



PERGAMON

Available at
www.ElsevierComputerScience.com

POWERED BY SCIENCE @ DIRECT®

Pattern Recognition 38 (2005) 637–649

PATTERN
RECOGNITION

THE JOURNAL OF THE PATTERN RECOGNITION SOCIETY

www.elsevier.com/locate/patcog

Scalable model-based cluster analysis using clustering features

Huidong Jin^{a,*,1}, Kwong-Sak Leung^b, Man-Leung Wong^c, Zong-Ben Xu^d

^aCSIRO Data Mining Research, GPO Box 664, Canberra ACT 2601, Australia

^bDepartment of Computer Science & Engineering, CUHK, Shatin, Hong Kong

^cDepartment of Computing and Decision Sciences, Lingnan University, Tuen Mun, Hong Kong

^dFaculty of Science, Xi'an Jiaotong University, 710049 Xi'an, P.R. China

Received 9 May 2003; received in revised form 15 July 2004; accepted 15 July 2004

Abstract

We present two scalable model-based clustering systems based on a Gaussian mixture model with independent attributes within clusters. They first summarize data into sub-clusters, and then generate Gaussian mixtures from their clustering features using a new algorithm—EMACF. EMACF approximates the aggregate behavior of each sub-cluster of data items in the Gaussian mixture model. It provably converges. The experiments show that our clustering systems run one or two orders of magnitude faster than the traditional EM algorithm with few losses of accuracy.

© 2004 Pattern Recognition Society. Published by Elsevier Ltd. All rights reserved.

Keywords: Cluster analysis; Data mining; Scalable; Gaussian mixture model; Expectation maximization; Clustering feature; Convergence

1. Introduction

With the explosive growth of data amassed from business, scientific and engineering disciplines, scalable cluster analysis and other data mining functionalities play a more and more important role [1–3]. Among many clustering techniques [1,4–6], model-based clustering techniques have attracted much research interest [2,7–11]. They can identify clusters of a variety of shapes and can handle complicated data sets with different kinds of attributes [12,13]. They have solid probabilistic foundations [14–17]. They have also been successfully applied to various real-life applications, such

as image segmentation [7], microarray gene expression data clustering [10], Web navigation pattern recognition [11], and OLAP aggregate query optimization [18]. This paper concentrates on scalable cluster analysis based on a Gaussian mixture model with independent attributes within each cluster.

Expectation maximization (EM) is an iterative algorithm for finding a maximum likelihood estimate (MLE) of a mixture model. It normally generates more accurate clustering results than hierarchical model-based clustering [19] and the incremental EM algorithm [3,17]. Though some attempts have been made to speed up the algorithm [12,16,17], EM and its extensions are still computationally expensive for large data sets, especially when they are too large to be stored in main memory. In particular, the lazy EM algorithm [20] evaluates the significance of each data item at scheduled iterations and then proceeds for several iterations actively using only the significant ones. However, its speedup factor is less than three. Moore [21] first used a KD-tree to cache sufficient statistics of interesting regions of data, and then applied EM to the KD-tree nodes. His algorithm handles low-dimensional data sets efficiently, but its

* Corresponding author. Tel.: +61 2 62167258;
fax: +61 2 62167111.

E-mail addresses: Warren.Jin@csiro.au (H. Jin),
ksleung@cse.cuhk.edu.hk (K.-S. Leung), mlwong@ln.edu.hk
(M.-L. Wong), zbxu@mail.xjtu.edu.cn (Z.-B. Xu).

¹ The work was submitted when the author was with Department of Computing and Decision Sciences, Lingnan University, Hong Kong.

performance degenerates dramatically as the dimensionality increases [21]. The scalable EM (SEM) algorithm [3] uses the extended EM (ExEM) algorithm to identify compressible data regions, and then only retains their sufficient statistics in order to load next batch of data. It needs to invoke ExEM many times, and hence its speedup factor is less than 10 [3].

In this paper, we present two scalable model-based clustering systems that can run one or two orders of magnitude faster than the traditional EM algorithm for the Gaussian mixture model. Moreover, there is little or no sacrifice in the clustering quality. They, using similar computational resources, can also generate significantly more accurate clustering results than the existing scalable model-based clustering systems. Their basic idea is to incrementally summarize a data set into sub-clusters first, and then generate a mixture estimate from their clustering features directly by a specifically designed EM algorithm—EM algorithm for clustering features (EMACF). EMACF works on the clustering features of sub-clusters. It is associated with a pseudo mixture model that approximates the aggregate behavior of each sub-cluster of data items in the Gaussian mixture model. Thus, it can efficiently generate good estimates of the Gaussian mixture model from the clustering features.

The rest of the paper is organized as follows. Two clustering systems, gEMACF and bEMACF, are proposed in Section 2. In Section 3, two data summarization procedures are presented to generate clustering features. In Section 4, EMACF is derived and analyzed. Section 5 describes the experimental setup and results, followed by discussion and conclusion in Section 6.

2. Two scalable model-based clustering systems

Given a data set $X = \{\mathbf{x}_1, \dots, \mathbf{x}_N\}$ of size N , model-based clustering techniques assume that each data item $\mathbf{x}_i = [x_{1i}, \dots, x_{Di}]^T \in \mathbb{R}^D$ is drawn from a K -component mixture model Φ :

$$p(\mathbf{x}_i|\Phi) = \sum_{k=1}^K p_k \phi(\mathbf{x}_i|\theta_k). \quad (1)$$

Here, $\phi(\mathbf{x}_i|\theta_k)$ is a *component density function* with parameters θ_k , and it represents a cluster, and p_k is the mixing proportion of the cluster ($0 < p_k < 1$ for $k = 1, \dots, K$, and $\sum_{k=1}^K p_k = 1$). Given Φ , a crisp clustering is got by assigning a data item \mathbf{x}_i to cluster k where its posterior probability reaches maximum, i.e., $k = \arg \max_l \{p_l \phi(\mathbf{x}_i|\theta_l)\}$.

A Gaussian mixture model follows Eq. (1) but each function $\phi(\mathbf{x}_i|\theta_k)$ indicates a multivariate Gaussian distribution. Gaussian mixture models can effectively approximate any distribution [3]. They have successfully been used in a variety of real-life applications [7,10,11,13,18]. Thus, research efforts on Gaussian mixture models are theoretically and practically important. In this paper, we concentrate on a parsimonious Gaussian mixture model where, conditional on

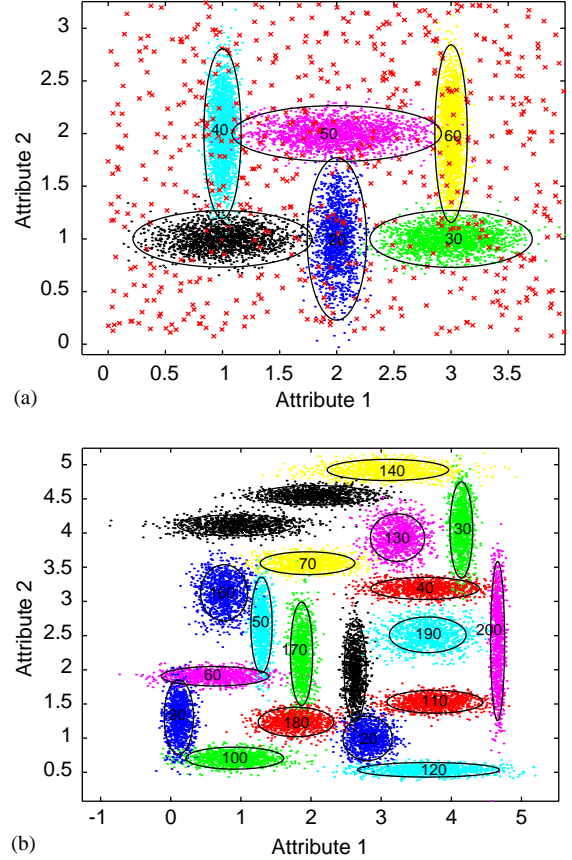


Fig. 1. Illustration of two synthetic data sets (20% plotted). A dot indicates a data item. An ellipse and its associated “o” indicate a contour and the center of a Gaussian distribution component, respectively. “x” indicates noise. (a) The first data set, and (b) the fourth data set.

clusters, attributes are independent [12]. Its component density function is

$$\phi(\mathbf{x}_i|\theta_k) = \prod_{d=1}^D \frac{\exp\{-(x_{di} - \mu_{dk})^2/2\sigma_{dk}\}}{(2\pi\sigma_{dk})^{1/2}},$$

where parameter θ_k consists of a mean vector $\mu_k = [\mu_{1k}, \dots, \mu_{Dk}]^T$ and a variance vector $\sigma_k = [\sigma_{1k}, \dots, \sigma_{Dk}]^T$. Two data sets generated according to the Gaussian mixture model are illustrated in Fig. 1. Given the number of clusters K , EM for the Gaussian mixture model estimates its parameters to maximize log-likelihood $L(\Phi) = \sum_{i=1}^N \log p(\mathbf{x}_i|\Phi)$ iteratively. It alternates between the following two steps.

