Vol.2013-DBS-157 No.23
Vol.2013-IFAT-111 No.23
2013/7/23

Experiment of Document Clustering by Triple-pass Leader-follower Algorithm without Any Information on Threshold of Similarity

Kazuaki Kishida[1,a)]

Abstract: The number of clusters has to be defined a priori in most clustering algorithms, but it is usually unknown in situations to which document clustering is applied. Therefore, it would be convenient if a clustering algorithm could be executed without any information on the number of clusters. This article attempts to develop such an algorithm by extending the leader-follower clustering algorithm, which is appropriate for the clustering of large-scale datasets. Specifically, a threshold value required for executing the leader-follower clustering algorithm is automatically estimated from some pairs of documents by scanning the document file one time before executing the standard leader-follower algorithm. In particular, the triple-pass algorithm in which cluster vectors are generated in the second scan and each document is allocated to the most similar cluster in the third scan is proposed. The experimental result suggests that the triple-pass leader-follower clustering algorithm is sufficiently effective and comparable with the hierarchical Dirichlet process (HDP) mixture model and with the spherical k-means algorithm with automatically estimating the number of clusters based on the cover-coefficient. The algorithm requires less computational iteration than the other two methods, and is thus cost effective.

Keywords: Document clustering, K-means algorithm, Leader-follower clustering algorithm, Hierarchical Dirichlet process mixture model

1. Introduction

A core element of text mining is document clustering, which has been widely used for organizing heterogeneous sets of documents such as news articles, web pages, e-mails and so on. In many cases, the number of clusters is unknown because document clustering is usually expected to work in an unsupervised manner without any predefined classification scheme, which is a major difference from text categorization. Unfortunately, standard techniques for dividing documents into several clusters in an unsupervised manner (e.g., k-means algorithm) usually require the number of clusters as a parameter to be determined a priori, and so may not be applied 'correctly' to a document collection for which the number of 'true' clusters is unknown.

This article aims to develop a clustering algorithm without any information on the number of clusters inherent in the target document collection, and more specifically, attempts to extend the leader-follower clustering algorithm [6] for this purpose. This algorithm is sometimes referred to in the literature as a 'method requiring no information on the number of clusters' (e.g., [16]), but actually, a threshold of proximity (or similarity) between a document and a cluster for determining whether the document is allocated to the cluster or not has to be specified a priori instead of the number of clusters. Accordingly, in the algorithm proposed by this article, an appropriate threshold is estimated at the first scan of the target file. Thereafter, by applying the standard leader-follower algorithm based on the estimated threshold, a set of clusters is generated. In other words, by automatically estimating the threshold used in the leader-follower clustering algorithm, the number of clusters inherent in the target dataset can be posteriorly obtained as a result of the clustering operation.

The reason why the leader-follower technique was selected is that it is suitable for clustering of large-scale document collections in terms of computational efficiency [10]. Actually, the method proposed here can generate final clusters after scanning the file containing a document set only three times (i.e., triple-pass clustering). In contrast, many more scans would usually be needed for iterative computation in the k-means methods or the EM algorithm for estimating parameters in probabilistic mixture models [13] and probabilistic latent semantic indexing (PLSI) [8], which can be used for document clustering (of course, the dominant efficiency of the leader-follower algorithm is not changed if the file is completely loaded in the main memory).

This article reports on an experiment of empirically examining the validity or quality of clusters generated by the triple-pass leader-follower clustering algorithm using a portion of the Reuters corpus (RCV1) [11]. The result showed that the algorithm was comparable with the spherical k-means algorithm and the hierarchical Dirichlet process (HDP) mixture model [14] in terms of the effectiveness. This means that the proposed method is highly cost-effective because it can generate clusters of ade-

[1] School of Library and Information Science, Keio University, Minato-ku, Tokyo 108–8345, Japan
[a)] kz_kishida@z8.keio.jp

quate quality after only three iterations without any information on the number of clusters.

2. Automatically Estimating the Number of Clusters in Document Clustering

Several techniques for automatically estimating the number of clusters inherent in a document collection have already been developed (e.g, [15]). Some of them were used in the experiment of this article as baselines for measuring the effectiveness of the triple-pass leader-follower clustering algorithm introduced in the next section.

