Journal of Machine Learning Research 4 (2003) 1001-1037 Submitted 3/03; Revised 7/03; Published 11/03

# A Unified Framework for Model-based Clustering

**Shi Zhong*** ZHONG@CSE.FAU.EDU
*Department of Computer Science and Engineering*
*Florida Atlantic University, Boca Raton, FL 33431, USA*
**Joydeep Ghosh** GHOSH@ECE.UTEXAS.EDU
*Department of Electrical and Computer Engineering*
*The University of Texas at Austin, Austin, TX 78712, USA*

**Editor:** Claire Cardie

## Abstract

Model-based clustering techniques have been widely used and have shown promising results in many applications involving complex data. This paper presents a unified framework for probabilistic model-based clustering based on a bipartite graph view of data and models that highlights the commonalities and differences among existing model-based clustering algorithms. In this view, clusters are represented as probabilistic models in a model space that is conceptually separate from the data space. For partitional clustering, the view is conceptually similar to the Expectation-Maximization (EM) algorithm. For hierarchical clustering, the graph-based view helps to visualize critical/important distinctions between similarity-based approaches and model-based approaches. The framework also suggests several useful variations of existing clustering algorithms. Two new variations—balanced model-based clustering and hybrid model-based clustering—are discussed and empirically evaluated on a variety of data types.

**Keywords:** Model-based Clustering, Similarity-based Clustering, Partitional Clustering, Hierarchical Agglomerative Clustering, Deterministic Annealing

## 1. Introduction

Clustering or segmentation of data is a fundamental data analysis step that has been widely studied across multiple disciplines for over 40 years (Hartigan, 1975; Jain and Dubes, 1988; Jain et al., 1999; Ghosh, 2003). In this paper we make a fundamental distinction between *discriminative* (or distance/similarity-based) approaches (Indyk, 1999; Scholkopf and Smola, 2001; Vapnik, 1998) and *generative* (or model-based) approaches (Blimes, 1998; Rose, 1998; Smyth, 1997) to clustering. With a few exceptions (Vapnik, 1998; Jaakkola and Haussler, 1999), this is not considered the primary dichotomy in the vast clustering literature—partitional vs. hierarchical is a more popular choice by far. We shall show that the discriminative vs. generative distinction leads to a useful understanding of existing clustering algorithms. In discriminative approaches, such as clustering via graph partitioning (Karypis et al., 1999), one determines a distance or similarity function between pairs of data objects, and then groups similar objects together into clusters. Parametric, model-based approaches, on the other hand, attempt to learn generative models from the data, with each model representing one particular cluster. In both categories, the most popular clustering techniques

∗. This work was done while Shi Zhong was a PhD student at The University of Texas at Austin.

©2003 Shi Zhong and Joydeep Ghosh.

include partitional clustering and hierarchical clustering (Hartigan, 1975; Jain et al., 1999). A partitional method partitions the data objects into $K$ (often specified *a priori*) groups according to some optimization criterion. The widely-used k-means algorithm is a classic example of partitional methods. A hierarchical method builds a hierarchical set of nested clusterings, with the clustering at the top level containing a single cluster of all data objects and the clustering at the bottom level containing $N$ singleton clusters (i.e., one cluster for each data object), where $N$ is the total number of data objects. The resulting hierarchy shows at each level which two clusters are merged together and the inter-cluster distance between them, and thus provides a good visualization tool.

In discriminative approaches, the most commonly used distance measures are Euclidean distance and Mahalanobis distance for data that can be represented in a vector space. The instance-based learning literature (Aha et al., 1991) provides several examples of scenarios where customized distance measures perform better than such generic ones. For high-dimensional text clustering, Strehl et al. (2000) studied the impact of different similarity measures and showed that Euclidean distances are not appropriate for this domain. For complex data types (e.g., variable length sequences), defining a good similarity measure is very much data dependent and often requires expert domain knowledge. For example, a wide variety of distance measures have been proposed for clustering sequences (Geva and Kerem, 1998; Kalpakis et al., 2001; Qian et al., 2001). Another disadvantage of similarity-based approaches is that calculating the similarities between all pairs of data objects is computationally inefficient, requiring a complexity of $O(N^2)$. Despite this disadvantage, discriminative methods such as graph partitioning and spectral clustering algorithms (Karypis et al., 1999; Dhillon, 2001; Meila and Shi, 2001; Ng et al., 2002; Strehl and Ghosh, 2002) have gained recent popularity due to their ability to produce desirable clustering results.

For model-based clustering approaches, the model type is often specified *a priori*, such as Gaussian or hidden Markov models (HMMs). The model structure (e.g., the number of hidden states in an HMM) can be determined by model selection techniques and parameters estimated using maximum likelihood algorithms, e.g., the EM algorithm (Dempster et al., 1977). Probabilistic model-based clustering techniques have shown promising results in a corpus of applications. Gaussian mixture models are the most popular models used for vector data (Symons, 1981; McLachlan and Basford, 1988; Banfield and Raftery, 1993; Fraley, 1999; Yeung et al., 2001); multinomial models have been shown to be effective for high dimensional text clustering (Vaithyanathan and Dom, 2000; Meila and Heckerman, 2001). By deriving a bijection between Bregman divergences and the exponential family of distributions, Banerjee et al. (2003b) have recently shown that clustering based on a mixture of components from any member of this vast family can be done in an efficient manner. For clustering more complex data such as time sequences, the dominant models are Markov Chains (Cadez et al., 2000; Ramoni et al., 2002) and HMMs (Dermatas and Kokkinakis, 1996; Smyth, 1997; Oates et al., 1999; Law and Kwok, 2000; Li and Biswas, 2002). Compared to similarity-based methods, model-based methods offer better interpretability since the resulting model for each cluster directly characterizes that cluster. Model-based partitional clustering algorithms often have a computational complexity that is "linear" in the number of data objects under certain practical assumptions, as analyzed in Section 2.4.

Existing works on model-based clustering largely concentrate on a specific model or application. A notable exception is the work of Cadez et al. (2000) who proposed an EM framework for partitional clustering with a mixture of probabilistic models. Essentially, this work is "*EM cluster-*

*ing*"[1] with an emphasis on clustering of non-vector data such as variable-length sequences. Their work, however, does not address model-based hierarchical clustering or specialized model-based partitional clustering algorithms such as the Self-Organizing Map (SOM) (Kohonen, 1997) and the Neural-Gas algorithm (Martinetz et al., 1993), both of which use a varying neighborhood function to control the assignment of data objects to different clusters.

This paper provides a characterization of all existing model-based clustering algorithms under a unified framework. The framework includes a bipartite graph view of model-based clustering, an information-theoretic analysis of model-based partitional clustering, and a view of model-based hierarchical clustering that leads to several useful extensions. Listed below are four main contributions of this paper:

1. We propose a bipartite graph view (Section 2.1) of data and models that provides a good visualization and understanding of existing model-based clustering algorithms—both partitional and hierarchical algorithms. For partitional clustering, the view is conceptually similar to the EM algorithm. For hierarchical clustering, it points out helpful distinctions between similarity-based approaches and model-based approaches.

2. We conduct an information-theoretic analysis of model-based partitional clustering that demonstrates the connections between many existing algorithms including k-means, EM clustering, SOM, and Neural-Gas from a deterministic annealing point of view (Section 2.2). Deterministic annealing has been used for clustering (Wong, 1993; Hofmann and Buhmann, 1997, 1998; Rose, 1998), but only on Gaussian models. Our analysis of model-based clustering algorithms from this perspective gives new insights into k-means and EM clustering, and provides model-based extensions of SOM and Neural-Gas algorithms. The benefits of this synthetic view are demonstrated through an experimental study on document clustering.

3. We present an analysis of model-based approaches vs. similarity-based approaches for hierarchical clustering (Section 2.3) that leads to several useful extensions of model-based hierarchical clustering, e.g., hierarchical cluster merging with extended Kullback-Leibler divergences.

4. The unified framework is used to obtain two new variations of model-based clustering—balanced clustering (Section 5) and hybrid clustering (Section 6), tailored to specific applications. Both variations show promising results in several case studies.

The organization of this paper is as follows. The next section presents the unified framework for model-based clustering and a synthetic view of existing model-based partitional and hierarchical clustering algorithms. Section 3 introduces several commonly used clustering evaluation criteria. Section 4 compares different models and different model-based partitional clustering algorithms for document clustering. Section 5 describes a generic balanced model-based clustering algorithm that produces clusters of high quality as well as of comparable sizes. Section 6 proposes a hybrid clustering idea to combine the advantages of both partitional and hierarchical model-based clustering methods. Experimental results show the effectiveness of the proposed algorithms. Section 7 summarizes some related work. Finally, Section 8 concludes this paper.

1. This term signifies a specific application of the more general EM algorithm (Dempster et al., 1977), where one treats the cluster identities of data objects as the hidden indicator variables and then tries to maximize the objective function in Equation 6 using the EM algorithm.

## 2. A Unified Framework for Model-based Clustering

In this section, we present a unifying bipartite graph view of probabilistic model-based clustering and demonstrate the benefits of having such a viewpoint. In Section 2.2, model-based partitional clustering is mathematically analyzed from a deterministic annealing perspective, which reveals relationships between generic model-based k-means, EM clustering, deterministic annealing, SOM, and Neural-Gas algorithms. Model-based hierarchical clustering is discussed in Section 2.3, where a distinction between model-based and similarity-based hierarchical clustering is made. Several practical issues, including complexity analysis, are discussed in Section 2.4.

### 2.1 A Bipartite Graph View

The bipartite graph view (Figure 1) assumes a set of $N$ data objects $X$ (e.g., sequences), represented by $x_1, x_2, ...,$ and $x_N$, and $K$ probabilistic generative models (e.g., HMMs), $\lambda_1, \lambda_2, ..., \lambda_K$, each corresponding to a cluster of data objects.[2] The bipartite graph is formed by connections between the data and model spaces. The model space usually contains members of a specific family of probabilistic models. A model $\lambda_y$ can be viewed as the generalized "centroid" of cluster $y$, though it typically provides a much richer description of the cluster than a centroid in the data space. A connection between an object $x$ and a model $\lambda_y$ indicates that the object $x$ is being associated with cluster $y$, with the connection weight (closeness) between them given by the log-likelihood $\log p(x|\lambda_y)$.

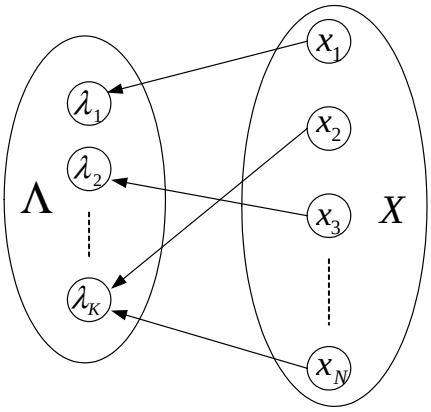



Figure 1: A bipartite graph view of model-based clustering.

Readers may immediately notice the conceptual similarity between this view and the EM algorithm (Dempster et al., 1977), which is a general algorithm for solving maximum likelihood estimation from incomplete data. Indeed, for partitional clustering, the cluster indices for data objects can be treated as missing data and the EM algorithm can be employed to estimate the model parameters that maximize the incomplete data likelihood $P(X|\Lambda)$. The bipartite graph view, however, is not equivalent to the EM algorithm for two reasons. First, it is not an algorithm; rather it provides

2. We interchangeably use $\lambda$ to represent a model as well as the set of parameters of that model. The set of all parameters used for modeling the whole dataset is represented by $\Lambda = \{\lambda_1, ..., \lambda_K\}$. Later in this paper, $\Lambda$ also includes cluster priors for soft clustering.

a visualization for model-based clustering. Second, it also encompasses hierarchical model-based clustering which does not involve incomplete data. The idea of representing clusters by models generalizes the standard k-means algorithm, where both data objects and cluster centroids are in the same data space. The models also provide a probabilistic interpretation of clusters, which is a desirable feature in many applications.

A variety of hard and soft assignment strategies can be designed by attaching to each connection an association probability based on the connection weights. For hard clustering these probabilities are either 1's or 0's. Intuitively, a suitable objective function is the sum of all connection weights (log-likelihoods) weighted by the association probabilities, which is to be maximized. Indeed, maximizing this objective function leads to a well-known hard clustering algorithm (Kearns et al., 1997; Li and Biswas, 2002; Banerjee et al., 2003b). We will show in the next section that soft model-based clustering can be obtained by adding entropy constraints to the objective function. Similar to deterministic annealing, a temperature parameter can be used to regulate the softness of data assignments.

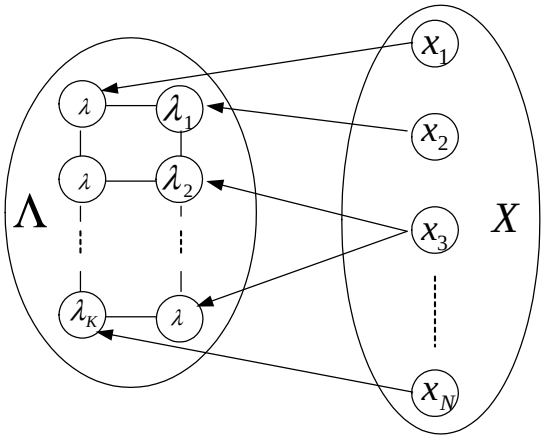



Figure 2: Model-based partitional clustering with an imposed logical neighborhood structure on cluster models.

A straightforward design of a model-based partitional clustering algorithm is to iteratively re-train models and re-partition data objects. This can be achieved by applying the EM algorithm to iteratively compute the (hidden) cluster identities of data objects in the E-step and estimate the model parameters in the M-step. However, alternative techniques can also be applied. Moreover, model-based partitional clustering based on the bipartite graph view allows for several extensions. Two examples are: (a) impose a structure on the $K$ models in the model space; (b) constrain the partitioning of data objects in the data space in certain ways. If a grid map structure is imposed on the relationship between models (Figure 2), one can get a SOM-like model-based partitional clustering algorithm and at the end of clustering process, the relative distance between different clusters should conform to the imposed topological ordering. An example can be found in work by Heskes (2001), where multinomial model-based SOM was used for market basket data analysis. For the second extension idea, we will introduce in Section 5 a balanced model-based clustering algorithm

which can produce *balanced* clusters, i.e., clusters with comparable number of data objects, and improve clustering quality by using balance constraints in the clustering process.

Alternatively, one can initialize $K = N$ and hierarchically merge clusters in the model space, resulting in a model-based hierarchical clustering algorithm (Figure 3). The difference from the standard single-link or complete-link hierarchical clustering algorithms is that the hierarchy is built in the model space using a suitable measure of divergence between models.