- (1) *E-step*: Given the model parameters at iteration j , compute the membership probability $t_{ik}^{(j)}$:

$$t_{ik}^{(j)} = \frac{p_k^{(j)} \phi(\mathbf{x}_i|u_k^{(j)}, \sigma_k^{(j)})}{\sum_{l=1}^K p_l^{(j)} \phi(\mathbf{x}_i|u_l^{(j)}, \sigma_l^{(j)})}. \quad (2)$$

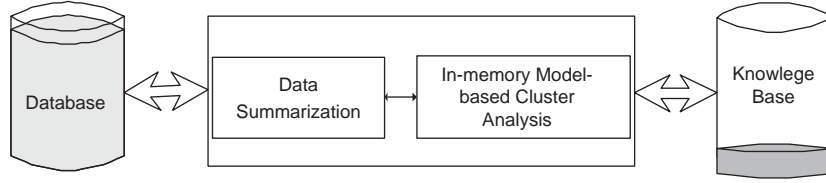


Fig. 2. A scalable model-based clustering framework.

(2) *M-step*: Given $t_{ik}^{(j)}$, update the mixture model parameters for $k = 1, \dots, K$ from the N data items:

$$p_k^{(j+1)} = \frac{1}{N} \sum_{i=1}^N t_{ik}^{(j)}, \quad (3)$$

$$\mu_k^{(j+1)} = \frac{\sum_{i=1}^N t_{ik}^{(j)} \mathbf{x}_i}{N \cdot p_k^{(j+1)}}, \quad (4)$$

$$\sigma_k^{(j+1)} = \frac{\sum_{i=1}^N t_{ik}^{(j)} (\mathbf{x}_i - \mu_k^{(j+1)}) \otimes (\mathbf{x}_i - \mu_k^{(j+1)})}{N \cdot p_k^{(j+1)}}, \quad (5)$$

where \otimes is the array multiplication operation, i.e., $(A \otimes B)_{ij} = a_{ij}b_{ij}$.

EM can generate very accurate results and is widely used in practice [12,16,18]. But it needs to scan the whole data set for each iteration, which prohibits it from handling large databases [2,3].

There are three strategies to scale-up iterative clustering algorithms such as EM [22]. The first is to analyze random samples from the data set. This is easy to achieve, but often performs badly due to sampling biases [1,23]. The second strategy is to analyze weighted samples. The weighted (pseudo) data items emulate the local distribution of the original data set [6,23]. This strategy requires a slight modification to traditional clustering techniques. However, as shown in Section 5, when this is used to scale-up model-based clustering, the performance depends on the sampling procedure and often degenerates greatly. The third one is to construct summary statistics of the large data set on which to base the desired analysis [3,4]. It usually involves several phases.

Our scalable model-based clustering framework falls into the third strategy. It is motivated by the following observations. In a scalable clustering system, we usually handle a sub-cluster of similar data items as an object in order to reduce computational resources. Within model-based clustering, a component density function essentially determines clustering results. Thus, for a sub-cluster of similar data items, a new pseudo-component density function should be introduced so as to remedy the possible loss of clustering quality caused by handling trivially the data items as their mean vector. Such a loss often happens in the second scaling-up strategy. A new mixture model, defined over the summary statistics, can approximate the aggregate behavior of each

sub-cluster of data items in the original one. Then, its associated model-based clustering algorithm, e.g., the one derived from the general EM algorithm [16], effectively generates a good estimate of the original mixture model from the summary statistics. Thus, as illustrated in Fig. 2, our framework consists of the following two phases.

- (1) *Data summarization*: A large data set is partitioned into mutually exclusive sub-clusters, and only summary statistics of these sub-clusters are retained in the main memory.
- (2) *In-memory model-based cluster analysis*: A mixture is generated from the summary statistics directly by the new EM algorithm associated with the pseudo-mixture model.

We introduce a *clustering feature* to serve as the summary statistics of a sub-cluster. The clustering feature includes variance information because the parsimonious Gaussian distribution functions embody variance vectors. For the m th sub-cluster, its *clustering feature* is a triplet $\mathbf{s}_m = \{n_m, v_m, \gamma_m\}$ ($m = 1, \dots, M$), where n_m is the number of data items in the m th sub-cluster, $v_m = [v_{1m}, \dots, v_{Dm}]^T = (1/n_m) \sum_{\mathbf{x}_i \in CF_m} \mathbf{x}_i$, and $\gamma_m = [\gamma_{1m}, \dots, \gamma_{Dm}]^T = (1/n_m) \sum_{\mathbf{x}_i \in CF_m} \mathbf{x}_i \otimes \mathbf{x}_i$. Here $\mathbf{x}_i \in CF_m$ indicates \mathbf{x}_i belongs to the m th sub-cluster. Similar to the one in Ref. [4], the clustering feature contains the zeroth, first, and second moments of the sub-cluster [1]. It has a simple additivity property [4,22]. This facilitates our incremental data summarization procedures. A grid-based and the BIRCH's data summarization procedures will be outlined in Section 3. In Section 4, we will derive EMACF which generates Gaussian mixtures from clustering features directly. Combining EMACF with these two data summarization procedures, we then can establish two scalable model-based clustering systems, called gEMACF and bEMACF, respectively.

3. Data summarization procedures

The data summarization procedure sums up similar data items into clustering features according to a definition of sub-clusters. This section outlines two possible data summarization procedures.

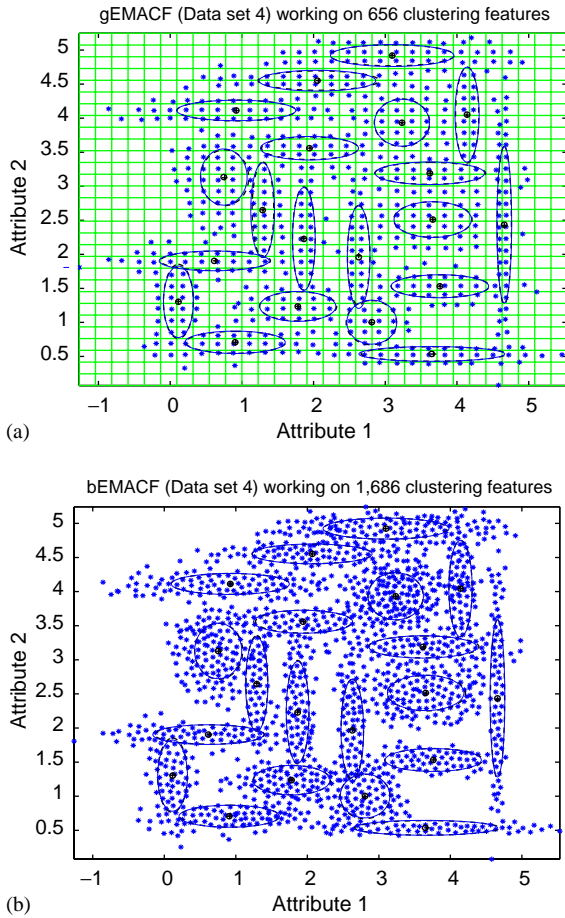


Fig. 3. Clusters generated by gEMACF and bEMACF for the fourth data set. Clustering features are indicated by “*”. An “o” and an ellipse indicate a generated Gaussian component, while a “+” and a dashed ellipse indicate an original one. (a) Clusters generated by gEMACF, and (b) clusters generated by bEMACF.

The grid-based data summarization procedure partitions a data set by imposing a multidimensional grid structure in the data space, and then incrementally sums up the data items within a cell into its clustering feature. That is, the data items within a cell form a sub-cluster. For simplicity, each attribute may be partitioned into several equal-width segments by grids. Thus, each cell has the same width in each attribute and has the same volume, as exemplified in Fig. 3(a). To operate within the given main memory, we only store clustering features for the non-empty cells in a clustering feature (CF) array: CF-array. This CF-array has a fixed number of entries, M , according to the given amount of main memory. When a new data item is input, we calculate in which cell it is located. Then we efficiently search for its associated entry in the CF-array by using a hash function [22]. If a corresponding entry is found, its clustering feature is updated to absorb the data item. Otherwise, a new

entry will be allocated to store the clustering feature of the cell.

Our grid-based data summarization procedure adaptively determines the cell width to make better use of the given main memory. At the beginning, the cell widths are initialized to some reasonable values. If the cell widths are very small, then the number of non-empty cells may be greater than the number of entries in the CF-array. When the entries in the CF-array are used up, the grid-based data summarization procedure increases the cell widths to squash all of the clustering features into the given main memory. During the adaptation procedure, the CF-array is rebuilt. To avoid reading through the whole data set again, the rebuilding procedure merges every two adjacent cells into one along the dimension with the smallest width. Hence, a new clustering feature is directly calculated from the two old ones due to the additivity property.

