Model-based Clustering with Soft Balancing

Shi Zhong
Dept. of Computer Science and Engineering
Florida Atlantic University
777 Glades Road, Boca Raton, FL 33431
zhong@cse.fau.edu

Joydeep Ghosh
Dept. of Electrical and Computer Engineering
The University of Texas at Austin
1 University Station, Austin, TX 78712-1084
ghosh@ece.utexas.edu

Abstract

Balanced clustering algorithms can be useful in a variety of applications and have recently attracted increasing research interest. Most recent work, however, addressed only hard balancing by constraining each cluster to have equal or certain minimum number of data objects. This paper provides a soft balancing strategy built upon a soft mixture-of-models clustering framework. This strategy constrains the sum of posterior probabilities for each cluster to be equal and thus balances the expected number of data objects in each cluster. We first derive soft model-based clustering from an information-theoretic viewpoint and then show that the proposed balanced clustering can be parameterized by a temperature parameter that controls the softness of clustering as well as that of balancing. As the temperature decreases, the actual resulting partitioning becomes more and more balanced. In the limit, when temperature becomes zero, the balancing becomes hard and the actual partitioning becomes perfectly balanced. The effectiveness of the proposed soft balanced clustering algorithm is demonstrated on both synthetic and real text data.

1 Introduction

Clustering algorithms have been widely studied across multiple disciplines for several decades [13, 15, 16]. A central goal of clustering is to group similar data objects together and to extract a compact representation for each group. Current clustering methods can be divided into discriminative (similarity-based) approaches [31, 14, 19] and generative (model-based) approaches [23, 4, 7]. Similarity-based approaches focus on partitioning data objects according to a data-pairwise (dis)similarity measure, whereas model-based approaches aim at estimating a set of generative models for each cluster, usually through a maximum likelihood approach. This paper is based on model-based clustering.

In many data mining applications, it is often desirable to have (approximately) balanced clusters. For example, pre-clustering is sometimes used to build an indexing structure to facilitate search in very large databases. When a query comes in, the search engine can first check the cluster representatives and then only search the individual records in those closely matched clusters. If the clusters are very skewed in size, the worst case search time can be close to the time needed to search over all data records. Therefore, balanced clusters can significantly reduce search time. Similarly, in a clustering of a large corpus of documents to generate topic hierarchies, balancing can greatly improve navigation by avoiding the generation of highly skewed hierarchies, with uneven depth in different parts of the "tree" hierarchy or having widely varying number of documents at the leaf nodes.

In addition to application requirements, balanced clustering is sometimes also helpful because it tends to decrease sensitivity to initialization and to avoid outlier clusters (highly under-utilized representatives) from forming, and thus has a beneficial regularizing effect. This is especially useful for k-means type algorithms, including the soft EM variant [5], which are increasingly prone to yielding unbalanced solutions as the input dimensionality increases. This problem is exacerbated when a large (tens or more) number of clusters are needed, and it is well known that both hard and soft k-means invariably result in some near-empty clusters in such scenarios [6, 3].

Many clustering algorithms favor balanced clusters even though there are no explicit balancing constraints in their objective functions. For example, in divisive hierarchical clustering, one can pick the largest cluster to split at each iteration, resulting in a set of relatively balanced clusters. Frequency sensitive competitive learning [1, 11] penalizes the distance to large clusters thus eliminates empty clusters. Spectral graph partitioning algorithms [9, 17, 24] use a modified minimum cut objective that favors balanced partitioning since exact minimum cut solution often leads to completely useless results (e.g., one point in one cluster and

all other points in the other in a bipartitioning). These algorithms, however, do not guarantee the level of balancing and have no principled way of adjusting it in the final clustering results.

The problem of clustering large scale data under constraints such as balancing has recently received attention in the data mining literature [6, 30, 27, 2, 33]. Since balancing is a global property, it is difficult to obtain near-linear time techniques to achieve this goal while retaining high cluster quality [12]. Banerjee and Ghosh [3] proposed a three-step framework for balanced clustering: sampling, balanced clustering of the sampled data, and populating the clusters with the remaining data points in a balanced fashion. This algorithm has relatively low complexity of $O(N\log(N))$ but relies on the assumption that the data itself is very balanced (for the sampling step to work). Here N is the number of data objects. Bradley et al. [6] developed a constrained version of the k-means algorithm. They constrained each cluster to be assigned at least a minimum number of data points at each iteration. The data assignment subproblem was formulated as a minimum cost flow problem, which has a high ($O(N^3)$) complexity. In a previous work [33], we presented a general framework for adapting any hard model-based clustering to provide balanced solutions. An efficient heuristic was developed to solve the completely balanced data assignment subproblem in $O(K^2N + KN\log N)$ time, where K is the number of clusters.