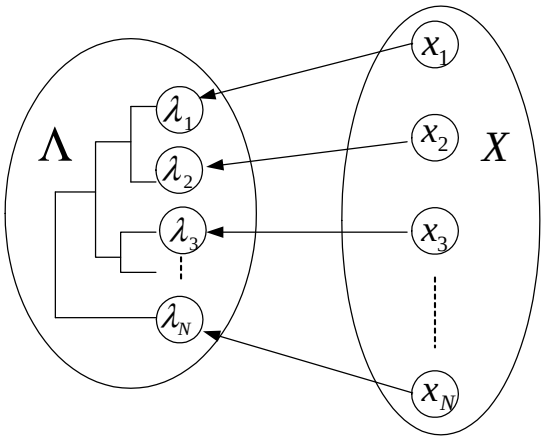



Figure 3: A graph view of model-based hierarchical clustering.

### 2.2 Model-based Partitional Clustering

In this section, we present a principled, information-theoretic analysis of model-based partitional clustering. The derivation process is similar to that of deterministic annealing (Rose, 1998) and the analysis provides a common view and useful generalization of existing algorithms, including k-means (MacQueen, 1967; Dermatas and Kokkinakis, 1996; Dhillon and Modha, 2001; Li and Biswas, 2002), EM clustering (McLachlan and Basford, 1988; Banfield and Raftery, 1993; Cadez et al., 2000; Meila and Heckerman, 2001), SOM (Kohonen, 1997), and Neural-Gas (Martinetz et al., 1993). The resulting algorithm involves a temperature parameter $T$, which governs the randomness of posterior data assignments. EM clustering corresponds to the special case $T = 1$ whereas the k-means clustering corresponds to $T = 0$.

Let the joint probability for a data object $x$ and a cluster $y$ be $P(x,y)$. We aim to maximize the expected log-likelihood

$$L = \sum_{x,y} P(x,y) \log p(x|\lambda_y) = \sum_{x} P(x) \sum_{y} P(y|x) \log p(x|\lambda_y) \ . \tag{1}$$

Note that in practice one typically uses a sample average to calculate $L$, i.e., $P(x) = \frac{1}{N}$, $\forall x \in X$. As $N$ goes to infinity, the sample average approaches the expected log-likelihood asymptotically.

Directly maximizing the objective function in Equation 1 over $P(y|x)$ and $\lambda_y$ leads to a known hard clustering algorithm (Kearns et al., 1997; Li and Biswas, 2002) that we call model-based k-means (*mk-means*). It is a generalized version of the standard k-means algorithm (MacQueen, 1967;

Lloyd, 1982) and iterates between the following two steps:

$$P(y|x) = \begin{cases} 1, & y = \arg\max_{y'} \log p(x|\lambda_{y'}); \\ 0, & \text{otherwise}, \end{cases} \tag{2}$$

and

$$\lambda_y = \arg\max_{\lambda} \sum_x P(y|x) \log p(x|\lambda) \ . \tag{3}$$

The posterior probability $P(y|x)$ in Equation 2 is actually conditioned on current parameters $\Lambda = \{\lambda_1, ..., \lambda_K\}$, but for simplicity we use $P(y|x)$ instead of $P(y|x, \Lambda)$ where there is no confusion. Equation 2 represents a hard data assignment strategy—each data object $x$ is assigned, with probability 1, to the cluster $y$ that gives the maximum $\log p(x|\lambda_y)$. When equi-variant spherical Gaussian models are used for vector data, the mk-means algorithm reduces to the standard k-means algorithm. It is well known that the k-means algorithm tends to quickly get stuck in a local solution. One way of alleviating this problem is to use soft assignments (Rose, 1998).

To introduce some randomness or softness to the data assignment step, we add entropy constraints to Equation 1. Let $X$ be the set of all data objects and $Y$ the set of all cluster indices. The new objective is

$$L_1 = L + T \cdot H(Y|X) - T \cdot H(Y) = L - T \cdot I(X;Y) \ , \tag{4}$$

where $H(Y) = -\sum_y P(y) \log P(y)$ is the cluster prior entropy,

$$H(Y|X) = -\sum_x P(x) \sum_y P(y|x) \log P(y|x)$$

the average posterior entropy, and $I(X;Y)$ the mutual information between $X$ and $Y$. The parameter $T$ is a Lagrange multiplier used to trade off between maximizing the average log-likelihood $L$ and minimizing the mutual information between $X$ and $Y$. If we fix $H(y)$, minimizing $I(X;Y)$ is equivalent to maximizing the average posterior entropy $H(Y|X)$, or maximizing the randomness of the data assignment.

Note that the added entropy terms do not change the model re-estimation formula in Equation 3 since the model parameters that maximize $L$ also maximize $L_1$. To solve for $P(y|x)$ under constraint $\sum_y P(y|x) = 1$, one can first construct the Lagrangian

$$\mathcal{L} = L_1 + \sum_x \xi_x (\sum_y P(y|x) - 1) \ ,$$

where $\xi$'s are Lagrange multipliers, and then let the partial derivative $\frac{\partial \mathcal{L}}{\partial P(y|x)} = 0$ . The resulting $P(y|x)$ is the well-known Gibbs distribution (Geman and Geman, 1984) given by

$$P(y|x) = \frac{P(y) p(x|\lambda_y)^{\frac{1}{T}}}{\sum_{y'} P(y') p(x|\lambda_{y'})^{\frac{1}{T}}} \ . \tag{5}$$

If $P(y)$ is not known *a priori*, we can estimate it from the data as $P(y) = \sum_x P(x) P(y|x)$ . Now we get a model-based clustering algorithm parameterized by the parameter $T$, which has a temperature interpretation in deterministic annealing (Rose, 1998). At a high temperature, the objective function (Equation 4) is smooth and there are very few local solutions (Rose, 1998). As $T$ decreases to zero, the posterior probabilities in Equation 5 become hard. A standard deterministic annealing

**Algorithm:** model-based clustering via deterministic annealing

**Input:** A set of $N$ data objects $X = \{x_1, ..., x_N\}$, model structure $\Lambda = \{\lambda_1, ..., \lambda_K\}$, temperature decreasing rate $\alpha, 0 < \alpha < 1$, and final temperature $T_f$ (usually a small positive value close to 0).

**Output:** Trained models $\Lambda$ and a partition of the data objects given by the cluster identity vector $Y = \{y_1, ..., y_N\}$, $y_n \in \{1, ..., K\}$ .

**Steps:**

1. Initialization: initialize the model parameters $\Lambda$ and set $T$ to be high (a large number);
2. Optimization: optimize the objective in Equation 4 by iterating between an E-step (Equation 5) and an M-step (Equation 3) until convergence;
3. Annealing: lower the temperature parameter $T^{(new)} = \alpha T^{(old)}$, go to step 4 if $T < T_f$, otherwise go back to step 2.
4. For each data object $x_n$, set $y_n = \arg\max_y P(y|x_n)$ .

Figure 4: Deterministic annealing algorithm for model-based clustering.

algorithm for model-based clustering can be constructed as shown in Figure 4. Note that at each temperature, the EM algorithm is used to maximize the objective in Equation 4, with cluster labels $Y$ being the hidden variables and Equation 5 and Equation 3 corresponding to the E-step and M-step, respectively.

It can be shown that plugging Equation 5 into Equation 4 and setting $T = 1$ reduces the objective function to

$$L_1^* = \sum_x P(x) \log \left( \sum_y P(y) p(x|\lambda_y) \right) , \tag{6}$$

which is exactly the (incomplete data log-likelihood) objective function that the EM clustering maximizes (Neal and Hinton, 1998). As $T$ goes to 0, Equation 5 reduces to Equation 2 and the algorithm reduces to mk-means, independent of the actual $P(y)$'s (unless they are 1 and 0's). For any $T > 0$, iterating between Equation 5 and Equation 3 gives a soft model-based clustering algorithm that maximizes the objective function in Equation 4 for a given $T$.

This analysis makes it clear that the mk-means and EM clustering can be viewed as two special stages of the deterministic annealing algorithm (Figure 4), with $T = 0$ and $T = 1$, respectively, and they optimize two different objective functions ($L$ vs. $L - I(X;Y)$). This interesting view of the relationship between mk-means and EM clustering is different from a traditional view in which k-means is regarded as a special case of EM clustering (Mitchell, 1997; Neal and Hinton, 1998). Since larger $T$ indicates smoother objective function and a smaller number of local solutions, theoretically the EM clustering ($T = 1$) should have a better chance of finding good local solutions than the mk-means algorithm ($T = 0$). This justifies using EM clustering results to initialize the mk-means, which can be viewed as a one-step deterministic annealing algorithm (temperature decreases one time from $T = 1$ to $T = 0$). This also reveals the inappropriateness of the practice of using k-means to initialize EM clustering. A safer approach is to start at a high $T$ and gradually reduce $T$ towards 0. Moreover, clustering by "melting" can also be achieved if done slowly and carefully (Wong, 1993).

A stochastic variant of the mk-means algorithm, *stochastic mk-means*, was described by Kearns et al. (1997) as *posterior assignment* (as opposed to k-means assignment and EM assignment). The basic idea is that each data object is *stochastically* (and entirely, not fractionally) assigned to one of the $K$ clusters according to the posterior probability $P(y|x)$. The stochastic mk-means can also be viewed as a sampled version of the EM clustering, where one uses a sampled E-step based on the posterior probabilities.

We now present a generalized batch version of the SOM and Neural-Gas algorithms in the context of model-based clustering and show they also can be interpreted from a deterministic annealing point of view. A distinct feature of SOM is the use of a topological map, in which each cluster has a fixed coordinate. Let the map location of cluster $y$ be $\phi_y$ and $K_\alpha(\phi_y, \phi_{y'}) = \exp\left(-\frac{\|\phi_y - \phi_{y'}\|^2}{2\alpha^2}\right)$ be the neighborhood function. Let $y(x) = \arg\max_y \log p(x|\lambda_y)$. The batch SOM algorithm amounts to iterating between Equation 3 and the following step:

$$P(y|x) = \frac{K_\alpha(\phi_y, \phi_{y(x)})}{\sum_{y'} K_\alpha(\phi_{y'}, \phi_{y(x)})} ,$$

where $\alpha$ is a parameter controlling the width of the neighborhood function and decreases gradually during the clustering process. Here $\alpha$ can be seen as a temperature parameter as in a deterministic annealing process. SOM can be viewed as using a constrained E-step, where the calculation of posteriors $P(y|x)$ is not only based on the actual $\log p(x|\lambda_y)$'s, but also constrained by the topological map structure. This mechanism gives SOM the advantage that all resulting clusters are related according to the pre-specified topological map.

The batch Neural-Gas algorithm differs from SOM and the algorithm in Figure 4, only in how $P(y|x)$ is calculated

$$P(y|x) = \frac{e^{-r(x,y)/\beta}}{\sum_{y'} e^{-r(x,y')/\beta}} ,$$

where $\beta$ is an equivalent temperature parameter and $r(x,y)$ is a function of cluster rank. For example, $r(x,y)$ takes value 0 if $y$ is the closest cluster centroid to $x$, value 1 if $y$ is the second closest centroid to $x$, and value $k-1$ if $y$ is the $k$-th closest centroid to $x$. It has been shown that the online Neural-Gas algorithm can converge faster and find better local solutions than the SOM and deterministic annealing algorithms for certain problems (Martinetz et al., 1993).

### 2.3 Model-based Hierarchical Clustering

For partitional clustering methods, the number of clusters needs to be specified *a priori*. This number, however, is often unknown in many clustering problems. Moreover, sometimes one prefers the clustering algorithm to return a series of nested clusterings for interactive analysis (Seo and Shneiderman, 2002). Hierarchical clustering techniques provide such an advantage. Although one can run k-means or EM clustering multiple times with different numbers of clusters, the returned clusterings are not guaranteed to be structurally related. Bottom-up hierarchical agglomerative clustering has been the most popular hierarchical method (Jain et al., 1999), although top-down methods have also been used, e.g., Steinbach et al. (2000).

Researchers usually do not discriminate between model-based and similarity-based approaches for hierarchical clustering algorithms. In contrast, we make a distinction between model-based

hierarchical methods and similarity-based ones. Ward's algorithm and centroid methods are model-based methods. The former selects two clusters whose merge maximizes the resulting likelihood, whereas the latter chooses the two clusters whose centroids are closest. Both methods use spherical Gaussians as the underlying models. On the other hand, single-link, complete-link, and average-link methods are all discriminative methods, since data-pairwise distances have to be calculated and form the basis for computing inter-cluster distances.

To design model-based hierarchical clustering algorithms, one first needs a methodology for identifying two clusters to merge at each iteration. To do this, we define a "distance"[3] measure between clusters (i.e., models) and then iteratively merge the closest pair of clusters. A traditional way is to choose the two clusters such that merging them results in the largest log-likelihood $\log P(X|\Lambda)$ (Fraley, 1999; Meila and Heckerman, 2001). The distance for this method can be defined as

$$D^W(\lambda_k,\lambda_j) = \log P(X|\Lambda_{before}) - \log P(X|\Lambda_{after}) \ , \tag{7}$$

where $\Lambda_{before}$ and $\Lambda_{after}$ are the set of all parameters before and after merging two models ($\lambda_k$ and $\lambda_j$), respectively. We call this measure (generalized) Ward's distance since this is exactly the Ward's algorithm (Ward, 1963) when equi-variant Gaussian models are used.