If Euclidean distance is used to define the similarity among data items within sub-clusters, we can employ existing distance-based clustering techniques [1], such as BIRCH [4], to generate sub-clusters from a data set. BIRCH uses the clustering feature and the CF-tree to summarize cluster representations [4]. It scans the data set to build an initial in-memory CF-tree, which can be viewed as a multilevel compression of the data set that tries to preserve its inherent clustering structure. The CF-tree is built dynamically as data items are input. A data item is inserted into the closest leaf entry. If, after insertion, the diameter of the sub-cluster stored in the leaf node is greater than a threshold value, then the leaf node and possibly other nodes are split. After inserting a new data item, its information is passed toward the root of the tree. The size of the CF-tree can be changed by modifying the threshold. If the amount of memory required for storing the CF-tree is greater than the given amount of main memory, then a larger threshold value is specified and the CF-tree is rebuilt. The rebuilding process is conducted without reading the data items again. This is similar to the insertion and node split in the construction of B^+ -trees. Therefore, for building the CF-tree, data items have to be read only once. Fig. 3(b) plots the clustering features generated by the BIRCH's data summarization procedure from the data set shown in Fig. 1(b). Once all data items are stored in the CF-tree, BIRCH applies a hierarchical agglomerative clustering algorithm to cluster the leaf nodes of the CF-tree [4]. If the clusters are not spherical in shape, e.g., the clusters in Fig. 1(b), BIRCH does not perform well because it uses the notion of radius or diameter to control the boundary of a cluster [1,3,22].

Both the BIRCH's and the grid-based data summarization procedures attempt to generate good clustering features using the restricted computational resources. Their computation complexity is linear with the number of data items, but the storage requirement is linear with the number of sub-clusters. They both read through the data set once. However, the former uses a tree indexing, while the latter employs a hash indexing. The former makes better use of memory,

while the latter is simpler to implement and manipulate, especially for low-dimensional data sets.

4. Derivation and analysis of EMACF

Before deriving EMACF for getting good estimates of the Gaussian mixture model, we establish a new pseudo-component density function. It is only based on to which sub-cluster a data item \mathbf{x}_i belongs since individual data are inaccessible at this stage. We also embed the sub-cluster variance into our pseudo-density function so as to pay more attention to denser data area as Gaussian distribution does.

Definition 1. For a data item \mathbf{x}_i in the m th sub-cluster, its probability in the pseudo-component density function ψ with parameters $\theta_k = \{\mu_k, \sigma_k\}$ is

$$\begin{aligned} p(\mathbf{x}_i \in \text{CF}_m | \theta_k) &\triangleq \psi(\mathbf{s}_m | \theta_k) \\ &= \prod_{d=1}^D \frac{\exp[-((\gamma_{dm} - v_{dm}^2) + (v_{dm} - \mu_{dk})^2)/2\sigma_{dk}]}{(2\pi)^{1/2} \sigma_{dk}^{1/2}}. \end{aligned} \quad (6)$$

In this pseudo-density function, the dispersion of a sub-cluster influences the probability of its associated data items. The smaller $(\gamma_{dm} - v_{dm}^2)$ is, the larger the probability is. In other words, the data item within a denser sub-cluster has a relatively larger probability, and vice versa. This point complies with the Gaussian mixture model which pays more attention to denser data areas. This pseudo-density function is equivalent to a Gaussian density function when $\gamma_{dm} - v_{dm}^2 = 0$, i.e., the sub-cluster variance is zero. However, it is not a genuine density function in general since its integral is less than 1.0. With this function, we can uniformly treat data items within each sub-cluster without accessing data items, and then reduce much computation time and main memory for large data sets. Furthermore, we shall see that its associated EM algorithm, EMACF, can approximate the aggregate behavior of each sub-cluster. Hence, the pseudo-density function is practicable.

With the pseudo-density function in Eq. (6), a K -component pseudo-mixture model Ψ for each data item $\mathbf{x}_i (\in \text{CF}_m)$ can be defined as

$$p(\mathbf{x}_i \in \text{CF}_m | \Psi) \triangleq p(\mathbf{s}_m | \Psi) = \sum_{k=1}^K p_k \psi(\mathbf{s}_m | \mu_k, \sigma_k). \quad (7)$$

Its log-likelihood that indicates the fitness of the mixture model is defined as

$$L(\Psi) = \log \left[\prod_{i=1}^N p(\mathbf{x}_i | \Psi) \right] = \sum_{m=1}^M n_m \log p(\mathbf{s}_m | \Psi). \quad (8)$$

Here N and M are the numbers of data items and sub-clusters, respectively. The pseudo-mixture model Ψ has the same

parameters as the Gaussian mixture model Φ . In addition, this model can approximate the aggregate behavior of a sub-cluster of data items in the Gaussian mixture model. Thus, good Gaussian mixtures Φ can be obtained by finding MLEs of Ψ .

We now derive EMACF to get MLEs of the pseudo-mixture model Ψ . The derivation is based on the general EM algorithm by interpreting cluster labels as ‘missing’ values. If \mathbf{x}_i is in the k th cluster, then its indicator vector $\mathbf{z}_i = [z_{1i}, \dots, z_{Ki}]^T$ equals 0 except that z_{ki} equals one. Then the complete data vector is $\mathbf{y}_i = [\mathbf{x}_i^T, \mathbf{z}_i^T]^T$. Its likelihood is $g(\mathbf{y}_i | \Psi) = p(\mathbf{x}_i | \mathbf{z}_i, \Psi) p(\mathbf{z}_i | \Psi) = \psi(\mathbf{x}_i | \theta_k) p_k = \prod_{k=1}^K [\psi(\mathbf{x}_i | \theta_k) p_k]^{z_{ki}}$. This holds because z_{ki} is either 0 or 1. For the N data items, we get the likelihood for the complete data set

$$\begin{aligned} g(\mathbf{y}_1, \dots, \mathbf{y}_N | \Psi) &= \prod_{i=1}^N \prod_{k=1}^K [\psi(\mathbf{x}_i | \theta_k) p_k]^{z_{ki}} \\ &= \prod_{m=1}^M \prod_{k=1}^K [\psi(\mathbf{s}_m | \theta_k) p_k]^{\bar{z}_{km} n_m}. \end{aligned} \quad (9)$$

It holds because, in the pseudo-mixture model Ψ , any \mathbf{x}_i in the m th sub-cluster has the same indicator vector denoted by $\bar{\mathbf{z}}_m = [\bar{z}_{1m}, \dots, \bar{z}_{Km}]^T$. In the E -step, we compute the expectation of the complete data log-likelihood, $Q(\Psi; \Psi^{(j)})$, conditional on the observed data $\{\mathbf{x}\}$ (which is replaced by $\{\mathbf{s}\}$ below) and the current parameter value $\Psi^{(j)}$.

$$\begin{aligned} Q(\Psi; \Psi^{(j)}) &= E[\log g(\{\mathbf{y}\} | \Psi) | \{\mathbf{x}\}, \Psi^{(j)}] \\ &= E[\log g(\{\mathbf{y}\} | \Psi) | \{\mathbf{s}\}, \Psi^{(j)}] \end{aligned} \quad (10)$$

$$\begin{aligned} &= \sum_{m=1}^M n_m \sum_{k=1}^K E[\bar{z}_{km} | \{\mathbf{s}\}, \Psi^{(j)}] [\log p_k \\ &\quad + \log \psi(\mathbf{s}_m | \mu_k^{(j)}, \sigma_k^{(j)})] \\ &\triangleq \sum_{m=1}^M n_m \sum_{k=1}^K r_{mk}^{(j)} [\log p_k + \log \psi(\mathbf{s}_m | \mu_k^{(j)}, \sigma_k^{(j)})]. \end{aligned} \quad (11)$$

The random variable \bar{z}_{km} corresponds to \bar{z}_{km} . The membership probability of $\mathbf{x}_i (\in \text{CF}_m)$ for the k th component, $r_{mk}^{(j)}$, is got according to Bayes’ rule:

$$\begin{aligned} r_{mk}^{(j)} &\triangleq E[\bar{z}_{km} | \{\mathbf{s}\}, \Psi^{(j)}] = p_{\Psi^{(j)}}(\bar{z}_{km} = 1 | \{\mathbf{s}\}) \\ &= \frac{p_k^{(j)} \psi(\mathbf{s}_m | \mu_k^{(j)}, \sigma_k^{(j)})}{\sum_{l=1}^K p_l^{(j)} \psi(\mathbf{s}_m | \mu_l^{(j)}, \sigma_l^{(j)})}. \end{aligned} \quad (12)$$