2.1 Cover-coefficient

The cover-coefficient-based concept clustering methodology (C^3M) by [2] and [3] works without a predetermined number of clusters by predicting it from a 'cover-coefficient', which measures the degree to which a given document is 'covered' by the other documents. The cover-coefficient is computed as $\delta_{ii'} = \sum_{j=1}^{M} \phi_{ij}\rho_{i'j}$ based on two quantities ϕ_{ij} and ρ_{ij} such that

$$\phi_{ij} = \frac{s_{ij}}{\sum_{j'=1}^{M} s_{ij'}}, \rho_{ij} = \frac{s_{ij}}{\sum_{i'=1}^{N} s_{i'j}} \quad (i = 1, \ldots, N; j = 1, \ldots, M),$$

where $s_{ij} = 1$ if the j-th index term appears in the i-th document and $s_{ij} = 0$ if not, and N indicates the number of documents included in the set and M means the total number of distinct terms contained in the N documents. When $i = i'$, δ_{ii} can be interpreted as a measure of 'uniqueness' of the i-th document. Actually, if the i-th document does not share any term with the other documents, then $\delta_{ii} = 1$, which means that the document is completely unique in the set. Conversely, when all terms in the i-th document appear in all the other documents, δ_{ii} becomes $1/N$, which is its minimum value. Therefore, the number of clusters (which is denoted by L) can be estimated as $L = \sum_{i=1}^{N} \delta_{ii}$ by using the 'uniqueness', and it is possible to execute various clustering algorithms such as k-means or PLSI based on the estimation.

2.2 HDP mixture model

Recently, PLSI [8] and latent Dirichlet allocation (LDA) [1] have often been used for discovering 'latent' topics included implicitly in document collections, and the 'topic model' empirically constructed by them can also provide a set of clusters to which documents belong. However, the number of topics (or clusters) has to be determined a priori before executing PLSI and LDA.

On the other hand, in the HDP mixture model [14] which can be considered as an extension of LDA, the number of topics is automatically estimated based on hyper-parameters in a Bayesian framework because this model is theoretically a mixture of an infinite number of components (corresponding to topics or clusters) and the actual number of components is empirically fixed in the process of parameter estimation for the given data. Therefore, the HDP mixture model would be a useful tool for document clustering when the number of clusters is unknown.

Similarly with LDA, it is assumed in HDP that a latent topic is assigned to each token in a document, and then a term appearing as the token is determined according to the latent topic. The assignment of topics and the determination of terms are supposed to be executed stochastically based on the Dirichlet distribution, which can be interpreted as a 'distribution of distributions'. After estimating the topics of all tokens included in the document collection under assumptions of the HDP mixture model, a set of document clusters can be obtained by considering the most frequently appearing topic in a document as a label of the cluster to which the document should belong.

3. Leader-follower Clustering Algorithm with Estimation of Threshold

As mentioned above, the leader-follower clustering algorithm [6] is suitable for processing large-scale datasets because clusters can be generated by scanning the target file (in the main memory or hard disk) only once or twice, and therefore, many researchers have adopted it for clustering large-scale document collections.

3.1 Basic Procedure of Leader-follower Clustering

A typical procedure of the double-pass leader-follower clustering [4] is to generate cluster vectors at the first scan of the file in online mode, and to allocate each document to a cluster at the second scan, as shown in Figure 1 [10].

1) Set the first document as the first cluster, and provide similarity threshold θ.
2) [Cluster vector generation stage (first scan)]
2-1) Read the next document. If no document remains, go to step 3).
2-2) Compute similarities between vectors of the document and of all current clusters. If the maximum exceeds θ, then the vector of the cluster with the maximum similarity is updated by using the document. Otherwise, the document is added to the set of clusters as a singleton (i.e., a new cluster).
2-3) Return to step 2-1).
3) [Document allocation stage (second scan)]
Read documents sequentially from the top of the file to the end, and allocate each document to the cluster with the maximum similarity.