The above method is not efficient, however, since to find the closest pair one needs to train a merged model for every pair of clusters and then evaluate the resulting log-likelihood. In practice, except for some specific models for which the Ward's distance can be efficiently computed (Fraley, 1999; Meila and Heckerman, 2001), the Kullback-Leibler (KL) distance measure which does not involve re-estimating models has been commonly used (Sinkkonen and Kaski, 2001; Ramoni et al., 2002). Exact KL divergence is difficult to calculate for complex[4] models. An empirical KL divergence (Juang and Rabiner, 1985) between two models $\lambda_k$ and $\lambda_j$ can be defined as

$$D^K(\lambda_k,\lambda_j) = \frac{1}{|X_k|}\sum_{x\in X_k}\left(\log p(x|\lambda_k) - \log p(x|\lambda_j)\right) \ , \tag{8}$$

where $X_k$ is the set of data objects being grouped into cluster $k$. This distance can be made symmetric by defining (Juang and Rabiner, 1985)

$$D_s^K(\lambda_k,\lambda_j) = \frac{D^K(\lambda_k,\lambda_j) + D^K(\lambda_j,\lambda_k)}{2} \ ,$$

or using the Jensen-Shannon divergence with $\pi_1 = \pi_2 = \frac{1}{2}$ (Lin, 1991):

$$D^{JS}(\lambda_k,\lambda_j) = \frac{1}{2}D^K(\lambda_k, \frac{\lambda_k+\lambda_j}{2}) + \frac{1}{2}D^K(\lambda_j, \frac{\lambda_k+\lambda_j}{2}) \ .$$

Compared to classical hierarchical agglomerative clustering (HAC) algorithms, KL divergence is analogous to the centroid method. It can be shown that when Gaussian models with equal covariance matrices are used, the KL divergence reduces to the Mahalanobis distance between two cluster means. Motivated by this observation as well as the single-link and complete-link HAC

3. This and several other quantities defined in this section that are used as merging criteria are termed "distance" only in a colloquial sense, since they may not satisfy the symmetry or triangle inequality properties needed of a metric. "Divergence" is the technically correct term in such situations.
4. A complex model has high representational power and is able to describe complex data, such as non-Gaussian vectors and variable length sequences.

algorithms, we propose several modified KL distances. Corresponding to single-link, we define a *minKL* distance as

$$D^m(\lambda_k,\lambda_j) = \min_{x\in X_k} (\log p(x|\lambda_k) - \log p(x|\lambda_j)) \ , \tag{9}$$

and corresponding to complete-link, we define a *maxKL* distance as

$$D^M(\lambda_k,\lambda_j) = \max_{x\in X_k} (\log p(x|\lambda_k) - \log p(x|\lambda_j)) \ . \tag{10}$$

Finally, to characterize high "boundary density" between two clusters for building complex-shaped clusters, we propose a *boundaryKL* distance measure

$$D^B(\lambda_k,\lambda_j) = \frac{1}{|B_k|} \sum_{x\in B_k} (\log p(x|\lambda_k) - \log p(x|\lambda_j)) \ , \tag{11}$$

where $B_k$ is the $\eta$ fraction of $X_k$ that have smallest $\log p(x|\lambda_k) - \log p(x|\lambda_j)$ values. A value of 0 for $\log p(x|\lambda_k) - \log p(x|\lambda_j)$ defines the "boundary" between cluster $k$ and $j$. This distance measure reduces to the minKL distance if $B_k$ contains only one data object, and to the KL distance if $B_k = X_k$. The minKL and maxKL measures are more sensitive to outliers than the KL distance since they are defined on only one specific data object. A favorable property of the minKL measure, however, is that hierarchical algorithms using this distance (analogous to single-link HAC methods) can produce arbitrary-shaped clusters. To guard against outliers but reap the benefits of single-link methods, we set $\eta$ to be around 10%. The experimental results in Section 6 demonstrate the effectiveness of this new distance measure.

Figure 5 describes a generic view of model-based HAC algorithm. Instances of this generic algorithm include existing model-based HAC algorithms that were first explored by Banfield and Raftery (1993) and Fraley (1999) with Gaussian models, later by Vaithyanathan and Dom (2000) with multinomial models for clustering documents, and more recently by Ramoni et al. (2002) with Markov chain models for grouping robot sensor time series. The first three works used the Ward's distance in Equation 7 and the fourth one employed the KL distance in Equation 8.

### 2.4 Practical Considerations

In general, the maximum likelihood estimation of model parameters in Equation 2 can itself be an iterative optimization process (e.g., estimation of HMMs), that needs appropriate initialization and may get into local minima. For the clustering algorithms to converge, sequential initialization has to be used. That is, the model parameters resulting from the previous clustering iteration should be used to initialize the current clustering iteration, to guarantee that the objective (Equation 1 or 4) does not decrease.

The second observation is that ML model estimation sometimes leads to a singularity problem, i.e., unbounded log-likelihood. This can happen for a continuous probability distribution for which $p(x|\lambda)$ is a probability density that can become unbounded even though $\int_x p(x|\lambda)dx = 1$. For example, for Gaussian models, this occurs when the covariance matrix becomes singular. For discrete distributions, this would not happen since $p(x|\lambda)$ is then upper-bounded by 1. The singularity problem is often dealt with in one of the following three ways: (a) restarting the whole clustering algorithm with a different initialization (Juang et al., 1986); (b) using maximum *a posteriori* estimation with an appropriate prior (Gauvain and Lee, 1994); (c) using constrained ML estimation, e.g., lower-bound the variance for spherical Gaussian models (Bishop, 1995).

**Algorithm:** model-based HAC

**Input:** A set of $N$ data objects $X = \{x_1, ..., x_N\}$, and model structure $\lambda$.

**Output:** An $N$-level cluster (model) hierarchy and hierarchical partition of the data objects, with $n$ models/clusters at the $n$-th level.

**Steps:**

1. Initialization: start with the $N$-th level, initialize each data object as a cluster itself and train a model for each cluster, i.e., $\lambda_n = \max_{\lambda} \log p(x_n|\lambda)$ ;

2a. Distance calculation: compute pairwise inter-cluster distances using an appropriate measure, e.g., one of the measures defined in Equations 7–11;

2b. Cluster merging: merge the two closest clusters (assume they are $k$ and $j$) and re-estimate a model from the merged data objects $X_k = X_k \cup X_j$, i.e., $\lambda_k = \max_{\lambda} \log P(X_k|\lambda)$ ;

3. Stop if all data objects have been merged into one cluster, otherwise go back to Step 2a.

Figure 5: Model-based hierarchical agglomerative clustering algorithm.

A third comment is on the performance of mk-means, stochastic mk-means, and EM clustering. In practice, it is common to see the condition $p(x|\lambda_{y(x)}) \gg p(x|\lambda_k), \forall k \neq y(x)$ (especially for complex models such as HMMs), which means that $P(k|x)$ in Equation 5 will be dominated by the likelihood values and be very close to 1 for $k = y(x)$, and 0 otherwise, provided that $T$ is small ($\leq 1$). This suggests that the differences between hard mk-means, stochastic mk-means, and EM clustering algorithms are often small, i.e., their clustering results will be similar in many practical applications.

Finally, let us look at the computational complexity for model-based clustering algorithms. First consider partitional clustering involving models for which the estimation of model parameters has a closed-form solution and does not need an iterative process, e.g., Gaussian, Multinomial, etc. For each iteration, the time complexity is linear in the number of data objects $N$ and the number of clusters $K$ for both the data assignment step and the model estimation step. The total complexity is $O(KNM)$, where $M$ is the number of iterations. For those models for which the estimation of parameters needs an iterative process (e.g., hidden Markov models with Gaussian observation density), the model estimation complexity is $O(KNM_1)$ for each clustering iteration, where $M_1$ is the number of iterations used in the model estimation process. In this case, the total complexity of the clustering process is $O(KNMM_1)$. Theoretically the number of iterations $M$ and $M_1$ could be very large and may increase a bit with $N$, but in practice the maximum number of iterations is typically set to be a constant based on empirical observations. In our experiments, the EM algorithm usually converges very fast (within 20 to 30 iterations when clustering documents).

The above analysis applies to mk-means, stochastic mk-means, and EM clustering. For the model-based deterministic annealing algorithm, there is an additional outer loop controlled by the decreasing temperature parameter. Therefore, a slower annealing schedule is computationally more expensive.

For model-based hierarchical agglomerative clustering, there are $i$ models/clusters at the $i$-th level and one starts from $N$ clusters at the bottom. The number of inter-cluster distances to be

calculated is $\frac{N(N-1)}{2}$ for the bottom ($N$-th) level and $i-1$ for the $i$-th level ($i < N$, other than the $N$-th level, one only needs to compute the distances between the merged cluster and other clusters). The total number of distances calculated for the whole hierarchy is

$$\frac{N(N-1)}{2} + (N-2) + (N-3) + \cdots + 1 \simeq O(N^2) \ .$$

Using the same logic, we can compute the total number of distance comparisons needed as

$$\frac{N(N-1)}{2} + \frac{(N-1)(N-2)}{2} + \cdots + \frac{2 \cdot 1}{2} \simeq O(N^3) \ ,$$

and the total complexity for model estimation as

$$N \cdot 1M_1 + 1 \cdot 2M_1 + 1 \cdot 3M_1 + \cdots + 1 \cdot NM_1 \simeq O(N^2 M_1) \ ,$$

where $M_1$ is the number of iterations used for model estimation. The complexity can be reduced to $O(N^2 \log N)$ for inter-cluster distance comparisons by using a clever data structure (e.g., heap) to store the comparison results (Jain et al., 1999). Clearly, a complexity of $O(N^3)$ or $O(N^2 \log N)$ is still too high for large datasets, which explains why model-based hierarchical clustering algorithms are not as popular as partitional ones. In areas where researchers do use hierarchical algorithms, model-specific tricks have often been used to further reduce the computational complexity (Fraley, 1999; Meila and Heckerman, 2001).

## 3. Clustering Evaluation

Comparative studies on clustering algorithms are difficult in general due to lack of universally agreed upon quantitative performance evaluation measures (Jain et al., 1999). Subjective (human) evaluation is often difficult and expensive, yet is still indispensable in many real applications. Objective clustering evaluation criteria include intrinsic measures and extrinsic measures (Jain et al., 1999). Intrinsic measures formulate quality as a function of the given data and similarities/models and are often the same as the objective function that a clustering algorithm explicitly optimizes. For example, the data likelihood objective was used by Meila and Heckerman (2001) to cluster text data using multinomial models. For low-dimensional vector data, the average (or summed) distance from cluster centers, e.g., the sum-squared error criteria used for the standard k-means algorithm, is a common criterion.

Extrinsic measures are commonly used when the category (or class) labels of data are known (but of course not used in the clustering process). In this paper, a class is a predefined ("true") data category but a cluster is a category generated by a clustering algorithm. Examples of external measures include the confusion matrix, classification accuracy, F1 measure, average purity, average entropy, and mutual information (Ghosh, 2003). There are also several other ways to compare two partitions of the same data set, such as the Rand index (Rand, 1971) and Fowlkes-Mallows measure (Fowlkes and Mallows, 1983) from the statistics community.

F1 measure is often used in information retrieval, where clustering serves as a way of improving the quality and accelerating the speed of search. The purity of a cluster is defined as the percentage of the majority category in the cluster. Entropy $H(\cdot)$ measures the category spread or uncertainty of a cluster and can be normalized to the range $[0,1]$ by dividing $\log K$, where $K$ is the number of classes. If all objects in a cluster come from one category, the purity is 1 and the normalized entropy

is 0. If a cluster contains an equal number of objects from each category, the purity is $1/K$ and the normalized entropy is 1.

It has been argued that the mutual information $I(Y;\hat{Y})$ between a *r.v.* $Y$, governing the cluster labels, and a *r.v.* $\hat{Y}$, governing the class labels, is a superior measure to purity or entropy (Dom, 2001; Strehl and Ghosh, 2002). Moreover, by normalizing this measure to lie in the range [0,1], it becomes relatively impartial to $K$. There are several choices for normalization based on the entropies $H(Y)$ and $H(\hat{Y})$. We shall follow the definition of normalized mutual information (*NMI*) using geometrical mean, $NMI = \frac{I(Y;\hat{Y})}{\sqrt{H(Y)\cdot H(\hat{Y})}}$, as given by Strehl and Ghosh (2002). The corresponding sample estimate is:

$$NMI = \frac{\sum_{h,l} n_{h,l} \log\left(\frac{n \cdot n_{h,l}}{n_h n_l}\right)}{\sqrt{\left(\sum_h n_h \log\frac{n_h}{n}\right)\left(\sum_l n_l \log\frac{n_l}{n}\right)}},$$

where $n_h$ is the number of data objects in class $h$, $n_l$ the number of objects in cluster $l$ and $n_{h,l}$ the number of objects in class $h$ as well as in cluster $l$. The *NMI* value is 1 when clustering results perfectly match the external category labels and close to 0 for a random partitioning.

In the simplest scenario where the number of clusters equals the number of categories and their one-to-one correspondence can be established, any of these external measures can be fruitfully applied. For example, when the number of clusters is small ($< 4$), the accuracy measure is intuitive and easy to understand. However, when the number of clusters differs from the number of original classes, the confusion matrix is hard to read and the accuracy difficult (or sometimes impossible) to calculate. In such situations, the *NMI* measure is better than purity and entropy measures, both of which are biased towards high $k$ solutions (Strehl et al., 2000; Strehl and Ghosh, 2002).

In this paper, several different measures are used, and we explain in each case study what measures we use and why.

## 4. A Case Study on Document Clustering

We recently performed an extensive comparative study of model-based approaches to document clustering (Zhong and Ghosh, 2003a). This section reports on a small subset of this study with an intent to highlight how the unified framework proves very helpful in such an endeavor. Therefore, details of the data sets, experimental setting and comparative results are relegated to Zhong and Ghosh (2003a) while we focus here on the experimental process. In particular, we compare two different probabilistic model types, namely mixtures of multinomials and of von-Mises Fisher distributions. For each model type, we instantiate the four generic model-based clustering algorithms (mk-means, stochastic mk-means, EM, and deterministic annealing) described in Section 2.2. The key observation is that the same pseudocode of mk-means (Figure 6) can be used for different models—only the model re-estimation segment (Step. 2a) needs to be changed. Thus, code development becomes easier and experimental settings can be automatically kept the same for different models to ensure a fair comparison.

The traditional vector space representation is used for text documents, i.e., each document is represented as a high dimensional vector of "word"[5] counts in the document. The dimensionality equals the number of words in the vocabulary used.

5. Used in a broad sense since it may represent individual words, stemmed words, tokenized words, or short phrases.

**Algorithm:** mk-means

**Input:** Data objects $X = \{x_1, ..., x_N\}$, and model structure $\Lambda = \{\lambda_1, ..., \lambda_K\}$.

**Output:** Trained model parameters $\Lambda$ and a partition of the data samples given by the cluster identity vector $Y = \{y_1, ...y_N\}$, $y_i \in \{1, ..., K\}$ .

**Steps:**

1. Initialization: initialize the model parameters $\Lambda$ and cluster identity vector $Y$;

2a. Model re-estimation: for each cluster $j$, let $X_j = \{x_i | y_i = j\}$, the parameters of each model $\lambda_j$ is re-estimated as $\lambda_j = \max_{\lambda} \sum_{x \in X_j} \log p(x|\lambda)$ ;

2b. Sample re-assignment: for each data sample $i$, set $y_i = \arg\max_j \log p(x_i|\lambda_j)$;

3. Stop if $Y$ does not change, otherwise go back to Step 2a.

Figure 6: Common mk-means template for document clustering case study.