These existing balanced clustering algorithms, however, mostly concentrated on hard balancing that constraints each cluster to have equal or certain minimum number of data objects. In some situations, strict balance is not required or desired. In this case, hard balancing does not offer the needed flexibility given the evidence that hard balancing can group two distant data points together [6].

The contribution of this paper is a soft balancing strategy built on a general soft model-based clustering framework. Instead of constraining the actual number of data objects in each cluster to be equal, we constrain the expected number of data objects in each cluster to be equal. This is realized by setting the sum of posterior probabilities to be equal to N/K for each cluster. The resulting algorithm is parameterized by a temperature parameter, which controls the softness of clustering as well as that of balancing. Therefore, it provides a knob for users to adjust the level of balancing for different applications. Also it turns out that this soft balancing strategy is computationally more efficient compared to hard balancing and has a time complexity of $O(KN)$. Finally, the proposed soft balancing is a general method that can be applied to any application for which good generative models exist.

The organization of this paper is as follows. Section 2 gives an information-theoretic derivation of soft model-based clustering. Section 3 details the soft balancing strategy built upon the general model-based clustering framework. Section 4 demonstrates the effectiveness of the proposed soft balanced clustering algorithms through experimental results on both synthetic and real text data. Finally, Section 5 summarizes this paper with some concluding remarks.

2 Soft Model-based Clustering

In this section, we present a principled, information-theoretic derivation of model-based clustering. The derivation process is similar to that of deterministic annealing [26] and provides a useful generalization of the standard mixture-of-models clustering [23, 4, 7]. The resulting algorithm is parameterized by a temperature parameter T, which governs the randomness of posterior data assignments. As we will see, the standard mixture-of-models clustering corresponds to the special case $T = 1$ whereas k-means clustering corresponds to the special case $T = 0$.

In model-based clustering, one estimates K models from N data objects, with each model representing a cluster. Let us define the loss function of assigning a data object x to a cluster y to be $-\log P(x|\lambda_y)$, which is the negative log-likelihood of x from model λ_y [20]. Let the joint probability between x and y be $P(x, y)$. We aim to minimize the expected loss

$$\begin{aligned} E &= -\sum_{x,y} P(x,y)\log p(x|\lambda_y) \\ &= -\sum_{x} P(x)\sum_{y} P(y|x)\log p(x|\lambda_y)\ , \end{aligned}$$

or equivalently, to maximize the expected log-likelihood

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

Note that in practice it is unavoidable to use a sample average to calculate (1), i.e., to set the prior $P(x)$ to be constant $1/N$. As N goes to infinity, the sample average approaches the expected log-likelihood asymptotically.

Directly maximizing (1) over $P(y|x)$ and λ_y leads to a generic model-based k-means algorithm [33], which iterates between the following two steps:

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

and

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

The posterior probability $P(y|x)$ in (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-variance spherical Gaussian models are used, this model-based k-means algorithm reduces to the standard k-means algorithm [22, 21]. 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.

To introduce some randomness/softness to the data assignment step, we add entropy terms to (1) to get an entropy-constrained objective function. Let X be the set of all data objects and Y the set of all cluster indices. The new objective is

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

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 tradeoff 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 the added entropy terms do not change the model re-estimation formula (3) since the model parameters that maximize L also maximize L_1. To solve for $P(y|x)$ under constraint $\sum_y P(y|x) = 1$, we first construct the Lagrangian $\mathcal{L} = L_1 + \sum_x \xi_x (\sum_y P(y|x) - 1)$ and then let the partial derivative $\frac{\partial \mathcal{L}}{\partial P(y|x)} = 0$. The resulting $P(y|x)$ is given by

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

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 [26]. A standard deterministic annealing algorithm for model-based clustering is shown in Fig. 1. Note that at each temperature, the EM algorithm [8] is actually used to maximize (4), with cluster labels Y being the hidden variable and (5) and (3) corresponding to E-step and M-step, respectively.

