x}_i, \Theta\right)\log P\left(\vec{x}_i \middle| c_k, \hat{\Theta}\right)$$

auxiliary function for cluster distributions

$$= \sum_{i=1}^{n}\sum_{k=1}^{K} \frac{P\left(\vec{x}_i \middle| c_k, \Theta\right)P\left(c_k \middle| \Theta\right)}{\sum_{l=1}^{K} P\left(\vec{x}_i \middle| c_l, \Theta\right)P\left(c_l \middle| \Theta\right)}\log P\left(\vec{x}_i \middle| c_k, \hat{\Theta}\right)$$

The EM Algorithm (cont.)

- M-step (Maximization)
  - Remember that
    - Maximize a function F with a constraint by applying Lagrange multiplier

By applying Lagrange Multiplier ℓ

Suppose that $F=\sum_{j=1}^{N} w_j \log y_j \Rightarrow \hat{F}=\sum_{j=1}^{N} w_j \log y_j + \ell\left(\sum_{j=1}^{N} y_j - 1\right)$

Constraint

$$\frac{\partial \hat{F}}{\partial y_j}=\frac{w_j}{y_j}+\ell=0 \Rightarrow \ell=-\frac{w_j}{y_j}\ \forall\ j$$

$$\ell\sum_{j=1}^{N} y_j=-\sum_{j=1}^{N} w_j \Rightarrow \ell=-\sum_{j=1}^{N} w_j$$

$$\therefore y_j=\frac{w_j}{\sum_{j=1}^{N} w_j}$$

Note :

$$\frac{\partial \log y_j}{\partial y_j}=\frac{1}{y_j}$$

The EM Algorithm (cont.)

- M-step (Maximization)
  - Maximize $\Phi_a\left(\Theta, \hat{\Theta}\right)$

auxiliary function for
mixture weights (or priors for Gaussians)

$$\overline{\Phi}_a\left(\Theta, \hat{\Theta}\right) = \Phi_a\left(\Theta, \hat{\Theta}\right) + l\left(\sum_{k=1}^{K} P\left(c_k \middle| \hat{\Theta}\right) - 1\right)$$

$$= \sum_{k=1}^{K} \sum_{i=1}^{n} \frac{P\left(\vec{x}_i \middle| c_k, \Theta\right) P\left(c_k \Theta\right)}{\sum_{l=1}^{K} P\left(\vec{x}_i \middle| c_l, \Theta\right) P\left(c_l \middle| \Theta\right)} \log P\left(c_k \middle| \hat{\Theta}\right) + l\left(\sum_{k=1}^{K} P\left(c_k \middle| \hat{\Theta}\right) - 1\right)$$

w_k $\quad$ y_k

r_k^i : the expected number of times that $\vec{x}_i$ falls in class c_k

$$\Rightarrow \hat{\pi}_k = P\left(c_k \middle| \hat{\Theta}\right) = \frac{w_k}{\sum_{k=1}^{K} w_k} = \frac{\sum_{i=1}^{n} \frac{P\left(\vec{x}_i \middle| c_k, \Theta\right) P\left(c_k \middle| \Theta\right)}{\sum_{l=1}^{K} P\left(\vec{x}_i \middle| c_l, \Theta\right) P\left(c_l \middle| \Theta\right)}}{\sum_{k=1}^{K} \sum_{i=1}^{n} \frac{P\left(\vec{x}_i \middle| c_k, \Theta\right) P\left(c_k \middle| \Theta\right)}{\sum_{l=1}^{K} P\left(\vec{x}_i \middle| c_l, \Theta\right) P\left(c_l \middle| \Theta\right)}} = \frac{\sum_{i=1}^{n} \frac{P\left(\vec{x}_i \middle| c_k, \Theta\right) P\left(c_k \middle| \Theta\right)}{\sum_{l=1}^{K} P\left(\vec{x}_i \middle| c_l, \Theta\right) P\left(c_l \middle| \Theta\right)}}{n}$$

The EM Algorithm (cont.)