### 4.1 Models

Multinomial models have been quite popular for text clustering (Meila and Heckerman, 2001), and we use the standard formulation for estimating the model parameters following McCallum and Nigam (1998), who use Laplace smoothing to avoid zero probabilities. The second model uses the von Mises-Fisher distribution, which is the analogue of the Gaussian distribution for directional data in the sense that it is the unique distribution of $L_2$-normalized data that maximizes the entropy given the first and second moments of the distribution (Mardia, 1975). There is a long-time folklore in the information retrieval community that the direction of a text vector is more important than its magnitude, leading to the practices of using cosine similarity, and of normalizing such vectors to unit length using $L_2$ norm. Thus a model for directional data seems worthwhile to consider. The pdf of a vMF distribution is

$$p(x|\lambda) = \frac{1}{Z(\kappa)} \exp\left(\kappa \cdot x^T \mu\right) ,$$

where $x$ is a $L_2$-normalized data vector, $\mu$ the $L_2$-normalized mean vector, and the Bessel function $Z(\kappa)$ a normalization term. The $\kappa$ measures the directional variance (or dispersion) and the higher it is, the more peaked the distribution is. The maximum likelihood estimation of $\mu$ is simple and given by $\mu = \frac{\sum x}{\|\sum x\|}$. The estimation of $\kappa$, however, is rather difficult due to the Bessel function involved (Banerjee and Ghosh, 2002a; Banerjee et al., 2003a). In a k-means clustering setting, if $\kappa$ is assumed to be the same for all clusters, then the clustering results do not depend on $\kappa$, which can be ignored. In this case, we can evaluate the average cosine similarity ($\frac{1}{N}\sum_x x^T \mu$, which is actually a displaced log-likelihood) as the objective (to be minimized). For EM clustering, the maximum likelihood solution has been derived by Banerjee et al. (2003a) including the computationally expensive updates for $\kappa$. In this work, for convenience, we use a simpler soft assignment scheme, which is discussed in Section 4.3. To use vMF models, the word-count document vectors are log(IDF)-weighted and then $L_2$-normalized. The IDF here stands for inverse document frequency. The log(IDF) weighting is a common practice in the information retrieval community used to de-emphasize the words that occur in too many documents. The weight for word $l$ is $\log \frac{N}{N_l}$, where $N$ is the number of documents and $N_l$ the number of documents that contain word $l$. The $L_2$ normalization is required since the

vMF distribution is a directional distribution defined on a unit hypersphere and does not capture any magnitude information.

### 4.2 Datasets

We used the 20-newsgroups dataset[6] and a number of datasets from the CLUTO toolkit[7] (Karypis, 2002). These datasets provide a good representation of different characteristics: the number of documents ranges from 204 to 19949, the number of terms from 5832 to 43586, the number of classes from 6 to 20, and the balance from 0.037 to 0.991. Here the balance of a dataset is defined as the ratio of the number of documents in the smallest class to the number of documents in the largest class. So a value close to 1(0) indicates a very (un)balanced dataset. A summary of all the datasets used in this section is shown in Table 1. Additional details on data characteristics and preprocessing are found work by Zhao and Karypis (2001) and Zhong and Ghosh (2003a).

| Data | Source | $n_d$ | $n_w$ | $k$ | $\bar{n}_c$ | Balance |
|---|---|---|---|---|---|---|
| *NG20* | 20 Newsgroups | 19949 | 43586 | 20 | 997 | 0.991 |
| *ohscal* | OHSUMED-233445 | 11162 | 11465 | 10 | 1116 | 0.437 |
| *hitech* | San Jose Mercury (TREC) | 2301 | 10080 | 6 | 384 | 0.192 |
| *k1b* | WebACE | 2340 | 21839 | 6 | 390 | 0.043 |
| *tr11* | TREC | 414 | 6429 | 9 | 46 | 0.046 |
| *tr23* | TREC | 204 | 5832 | 6 | 34 | 0.066 |
| *tr41* | TREC | 878 | 7454 | 10 | 88 | 0.037 |
| *tr45* | TREC | 690 | 8261 | 10 | 69 | 0.088 |

Table 1: Summary of text datasets. (For each dataset, $n_d$ is the total number of documents, $n_w$ the total number of words, $k$ the number of classes, and $\bar{n}_c$ the average number of documents per class.)

### 4.3 Experiments

For simplicity, we introduce some abbreviations: when instantiated with the multinomial model, the four algorithms—mk-means, stochastic mk-means, EM, and deterministic annealing—will be referred to as k-multinomials (*kmnls*), stochastic k-multinomials (*skmnls*), mixture-of-multinomials (*mixmnls*), and multinomial-based deterministic annealing (*damnls*), respectively. For vMF-based algorithms, the corresponding abbreviated names are *kvmfs*, *skvmfs*, *softvmfs*, and *davmfs*. We use *softvmfs* instead of *mixvmfs* for the soft vMF-based algorithm for the following reason. As mentioned previously, the estimation of parameter $\kappa$ in a vMF model is difficult but is needed for the mixture-of-vMFs algorithm. As a simple heuristic, we use $\kappa_m = 20m$, where $m$ is the iteration number. So $\kappa$ is set to be a constant for all clusters at each iteration, and gradually increases over iterations.

The *davmfs* algorithm uses an exponential schedule for the (equivalent) inverse temperature parameter $\kappa$, i.e., $\kappa_{m+1} = 1.1\kappa_m$, starting from 1 and up to 500. For the *damnls* algorithm, an inverse temperature parameter $\gamma = 1/T$ is created to parameterize the E-step of *mixmnls*. The annealing schedule for $\gamma$ is set to $\gamma_{m+1} = 1.3\gamma_m$, and $\gamma$ starts from 0.5 and can go up to 200.

6. http://kdd.ics.uci.edu/databases/20newsgroups/20newsgroups.html.
7. http://www.cs.umn.edu/∼karypis/CLUTO/files/datasets.tar.gz.

For all the model-based algorithms, we use a relative convergence criterion—when the likelihood objective changes less than 0.01% for the multinomial models or the average cosine similarity less than 0.1% for the vMF models, the iterative process is treated as converged. For all situations except the vMF models on the *NG20* dataset, the clustering process converges in 20 or fewer iterations on average. The largest average number of iterations needed is 38, for the *skvmfs* algorithm running on *NG20* with 30 clusters. Each experiment is run ten times, each time starting from a random balanced partition of documents. The averages and standard deviations of the normalized mutual information results are reported. We use *NMI* measure since the class labels of each document are available and the number of clusters is relatively large. Recall that *NMI* measures how well the clustering results match existing category labels. We also include the results for one state-of-the-art graph partitioning approach to document clustering—CLUTO (Karypis, 2002). We use the *vcluster* algorithm in the CLUTO toolkit with default setting. The algorithm is run ten times, each time with randomly ordered documents. Note that results of regular k-means (using Euclidean distance) are not included since this is well known to perform miserably for high-dimensional text data (Strehl et al., 2000).

### 4.4 Discussion

Table 2 shows the *NMI* results on the *NG20* and *ohscal* datasets, across different number of clusters for each dataset. All numbers in the table are shown in the format *average* $\pm$ *1 standard deviation*. Boldface entries highlight the best performance in each column. The number of clusters $K$ does not seem to affect much the relative comparison between different algorithms, at least for the range of $K$ we have experimented with in this study. This is also the case for other datasets (Zhong and Ghosh, 2003a). Therefore, to save space, we show the *NMI* results on all other datasets for one specific $K$ only in Table 3.

| | *NG20* | | | *ohscal* | | |
|---|---|---|---|---|---|---|
| $K$ | 10 | **20** | 30 | 5 | **10** | 15 |
| kmnls | $.50 \pm .02$ | $.53 \pm .03$ | $.53 \pm .02$ | $.37 \pm .01$ | $.37 \pm .02$ | $.37 \pm .01$ |
| skmnls | $.51 \pm .02$ | $.53 \pm .03$ | $.54 \pm .02$ | $.37 \pm .01$ | $.37 \pm .02$ | $.37 \pm .02$ |
| mixmnls | $.52 \pm .02$ | $.54 \pm .03$ | $.54 \pm .02$ | $.37 \pm .01$ | $.37 \pm .02$ | $.38 \pm .02$ |
| damnls | $.55 \pm .03$ | $.57 \pm .02$ | $.55 \pm .02$ | $.38 \pm .01$ | $.39 \pm .02$ | $.39 \pm .02$ |
| kvmfs | $.53 \pm .02$ | $.54 \pm .01$ | $.52 \pm .01$ | $.40 \pm .03$ | $.43 \pm .03$ | $.41 \pm .01$ |
| skvmfs | $.53 \pm .03$ | $.55 \pm .01$ | $.54 \pm .02$ | $.39 \pm .02$ | $.44 \pm .02$ | $.41 \pm .01$ |
| softvmfs | $.55 \pm .02$ | $.57 \pm .02$ | $.56 \pm .01$ | $.40 \pm .02$ | $.44 \pm .02$ | $.41 \pm .01$ |
| davmfs | $\mathbf{.57 \pm .03}$ | $\mathbf{.59 \pm .02}$ | $.57 \pm .01$ | $.41 \pm .01$ | $\mathbf{.47 \pm .02}$ | $\mathbf{.45 \pm .01}$ |
| CLUTO | $.55 \pm .02$ | $.58 \pm .01$ | $\mathbf{.58 \pm .01}$ | $\mathbf{.44 \pm .01}$ | $.44 \pm .02$ | $.44 \pm .01$ |

Table 2: *NMI* results on *NG20* and *ohscal* dataset

The unified framework allows us to distinguish the effects of the assignment strategy from the impact of the probability models. In this study, the vMF-based algorithms fare better than the multinomial-based ones, and significantly so for the smaller datasets in Table 3, indicating that the directional characteristic is important in the high-dimensional vector space representation of text documents and working on normalized document vectors produces promising clustering results. For a given model, soft algorithms perform better than hard ones, but the gain is only marginal for large datasets or those with well-separated clusters. However, a study of the run times given by

| | *hitech* | *k1b* | *tr11* | *tr23* | *tr41* | *tr45* |
|---|---|---|---|---|---|---|
| $K$ | 6 | 6 | 9 | 6 | 10 | 10 |
| kmnls | $.23 \pm .03$ | $.55 \pm .04$ | $.39 \pm .07$ | $.15 \pm .03$ | $.49 \pm .03$ | $.43 \pm .05$ |
| skmnls | $.23 \pm .04$ | $.55 \pm .05$ | $.39 \pm .08$ | $.15 \pm .02$ | $.50 \pm .04$ | $.43 \pm .05$ |
| mixmnls | $.23 \pm .03$ | $.56 \pm .04$ | $.39 \pm .07$ | $.15 \pm .03$ | $.50 \pm .03$ | $.43 \pm .05$ |
| damnls | $.27 \pm .01$ | $.61 \pm .04$ | $.61 \pm .02$ | $.31 \pm .03$ | $.61 \pm .05$ | $.56 \pm .03$ |
| kvmfs | $.28 \pm .02$ | $.60 \pm .03$ | $.52 \pm .03$ | $.33 \pm .05$ | $.59 \pm .03$ | $.65 \pm .03$ |
| skvmfs | $.29 \pm .02$ | $.60 \pm .02$ | $.57 \pm .04$ | $.34 \pm .05$ | $.62 \pm .03$ | $.65 \pm .05$ |
| softvmfs | $.29 \pm .01$ | $.60 \pm .04$ | $.60 \pm .05$ | $.36 \pm .04$ | $.62 \pm .05$ | $.66 \pm .03$ |
| davmfs | $.30 \pm .01$ | $\mathbf{.67 \pm .04}$ | $.66 \pm .04$ | $.41 \pm .03$ | $\mathbf{.69 \pm .02}$ | $\mathbf{.68 \pm .05}$ |
| CLUTO | $\mathbf{0.33 \pm .01}$ | $.62 \pm .03$ | $\mathbf{.68 \pm .02}$ | $\mathbf{.43 \pm .02}$ | $.67 \pm .01$ | $.62 \pm .01$ |

Table 3: *NMI* Results on *hitech*, *k1b*, *tr11*, *tr23*, *tr41*, and *tr45* datasets

Zhong and Ghosh (2003a) indicates that algorithms using soft assignment take (slightly) longer time than those using hard assignments. While the kvmfs algorithm is the fastest overall, deterministic annealing is much slower for vMF distributions.

## 5. Balanced Model-based Clustering

The problem of clustering large scale data under constraints such as balancing has recently received attention in the data mining literature (Bradley et al., 2000; Tung et al., 2001; Banerjee and Ghosh, 2002b; Strehl and Ghosh, 2003; Zhong and Ghosh, 2003b). Balanced solutions yield comparable numbers of objects in each cluster, and are desirable in a variety of applications (Zhong and Ghosh, 2003b). However, since balancing is a global property, it is difficult to obtain near-linear time techniques to achieve this goal while retaining high cluster quality. In this section we show how balancing constraints can be readily incorporated into the unified framework. Essentially, one needs to perform a balanced partitioning of the bipartite graph (Figure 1) at each iteration of the EM algorithm, i.e., use a balanced E-step. The suggested approach can be easily generalized to handle partially balanced assignments and specific percentage assignment problems.

### 5.1 Balanced Model-based K-means

Since we focus on balanced hard clustering, the posteriors are either 1's or 0's. For simplicity, let $z_{nk}$ be a binary assignment variable with a value of 1 indicating that data object $x_n$ is assigned to cluster $k$. The completely balanced mk-means clustering problem can then be written as:

$$\begin{aligned} \max_{\{\lambda,z\}} \quad & \sum_{n,k} z_{nk} \log p(x_n|\lambda_k) \\ \text{s.t.} \quad & \textstyle\sum_k z_{nk} = 1, \forall n; \ \sum_n z_{nk} = N/K, \forall k; \\ & z_{nk} \in \{0,1\}, \forall n,k. \end{aligned}$$

If $N/K$ is not an integer, one can round it to the closest integer and make slight changes so that $\sum_{n,k} z_{nk} = N$ holds. This problem is decomposed into two subproblems corresponding to the E-step

**Algorithm:** Iterative greedy bipartitioning

**Input:** Log-likelihood matrix $w_{nk} = \log p(x_n|\lambda_k), n = 1,...,N,\ k = 1,...,K$ .

**Output:** A partition matrix $Z$ that satisfies $z_{nk} \in \{0,1\}$, $\sum_k z_{nk} = 1, \forall n$ and $\sum_n z_{nk} = \frac{N}{K}, \forall k$ .

**Steps:**

1. Initialization: set $j = 1$, $J = \{1,...,N\}$ and $z_{nk} = 0, \forall n,k$ ;
2. Calculating log-likelihood difference vector: let $dv_n = w_{nj} - \max_{k>j} w_{nk}, \forall n \in J$ and $dv = \{dv_n\}_{n \in J}$ ;
3. Bipartitioning using sorted difference vector: sort $dv$ in descending order and assign the top $\frac{N}{K}$ objects to cluster $j$, i.e., set $z_{nj} = 1, \forall n \in I$, where $I$ is the set of indices corresponding to the top $\frac{N}{K}$ objects;
4. Stop if $j = K$, otherwise let $j = j+1$, $J = J - I$ and go back to Step 2.