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

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

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 Λ 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 Λ and set T to be high (a large number);
2. Optimization: optimize (4) by iterating between (5) and (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 1. Deterministic annealing algorithm for model-based clustering.

which is exactly the (incomplete data log-likelihood) objective function that the standard mixture-of-models clustering maximizes. It is a common practice to refer to the mixture-of-models clustering that maximizes (6) as EM clustering. As T goes to 0, (5) reduces to (2) and the algorithm reduces to model-based k-means, independent of the actual $P(y)$'s (unless they are 1 and 0's). For any $T > 0$, iterating between (5) and (3) gives a soft model-based clustering algorithm that maximizes (4) for a given T.

This analysis makes it clear that model-based k-means and EM clustering can be viewed as two special stages of the deterministic annealing process, with $T = 0$ and $T = 1$, respectively, and they optimize two different objective functions (L vs. $L - I(X;Y)$). 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 model-based k-means algorithm ($T = 0$). So it makes sense to use the EM clustering results to initialize the model-based k-means, which has been heuristically used in practice (e.g., using mixture-of-Gaussians to initialize standard k-means). It can be viewed as a one-step deterministic annealing algorithm (temperature decreases one time from $T = 1$ to $T = 0$). Of course, a better approach is always to start at a high $T \gg 1$ and gradually reduce T toward 0.

The computational complexity for all the algorithms described above is linear in the number of data samples N, provided that we use a constant (maximum) number of iter-

ations and a model training algorithm that has linear complexity.

3 Model-based clustering with soft balancing

In previous work [33], we have built a generic balanced hard clustering algorithm, based upon model-based k-means. In this section we shall show how we can build soft balancing based on the soft model-based clustering framework presented in the previous section.

Instead of enforcing the exact number of data objects grouped into each cluster to be equal, we constrain the sum of posterior probabilities for each cluster to be equal to N/K, which equalizes the expected number of objects assigned to each cluster. This is a soft balancing constraint in that the actual partitioning can be unbalanced. Our experimental results show that the softness of balancing can be characterized by T, the same parameter that parameterizes the softness of clustering.

After taking into account the soft balancing constraints

$$\sum_x P(y|x) = \frac{N}{K},\ \forall y\ , \tag{7}$$

we construct the Lagrangian w.r.t. $P(y|x)$ as

$$\begin{aligned}\mathcal{L} &= L_1 + \sum_x \xi_x (\sum_y P(y|x) - 1) \\ &\quad + \sum_y \eta_y \left(\sum_x P(y|x) - M \right),\end{aligned}$$

where ξ_x and η_y are Lagrange multipliers. Taking the derivative $\frac{\partial \mathcal{L}}{\partial P(y|x)} = 0$, and after some algebra, we get

$$P(y|x) = \frac{P(y)\left[e^{\eta_y} p(x|\lambda_y)\right]^{\frac{1}{T}}}{\sum_{y'} P(y')\left[e^{\eta_{y'}} p(x|\lambda_{y'})\right]^{\frac{1}{T}}}\ . \tag{8}$$

For balance clustering, it makes sense to set $P(y)$ to be $1/K$, which eliminates $P(y)$ from (8). For simplicity, let $\beta_y = e^{\eta_y}$. Plugging (8) into (7), we get

$$\sum_x \frac{\left[\beta_y p(x|\lambda_y)\right]^{\frac{1}{T}}}{\sum_{y'} \left[\beta_{y'} p(x|\lambda_{y'})\right]^{\frac{1}{T}}} = \frac{N}{K}\ . \tag{9}$$

To solve for β_y's in (9), we take an iterative optimization approach since a closed form solution is not available. The iterative formula for β_y can be derived from (9) as

$$\beta_y^{(t+1)} = \left(\frac{N/K}{\sum_x \frac{p(x|\lambda_y)^{\frac{1}{T}}}{\sum_{y'} \left(\beta_{y'}^{(t)} p(x|\lambda_{y'})\right)^{\frac{1}{T}}}} \right)^T , \tag{10}$$

where t is the iteration number. We use $\beta_y^{(0)} = 1, \forall y$. To avoid possible underflow problem with very small likelihood $p(x|\lambda_y)$, we operate on log-likelihood $\log p(x|\lambda_y)$ and $\log \beta_y$ and use the following implementation:

$$\begin{aligned}\log \beta_y^{(t+1)} &= T \cdot \log \left(\frac{N}{K}\right) - \\ &\quad T \cdot \log \left(\sum_x \frac{e^{\frac{1}{T}\log p(x|\lambda_y)}}{\sum_{y'} e^{\frac{1}{T}\left(\log \beta_{y'}^{(t)} + \log p(x|\lambda_{y'})\right)}} \right).\end{aligned} \tag{11}$$