In the M -step, we maximize $Q(\Psi; \Psi^{(j)})$ in Eq. (11) with respect to Ψ . To re-estimate the mixing proportion p_k , we introduce a Lagrange multiplier λ to handle the constraint $\sum_{k=1}^K p_k = 1$. Differentiating $Q(\Psi; \Psi^{(j)}) - \lambda(\sum_{k=1}^K p_k - 1)$

with respect to p_k , we get

$$\sum_{m=1}^M \frac{n_m r_{mk}^{(j)}}{p_k} - \lambda = 0 \quad \text{for } k = 1, \dots, K.$$

Summing up these K equations together, one has $\lambda = N$, and then gets the re-estimate of p_k , $\hat{p}_k = (1/N) \sum_{m=1}^M n_m r_{mk}^{(j)}$. For the other parameters, their partial derivatives on the pseudo-component density function are

$$\frac{\partial \log \psi(\mathbf{s}_m | \mu_k, \sigma_k)}{\partial \mu_{dk}} = \frac{\gamma_{dm} - \mu_{dk}}{\sigma_{dk}}$$

and

$$\frac{\partial \log \psi(\mathbf{s}_m | \mu_k, \sigma_k)}{\partial \sigma_{dk}} = \frac{\gamma_{dm} - 2\mu_{dk}\gamma_{dm} + \mu_{dk}^2}{2\sigma_{dk}^2} - \frac{1}{2\sigma_{dk}}.$$

Differentiating $Q(\Psi; \Psi^{(j)})$ with respect to μ_{dk} and equating the partial differential to zero gives

$$\frac{\partial Q(\Psi; \Psi^{(j)})}{\partial \mu_{dk}} = \sum_{m=1}^M n_m r_{mk}^{(j)} \frac{1}{\sigma_{dk}} (\gamma_{dm} - \mu_{dk}) = 0.$$

This gives the re-estimate of μ_{dk} as

$$\hat{\mu}_{dk} = \frac{\sum_{m=1}^M n_m r_{mk}^{(j)} \gamma_{dm}}{\sum_{m=1}^M n_m r_{mk}^{(j)}}.$$

So, the new cluster center $\hat{\mu}_k$ is a weighted average of the sub-cluster means. Similarly, differentiating $Q(\Psi; \Psi^{(j)})$ with respect to σ_{dk} and equating it to zero leads to

$$\hat{\sigma}_{dk} = \frac{\sum_{m=1}^M n_m r_{mk}^{(j)} (\gamma_{dm} - 2\mu_{dk}\gamma_{dm} + \mu_{dk}^2)}{\sum_{m=1}^M n_m r_{mk}^{(j)}}. \quad (13)$$

It is worth pointing out that this re-estimate approximates the aggregate behavior of a sub-cluster of data items in the Gaussian mixture model. Each \mathbf{x}_i in the m th sub-cluster is similar to one another, and then has a similar $t_{ik}^{(j)}$ in Eqs. (2)–(5). We approximate these $t_{ik}^{(j)}$ with $r_{mk}^{(j)}$ in Eq. (5), and see the aggregate behavior of the sub-cluster in the Gaussian mixture model as follows:

$$\begin{aligned} \sigma_{dk}^{(j+1)} &= \frac{\sum_{m=1}^M \sum_{\mathbf{x}_i \in \text{CF}_m} t_{ik}^{(j)} (x_{di} - \mu_{dk}^{(j+1)})^2}{\sum_{m=1}^M \sum_{\mathbf{x}_i \in \text{CF}_m} t_{ik}^{(j)}} \\ &\approx \frac{\sum_{m=1}^M \sum_{\mathbf{x}_i \in \text{CF}_m} r_{mk}^{(j)} (x_{di} - \mu_{dk}^{(j+1)})^2}{\sum_{m=1}^M \sum_{\mathbf{x}_i \in \text{CF}_m} r_{mk}^{(j)}} \\ &= \frac{\sum_{m=1}^M n_m r_{mk}^{(j)} [\gamma_{dm} - 2\gamma_{dm}\mu_{dk}^{(j+1)} + (\mu_{dk}^{(j+1)})^2]}{\sum_{m=1}^M n_m r_{mk}^{(j)}}. \end{aligned} \quad (14)$$

Thus, the re-estimate of EMACF approximates the re-estimate in the traditional EM algorithm for the Gaussian

mixture model. We summarize EMACF in terms of vectors as follows.

- (1) *Initialization*: Initialize the parameters in the mixture model, $p_k^{(j)} (> 0)$, $\mu_k^{(j)}$, and $\sigma_k^{(j)} (> 0)$ ($k = 1, \dots, K$), and set the current iteration j to 0.
- (2) *E-step*: Given the mixture model parameters $\Psi^{(j)}$, compute the membership probability $r_{mk}^{(j)}$ for each sub-cluster:

$$r_{mk}^{(j)} = \frac{p_k^{(j)} \psi(\mathbf{s}_m | \mu_k^{(j)}, \sigma_k^{(j)})}{\sum_{l=1}^K p_l^{(j)} \psi(\mathbf{s}_m | \mu_l^{(j)}, \sigma_l^{(j)})}. \quad (15)$$

- (3) *M-step*: Given $r_{mk}^{(j)}$, update the mixture model parameters for $k = 1, \dots, K$:

$$p_k^{(j+1)} = \frac{1}{N} \sum_{m=1}^M n_m r_{mk}^{(j)}, \quad (16)$$

$$\mu_k^{(j+1)} = \frac{\sum_{m=1}^M n_m r_{mk}^{(j)} \gamma_{dm}}{\sum_{m=1}^M n_m r_{mk}^{(j)}} = \frac{\sum_{m=1}^M n_m r_{mk}^{(j)} \gamma_{dm}}{N \cdot p_k^{(j+1)}}, \quad (17)$$

$$\begin{aligned} \sigma_k^{(j+1)} &= \frac{\sum_{m=1}^M n_m r_{mk}^{(j)} (\gamma_{dm} - 2\mu_k^{(j+1)} \gamma_{dm} + \mu_k^{(j+1)2})}{N \cdot p_k^{(j+1)}}. \end{aligned} \quad (18)$$

- (4) *Termination*: If $|L(\Psi^{(j+1)}) - L(\Psi^{(j)})| \geq \varepsilon |L(\Psi^{(j)})|$, set j to $j + 1$ and go to step (2).

Though EMACF embodies the variance information explicitly in its *E-step* and *M-step*, it involves only several equations (Eqs. (15)–(18)) and is still easy for implementation. Furthermore, it can surely terminate as supported by the following theorem.

Theorem 2. If $\gamma_{dm} - \gamma_{dm}^2 \geq \zeta > 0$ for $d = 1, \dots, D$ and $m = 1, \dots, M$, then the log-likelihood $L(\Psi)$ for EMACF converges monotonically to a value $L(\Psi^*)$.

Proof. Since $\{p_k\}$ and $\{\sigma_{dk}\}$ are initialized with values larger than zero, they will always keep positive according to Eqs. (15)–(18). In particular, according to Eq. (18),

$$\begin{aligned} \sigma_{dk}^{(j+1)} &= \frac{\sum_{m=1}^M n_m r_{mk}^{(j)} [(\gamma_{dm} - \gamma_{dm}^2) + (\gamma_{dm} - \mu_{dk}^{(j+1)})^2]}{N \cdot p_k^{(j+1)}} \\ &\geq \frac{\sum_{m=1}^M n_m r_{mk}^{(j)} \zeta}{N \cdot p_k^{(j+1)}} = \zeta > 0. \end{aligned}$$

Then, the algorithm is feasible. We now prove that the log-likelihood value does not decrease. As we can see in the derivation procedure, the Q -function value does not decrease, i.e., $Q(\Psi^{(j+1)}; \Phi^{(j)}) \geq Q(\Psi^{(j)}; \Phi^{(j)})$. Moreover,

EMACF is derived from the general EM algorithm, and then EMACF is its instance. Thus, the fact that the Q -function value does not decrease implies the log-likelihood value does not decrease, i.e., $L(\Psi^{(j+1)}) \geq L(\Psi^{(j)})$ (the proof see, e.g., [16, p. 83]).