Figure 7: Iterative greedy bipartitioning algorithm.

and M-step in the EM algorithm, respectively. The balanced data assignment subproblem is

$$\begin{array}{ll} \max\limits_{\{z\}} & \sum\limits_{n,k} z_{nk} \log p(x_n|\lambda_k) \\ \text{s.t.} & \sum_k z_{nk} = 1, \forall n;\ \sum_n z_{nk} = N/K, \forall k; \\ & z_{nk} \in \{0,1\}, \forall n,k. \end{array} \tag{12}$$

This is an integer programming problem, which is NP-hard in general. Fortunately, this integer programming problem is special in that it has the same optimum as its corresponding real relaxation (Bradley et al., 2000), which is a linear programming problem. The best known exact algorithm to solve this linear programming problem is an improved interior point method that has a complexity of $O(N^3K^3/\log(NK))$, according to Anstreicher (1999).

To make this clustering algorithm scalable to large database, we seek approximate solutions (to the optimization problem in Equation 12) that can be obtained in time better than $O(N^2)$. We propose an iterative greedy bipartitioning algorithm (Figure 7) that assigns $N/K$ data objects to one of the $K$ clusters at each iteration in a locally optimal fashion.

The motivation behind this heuristic is that it solves the balanced assignment problem (Equation 12) exactly for $K = 2$. In other words, if there are just two clusters, one simply sorts the difference vector $dv_n = \log p(x_n|\lambda_1) - \log p(x_n|\lambda_2), n = 1,...,N$ in descending order and assigns the first $N/2$ objects to cluster 1 and the second half to cluster 2. It is easy to show that this gives a $\{\frac{N}{2}, \frac{N}{2}\}$ bipartition that maximizes the objective in Equation 12. For $K > 2$, a greedy bipartition is conducted at each iteration that separates the data objects for one cluster from all the others in such a way that the objective in Equation 12 is locally maximized. It is trivial to show that the $j$-th iteration of the algorithm in Figure 7 gives a locally optimal $\{\frac{N}{K}, \frac{(K-j)N}{K}\}$ bipartition that assigns $\frac{N}{K}$ objects to the $j$-th cluster.

Let us now look at the time complexity of this algorithm. Let $N_j = \frac{(K+1-j)N}{K}$ be the length of the difference vector computed at the $j$-th iteration. Calculating the difference vectors takes $\sum_j N_j(K-j) \simeq O(K^2N)$ time and sorting them takes $\sum_j N_j \log N_j \simeq O(KN \log N)$ time. The total

time complexity is $O(K^2N + KN\log N)$ for the greedy bipartitioning algorithm and $O(K^2MN + KMN\log N)$ for the resulting balanced clustering algorithm, where $M$ is the number of clustering iterations. The greedy nature of the algorithm stems from the imposition of an arbitrary ordering of the clusters using $j$. So one should investigate the effect of different orderings. In the experiments, the ordering is done at random in each experiment, multiple experiments are run and the variation in results is inspected. The results exhibit no abnormally large variations and suggest that the effect of ordering is small.

A post-processing refinement can be used to improve cluster quality when approximate rather than exact balanced solutions are acceptable. This is achieved by letting the results from completely balanced mk-means serve as an initialization for the regular mk-means. Since the regular mk-means has relatively low complexity of $O(KMN)$, this extra overhead is low. The experiments reported in this section reflect a "full" refinement in the sense that the regular mk-means in the refinement step is run until convergence. Alternatively, partial refinement such as one round of ML re-assignment can be used and is expected to give an intermediate result between the completely balanced one and the "fully" refined one. In the experimental results, intermediate results are not shown but they will be bounded from both sides by the completely balanced and the "fully" refined results.

The refinement step can be viewed from a second perspective—results from completely balanced clustering serve as an initialization to regular mk-means clustering. From this point of view, the completely balanced data assignment generates better initial clusters than random initialization according to our experimental results.

### 5.2 Results on Real Text Data

We used the *NG20* dataset described in Section 4.2, and two types of models, vMFs and multinomials. For each model type, we compare the balanced mk-means with regular mk-means clustering in terms of balance, objective value and mutual information with original labels, over different number of clusters. The balance of a clustering is defined as the normalized entropy of cluster size distribution of the clustering, i.e.,

$$N_{entro} = -\frac{1}{\log K}\sum_{k=1}^{K}\frac{N_k}{N}\log\left(\frac{N_k}{N}\right),$$

where $N_k$ is the number of data objects in cluster $k$. A value of 1 means perfectly balanced clustering and 0 extremely unbalanced clustering. The average log-likelihood of a clustering is given by $\frac{1}{N}\sum_x\sum_{l=1}^{d}x^{(l)}\log P_l^{(y)}$ for multinomial models and by $\frac{1}{N}\sum_x x^T\mu_y$ for von Mises-Fisher models, where $y = \arg\max_{y'} p(x|\lambda_{y'})$ and $d$ is the dimensionality of document vectors. Other experimental settings are the same as in Section 4.3.

Figure 8 show the results on the *NG20* dataset, with results for multinomial models on the left column and those for vMF models on the right. The first row shows balance results (normalized entropy), the second row average log-likelihood (*ALL*) values and the last row normalized mutual information (*NMI*) values. All results are shown as $average \pm 1$ *standard deviation* over 20 runs.

In all cases, the completely balanced clustering algorithms produce worse clusterings in terms of both the *ALL* and *NMI* measures since perfect balancing is too strict a constraint. But the balanced clustering algorithms, with refinement, perform either comparably or significantly better than regular mk-means in terms of both *ALL* and *NMI* results, and provide significantly more balanced

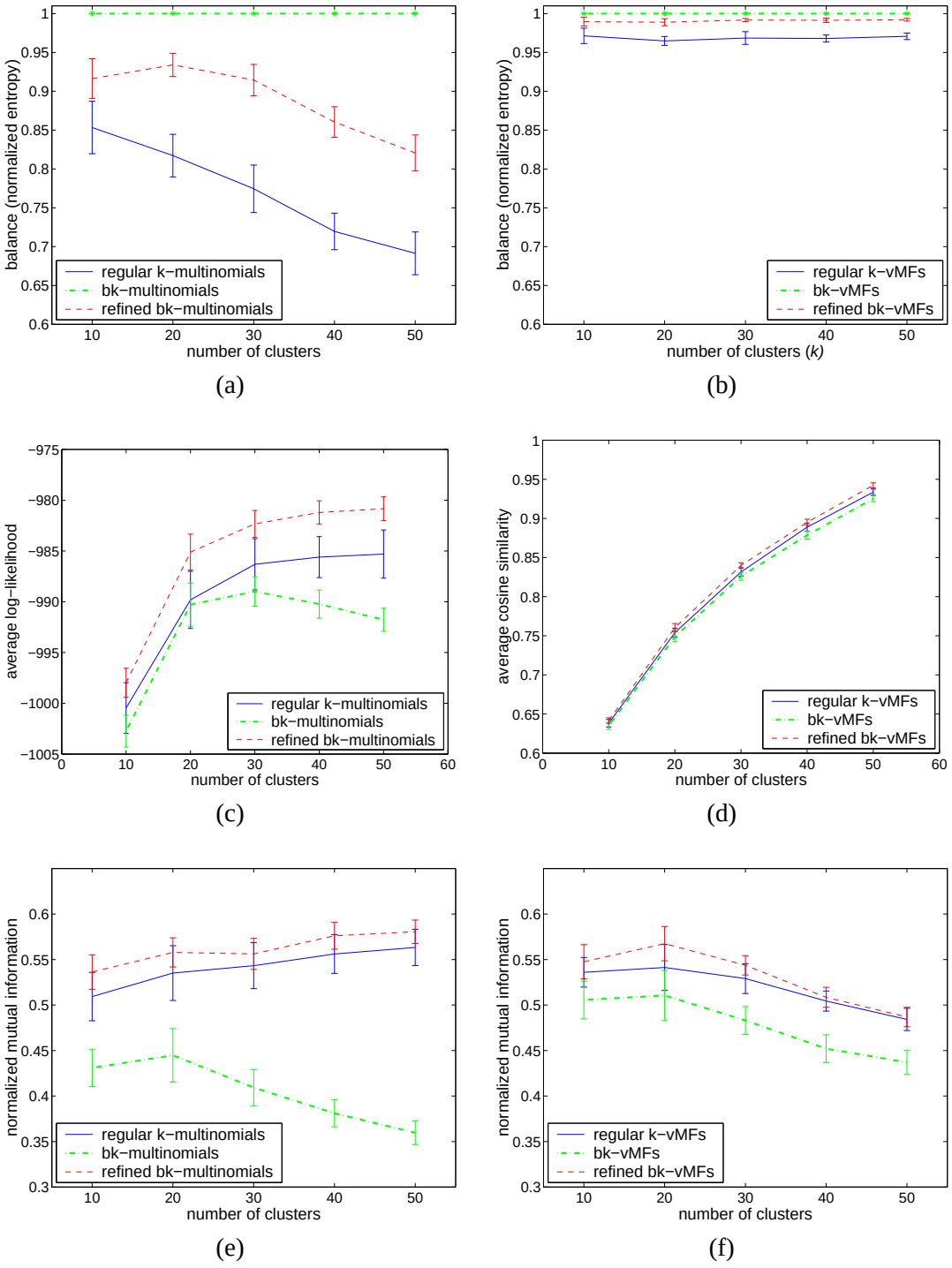



Figure 8: Results on the NG20 dataset: balance results for (a) multinomial models and (b) vMF models; log-likelihood results for (c) multinomial models and (d) vMF models; mutual information results for (e) multinomial models and (f) vMF models.

clusterings than the regular mk-means. Comparing multinomial models with vMF ones, we see that the vMF-based algorithms produce much more balanced clusterings for all datasets.

Note that the balanced variation of model-based clustering is generic since it is applied on the unified framework. One can simply plug in different models for different applications. For conciseness, we have only shown results on one dataset here. More experimental results can be found in an earlier paper (Zhong and Ghosh, 2003b).

## 6. Hybrid Model-based Clustering

This section presents a hybrid methodology that combines the advantages of partitional and hierarchical methods. The idea is a "reverse" of the "Scatter/Gather" approach (Cutting et al., 1992) and has been used by Vaithyanathan and Dom (2000) and Karypis et al. (1999). We shall first analyze the advantages of this hybrid approach for model-based clustering, and then present a new variation called hierarchical meta-clustering in Section 6.1. Two case studies are presented in Section 6.2 and 6.3 to show the benefits of hybrid model-based clustering algorithms. The key observation, however, is that the hybrid methodology is built on top of the generic partitional model of Section 2.2 and thus inherits its generality.

Figure 9 shows a generic model-based hybrid algorithm. We first cluster the data into $K_0$ (greater than the "natural" number of clusters $K$, which may be unknown) groups, that is, cluster the data into fine granularity, using a partitional method discussed in Section 2.2. For model-based clustering, this means that we now have $K_0$ models. The first step can be viewed as compressing/coarsening of the data. In the second step, we run the HAC algorithm starting from the $K_0$ clusters to iteratively merge the clusters that are closest to each other until all data objects are in one cluster or the process is stopped by a user. This hybrid approach returns a series of nested clusterings that can be either interactively analyzed by the user or evaluated with an optimization criterion. Note the methodology is not necessarily limited to model-based clustering (Karypis et al., 1999) although we only intend to show the benefits of model-based approaches in this paper.

This hybrid algorithm is a practical substitute for hierarchical agglomerative clustering algorithm since it keeps some hierarchical structure and the visualization benefit but reduces computational complexity from $O(N^3)$ to $O(K_0^3 + K_0 N M_1)$.[8] We can reasonably set $K_0$ to be a constant times $K$ to obtain the same complexity as model-based partitional clustering (assuming $K_0 \ll N$). It can also be used to improve partitional clustering algorithms by starting at $K_0$ clusters and iteratively merge back to $K$ clusters. This method has proven to be effective for graph partitioning techniques to generate high quality partitions (Karypis et al., 1999). An intuitive explanation is that the second merging step fine-tunes and improves the initial flat clusters. Experimental results in Section 6.3 show the effectiveness of the hybrid clustering approach on time series clustering with hidden Markov models.

In the next section, we introduce a particular variation of this hybrid algorithm—a hierarchical meta-clustering algorithm.

8. The number of distance comparisons is $O(K_0^3)$ and the complexity for estimating models is $O(K_0 N M_1)$ following the same analysis in Section 2.4. Recall that $M_1$ is the number of iterations for model training.

**Algorithm:** model-based hybrid partitional-hierarchical clustering

**Input:** Data objects $X = \{x_1, ..., x_n\}$, model structure $\lambda$ and hierarchy depth $K_0$.

**Output:** A $K_0$ level cluster (model) hierarchy and hierarchical partition of the data objects, with $i$ clusters at the $i$-th level.

**Steps:**

1. Flat partitional clustering: partition data objects into $K_0$ clusters using one of the model-based partitional clustering algorithms discussed in Section 2.2;

2a. Distance calculation: compute pairwise inter-cluster distances using one of the measures defined in Equations 8–11, and identify the closest cluster pair;

2b. Cluster merging: merge the two closest clusters (assume they are $j$ and $j'$) and re-estimate a model for the merged data objects $X_j = X_j \cup X_{j'}$, i.e., let $\lambda_j = \arg\max_{\lambda} \log P(X_j|\lambda)$ ;

3. Stop if all data objects have been merged into one cluster or user-specified number of clusters is reached, otherwise go back to Step 2a.

Figure 9: Model-based hybrid partitional-hierarchical clustering algorithm.