Now we have a soft balancing strategy parameterized by the temperature parameter T. Experimental results (see Fig. 2) show that the iterative estimation of $\log \beta_y$ converges fast for a high T but can be very slow for a low T. To avoid long estimation time for a low temperature (e.g., $T = 0.01$), we take an annealing approach for computing $\log \beta_y$. That is, starting from a high temperature (e.g., $T = 0.1$) and quickly lower the temperature toward $T = 0.01$. At every temperature we run a small number of iterations and initialize $\log \beta_y$'s using the values computed from previous temperature. As shown in Fig. 2(d), this annealing method can converge fast even for low temperature settings.

Compared to hard balancing, soft balancing for model-based clustering can be solved exactly and efficiently using the iterative strategy described above. If we use a fixed number of maximum iterations, the time complexity for computing $\log \beta$'s will be $O(KN)$. In hard balancing [6, 33], a linear programming problem has to be solved and the time complexity is $O(K^3N^3)$ for an exact solution and $O(K^2N + KN \log N)$ even for an approximate solution. This is not surprising since for hard clustering, the posterior probabilities $P(y|x)$ are either 0's or 1's , creating a much harder integer programming problem.[1] For this reason, we recommend using a temperature T away from 0 (e.g., let $T > \delta$, where $\delta > 0$ is a small positive number) for our soft balancing strategy.

4 Experimental results and discussions

In this section, we first introduce several criteria used to evaluate balanced clustering results, followed by results and discussions on both synthetic datasets and real text document datasets.

4.1 Clustering evaluation

To evaluate the performance of our balanced clustering algorithms, we use three criteria—balance, objective function value, and mutual information between cluster labels and class labels (if they exist).

[1] Fortunately, the resulting integer programming problem can be reduced to a linear programming problem [6]. Even so, the complexity is still high.

We measure the balance of a clustering by normalized entropy (of the distribution of cluster sizes) that is defined as

$$N_{entro} = -\frac{1}{\log K}\sum_{j=1}^{K}\frac{N_j}{N}\log\left(\frac{N_j}{N}\right), \qquad (12)$$

where N_j is the number of data samples in cluster j. A normalized entropy of 1 means perfectly balanced clustering and 0 extremely unbalanced clustering.

The expected log-likelihood objective (1) is used as an internal measure of clustering quality. For the text datasets, where class labels are available, we calculate a normalized mutual information (NMI) criterion as an external measure of how well the clustering results conform to existing class labels. There are several choices for normalization; we shall follow the definition given in [28]:

$$NMI = \frac{\sum_{h,l} n_{h,l}\log\left(\frac{N\cdot n_{h,l}}{n_h n_l}\right)}{\sqrt{\left(\sum_h n_h \log\frac{n_h}{N}\right)\left(\sum_l n_l \log\frac{n_l}{N}\right)}}, \qquad (13)$$

where n_h is the number of data samples in class h, n_l the number of samples in cluster l and $n_{h,l}$ the number of samples 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. This is a better measure than purity or entropy which are both biased towards high K solutions [29, 28].

4.2 Results on synthetic datasets

We first tested the soft balanced clustering algorithms on a synthetic dataset—the t4 dataset (Fig. 3(a)) included in the CLUTO toolkit [18]. There are no ground truth labels for this dataset but there are six natural clusters plus a lot of noise according to human judgment. The best algorithm that can identify all the six natural clusters uses a hybrid partitional-hierarchical approach [19, 18]. It partitions the data into a large number (e.g., 30) of clusters and then merges them back to a proper granularity level.

We intend to use our balanced clustering algorithm to get a partition of 30 clusters. These fine granularity clusters can be merged using a hierarchical clustering algorithm to form a cluster hierarchy, for further interactive analysis. But here we are only concerned with the partitional step.

Spherical Gaussian distributions are used to model clusters. For a spherical Gaussian model, the covariance matrix is a constant times identity matrix and the constant is the variance of the Gaussian model which is constrained to be the same for every dimension. The pdf of a spherical Gaussian model can be written as

$$p(x|\lambda) = \frac{1}{(\sqrt{2\pi}\sigma)^d}\exp\left(-\frac{\|x-\mu\|^2}{2\sigma^2}\right), \qquad (14)$$

where $\lambda = \{\mu, \sigma\}$, μ is the mean vector, σ the standard deviation, and d the number of dimensions.