Then we prove the log-likelihood of EMACF has an upper bound. As shown above, $\sigma_{dk}^{(j)} \geq \zeta > 0$ for any j , thus we have $\psi(\mathbf{x}_i \in \text{CF}_m | \theta_k^{(j)}) = \psi(\mathbf{s}_m | \theta_k^{(j)}) \leq \prod_{d=1}^D 1 / \sqrt{2\pi\sigma_{dk}^{(j)}} \leq (2\pi\zeta)^{-D/2}$. Hence, $L(\Psi^{(j)}) = \sum_{m=1}^M n_m \log \left[\sum_{k=1}^K p_k^{(j)} \psi(\mathbf{s}_m | \theta_k^{(j)}) \right] \leq \sum_{m=1}^M n_m \log \left[\sum_{k=1}^K p_k^{(j)} (2\pi\zeta)^{-\frac{D}{2}} \right] = (ND/2) \log \frac{1}{2\pi\zeta}$ is bounded. Thus, $L(\Psi^{(j)})$ converges monotonically to a value $L(\Psi^*)$. It completes the proof. \square

The prerequisite of Theorem 2 is easily satisfied. With the Jensen's inequality [16], $\gamma_{dm} = \frac{1}{n_m} \sum_{\mathbf{x}_i \in \text{CF}_m} x_{di}^2 \geq \left[(1/n_m) \sum_{\mathbf{x}_i \in \text{CF}_m} x_{di} \right]^2 = v_{dm}^2$. The inequality strictly holds if there exist \mathbf{x}_i and \mathbf{x}_j in the m th sub-cluster having $x_{di} \neq x_{dj}$. In other words, if, for any m and d , there exist two different data items in the m th sub-cluster having the different d th elements, then the prerequisite is satisfied.

Let us have a brief discussion of the complexity of EMACF. In the E -step, it needs to calculate $M \times K$ membership probabilities $r_{mk}^{(j)}$. For each $r_{mk}^{(j)}$, it calculates the probability of each sub-cluster in each pseudo-component distribution according to Eq. (6). It involves $O(D)$ arithmetic operations. Thus the E -step takes $O(MKD)$ operations. Similarly, the M -step of EMACF takes $O(MKD)$ operations. In a word, the computational complexity of EMACF is $O(MKDI)$ where I is the number of iterations. The maximal number of iterations is usually set as a constant [20], say, 500 in this paper. The total storage requirement of EMACF is $2MD + MK + 2KD + K + M$ floating point numbers. Thus, the computation and the storage complexity of EMACF are linear with respect to the number of sub-clusters M , the number of clusters K , and the data dimensionality D .

5. Experimental results

5.1. Methodology and synthetic data

To study the performance of our proposed systems, gEMACF and bEMACF, we compare them with the following model-based clustering systems.

The iEM algorithm, which is the traditional EM algorithm for the Gaussian mixture model as described in Section 2 and 'i' indicates that the data attributes are statistically independent within each cluster.

The sampling iEM system, which is iEM working on, if not specified, 5% random samples, and is referred to as *sampiEM* hereinafter.

The gEMAWS and bEMAWS systems, which are EMAWS working on the clustering features generated by the grid-based and the BIRCH's data summarization procedures respectively, but without considering the variance information [22]. EMAWS can be viewed as iEM handling each data item in the same way as the mean vector of its associated sub-cluster. These mean vectors, with the cardinalities of their associated sub-clusters as weights, can surely approximate the local distribution of the original data set. Hence, gEMAWS and bEMAWS can be regarded as density-biased-sampling model-based clustering techniques [6]. Both follow the second scaling-up strategy as discussed in Section 2.

The gExEM and bExEM systems, which are ExEM working on the clustering features generated by the grid-based and the BIRCH's data summarization procedures, respectively. ExEM, the core algorithm of SEM [3], considers the covariance information only in the M -step. Furthermore, different from EMACF, ExEM is derived in a heuristic way and it is not easy to ascertain its convergence. SEM invokes ExEM to identify the compressible regions of data in the memory, and then compress these regions and read in more data. In order to squash all the data into the memory, SEM has to invoke ExEM many times, and this leads to its speedup factor being smaller than 10 with respect to the traditional EM algorithm [3]. On the other hand, both gExEM and bExEM invoke ExEM only once, and thus can run much faster than SEM. They are mainly designed to make a fair comparison between ExEM and EMACF.

All the algorithms were coded in MATLAB and experiments were conducted on a Sun Enterprise E4500 server. EMACF, ExEM, EMAWS, and iEM were initialized with the cluster centers generated by K -means from 4000 random samples. They were terminated if the successive log-likelihood modification was within 10^{-5} of the current value as in Refs. [19,20]. All experimental results reported were averaged on 10 independent runs. The data summarization procedures were set to generate at most 4000 clustering features and used about 8 Mb main memory. Hence, EMACF, ExEM, and EMAWS only needed memory to store 4000 clustering features, respectively. In contrast, there was no restriction on the amount of the main memory used for both iEM and sampiEM in our experiments.

We generated three groups of synthetic data sets based on random Gaussian mixtures with independent attributes in each component. The first group has two data sets in a two-dimensional space. For each mixture model, the means of the Gaussian components are located on grids, and are $[1, 1]^T, \dots, [\lceil \sqrt{K} \rceil, 1]^T, [1, 2]^T, \dots, [\lceil \sqrt{K} \rceil, \lceil \sqrt{K} \rceil]^T$. The mixing proportion p_k varies in $[\frac{1}{100K}, \frac{3}{K}]$, and the cluster sizes can be very skew. The variance for each attribute falls into $[0.001, 0.5]$. The first data set is exemplified in

Table 1
Parameters of 10 synthetic data sets

Data set		N	D	K
Group 1	1	60,000	2	6
	2	480,000	2	16
Group 2	3	100,000	2	9
	4	100,000	2	20
	5	120,000	2	31
	6	120,000	2	41
	7	100,000	3	10
	8	100,000	4	10
	9	100,000	5	10
	10	100,000	6	10

N , D , and K indicate the number of data items, the data dimensionality, and the number of clusters, respectively.

Fig. 1(a). The second group of eight data sets are also generated according to random Gaussian mixtures. The main difference from the first group is the method of generating the mean vectors. Two mean vectors are generated together to ensure that their Euclidean distance is 1.0. Hence, these two clusters are very close and not well separated. A typical data set is illustrated in Fig. 1(b). For these 10 data sets as summarized in Table 1, the number of data items N ranges from 60,000 to 480,000, the data dimensionality D ranges from 2 to 6, and the number of clusters K ranges from 6 to 41. The third group of eight data sets are generated, similar to the second group, according to a random Gaussian mixture. This mixture has 10 components in a four-dimensional space. These data sets differ in their numbers of data items, which increase exponentially from 6250 to 800,000.

We used the clustering accuracy to measure a generated mixture for the synthetic data. Comparing with the clustering results generated by original mixtures, the clustering accuracy is defined as the proportion of data items that are correctly clustered by the generated mixture [19]. Since all the systems finally generate Gaussian mixtures, another natural evaluation metric is their log-likelihood values. For ease of reading, we averaged the log-likelihood values over the samples.

5.2. Sensitivity examination

Because it is easy to manipulate the shapes and the granularities of the sub-clusters in the grid-based data summarization procedure, we use gEMACF, gExEM, and gEMAWS to examine the sensitivity of EMACF, ExEM, and EMAWS to the structures and the sizes of sub-clusters, respectively. In addition, sampiEM, is also examined. The first data set shown in Fig. 1(a) is taken as an example to examine the sensitivity. Fig. 4 summarizes the clustering accuracy of the four clustering systems for different data summarization or sampling results, which are determined by different grid

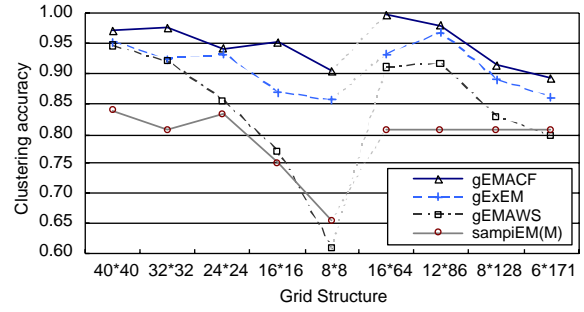


Fig. 4. The clustering accuracy of four model-based clustering systems for different data summarization or sampling results.

structures. For the first 40×40 grid structure, we partition two attributes into 40 equal-width segments, respectively. Here, sampiEM(M) refers to sampiEM working on M random samples where M is the total number of sub-clusters. Hence, these four clustering systems spend similar execution time.

For the first five grid structures, the segment numbers for each attribute are 40, 32, 24, 16, and 8, respectively. As shown in Fig. 4, the clustering accuracy of gEMACF decreases gradually from 97.6% to 90.5%, while the clustering accuracy values of gExEM, gEMAWS, and sampiEM(M) decrease quickly from 95.3% to 85.6%, from 94.6% to 60.9%, and from 83.9% to 65.5%, respectively. The clustering accuracy of gEMACF is usually much higher than that of its three counterparts for each grid structure. Moreover, its clustering accuracy is still acceptable when the sub-cluster granularity is reasonably small. The last four grid structures in Fig. 4 can lead to very skew sub-clusters. For example, the 8×128 grid structure divides the two attributes into 8 and 128 segments, respectively. Hence, the cell width is about 16 times larger than the cell height, and the cell is very skew. For this grid structure, the clustering accuracy of gEMACF is at least 2.3% higher than that of its three counterparts. For these four grid structures, the clustering accuracy of gEMACF decreases gradually from 99.8% to 89.2%. On average, the clustering accuracy values of gEMACF, gExEM, gEMAWS, and sampiEM(M) are, respectively, 94.8%, 91.0%, 84.0%, and 79.1% for the 9 grid structures. The one-tailed paired Student's t -test indicates that gEMACF statistically significantly outperforms its three counterparts at the 0.01 level. Moreover, the performance of gEMACF is not sensitive to the data summarization results.