### 6.1 Hierarchical Meta-clustering

Let us first introduce a composite model $\lambda_{jj'} = \{\lambda_j, \lambda_{j'}\}$ for the cluster merged from cluster $j$ and $j'$ and define the likelihood of a data object $x$ given this model as

$$p(x|\lambda_{jj'}) = \max\{p(x|\lambda_j), p(x|\lambda_{j'})\} . \tag{13}$$

Let us call $\lambda_j$ and $\lambda_{j'}$ children of the composite model $\lambda_{jj'}$. For the set of data objects in the merged cluster $X_{jj'} = X_j \cup X_{j'}$, we then have

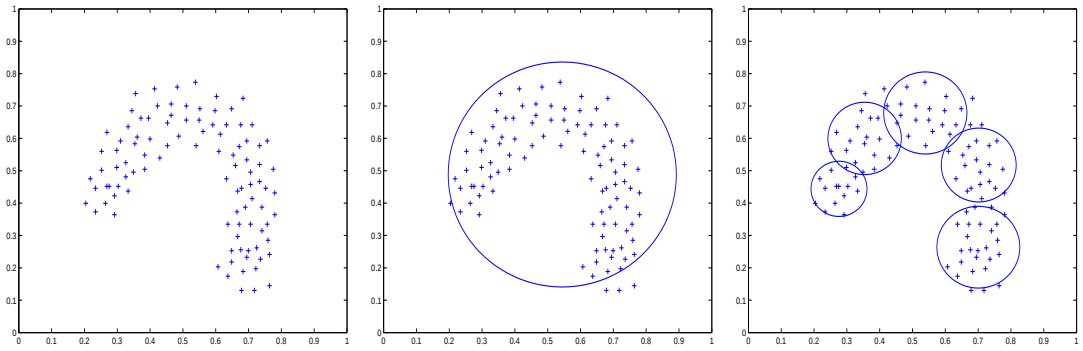
$$\log p(X_{jj'}|\lambda_{jj'}) = \log p(X_j|\lambda_j) + \log p(X_{j'}|\lambda_{j'}) . \tag{14}$$

Furthermore, we define the distance between two composite models $\lambda_a = \{\lambda_{a_1}, \lambda_{a_2}, ...\}$ and $\lambda_b = \{\lambda_{b_1}, \lambda_{b_2}, ...\}$ to be

$$D(\lambda_a, \lambda_b) = \min_{\substack{\lambda \in \lambda_a \\ \lambda' \in \lambda_b}} D(\lambda, \lambda') . \tag{15}$$

Two immediate benefits result from the above design. First, no model (parameter) re-estimation is needed after merging two clusters, since a composite model is simply represented by the parameters of its children. From Equation 14, it can be seen that now the cluster merging does not change the likelihood $P(X|\Lambda)$, which also means that the Ward's distance (Equation 7) cannot be used in this case. Second, a composite model can be used to characterize complex clusters that a single model represents poorly. For example, a (rotated) u-shape cluster (Figure 10) cannot be accurately modeled by a single Gaussian but can be approximated by a mixture of Gaussians. Using a single Gaussian model loses the u-shape structure of the cluster whereas concatenating five spherical Gaussian clusters gives a good representation, as shown in Figure 10(b) & (c).

Using the composite models defined in Equation 13 and the inter-cluster distances in Equation 15, we get a hierarchical meta-clustering algorithm, which is equivalent to treating each initial



(a) (b) (c)

Figure 10: (a) A u-shape cluster. (b) Using a single spherical Gaussian model for the cluster. (c) Using a union of five spherical Gaussian models for the cluster. (Each circle shows the isocontour of a Gaussian model at two times the standard deviation.)

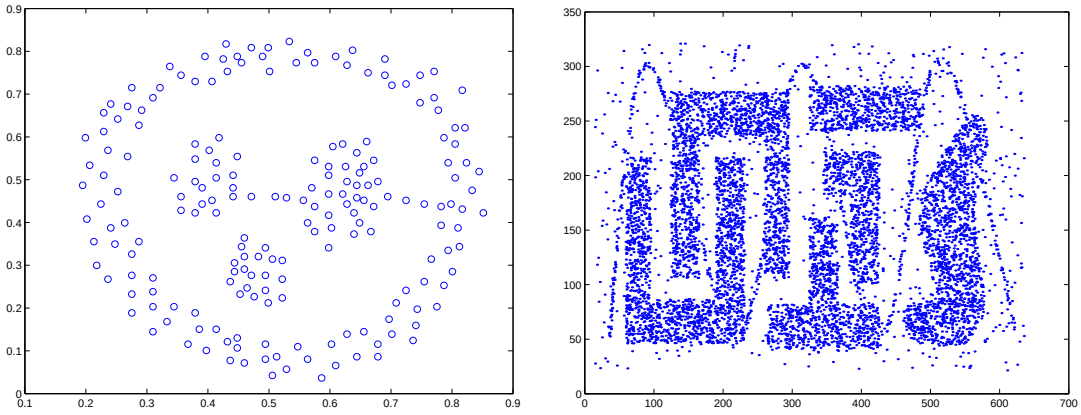
cluster as a meta-object and applying the traditional single-link hierarchical clustering algorithm to group the meta-objects. Obviously, nothing prevents us from using a different hierarchical method (e.g., complete-link, average-link, etc.) to cluster the meta-objects, by suitably modifying the measure definition in Equation 15. Each hierarchical method can be desirable in different applications.

The hierarchical meta-clustering algorithms can be seen as a combination of model-based flat clustering and discriminative hierarchical methods. Compared to using a single complex model, we favor this strategy of merging simple models to form complex clusters when a single complex model is difficult to define and to train. For example, what is the distribution for the u-shape cluster in Figure 10(a)? Furthermore, it is almost impossible to avoid poor local solutions even if we can define such a complex distribution.

It is sometimes helpful to produce approximately balanced clusters in the first (partitional) step. This may not be immediately clear, so let us again look at the u-shape cluster in Figure 10(a) as an example. Suppose we divide the data into five clusters in the flat clustering step. Using the k-means algorithm, we may get a very unbalanced clustering that contains one big cluster as in Figure 10(b) and four other near empty clusters (not shown), or a balanced solution as in Figure 10(c). While merging the five clusters back to one in either solution leads to the same set of data objects, we prefer the latter solution since it provides a useful hierarchy, disclosing the u-shape structure of the cluster.

### 6.2 Results on Synthetic 2-D Spatial Data

We tested our hybrid algorithm on two synthetic but difficult datasets: the d4 dataset in Figure 11(a), which contains 200 artificially generated data points, and the t4 dataset in Figure 11(b) included in the CLUTO toolkit (Karypis, 2002), which contains 8000 data points. There are no ground truth labels for these datasets but there are four natural clusters for the d4 dataset and six for the t4 dataset (plus some noise/outliers) according to human judgment. It is obvious that most of these natural clusters can not be well-modeled by a single Gaussian. In fact, it is difficult to propose a model that fits all of the arbitrary-shaped 2-D clusters.



(a) (b)

Figure 11: Synthetic 2-D datasets: (a) d4 dataset; (b) t4 dataset.

At first sight, one would probably turn to graph partitioning approaches to get good clustering results. Indeed, both traditional k-means and HAC algorithms fail miserably on these two datasets, whereas the spectral clustering algorithms (Kannan et al., 2000; Ng et al., 2002) identify the four natural clusters in d4, and a hybrid graph partitioning approach (Karypis et al., 1999; Karypis, 2002) produces all six natural clusters in t4. The hybrid graph partitioning algorithm first partitions the data into a large number of clusters and then merges (and refines) them back to a proper granularity level. In this section, we demonstrate that we can achieve the same intuitive clusters more efficiently with the proposed hybrid model-based approaches. The models used in this case are equi-variant spherical Gaussians, which actually result in an improved hierarchical k-means algorithm.

In the first step of our hybrid algorithm, we use a balanced version of the k-means algorithm (Zhong and Ghosh, 2003b) to partition the data into fine granularity, 12 clusters for the d4 dataset and 30 clusters for the t4 dataset.[9] When varying the number of clusters between 12 and 20 (for d4) and between 25 and 40 (for t4), we have observed that the final hybrid clustering results are relatively insensitive. Figure 12(a) & (b) show the balanced results.[10] It can be seen that when clustering the data into fine enough granularity, we get pure clusters (i.e., clusters that do not mix objects from different natural clusters). The balance constraint helps restrict each cluster to be well defined (not empty or too large). The resulting stable, well-defined clusters form a good basis for the next step, hierarchical merging. Currently the number of clusters is user-selected; automatic methods for model selection will be investigated in the future.

In the second step, we compute the cluster pairwise distances using the boundaryKL measure (Equation 11) with the parameter $\eta$ set to 0.1, and then apply single-link hierarchical meta-clustering to construct a meta-cluster hierarchy. We also observe that the final clustering results are relatively insensitive to the change of $\eta$ between 0.05 and 0.2. Figure 12(c) & (d) show the meta-cluster

9. We heuristically select $K_0$ to be a constant times $K$ (the "true" number of clusters which is either estimated empirically or known from prior knowledge). For hierarchical meta-clustering where actual clusters are represented by multiple simple models, we used roughly $3K$ to $6K$ flat clusters. For regular hybrid clustering (where we retrain models after merging two clusters), we used $2K$ flat clusters. Note these numbers are unavoidably heuristic since in reality the number $K$ itself needs to be estimated (see more discussion in Section 8).
10. We reused the symbols and colors in the figure, but each cluster is represented by a unique combination of symbol and color.

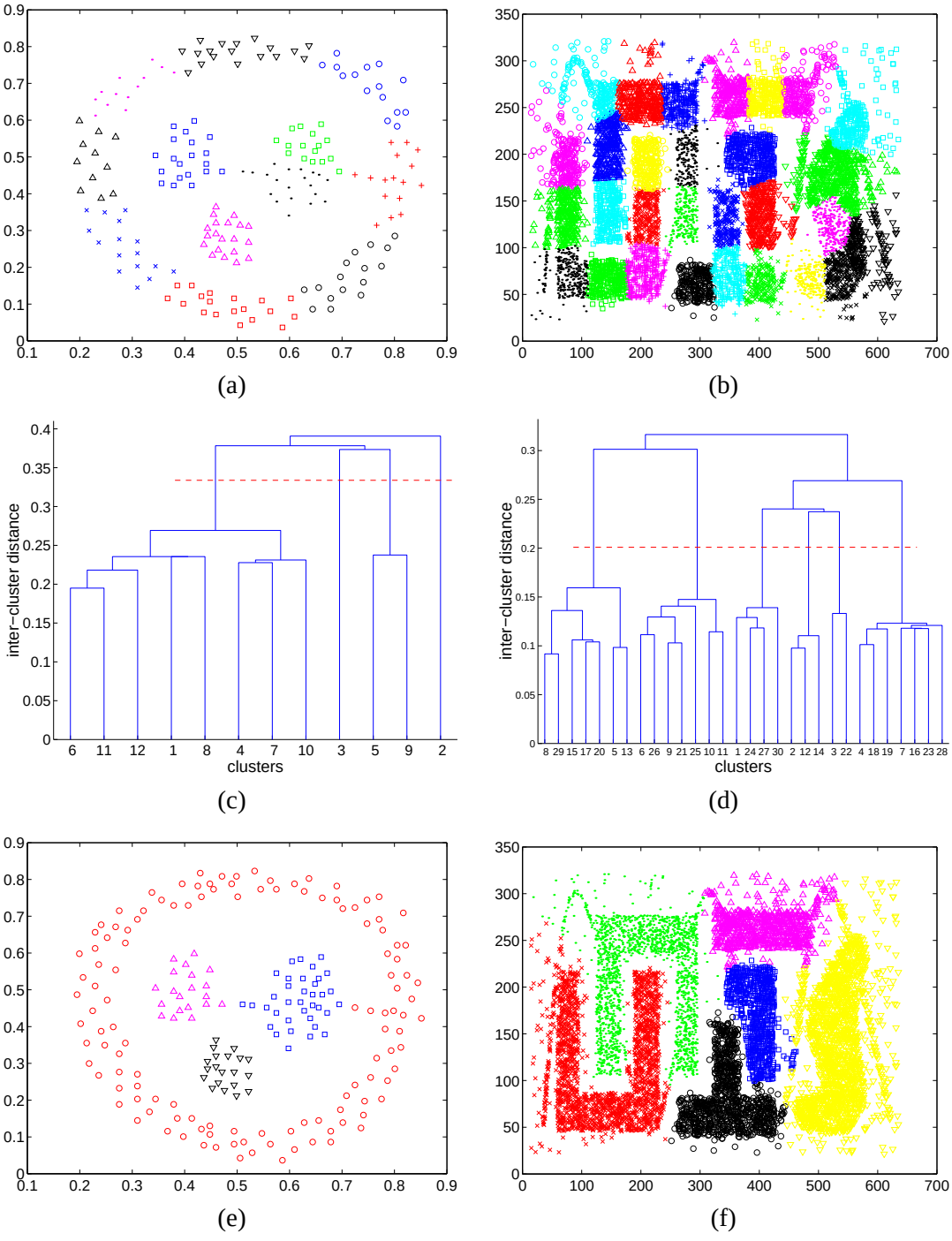



Figure 12: Results on the d4 and t4 datasets: balanced clustering of (a) d4 dataset into 12 clusters and (b) t4 dataset into 30 clusters; meta-cluster hierarchy for (c) d4 dataset and (d) t4 dataset using boundaryKL distances; hybrid clustering results for (e) d4 dataset in 4 clusters and (f) t4 dataset in 6 clusters.

hierarchies for the d4 and t4 datasets, respectively. From the hierarchies, it is evident that there are 4 clusters in d4 and 6 in t4. When slicing the hierarchies at an appropriate granularity level, we see from Figure 12(e) & (f) that the hierarchical meta-clustering produces decent natural clusters. These results suggest that hierarchical meta-clustering is useful in building a meta-cluster hierarchy (i.e., a hierarchical clustering of clusters) and finding the "right" number of clusters.

### 6.3 Results on Synthetic and EEG Time-series

In this case study, we used three datasets—two synthetic (hidden Markov model-generated) datasets and a real EEG dataset. It is worth noting that the simplistic approach of converting a time-series into a fixed-length vector (for example, so that we can use regular k-means) by time delay embedding is problematic in general because it leads to correlated components, has problems dealing with time warping, alignment and with variable length sequences, etc. The first synthetic dataset, *syn3*, contains three clusters and 60 sequences of length $T = 200$. The first 40 sequences are generated from two continuous HMM models (HMM1 and HMM2), 20 from each, same as in work by Smyth (1997). Both models have two hidden states and use the same priors and observation parameters. The priors are uniform and the observation distribution is univariate Gaussian with mean $\mu = 3$ and variance $\sigma^2 = 1$ for hidden state 1, and with mean $\mu = 0$ and variance $\sigma^2 = 1$ for hidden state 2. The state transition parameters of HMM1 and HMM2 are $A_1 = \begin{bmatrix} 0.6 & 0.4 \\ 0.4 & 0.6 \end{bmatrix}$ and $A_2 = \begin{bmatrix} 0.4 & 0.6 \\ 0.6 & 0.4 \end{bmatrix}$, respectively. The remaining 20 sequences are composed of uniformly distributed random numbers, which can be seen as generated from a special HMM model that has only one state and uniform observation distribution. The second synthetic dataset, *syn3-50*, is simply a subset of the first one, containing only the first 50 time points of each sequence in *syn3*.