The balanced clustering results are shown in Fig. 3. As can be seen from the histogram distribution of cluster sizes in Fig. 3(d), model-based clustering with soft balancing can generate more balanced results (higher normalized entropy) than model-based clustering without balancing. Furthermore, low temperature (Fig. 3(d)) leads to more balanced actual partitioning, as well as better clustering quality in terms of the average log-likelihood measure. Fig. 3(c) indicates that our algorithm generates unbalanced solutions when the temperature is high.

4.3 Results on text datasets

We used two datasets from the CLUTO toolkit[2] [18]. A summary of the datasets is shown in Table 1. The traditional vector space representation is used for text documents, i.e., each document is represented as a high dimensional vector of "word"[3] counts in the document. The dimensionality equals the number of words in the vocabulary used.

Table 1. Summary of text datasets. (For each dataset, n_d is the total number of documents, n_w the total number of words, and K the number of classes.)

Data	Source	n_d	n_w	K
classic	CACM/CRANFIELD/ CISI/MEDLINE	7094	41681	4
tr11	TREC	414	6429	9

All the datasets have already been preprocessed [32]. We further removed those words that appear in two or fewer documents. The *classic* dataset was obtained by combining the CACM, CISI, CRANFIELD, and MEDLINE abstracts that were used in the past to evaluate various information retrieval systems[4]. CACM consists of 3,203 abstracts from computer systems papers, CISI consists of 1,460 abstracts from information retrieval papers, MEDLINE consists of 1,033 abstracts from medical journals, and CRANFIELD consists of 1,398 abstracts from aeronautical systems papers. The datasets *tr11* was derived from TREC collections.

4.3.1 Experimental setting

We use multinomial models in the model-based clustering of text documents. A multinomial model for cluster y represents a document x by a multinomial distribution of the

[2] http://www.cs.umn.edu/∼karypis/CLUTO/files/datasets.tar.gz.

[3] Used in a broad sense since it may represent individual words, stemmed words, tokenized words, short phrases, etc.

[4] Available from ftp://ftp.cs.cornell.edu/pub/smart.

words in document x

$$P(x|\lambda_y) = \prod_l P_y(w_l)^{x^{(l)}} \ , \qquad (15)$$

where $x^{(l)}$ is the number of times word w_l occurs in document x. To make the likelihood comparable for documents of different lengths, we normalize the log-likelihood according to a fixed length $\bar{L}_d$ (which is chosen to be the average length of all documents)

$$\log P(x|\lambda_y) = \frac{\bar{L}_d}{\sum_l x^{(l)}} \sum_l x^{(l)} \log P_y(w_l)$$

The parameters $P_y(w_l)$'s can be estimated by counting the number of documents in each cluster and the number of times w_l occurs in all documents in cluster y [25]. With Laplacian smoothing, the parameter estimation of multinomial models amounts to

$$P_y(w_l) = \frac{1 + \sum_x P(y|x)x^{(l)}}{\sum_l \left(1 + \sum_x P(y|x)x^{(l)}\right)} \ . \qquad (16)$$

4.3.2 Results and discussions

Fig. 2 shows how the number of iterations needed to estimate the $\log \beta_y$'s in (11) changes with different temperatures. Clearly, high temperature leads to faster estimation whereas the iterative process can take a very long time for low temperatures (e.g., Fig. 2(a)). As mentioned in Section 3, we can employ an annealing approach just for computing $\log \beta_y$'s. Fig. 2(d) shows that by running a small number of iterations at a sequence of decreasing temperatures, we were able to compute an approximate (very close) solution for $\log \beta_y$'s in 100 iterations, which is much smaller than the number of iterations needed in Fig. 2(a).

The soft balanced clustering results for *tr11* and *classic* datasets are shown in Fig. 4, with results for *tr11* on the left column and results for *classic* on the right. The first row shows balance measures, the second row average log-likelihood measures, and the last row normalized mutual information measures across seven different temperatures. The x-axis is on a log(T) scale, which provides a better visualization than using the T scale. The vertical bar at each temperature shows $\pm$ 1 $standard\ deviation$ over 10 runs of each experiment (with random initialization for each run). The balance of clusterings has a general trend of going down with increasing temperature. But the clustering quality, in terms of both average log-likelihood and normalized mutual information, seems to peak somewhere in the middle. This is intuitive in that toward very low temperature setting, a highly balanced clustering starts grouping distant data objects together, whereas toward the other end, highly fuzzy posterior assignments fail to discriminate between different clusters and thus fail to generate a good partitioning.