- M-step (Maximization)
  - Maximize $\Phi_b(\Theta, \hat{\Theta})$

auxiliary function for
(multivariate) Gaussian Means and Variances

$$P(\vec{x}_i | c_k;\Theta) = \frac{1}{\sqrt{(2\pi)^m |\Sigma_k|}} \exp\left(-\frac{1}{2}(\vec{x}_i - \vec{\mu}_k)^T \Sigma_k^{-1} (\vec{x}_i - \vec{\mu}_k)\right)$$

$$\Phi_b(\Theta, \hat{\Theta}) = \sum_{i=1}^{n} \sum_{k=1}^{K} \frac{P(\vec{x}_i | c_k, \Theta) P(c_k | \Theta)}{\sum_{l=1}^{K} P(\vec{x}_i | c_l, \Theta) P(c_l | \Theta)} \log P(\vec{x}_i | c_k, \hat{\Theta})$$

Let $w_{k,i} = \frac{P(\vec{x}_i | c_k, \Theta) P(c_k | \Theta)}{\sum_{l=1}^{K} P(\vec{x}_i | c_l, \Theta) P(c_l | \Theta)}$ and $\log P(\vec{x}_i | c_k;\Theta) =$

$$-\frac{m}{2} \cdot \log(2\pi) - \frac{1}{2} \log |\Sigma_k| - \frac{1}{2}(\vec{x}_i - \vec{\mu}_k)^T \Sigma_k^{-1} (\vec{x}_i - \vec{\mu}_k)$$

$$\Rightarrow \Phi_b(\Theta, \hat{\Theta}) = -\sum_{i=1}^{n} \sum_{k=1}^{K} w_{k,i} \left[\frac{1}{2} \log |\Sigma_k| + \frac{1}{2}(\vec{x}_i - \hat{\vec{\mu}}_k)^T \hat{\Sigma}_k^{-1} (\vec{x}_i - \hat{\vec{\mu}}_k)\right] + D$$

constant

The EM Algorithm (cont.)

- M-step (Maximization)
  - Maximize $\Phi_b(\Theta, \hat{\Theta})$ with respect to $\vec{\mu}_k$

$$\frac{d\,(\boldsymbol{x}^T \boldsymbol{C}\boldsymbol{x})}{d\boldsymbol{x}} = \left(\boldsymbol{C} + \boldsymbol{C}^T\right)\boldsymbol{x}$$
and Σ_k^{-1} is symmetric here

$$\Phi_b(\Theta, \hat{\Theta}) = -\sum_{i=1}^{n}\sum_{k=1}^{K} w_{k,i}\left[\tfrac{1}{2}\log\left|\hat{\Sigma}_k\right| + \tfrac{1}{2}\left(\vec{x}_i - \hat{\vec{\mu}}_k\right)^T \hat{\Sigma}_k^{-1}\left(\vec{x}_i - \hat{\vec{\mu}}_k\right)\right] + D$$

$$\frac{\partial \Phi_b(\Theta, \hat{\Theta})}{\partial \hat{\vec{\mu}}_k} = -\sum_{i=1}^{n} w_{k,i} \cdot \tfrac{1}{2} \cdot (2) \cdot \hat{\Sigma}_k^{-1}\left(\vec{x}_i - \hat{\vec{\mu}}_k\right)(-1) = 0$$

$$\Rightarrow \hat{\vec{\mu}}_k = \frac{\sum_{i=1}^{n} w_{k,i} \cdot \vec{x}_i}{\sum_{i=1}^{n} w_{k,i}} = \frac{\sum_{i=1}^{n} \frac{P(\vec{x}_i | c_k, \Theta) P(c_k | \Theta)}{\sum_{l=1}^{K} P(\vec{x}_i | c_l, \Theta) P(c_l | \Theta)} \cdot \vec{x}_i}{\sum_{i=1}^{n} \frac{P(\vec{x}_i | c_k, \Theta) P(c_k | \Theta)}{\sum_{l=1}^{K} P(\vec{x}_i | c_l, \Theta) P(c_l | \Theta)}}$$

r_k^i : the expected number of times that $\vec{x}_i$ falls in class c_k

The EM Algorithm (cont.)