The EEG dataset, *EEG2*, is extracted from the UCI KDD Archive (Hettish and Bay, 1999) and contains measurements from an electrode (F4) on the scalp. There are 20 measurements from two subjects, a control subject and an alcoholic subject, 10 from each. Each measurement is sampled at 256Hz for 1 second, producing a sequence length of 256. Figure 13 shows the 20 sequence objects. The goal is to group the time series from the same subject together into one cluster. We model each cluster with a univariate HMM. EEG signals are believed to be highly correlated with the sleep stages of human brain cells. The number of sleep stages is about 6 according to Geva and Kerem (1998).

The number of hidden states in the HMMs is manually chosen. We used five hidden states for the synthetic datasets and eight for the EEG dataset. The heuristic here is to choose a number in the high end of an expected range (so the models possess enough representational power to characterize the data). Also in the experiments we use classification accuracy as the evaluation criterion, assuming that the number of clusters is known *a priori*. The class label of a cluster is defined as the most popular class in the cluster. The accuracy measures the percentage of sequences that have correct class labels. The accuracy criterion is chosen here because it is easy to understand and also convenient to compare to the classification accuracy described in an earlier paper (Zhong and Ghosh, 2002) for the same data set.

#### 6.3.1 DISCUSSION

We compare five HMM-based clustering algorithms on the three datasets described above. Three of them are partitional algorithms instantiated from the three generic model-based partitional clustering

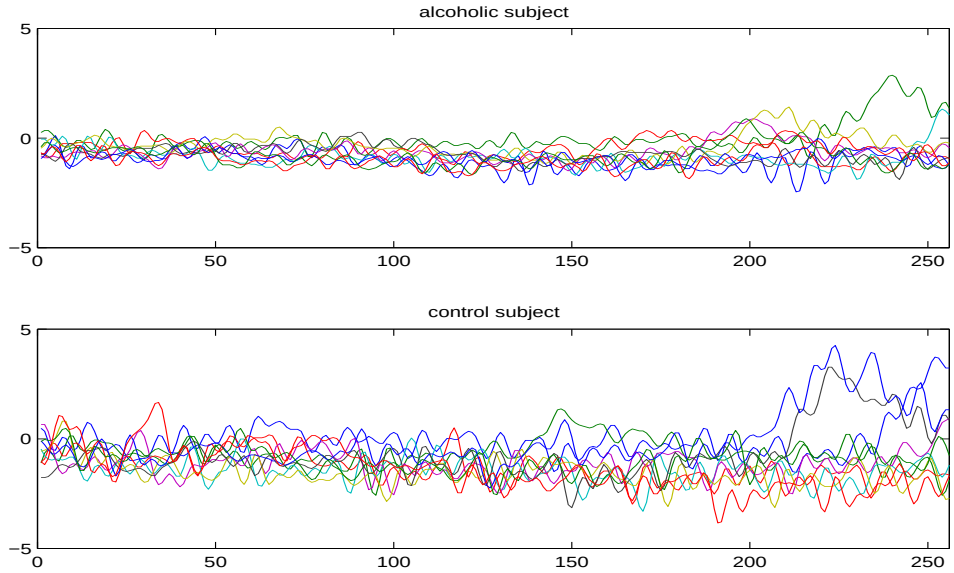



Figure 13: EEG data objects for one alcoholic subject and one control subject.

algorithms discussed in Section 2.2. By plugging in HMM models, we get HMM-based k-means (k-HMMs), stochastic k-HMMs, and mixture-of-HMMs (moHMMs), respectively. Two instantiated hybrid algorithms are

- *Hier-moHMMs:* the hybrid clustering algorithm (Figure 9) with the moHMMs algorithm used in the first step and the KL distance used for the second step;
- *Hier-k-HMMs:* the hybrid clustering algorithm (Figure 9) with the k-HMMs algorithm used in the flat clustering step and the KL distance used for the second step.

Clustering accuracy results are shown in Table 4. We use $K_0 = 2K$ flat clusters in the partitional step and run each algorithm 20 times (with different random initializations[11]) and report the averages and standard deviations. Boldface font indicates the best performance. The results are presented in the form $average \pm 1$ *standard deviation*. All three partitional methods perform comparably well and none is consistently better than the others across three datasets. Hybrid approaches clearly outperform partitional ones on all datasets. The hier-k-HMMs is better than hier-moHMMs for short sequences (*syn3-50*). Sophisticated models have been used to classify these EEG time sequences (Zhong and Ghosh, 2002) and the best classification accuracy (using five-fold cross validation) is around 90%. Therefore, the average accuracy of the hier-moHMMs algorithm on the EEG dataset (88.5%) is very good.

All methods have large standard deviations, indicating that the random initialization has a big effect on the clustering results, given the small data size. How to further reduce the initialization effect for HMM models remains an interesting task. Despite the high variances, we have run *t*-tests to measure the significance of our results, and observed that the best hybrid result in each column significantly outperforms all partitional results (at $p < 0.05$ level). A final comment is that

11. For the initialization method, we first train a global model using all sequences and then modify/train it for $K_0$ random balanced partitions of all sequences to get $K_0$ initial models.

| Approaches | Datasets | | |
|---|---|---|---|
| | *syn3* | *syn3-50* | *EEG256-2* |
| k-HMMs | $(75.2 \pm 17.6)\%$ | $(76 \pm 12.1)\%$ | $(81.5 \pm 14.2)\%$ |
| stochastic k-HMMs | $(85.8 \pm 19.5)\%$ | $(74.2 \pm 15.8)\%$ | $(82.7 \pm 14)\%$ |
| moHMMs | $(66.5 \pm 12.7)\%$ | $(71 \pm 10)\%$ | $(84.7 \pm 12.2)\%$ |
| hier-k-HMMs | $(86 \pm 18.3)\%$ | $(\mathbf{88.2 \pm 3.1})\%$ | $(85.2 \pm 10.2)\%$ |
| hier-moHMMs | $(\mathbf{90.5 \pm 15.4})\%$ | $(83.2 \pm 7.2)\%$ | $(\mathbf{88.5 \pm 8.3})\%$ |

Table 4: Clustering accuracy results on two synthetic datasets and one EEG dataset

the power of our hybrid model-based clustering lies in the suitability of the models used; a regular hybrid algorithm (e.g., regular k-means followed by hierarchical clustering) will not work well for time-series clustering.

## 7. Related Work

The majority of model-based clustering methods are based on the maximum likelihood formulation (Symons, 1981; McLachlan and Basford, 1988; Banfield and Raftery, 1993). Early work focused on different types of normal distributions. For example, Banfield and Raftery (1993) discussed clustering with a mixture of constrained Gaussian models, and Fraley (1999) described efficient hierarchical algorithms for special cases of Gaussian models.

Smyth (1997) applied a mixture of HMMs to cluster synthetic sequences. Cadez et al. (2000) then extended the work to a probabilistic framework for model-based partitional clustering using the EM algorithm. They advocate a mixture of generative models for clustering irregular data that are non-Gaussian and difficult or impossible to handle in the traditional vector space, e.g., sequences of different lengths. As mentioned in Section 1, their work is basically EM clustering, with an emphasis on applications to non-vector data. Their work partly motivates the work in this paper, which addresses all model-based clustering algorithms, including hierarchical methods.

Kalton et al. (2001) presented a two-step view of iterative clustering process—data assignment step and supervised learning step. Compared to the EM algorithm, the supervised learning step corresponds to the M-step but seems to be more general (need not be a maximum likelihood method). A potential difficulty, however, is that the two-step process with an arbitrary supervised learning method may not converge. For model-based partitional clustering, the convergence is guaranteed by the EM algorithm (provided that the maximum number of iterations we use is large enough).

Kamvar et al. (2002) presented interpretations of several classical hierarchical agglomerative clustering algorithms from a model-based standpoint. They intended to fit HAC algorithms into a standard model-based hierarchical clustering framework and discovered the corresponding model for each of the four agglomerative algorithms (Ward, single-link, complete-link, and average-link). However, only the Ward algorithm fits a natural model interpretation, while the other three match either a contrived model or an approximate one. From our viewpoint, the Ward algorithm is indeed a model-based approach in that it assumes spherical Gaussian models for each cluster and as a result each cluster can be represented by its mean. The latter three, however, are discriminative algorithms since they are based on data-pairwise similarities or distances. This explains the difficulty of determining suitable underlying generative models for them.

Deterministic annealing has been successfully used in a wide range of applications (Rose, 1998). But its applications to clustering have been largely restricted to vector data (Rose et al., 1993; Hofmann and Buhmann, 1997). Hofmann and Buhmann (1998) provided a unified treatment of SOM, Neural-Gas, and deterministic annealing algorithms for vector quantization applications (Gersho and Gray, 1992). They showed that the three types of algorithms are three different implementations of a continuation method (Allgower and Georg, 1990) for vector quantization, with different competitive learning rules. None of these work, however, have analyzed probabilistic model-based clustering or demonstrated the relationship between model-based k-means and EM clustering from an annealing perspective.

Hybrid algorithms were used by Cutting et al. (1992) in the famous "Scatter/Gather" approach, for reducing computational complexity. They applied the HAC algorithm on a small sampled dataset, fed the resulting clusters as initial cluster seeds to the k-means algorithm, and then ran the k-means on the entire dataset. A similar idea has also been explored by Meila and Heckerman (2001), where HAC was used to supply initial cluster centers for a subsequent EM clustering step. Vaithyanathan and Dom (2000) used a two-step hierarchical method for clustering documents: they apply a flat (i.e., partitional) clustering algorithm and then use feature selection and model selection methods to build a cluster hierarchy. Their method is basically the generic algorithm in Figure 9 instantiated with multinomial models and Ward's inter-cluster distance.

Another related work is the classic ISODATA algorithm (Hall and Ball, 1967), which performs a further refinement by splitting and merging the clusters obtained using the standard k-means algorithm. Clusters are merged if either the number of members in a cluster is less than a certain threshold or if the centers of two clusters are closer than a certain threshold. A cluster is split into two if its standard deviation exceeds a predefined value. This method needs several user-specified thresholds and assumes spherical clusters.

The multi-level graph partitioning algorithms (Karypis et al., 1999; Karypis, 2002) are similarity-based clustering approaches with a hybrid flavor. For example, the CLUTO toolkit (Karypis, 2002) allows the user to specify a large number of initial clusters, which are agglomeratively merged into the final desired number of clusters. This similarity-based method falls short in terms of theoretical complexity and interpretability compared to the hybrid model-based clustering method described in this paper.

## 8. Concluding Remarks

We have presented a unified framework for model-based clustering that provides a richer understanding of existing model-based clustering algorithms. The framework is applicable in any application domain for which good probabilistic models exist. Several related model-based partitional clustering algorithms, including both soft and hard k-means type methods, have been analyzed in detail, and their relationships explained from a deterministic annealing point of view. A comparative study on document clustering is conducted to show the usefulness of such a view. We have also designed several inter-cluster distances, leading to useful variations of model-based hierarchical clustering algorithms. Clear distinction between model-based and similarity-based hierarchical algorithms helps one gain a better understanding of existing hierarchical algorithms.

We have proposed two new variations of model-based clustering based on the unified framework—balanced clustering and hybrid clustering—to improve clustering results in certain applications. The

effectiveness of these model-based clustering algorithms is highlighted by experimental comparisons on several synthetic and real datasets.

A question we have not addressed is how to choose the final number of clusters, in either the partitional or the hierarchical/hybrid procedures. This is an old yet important problem for which a universally satisfactory answer is yet to be obtained. Bayesian model selection techniques (Schwarz, 1978; Banfield and Raftery, 1993; Fraley and Raftery, 1998) have been investigated extensively. Most simple criteria such as BIC (Bayesian Information Criterion) or AIC (Akaike Information criterion) either overestimate or underestimate the number of clusters, which severely limits their practical usability. Monte Carlo estimation of the posterior likelihood (Smyth, 1997) is more accurate but computationally expensive. Cross-validation methods can be effective when there is enough data; criteria such as the hold-out likelihood (Meila and Heckerman, 2001; Smyth, 1997) evaluated on a hold-out validation dataset often prove to be helpful.

It is often inappropriate to use model selection methods to find the number of clusters when models used are not a good description of the clusters. For example, the natural clusters in the synthetic 2-D datasets cannot be modeled by Gaussians. Attempts to estimate the number of Gaussians in a mixture of Gaussians for clustering the data will lead to a very high value of $K$. The use of hierarchical clustering alleviates the need to select $K$ since the user can interact with a hierarchical set of clusterings and choose the "best" one. Needless to say, in many clustering problems only a human can give the best domain-specific judgment.

An important related issue is how to choose a suitable model family. Highly expressive models are difficult to train (because of too many poor local solutions) and hard to interpret. This paper encourages researchers to start with simple models and combine them with hierarchical merging methods to characterize complex clusters.

Model-based partitional clustering algorithms can be made online, which is a promising feature in stream data mining applications. The competitive learning method, widely used in neural network literature, provides a way of constructing online k-means algorithms. It has been employed by Banerjee and Ghosh (2002a), Law and Kwok (2000), and Sinkkonen and Kaski (2001) for online model-based clustering of text documents, sequences, and gene expressions, respectively. This type of algorithm certainly deserves further investigation.

Another promising future direction is to examine the possibility of combining model-based clustering methods with discriminative ones. There has already been some preliminary work in this direction. For example, researchers have started constructing similarity measures from generative models (Amari, 1995; Jaakkola and Haussler, 1999; Tipping, 1999; Tsuda et al., 2003), but their impact on clustering performance is yet to be fully understood. A possible approach is to combine bipartite graph partitioning with model training based on Figure 1. In partitional clustering, hard assignments of data objects to models corresponds to a partitioning of the bipartite graph (Figure 1) with the constraint that each partition contains exactly one model vertex. In fact, the hard data assignment used in the mk-means algorithm is equivalent to a constrained minimum cut of the bipartite graph into $K$ partitions. For model-based hierarchical agglomerative clustering algorithms, merging two clusters corresponds to a partitioning of the graph into $K-1$ clusters in which one partition contains exactly two model vertices and all other partitions have one model vertex each. These connections may lead to a way of combining the model-based method with graph partitioning algorithms and deserve more investigation in the future. Further research may reveal new, robust clustering algorithms and insightful connections between generative and discriminative approaches.

## Acknowledgments

We thank the anonymous reviewers and Claire Cardie for helpful comments that have significantly improved the technical quality of this paper. This research was supported in part by an IBM Faculty Partnership Award from IBM/Tivoli and IBM ACAS, and by NSF grant IIS-0307792.

## References

D. W. Aha, D. Kibler, and M. Albert. Instance-based learning algorithms. *Machine Learning*, 6: 37–66, 1991.

E. L. Allgower and K. Georg. *Numerical Continuation Methods: An Introduction*. Springer-Verlag, Berlin Heidelberg, 1990.