5 Concluding remarks

Several recent balanced clustering algorithms focused on hard balancing. Acknowledging that balance is not the central goal of a clustering algorithm, we have presented a soft balancing strategy built upon a general soft model-based clustering framework, parameterized by a temperature parameter. It offers a continuous spectrum of solutions depending on where the temperature knob is set. At one end of the spectrum (zero temperature) is the hard balancing where all posterior probabilities are either 0 or 1. Toward the other end (high temperature), we can get very unbalanced clusterings even though the expected number of data objects in each cluster is constrained to be equal. Experimental results show that clusterings of the highest quality often occur in the middle of the spectrum.

The proposed soft balancing also has a computational advantage compared to hard balanced clustering algorithms. With a complexity linear in the number of data objects, it is better positioned for large scale data mining applications.

Since the proposed algorithm is built on a general model-based clustering framework, it can be readily applicable to other applications. For example, balanced time series clustering can be constructed with appropriate models (e.g., Markov chains or hidden Markov models), and can be useful in financial world for building balanced portfolios.

Finally, there are several model selection methods [10] for estimating number of clusters in a general model-based clustering framework. The interaction between model selection and balancing can be investigated.

References

[1] S. C. Ahalt, A. K. Krishnamurthy, P. Chen, and D. E. Melton. Competitive learning algorithms for vector quantization. *Neural Networks*, 3(3):277–290, 1990.

[2] 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 2002.

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

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

[5] 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.

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

[7] 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.

[8] 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.

[9] 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.

[10] 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.

[11] A. S. Galanopoulos, R. L. Moses, and S. C. Ahalt. Diffusion approximation of frequency sensitive competitive learning. *IEEE Trans. Neural Networks*, 8(5):1026–1030, September 1997.

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

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

[14] 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.

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

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

[17] 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.

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

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

[20] 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.

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

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

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

[24] 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.

[25] K. Nigam. *Using Unlabeled Data to Improve Text Classification*. PhD thesis, School of Computer Science, Carnegie Mellon University, May 2001.

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

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

[28] 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.

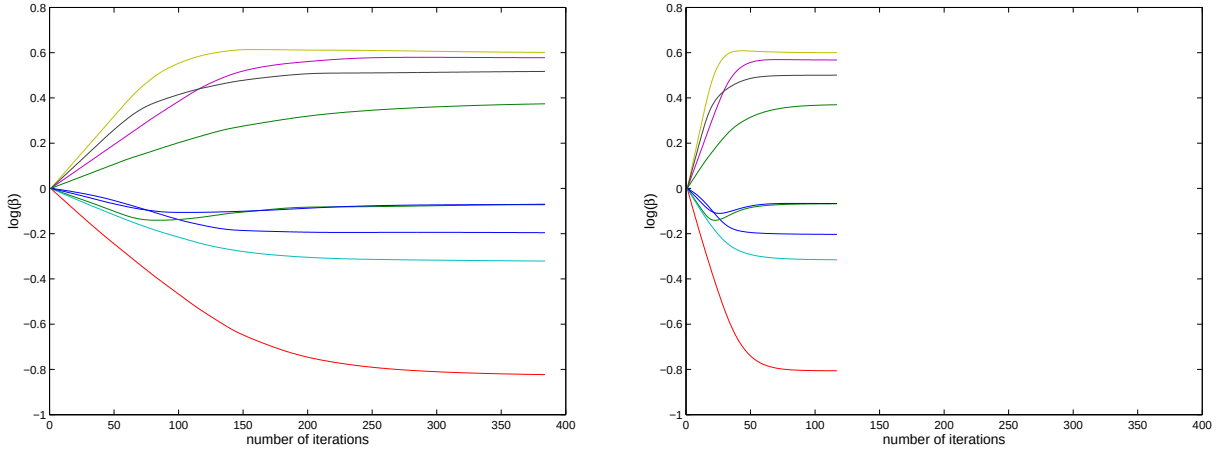
[29] A. Strehl, J. Ghosh, and R. J. Mooney. Impact of similarity measures on web-page clustering. In *AAAI Workshop on AI for Web Search*, pages 58–64, July 2000.

[30] 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.

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

[32] 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.

[33] 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 2003.