5.3. Scalability

The second set of experiments are mainly designed to analyze the scalability of gEMACF and bEMACF. We compare them with iEM, sampiEM, bExEM, and bEMAWS. Their performance on the third group of 8 data sets is shown in Fig. 5. To show the scalability clearly, logarithm axes are used in Fig. 5(a).

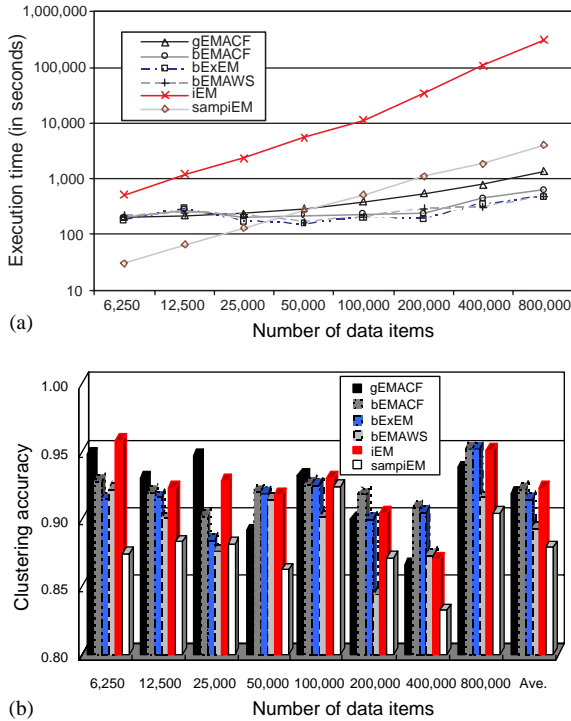


Fig. 5. Performance of six model-based clustering systems on eight four-dimensional data sets. (a) Execution time, and (b) clustering accuracy.

As shown in Fig. 5(a), gEMACF takes 211.6 s for the data set with 6250 data items, and takes 1344.3 s for the data set with 800,000 data items. The bEMACF system takes 197.4 and 611.0 s for these two data sets, respectively. The execution time of the two systems increases very slowly with the number of data items. Both systems scale up well with data size. The bEMACF system runs slightly faster than gEMACF because of their different data summarization procedures. For example, for the largest data set, the grid-based and the BIRCH's data summarization procedures take 216.3 and 985.4 s, respectively. Furthermore, as shown in Fig. 5(b), gEMACF and bEMACF alternatively reach the highest clustering accuracy among the six systems except for the smallest data set. The average clustering accuracy values on the 8 data sets of gEMACF and bEMACF are, respectively, 91.9% and 92.2%. Both perform quite well.

The execution time of iEM increases from 512.0 s for the smallest data set to 307,654.6 s for the largest one. The execution time of iEM increases almost linearly with the data size, because iEM has no restriction on the amount of main memory used in our implementation. For the other data sets, their speedup factors range from 2.4 to 503.5. Thus, both gEMACF and bEMACF can run one or two orders of magnitude faster than iEM. In addition, the speedups are obtained without sacrifice of clustering quality. As shown in Fig. 5(b), both gEMACF and bEMACF usually generate similar ac-

curate results as iEM does. Though their average clustering accuracy is slightly lower than the value of 92.3% for iEM, the one-tailed paired Student's *t*-test does not indicate that there exists significant difference among the three systems at the 0.05 level.

The execution time of sampiEM ranges from 30.9 to 4050.0 s as plotted in Fig. 5(a). The execution time increases linearly with the data size, because sampiEM also has no restriction on the amount of main memory used. Our scalable clustering systems need longer execution time than sampiEM for the first three smallest data sets. However, the data summarization overhead becomes relatively small as the data size increases. The execution time ratios of bEMACF and gEMACF to sampiEM are, respectively, 1:6.6 and 1:3.0 for the largest data set. Furthermore, as shown in Fig. 5(b), sampiEM often generates the worst Gaussian mixtures among the six systems. Its average clustering accuracy is 87.9%, which is statistically worse than that of bEMACF and gEMACF at the 0.05 level.

The execution time of bExEM and bEMAWS ranges from 183.6 to 492.1 s, and both run as fast as bEMACF. However, as plotted in Fig. 5(b), bExEM and bEMAWS usually generate worse clustering results than bEMACF and gEMACF. The average clustering accuracy of bEMAWS is 89.4%, which is significantly lower than that of bEMACF and gEMACF at the 0.05 level. The average clustering accuracy of bExEM is 91.5%. Though it is only 0.7% lower than that of bEMACF, the one-tailed paired *t*-test shows that the difference is significant at the 0.05 level.

5.4. Clustering quality

The third set of experiments is designed to examine the clustering quality of bEMACF and gEMACF, in comparison with iEM, sampiEM, bExEM, and bEMAWS. The clustering accuracy and the execution time of the six systems for the first 10 synthetic data sets are plotted in Fig. 6. To clearly show the execution time, a logarithm axis is used in Fig. 6(b). Two typical Gaussian mixtures generated by gEMACF and bEMACF are illustrated in Figs. 3(a) and (b), respectively. Both are very close to the original one.

As shown in Fig. 6(a), bEMACF usually generates more accurate clustering results than iEM does. For the 10 data sets, the clustering accuracy of bEMACF is higher than that of iEM except for the first and the seventh data sets. On average, the clustering accuracy of bEMACF is 89.3%, which is 1.4% higher than the accuracy of 87.9% for iEM. This situation may be caused by two factors. One is that the BIRCH's data summarization procedure can capture the clustering structure very well. The other one is that a smaller number of clustering features may cause less local maxima in the log-likelihood space. As shown in Fig. 6(b), bEMACF runs 18.7–262.3 times faster than iEM on the 10 data sets, which accords with the comparison results in Section 5.3. As shown in Fig. 6(a), gEMACF sometimes outperforms the other five systems, e.g., on the first three data sets. Except

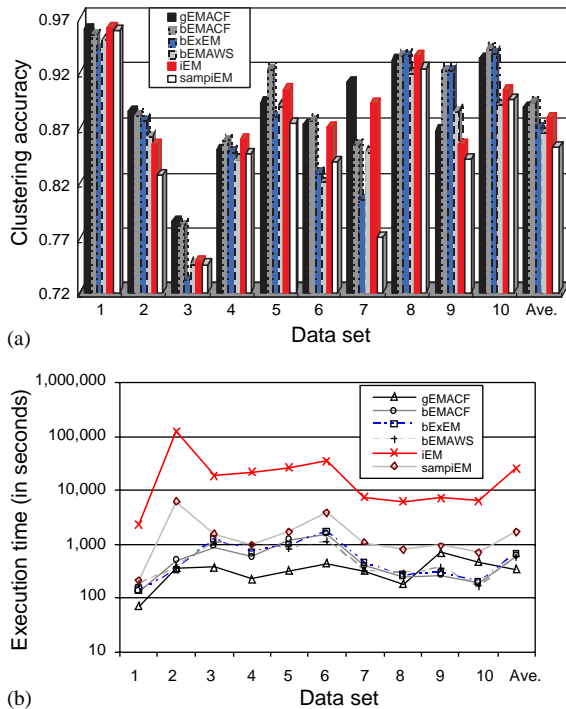


Fig. 6. Performance of six model-based clustering systems on 10 synthetic data sets. (a) Clustering accuracy, and (b) execution time.

the fourth, the fifth, and the eighth data sets, gEMACF generates better Gaussian mixtures than iEM. The average clustering accuracy on the 10 data sets of gEMACF is 88.9%, which is slightly better than the accuracy of 87.9% for iEM and comparable with the accuracy of 89.3% for bEMACF. The speedup factors of gEMACF to iEM range from 10.5 to 352.6. Hence, both gEMACF and bEMACF run one or two orders of magnitude faster than iEM without loss of accuracy.

The average clustering accuracy of sampiEM over the 10 data sets is 85.3%, which is at least 2.6% lower than that of bEMACF, gEMACF, and iEM. According to the one-tailed paired *t*-test, the performance of sampiEM is significantly worse than that of the three systems at the 0.05 level. Moreover, in comparison with bEMACF and gEMACF, sampiEM runs 1.4–17.4 times longer.