S. Amari. Information geometry of the EM and em algorithms for neural networks. *Neural Networks*, 8(9):1379–1408, 1995.

K. M. Anstreicher. Linear programming in $o([n^3/\log n]l)$ operations. *SIAM Journal on Optimization*, 9(4):803–812, 1999.

A. Banerjee, I. Dhillon, S. Sra, and J. Ghosh. Generative model-based clustering of directional data. In *Proc. 9th ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining*, pages 19–28, 2003a.

A. Banerjee and J. Ghosh. Frequency sensitive competitive learning for clustering on high-dimensional hyperspheres. In *Proc. IEEE Int. Joint Conf. Neural Networks*, pages 1590–1595, May 2002a.

A. Banerjee and J. Ghosh. On scaling up balanced clustering algorithms. In *Proc. 2nd SIAM Int. Conf. Data Mining*, pages 333–349, April 2002b.

A. Banerjee, S. Merugu, I. Dhillon, and J. Ghosh. Clustering with Bregman divergences. Technical Report TR-03-19, Department of Computer Science, University of Texas at Austin, 2003b.

J. D. Banfield and A. E. Raftery. Model-based Gaussian and non-Gaussian clustering. *Biometrics*, 49(3):803–821, September 1993.

C. M. Bishop. *Neural Networks for Pattern Recognition*. Oxford University Press, 1995.

J. A. Blimes. A gentle tutorial of the EM algorithm and its application to parameter estimation for Gaussian mixture and hidden Markov models. Technical report, University of California at Berkeley, April 1998.

P. S. Bradley, K. P. Bennett, and A. Demiriz. Constrained k-means clustering. Technical Report MSR-TR-2000-65, Microsoft Research, Redmond, WA, 2000.

I. V. Cadez, S. Gaffney, and P. Smyth. A general probabilistic framework for clustering individuals and objects. In *Proc. 6th ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining*, pages 140–149, 2000.

D. Cutting, D. Karger, J. Pedersen, and J. W. Tukey. Scatter/Gather: A cluster-based approach to browsing large document collections. In *Proc. ACM SIGIR*, pages 318–329, 1992.

A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum-likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society B*, 39(1):1–38, 1977.

E. Dermatas and G. Kokkinakis. Algorithm for clustering continuous density HMM by recognition error. *IEEE Trans. Speech and Audio Processing*, 4(3):231–234, May 1996.

I. S. Dhillon. Co-clustering documents and words using bipartite spectral graph partitioning. In *Proc. 7th ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining*, pages 269–274, 2001.

I. S. Dhillon and D. S. Modha. Concept decompositions for large sparse text data using clustering. *Machine Learning*, 42(1):143–175, 2001.

B. Dom. An information-theoretic external cluster-validity measure. Technical Report RJ10219, IBM, 2001.

E. B. Fowlkes and C. L. Mallows. A method for comparing two hierarchical clusterings. *Journal of the American Statistical Association*, 78(383):553–569, 1983.

C. Fraley. Algorithms for model-based Gaussian hierarchical clustering. *SIAM Journal on Scientific Computing*, 20(1):270–281, 1999.

C. Fraley and A. E. Raftery. How many clusters? Which clustering method? Answers via model-based analysis. *The Computer Journal*, 41(8):578–588, 1998.

J.-L. Gauvain and C.-H. Lee. Maximum a posteriori estimation for multivariate Gaussian mixture observations of Markov chains. *IEEE Trans. Speech and Audio Processing*, 2(2):291–298, April 1994.

S. Geman and D. Geman. Stochastic relaxation, Gibbs distribution and the Bayesian restoration of images. *IEEE Trans. Pattern Anal. Machine Intell.*, 6(6):721–741, November 1984.

A. Gersho and R. M. Gray. *Vector Quantization and Signal Processing*. Kluwer Academic Publisher, Boston, MA, 1992.

A. B. Geva and D. H. Kerem. Brain state identification and forecasting of acute pathology using unsupervised fuzzy clustering of EEG temporal patterns. In H.-N. Teodorescu, A. Kandel, and L. C. Jain, editors, *Fuzzy and Neuro-Fuzzy Systems in Medicine*, chapter 3, pages 57–93. CRC Press, 1998.

J. Ghosh. Scalable clustering. In N. Ye, editor, *Handbook of Data Mining*, pages 341–364. Lawrence Erlbaum Assoc., 2003.

D. J. Hall and G. B. Ball. A clustering technique for summarizing multivariate data. *Behavioral Science*, 12:153–155, 1967.

J. A. Hartigan. *Clustering Algorithms*. John Wiley & Sons, 1975.

T. Heskes. Self-organizing maps, vector quantization, and mixture modeling. *IEEE Trans. Neural Networks*, 12(6):1299–1305, November 2001.

S. Hettish and S. D. Bay. The UCI KDD archive [http://kdd.ics.uci.edu]. Irvine, CA: University of California, Department of Information and Computer Science, 1999.

T. Hofmann and J. Buhmann. Pairwise data clustering by deterministic annealing. *IEEE Trans. Pattern Anal. Machine Intell.*, 19(1):1–14, 1997.

T. Hofmann and J. Buhmann. Competitive learning algorithms for robust vector quantization. *IEEE Trans. Signal Processing*, 46(6):1665–1675, June 1998.

P. Indyk. A sublinear-time approximation scheme for clustering in metric spaces. In *40th Annual IEEE Symp. Foundations of Computer Science*, pages 154–159, 1999.

T. S. Jaakkola and D. Haussler. Exploiting generative models in discriminative classifiers. In M. Kearns, S. Solla, and D. Cohn, editors, *Advances in Neural Information Processing Systems*, volume 11, pages 487–493. MIT Press, 1999.

A. K. Jain and R. C. Dubes. *Algorithms for Clustering Data*. Prentice Hall, New Jersey, 1988.

A. K. Jain, M. N. Murty, and P. J. Flynn. Data clustering: A review. *ACM Computing Surveys*, 31 (3):264–323, 1999.

B.-H. Juang, S. E. Levinson, and M. M. Sondhi. Maximum likelihood estimation for multivariate mixture observations of Markov chains. *IEEE Trans. Information Theory*, 32(2):307–309, 1986.

B.-H. Juang and L. R. Rabiner. A probabilistic distance measure for hidden Markov models. *AT&T Technical Journal*, 64(2):391–408, 1985.

K. Kalpakis, D. Gada, and V. Puttagunta. Distance measures for effective clustering of ARIMA time-series. In *Proc. IEEE Int. Conf. Data Mining*, pages 273–280, 2001.

A. Kalton, P. Langley, K. Wagstaff, and J. Yoo. Generalized clustering, supervised learning, and data assignment. In *Proc. 7th ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining*, pages 299–304, 2001.

S. Kamvar, D. Klein, and C. Manning. Interpreting and extending classical agglomerative clustering algorithms using a model-based approach. In *Proc. 19th Int. Conf. Machine Learning*, pages 283–290, 2002.

R. Kannan, S. Vempala, and A. Vetta. On clusterings —good, bad and spectral. In *41st Annual IEEE Symp. Foundations of Computer Science*, pages 367–377, 2000.

G. Karypis. *CLUTO - A Clustering Toolkit*. Dept. of Computer Science, University of Minnesota, May 2002.

G. Karypis, E.-H. Han, and V. Kumar. Chameleon: Hierarchical clustering using dynamic modeling. *Computer*, 32(8):68–75, 1999.

M. Kearns, Y. Mansour, and A. Y. Ng. An information-theoretic analysis of hard and soft assignment methods for clustering. In *Proc. 13th Conf. Uncertainty in Artificial Intelligence*, pages 282–293, 1997.

T. Kohonen. *Self-Organizing Map*. Springer-Verlag, New York, 1997.

M. H. Law and J. T. Kwok. Rival penalized competitive learning for model-based sequence clustering. In *Proc. IEEE Int. Conf. Pattern Recognition*, pages 195–198, 2000.

C. Li and G. Biswas. Applying the hidden Markov model methodology for unsupervised learning of temporal data. *International Journal of Knowledge-based Intelligent Engineering Systems*, 6 (3):152–160, July 2002.

J. Lin. Divergence measures based on the shannon entropy. *IEEE Trans. Information Theory*, 37: 145–151, 1991.

S. P. Lloyd. Least squares quantization in PCM. *IEEE Trans. Information Theory*, IT-28:129–137, March 1982.

J. MacQueen. Some methods for classification and analysis of multivariate observations. In *Proc. 5th Berkeley Symp. Math. Statistics and Probability*, pages 281–297, 1967.

K. V. Mardia. Statistics of directional data. *J. Royal Statistical Society. Series B (Methodological)*, 37(3):349–393, 1975.

T. M. Martinetz, S. G. Berkovich, and K. J. Schulten. "Neural-Gas" network for vector quantization and its application to time-series prediction. *IEEE Trans. Neural Networks*, 4(4):558–569, July 1993.

A. McCallum and K. Nigam. A comparison of event models for naive Bayes text classification. In *AAAI Workshop on Learning for Text Categorization*, pages 41–48, 1998.

G. McLachlan and K. Basford. *Mixture Models: Inference and Applications to Clustering*. Marcel Dekker, New York, 1988.

M. Meila and D. Heckerman. An experimental comparison of model-based clustering methods. *Machine Learning*, 42:9–29, 2001.

M. Meila and J. Shi. Learning segmentation by random walks. In T. K. Leen, T. G. Dietterich, and V. Tresp, editors, *Advances in Neural Information Processing Systems 13*, pages 873–879. MIT Press, 2001.

T. Mitchell. *Machine Learning*. McGraw Hill, 1997.

R. Neal and G. Hinton. A view of the EM algorithm that justifies incremental, sparse and other variants. In M. I. Jordan, editor, *Learning in Graphical Models*, pages 355–368. Kluwer Academic Publishers, 1998.

A. Y. Ng, M. I. Jordan, and Y. Weiss. On spectral clustering: analysis and an algorithm. In T. G. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems*, volume 14, pages 849–856. MIT Press, 2002.

T. Oates, L. Firoiu, and P. R. Cohen. Clustering time series with hidden Markov models and dynamic time warping. In *IJCAI Workshop on Neural, Symbolic, and Reinforcement Methods for Sequence Learning*, Stockholm, Sweden, 1999.

J. Qian, M. Dolled-Filhart, J. Lin, H. Yu, and M. Gerstein. Beyond synexpression relationships: Local clustering of time-shifted and inverted gene expression profiles identifies new, biologically relevant interactions. *Journal of Molecular Biology*, 314:1053–1066, 2001.

M. Ramoni, P. Sebastiani, and P. Cohen. Bayesian clustering by dynamics. *Machine Learning*, 47: 91–121, 2002.

W. M. Rand. Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical Association*, 66(366):846–850, 1971.

K. Rose. Deterministic annealing for clustering, compression, classification, regression, and related optimization problems. *Proceedings of IEEE*, 86(11):2210–2239, 1998.

K. Rose, E. Gurewitz, and G. C. Fox. Constrained clustering as an optimization method. *IEEE Trans. Pattern Anal. Machine Intell.*, 15:785–794, August 1993.

B. Scholkopf and A. Smola. *Learning with Kernels*. MIT Press, 2001.

G. Schwarz. Estimating the dimension of a model. *Annals of Statistics*, 6:461–464, 1978.

J. Seo and B. Shneiderman. Interactively exploring hierarchical clustering results. *Computer*, 35 (7):80–86, 2002.

J. Sinkkonen and S. Kaski. Clustering based on conditional distributions in an auxiliary space. *Neural Computation*, 14:217–239, 2001.

P. Smyth. Clustering sequences with hidden Markov models. In M. C. Mozer, M. I. Jordan, and T. Petsche, editors, *Advances in Neural Information Processing Systems*, volume 9, pages 648–654. MIT Press, 1997.

M. Steinbach, G. Karypis, and V. Kumar. A comparison of document clustering techniques. In *KDD Workshop on Text Mining*, Boston, MA, August 2000.

A. Strehl and J. Ghosh. Cluster ensembles —a knowledge reuse framework for combining partitions. *Journal of Machine Learning Research*, 3:583–617, 2002.

A. Strehl and J. Ghosh. Relationship-based clustering and visualization for high-dimensional data mining. *INFORMS Journal on Computing: Special Issue on Web Mining*, 15(2):208–230, 2003.

A. Strehl, J. Ghosh, and R. J. Mooney. Impact of similarity measures on web-page clustering. In *AAAI Workshop on AI for Web Search*, pages 58–64, July 2000.

M. Symons. Clustering criteria and multivariate normal mixtures. *Biometrics*, 37:35–43, 1981.

M. E. Tipping. Deriving cluster analytic distance functions from Gaussian mixture models. In *Proc. 9th IEEE Int. Conf. Artificial Neural Networks*, volume 2, pages 815–820, 1999.

K. Tsuda, M. Kawanabe, and K.-R. Müller. Clustering with the fisher score. In S. Becker, S. Thrun, and K. Obermayer, editors, *Advances in Neural Information Processing Systems*, volume 15. MIT Press, 2003.

A. K. H. Tung, R. T. Ng, L. V. D. Lakshmanan, and J. Han. Constraint-based clustering in large databases. In *Proc. 8th Int. Conf. Database Theory*, pages 405–419, 2001.

S. Vaithyanathan and B. Dom. Model-based hierarchical clustering. In *Proc. 16th Conf. Uncertainty in Artificial Intelligence*, pages 599–608, July 2000.

V. Vapnik. *Statistical Learning Theory*. John Wiley, New York, 1998.

J. H. Ward. Hierarchical groupings to optimize an objective function. *Journal of the American Statistical Association*, 58:234–244, 1963.

Y. F. Wong. Clustering data by melting. *Neural Computation*, 5(1):89–104, 1993.

K. Y. Yeung, C. Fraley, A. Murua, A. E. Raftery, and W. L. Ruzzo. Model-based clustering and data transformations for gene expression data. *Bioinformatics*, 17(10):977–987, July 2001.

Y. Zhao and G. Karypis. Criterion functions for document clustering: experiments and analysis. Technical Report #01-40, Department of Computer Science, University of Minnesota, November 2001.

S. Zhong and J. Ghosh. HMMs and coupled HMMs for multi-channel EEG classification. In *Proc. IEEE Int. Joint Conf. Neural Networks*, pages 1154–1159, May 2002.

S. Zhong and J. Ghosh. A comparative study of generative models for document clustering. In *SIAM Int. Conf. Data Mining Workshop on Clustering High Dimensional Data and Its Applications*, San Francisco, CA, May 2003a.

S. Zhong and J. Ghosh. Scalable, balanced model-based clustering. In *Proc. 3rd SIAM Int. Conf. Data Mining*, pages 71–82, San Francisco, CA, May 2003b.