(a) $T = 0.01$ (b) $T = 0.04$

(c) $T = 0.1$ (d) $T = 0.1 \rightarrow 0.04 \rightarrow 0.01$

Figure 2. Iterative estimation of $\log \beta$'s for nine multinomial models trained on *tr11* dataset for different temperatures (each curve corresponds to one of the nine clusters/models): (a) the estimation takes more than 350 iterations to converge for $T = 0.01$; the estimation converges faster for (b) $T = 0.04$ and ever faster for (c) $T = 0.1$; (d) an annealing strategy is used to accelerate convergence for $T = 0.01$—we iterate 30 times for $T = 0.1$, then 30 times for $T = 0.04$, and finally 40 times for $T = 0.1$.

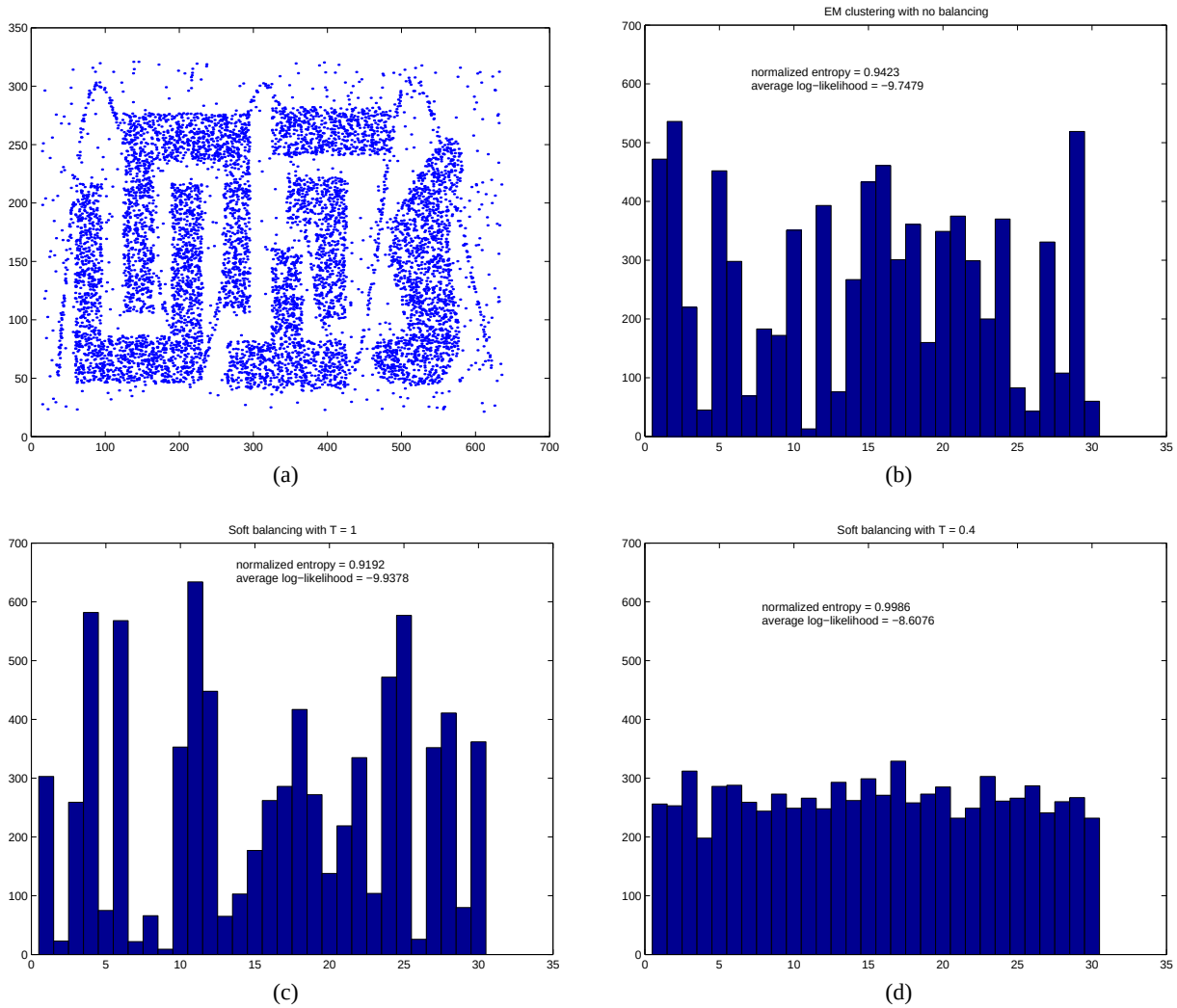


Figure 3. Balanced clustering results on *t4* dataset: (a) *t4* dataset; histogram distribution of cluster sizes when clustering *t4* into 30 groups using (b) EM clustering with spherical Gaussian models and no balancing; (c) mixture-of-spherical Gaussians clustering with soft balancing (T=1); (d) mixture-of-spherical Gaussians clustering with soft balancing (T=0.4).