The average clustering accuracy of bEMAWS is 86.5%. It is 2.8% lower than the value of 89.3% for bEMACF and 2.4% lower than the value of 88.9% for gEMACF. Though bEMAWS spends comparable execution time, its clustering accuracy is significantly worse than that of bEMACF and gEMACF at the 0.05 level. The average clustering accuracy of bExEM is 87.1%. It is 2.2% and 1.8% lower than that of bEMACF and gEMACF, respectively. Though bExEM spends similar execution time, its clustering accuracy is significantly worse than that of bEMACF at the 0.05

level. Thus, using comparable computational resources, both gEMACF and bEMACF can generate significantly more accurate clustering results than both bEMAWS and bExEM.

5.5. Application to three real-life data sets

We study the performance of the model-based clustering systems on three real-life data sets. The first one, Forest CoverType Data, is downloaded from the UCI KDD Archive (<http://kdd.ics.uci.edu>). It describes forest cover for 30×30 m cells obtained from US Forest Service Region 2 Resource Information System data. It has 581,012 data items. In the experiments, we use five quantitative attributes of Elevation, Aspect, Horizontal Distance to Hydrology, Horizontal Distance to Roadways, and Horizontal Distance to Fire Points. The second one, the Census-Income Database, is also downloaded from the UCI KDD archive. It contains weighted census data extracted from the 1994 and 1995 population surveys of the US Census Bureau. The data set contains 299,285 data items. We use three continuous attributes including Age, Dividends from Stocks, and Weeks Worked in Year. The last one is the California housing data, downloaded from www.spatial-statistics.com. It describes all of the block groups in California from the 1990 US Census, and has eight numeric attributes and 20,640 data items. All of the attribute values are scaled into the interval from 0 to 3. Based on some preliminary experiments, we choose Gaussian mixture models with 15, 10, and 7 components to describe these three data sets, respectively. Since gEMACF performs as well as bEMACF, we only give the results of bEMACF. Table 2 lists the performance of bEMACF, bExEM, bEMAWS, iEM, and sampiEM on the three data sets in terms of both quality and time, as well as their standard deviations. The average log-likelihood indicates the quality of a generated Gaussian mixture.

For the large Forest CoverType Data, we only draw 15% random data items for iEM, since iEM needs too much time for the whole data set. In fact, even sampiEM(15%) takes 53,475.2 s and sampiEM(5%) takes 17,872.4 s on average. However, bEMACF takes about 2064.9 s. It runs, respectively, 25.9 times faster than sampiEM(15%), and 8.7 times faster than sampiEM(5%). Furthermore, the average log-likelihood values of bEMACF, sampiEM(15%), and sampiEM(5%) are -3.242 , -3.250 , and -3.252 , respectively. So, bEMACF can generate slightly more accurate Gaussian mixtures than both sampiEM(15%) and sampiEM(5%), though there is no significant difference at the 0.05 level. Though bExEM runs fastest and spends only 1689.4 s on average, its average log-likelihood is -3.256 . It is not significantly different from that of sampiEM(15%), but it is significantly worse than that of bEMACF at the 0.05 level as indicated by the one-tailed *t*-test. Similarly, bEMAWS spends 1732.4 s on average. Its average log-likelihood is -3.394 , which significantly lags behind the other four systems at the 0.05 level as indicated by the one-tailed *t*-test.

Table 2
The performance of five clustering systems on three real-life data sets

Real-life data	<i>N</i>	<i>D</i>	<i>K</i>	<i>M</i>	Measures	bEMACF	bExEM	bEMAWS	iEM	sampiEM (5%)
Forest CoverType Data	581,012	5	15	3186	Log-likelihood time (s)	-3.242 ± 0.019 2064.9 ± 891.2	-3.256 ± 0.019 1689.4 ± 618.9	-3.394 ± 0.023 1732.4 ± 376.9	-3.250* ± 0.018 53,475.2 ± 10,166.3	-3.252 ± 0.022 17,672.4 ± 5198.4
Census-Income Database	299,285	3	10	3836	Log-likelihood time (s)	-0.747 ± 0.003 1361.3 ± 371.6	-0.750 ± 0.005 1205.3 ± 362.2	-0.751 ± 0.006 1400.8 ± 476.1	-0.747 ± 0.008 209,865.6 ± 35,892.2	-0.750 ± 0.007 4300.7 ± 1996.4
California housing data	20,640	8	7	2907	Log-likelihood time (s)	3.512 ± 0.101 684.9 ± 113.2	3.462 ± 0.166 719.9 ± 468.1	3.424 ± 0.127 512.6 ± 143.8	3.804 ± 0.162 4452.6 ± 691.40	3.380 ± 0.211 263.8 ± 59.8

N, *D*, *K*, and *M* indicate the number of data items, the data dimensionality, the number of clusters, and the number of sub-clusters, respectively. For the Forest CoverType Data, iEM runs on 15% samples as indicated by “*” in the second row.

For the Census-Income Database, the average log-likelihood of bEMACF is -0.747 , which is the best among the four systems for the real-life data set. It is interesting to see that the average log-likelihood of iEM is also -0.747 . Thus, the mixtures generated by bEMACF and iEM have the same quality. However, iEM runs 154.2 times longer than bEMACF. Hence, bEMACF generates Gaussian mixtures as accurate as iEM, but runs two orders of magnitude faster. The average log-likelihood of sampiEM is -0.750 , which is 0.003 lower than that of bEMACF. The one-tailed *t*-test indicates that the difference is significant at the 0.05 level. Furthermore, sampiEM spends around 3.2 times longer than that of bEMACF. For this data set, bExEM and bEMAWS, respectively, spend 1205.3 and 1400.8 s, which are comparable with bEMACF. However, their average log-likelihood values are -0.750 and -0.751 , respectively. They are at least 0.003 lower than the value of -0.747 for bEMACF. The difference is significant at the 0.05 level according to the one-tailed *t*-test. Thus, bEMACF significantly outperform bExEM, bEMAWS, and sampiEM in terms of both execution time and clustering quality.

For the eight-dimensional California housing data, the average log-likelihood values of bEMACF, bExEM, bEMAWS, iEM, and sampiEM are 3.512, 3.462, 3.424, 3.804, and 3.380, respectively. Their average execution time is 684.9, 719.9, 512.6, 4452.6, and 263.8 s, respectively. Though the clustering quality of bEMACF is lower than that of iEM, bEMACF runs 6.5 times faster than iEM. Compared with bExEM, bEMACF runs a little bit faster and generates more accurate results. For this moderate data set, bEMACF spends slightly longer time than bEMAWS and sampiEM. However, the one-tailed *t*-test indicates that the mixture model quality of bEMACF is significantly better than that of bEMAWS and sampiEM at the 0.05 level. For this data set, the advantage of bEMACF is not so apparent partially because the data size is not very large.

6. Discussion and conclusion

Based on our scalable model-based clustering framework, we have established two scalable clustering systems for the parsimonious Gaussian mixture model with independent attributes in each cluster. They first summarize a data set into disjoint sub-clusters, and then generate Gaussian mixtures using our new model-based clustering algorithm—EMACF. EMACF takes account of the cardinality, mean, and variance of each sub-cluster, and approximates its aggregate behavior in the Gaussian mixture model. It is theoretically convergent and is empirically not sensitive to the data summarization results. The two proposed clustering systems, gEMACF and bEMACF, are EMACF working on clustering features generated by the grid-based and the BIRCH's data summarization procedures, respectively. Comparison results on both synthetic and real-life data have shown that both systems run one or two orders of magnitude faster than the

traditional expectation maximization algorithm with few or no loss of clustering quality. They, using comparable computational resources, can generate significantly more accurate clustering results than the existing scalable model-based clustering techniques.

Based on the same principle presented in this paper, new model-based clustering algorithms can be developed to generate clusters effectively from summary statistics. Combining these algorithms with some sophisticated data summarization procedures, new scalable model-based clustering systems may be established to analyze large complicated data, such as time sequences and data sets with different kinds of attributes. We are currently working on this direction. We are also interested in some effective approaches to automatically determine the number of clusters for large data sets.

7. Summary

Scalability, one of the most significant challenges of data mining, addresses the problem of processing large data sets with limited computational resources, e.g., main memory and computation time. In this paper, we present two scalable clustering systems based on a Gaussian mixture model with independent attributes in each component. The basic idea is as follows: first, a data set is summarized into sub-clusters; then, clusters are directly generated from the clustering features of the sub-clusters by our specifically designed Expectation Maximization (EM) algorithm—EMACF. EMACF embodies the cardinality, mean, and variance of each sub-cluster. It may approximate the aggregate behavior of each sub-cluster of data items in the Gaussian mixture model. EMACF is proved to converge to local maxima. It is linear with respect to the number of sub-clusters.