- M-step (Maximization)
  - Maximize $\Phi_b(\Theta, \hat{\Theta})$ with respect to Σ_k

$$\Phi_b(\Theta, \hat{\Theta}) = -\sum_{i=1}^{n}\sum_{k=1}^{K} w_{k,i}\left[\frac{1}{2}\log\left|\hat{\Sigma}_k\right| + \frac{1}{2}(\vec{x}_i - \hat{\vec{\mu}}_k)^T \hat{\Sigma}_k^{-1}(\vec{x}_i - \hat{\vec{\mu}}_k)\right] + D$$

$$\frac{\partial \Phi_b(\Theta, \hat{\Theta})}{\partial \hat{\Sigma}_k} = -\sum_{i=1}^{n}\frac{1}{2}\cdot w_{k,i}\left[\left|\hat{\Sigma}_k\right|^{-1}\cdot\left|\hat{\Sigma}_k\right|\cdot\hat{\Sigma}_k^{-1} - \Sigma_k^{-1}(\vec{x}_i - \hat{\vec{\mu}}_k)(\vec{x}_i - \hat{\vec{\mu}}_k)^T\Sigma_k^{-1}\right] = 0$$

$$\Rightarrow \sum_{i=1}^{n} w_{k,i}\cdot\hat{\Sigma}_k^{-1} = \sum_{i=1}^{n} w_{k,i}\cdot\hat{\Sigma}_k^{-1}(\vec{x}_i - \hat{\vec{\mu}}_k)(\vec{x}_i - \hat{\vec{\mu}}_k)^T\hat{\Sigma}_k^{-1}$$

$$\Rightarrow \sum_{i=1}^{n} w_{k,i}\cdot\hat{\Sigma}_k\hat{\Sigma}_k^{-1}\hat{\Sigma}_k = \sum_{i=1}^{n} w_{k,i}\cdot\hat{\Sigma}_k\hat{\Sigma}_k^{-1}(\vec{x}_i - \hat{\vec{\mu}}_k)(\vec{x}_i - \hat{\vec{\mu}}_k)^T\hat{\Sigma}_k^{-1}\hat{\Sigma}_k$$

$$\Rightarrow \sum_{i=1}^{n} w_{k,i}\cdot\hat{\Sigma}_k = \sum_{i=1}^{n} w_{k,i}\cdot(\vec{x}_i - \hat{\vec{\mu}}_k)(\vec{x}_i - \hat{\vec{\mu}}_k)^T$$

$$\frac{d[\det(\boldsymbol{X})]}{d\boldsymbol{X}} = \det(\boldsymbol{X})\cdot\boldsymbol{X}^{-T}$$

and Σ_k is symmetric here

$$\frac{d\,(\boldsymbol{a}^T\boldsymbol{X}^{-1}\boldsymbol{b})}{d\boldsymbol{X}} = -\boldsymbol{X}^{-1}\boldsymbol{a}\boldsymbol{b}^T\boldsymbol{X}^{-1}$$

$$\Rightarrow \hat{\Sigma}_k = \frac{\sum_{i=1}^{n} w_{k,i}\cdot(\vec{x}_i - \hat{\vec{\mu}}_k)(\vec{x}_i - \hat{\vec{\mu}}_k)^T}{\sum_{i=1}^{n} w_{k,i}} = \frac{\sum_{i=1}^{n}\frac{P(\vec{x}_i|c_k,\Theta)P(c_k|\Theta)}{\sum_{l=1}^{K}P(\vec{x}_i|c_l,\Theta)P(c_l|\Theta)}\cdot(\vec{x}_i - \hat{\vec{\mu}}_k)(\vec{x}_i - \hat{\vec{\mu}}_k)^T}{\sum_{i=1}^{n}\frac{P(\vec{x}_i|c_k,\Theta)P(c_k|\Theta)}{\sum_{l=1}^{K}P(\vec{x}_i|c_l,\Theta)P(c_l|\Theta)}}$$

The EM Algorithm (cont.)

- The initial cluster distributions can be estimated using the *K*-means algorithm

- The procedure terminates when the likelihood function $P(X|\Theta)$ is converged or maximum number of iterations is reached