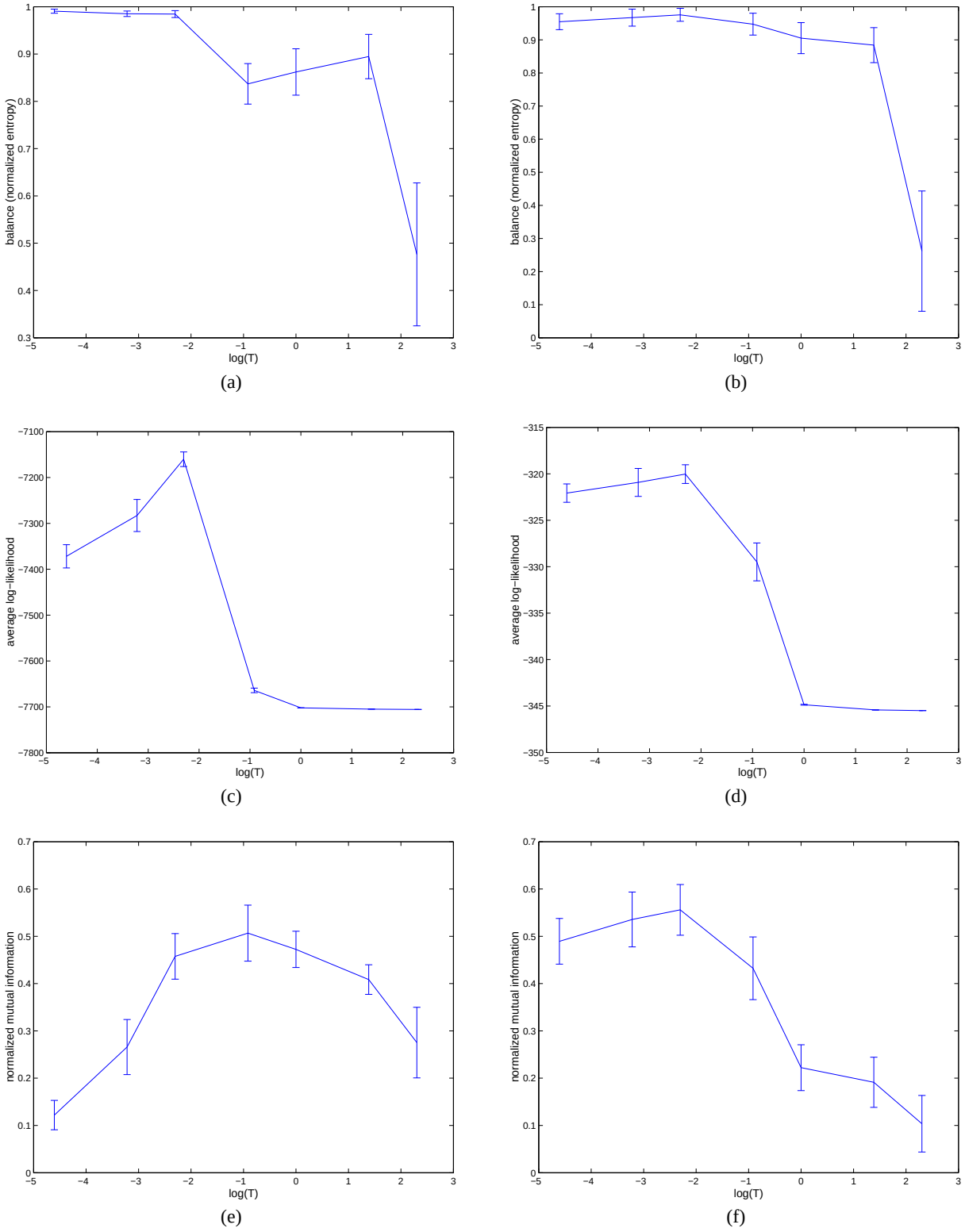


Figure 4. Soft balanced clustering results for *tr11* and *classic* datasets: balance (normalized entropy) results for (a) *tr11* and (b) *classic*; average log-likelihood results for (c) *tr11* and (d) *classic*; normalized mutual information results for (e) *tr11* and (f) *classic*.