Combining with an adaptive grid-based data summarization and the BIRCH's data summarization procedures, EMACF is used to construct two scalable model-based clustering systems: gEMACF and bEMACF. A series of experiments are conducted on both synthetic and real-life data sets. Both bEMACF and gEMACF usually run one or two orders of magnitude faster than the traditional EM algorithm for the Gaussian mixture model. Though there is sometimes a slight loss of clustering quality, it is not statistically significant at the 0.05 level. The two systems may run faster and can generate much more accurate clustering results than the random sampling EM algorithm. The two clustering systems, using comparable computational resources, generate significantly more accurate clustering results than the existing scalable model-based clustering techniques at the 0.05 level.

Acknowledgements

The authors appreciate the anonymous referees for their valuable comments to strengthen the paper, and T. Zhang,

R. Ramakrishnan, M. Livny, and V. Ganti for the BIRCH source code. The work was partially supported by RGC Grant CUHK 4212/01E of Hong Kong, Lingnan University direct grant (RES-021/200), RGC Grant LU 3009/02E of Hong Kong, and the Nature Science Foundation Project (No. 10371097) of China.

References

- [1] J. Han, M. Kamber, *Data Mining: Concepts and Techniques*, Morgan Kaufmann Publishers, San Francisco, CA, USA, 2001.
- [2] H.-D. Jin, K.-S. Leung, M.-L. Wong, Scaling-up model-based clustering algorithm by working on clustering features, in: *Proceedings of the Third International Conference on Intelligent Data Engineering and Automated Learning*, 2002, pp. 569–575.
- [3] P. Bradley, U. Fayyad, C. Reina, Clustering very large databases using EM mixture models, in: *Proceedings of the 15th International Conference on Pattern Recognition*, vol. 2, 2000, pp. 76–80.
- [4] T. Zhang, R. Ramakrishnan, M. Livny, BIRCH: a new data clustering algorithm and its applications, *Data Min. Knowl. Disc.* 1 (2) (1997) 141–182.
- [5] H.-D. Jin, W. Shum, K.-S. Leung, M.-L. Wong, Expanding self-organizing map for data visualization and cluster analysis, *Inform. Sci.* 163 (1–3) (2004) 157–173.
- [6] C. Palmer, C. Faloutsos, Density biased sampling: an improved method for data mining and clustering, in: *Proceedings of the 2000 ACM International Conference on Management of Data*, 2000, pp. 82–92.
- [7] S. McKenna, S. Gong, Y. Raja, Modelling facial colour and identity with Gaussian mixtures, *Pattern Recognition* 31 (1998) 1883–1892.
- [8] C. Fraley, Algorithms for model-based Gaussian hierarchical clustering, *SIAM J. Sci. Comput.* 20 (1) (1999) 270–281.
- [9] M. Figueiredo, A.K. Jain, Unsupervised learning of finite mixture models, *IEEE Trans. Pattern Anal. Mach. Intell.* 24 (3) (2002) 381–396.
- [10] G. McLachlan, R. Bean, D. Peel, A mixture model-based approach to the clustering of microarray expression data, *Bioinformatics* 18 (2002) 413–422.
- [11] I. Cadez, D. Heckerman, C. Meek, P. Smyth, S. White, Model-based clustering and visualization of navigation patterns on a web site, *Data Min. Knowl. Disc.* 7 (4) (2003) 399–424.
- [12] G. Celeux, G. Govaert, Gaussian parsimonious clustering models, *Pattern Recognition* 28 (5) (1995) 781–793.
- [13] P. Cheeseman, J. Stutz, Bayesian classification (AutoClass): theory and results, in: U. Fayyad et al. (Eds.), *Advances in Knowledge Discovery and Data Mining*, Menlo Park, CA, USA, 1996, pp. 153–180.
- [14] A. Dempster, N. Laird, D. Rubin, Maximum-likelihood from incomplete data via the EM algorithm, *J. Roy. Statist. Soc. Ser. B* 39 (1977) 1–38.
- [15] J. Ma, L. Xu, M. Jordan, Asymptotic convergence rate of the EM algorithm for Gaussian mixtures, *Neural Comput.* 12 (12) (2000) 2881–2907.
- [16] G. McLachlan, T. Krishnan, *The EM Algorithm and Extensions*, Wiley, New York, 1997.

- [17] R. Neal, G. Hinton, A view of the EM algorithm that justifies incremental, sparse, and other variants, in: M. Jordan (Ed.), *Learning in Graphical Models*, MIT Press, Cambridge, MA, 1999, pp. 355–368.
- [18] J. Shanmugasundaram, U. Fayyad, P. Bradley, Compressed data cubes for OLAP aggregate query approximation on continuous dimensions, in: *Proceedings of the Fifth ACM SIGKDD*, San Diego, USA, 1999, pp. 223–232.
- [19] M. Meilă, D. Heckerman, An experimental comparison of model-based clustering methods, *Mach. Learn.* 42 (1/2) (2001) 9–29.
- [20] B. Thiesson, C. Meek, D. Heckerman, Accelerating EM for large databases, *Mach. Learn.* 45 (2001) 279–299.
- [21] A. Moore, Very fast EM-based mixture model clustering using multiresolution KD-trees, in: M. Kearns, D. Cohn (Eds.), *Advances in Neural Information Processing Systems*, vol. 11, 1999, pp. 543–549.
- [22] H.-D. Jin, Scalable model-based clustering algorithms for large databases and their applications, Ph.D. Thesis, The Chinese University of Hong Kong, Shatin, N.T., Hong Kong, August 2002.
- [23] W. DuMouchel, C. Volinsky, T. Johnson, C. Cortes, D. Pregibon, Squashing flat files flatter, in: *Proceedings of the Fifth ACM SIGKDD*, San Diego, CA, USA, 1999, pp. 6–15.

About the Author—HUIDONG JIN received his B.Sc. degree from the Department of Applied Mathematics in 1995, and his M.Sc. degree from the Institute of Information and System Sciences in 1998, both from Xi'an Jiaotong University, China. In 2002, he got his Ph.D. degree in Computer Science and Engineering from the Chinese University of Hong Kong. He is currently with Division of Mathematical and Information Sciences, CSIRO, Australia. His research interests are data mining, pattern recognition, and soft computing. He is a member of the ACM and the IEEE.

About the Author—KWONG-SAK LEUNG received his B.Sc. (Eng.) and Ph.D. degrees in 1977 and 1980, respectively, from the University of London, Queen Mary College. He worked as a senior engineer on contract R&D at ERA Technology and later joined the Central Electricity Generating Board to work on nuclear power station simulators in England. He joined the Computer Science and Engineering Department at the Chinese University of Hong Kong in 1985, where he is currently professor and chairman of the Department. His research interests are in soft computing including evolutionary computation, neural computation, probabilistic search, information fusion and data mining, fuzzy data and knowledge engineering. He has published over 180 papers and two books in fuzzy logic and evolutionary computation. He has been chair and member of many program and organizing committees of international conferences. He is in the Editorial Board of Fuzzy Sets and Systems and an associate editor of International Journal of Intelligent Automation and Soft Computing. He is a senior member of the IEEE, a chartered engineer, a member of IEE and ACM and a fellow of HKCS and HKIE.

About the Author—MAN-LEUNG WONG is an associate professor at the Department of Computing and Decision Sciences of Lingnan University, Tuen Mun, Hong Kong. Before joining the university, he worked as an assistant professor at the Department of Systems Engineering and Engineering Management, the Chinese University of Hong Kong and the Department of Computing Science, Hong Kong Baptist University. He worked as a research engineer at the Hypercom Asia Ltd. in 1997. His research interests are evolutionary computation, data mining, machine learning, knowledge acquisition, and approximate reasoning. He has authored and co-authored over 50 papers and one book in these areas. He received his B.Sc., M.Phil., and Ph.D. in computer science from the Chinese University of Hong Kong in 1988, 1990, and 1995, respectively. He is a member of the IEEE and the ACM.

About the Author—ZONG-BEN XU received the M.S. degree in mathematics in 1981 and the Ph.D. degree in applied mathematics in 1987 from Xi'an Jiaotong University, China. He has been with the Faculty of Science and Institute for Information and System Sciences at Xi'an Jiaotong University since 1982, and now serves as a Ph.D. supervisor in mathematics and computer science, dean of the Faculty of Science, and director of the Institute for Information and System Sciences. He has published four monographs and more than 110 academic papers on nonlinear functional analysis, optimization techniques, neural networks, evolutionary computation, and data mining. Dr. Xu holds the title "Owner of Chinese Ph.D. Degree Having Outstanding Achievements" by the Chinese State Education Commission (CSEC) and the Academic Degree Commission of the Chinese Council in 1991, and get scientific awards from the university and CSEC several times. He is a member of the New York Academy of Sciences.