Fig. 1 Double-pass Leader-follower Clustering Algorithm

In the case of document clustering, the cosine coefficient is usually used as a metric of the similarity between two vectors of a document and a cluster, and the weight of j-th term in document d_i is adopted as the j-th element of document vector $\mathbf{d}_i$. Whereas the weight is often computed based on a tf-idf weighting formula (e.g., see [10]), this article employs simple tf (term frequency) as the element in order to prevent the inference of similarity threshold from becoming too complicated. Actually, by using tf vectors, each cluster vector can be simply computed as a summation of vectors of documents belonging to the cluster such that

$$\mathbf{c}_k = \sum_{i:d_i \in C_k} \mathbf{d}_i, \tag{1}$$

where $\mathbf{c}_k$ denotes the vector of cluster C_k (note that C_k is formally defined as the k-th set of some documents), and the cosine similarity between document d_i and cluster C_k in step 2-2) of Figure 1 is formally computed as

$$s(i,k) = \cos(\mathbf{d}_i, \mathbf{c}_k) = \frac{\mathbf{d}_i^T \mathbf{c}_k}{\|\mathbf{d}_i\| \|\mathbf{c}_k\|} = \frac{1}{\|\mathbf{d}_i\|} \sum_{h:d_h \in C_k} \frac{\mathbf{d}_i^T \mathbf{d}_h}{\| \sum_{h:d_h \in C_k} \mathbf{d}_h \|}.$$

The update of cluster vectors in step 2-2) is also based on Equation (1).

In order to compensate for omitting the idf factor in document vectors, index terms with high document frequency were removed in the experiment of this article (see below). This can be considered to be an implementation of feature selection.

3.2 Inference of Similarity Threshold

Intuitively, there would appear to be no way of knowing the 'true' value of a threshold leading to the optimal number of clusters in an unsupervised manner. However, it is certain that the 'density' of the entire collection should be taken into account at least when determining the threshold. For example, in a sparse collection like that in Figure 2(a), a small threshold may correctly identify clusters whereas the documents would be erroneously merged into a big cluster by a small threshold in a less sparse collection of Figure 2(b).

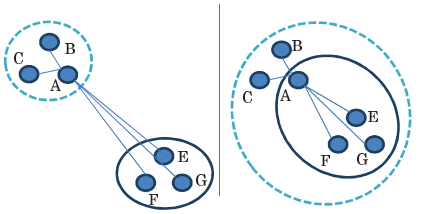

Fig. 2 Sparse and less sparse collections

More precisely, in the situation of Figure 2, it is assumed that the documents are processed in the order of E, F, G, A, B and C, and that document A is merged into a cluster { E, F, G } because the similarity between them exceeds the small threshold in Figure 2(b). For separating the set of documents into two groups { A, B, C } and { E, F, G } in Figure 2(b), it may be a good decision to use the average of five similarity scores between A and the other five documents as the threshold because the average may be larger than the similarities between A and each element of { E, F, G } due to the effect of B and C in computing the average. The average of similarity scores between document d_i and the other documents located in the 'vicinity' or 'neighbors' of d_i is denoted by $\tau_i(V_i)$ where V_i indicates the set of documents in the vicinity, which is formally defined such that

$$\tau_i(V_i) = \frac{1}{|V_i|} \sum_{h:d_h \in V_i} \cos(\mathbf{d}_i, \mathbf{d}_h) = \frac{1}{\|\mathbf{d}_i\|} \mathbf{d}_i^T \left(\frac{1}{|V_i|} \sum_{h:d_h \in V_i} \frac{\mathbf{d}_h}{\|\mathbf{d}_h\|} \right).$$

Note that if $V_i = C_k$, then $s(i,k) = \cos(\mathbf{d}_i, \mathbf{c}_k) \simeq \tau_i(V_i)$.

Unfortunately, the vicinity of each document as shown in Figure 2 is unknown in the situation assumed in this article, and there is no theoretically sound framework for estimating it because the vicinity is only an ambiguous and hypothetical concept. On the other hand, for practical convenience, the quantity

$$\omega_i(n) \equiv \frac{1}{n}[\cos(\mathbf{d}_i, \mathbf{d}_{i+1}) + \ldots + \cos(\mathbf{d}_i, \mathbf{d}_{i+n})],$$

could be considered as an estimate of $\tau_i(V_i)$ (where $n < N$) because it is possible to compute $\omega_i(n)$ by a single scan of the document file where it is assumed that d_i is located at the i-th position in the file. If the main memory is not large enough, then n can be decreased so as to store n vectors in it. Of course, $\{d_{i+1}, \ldots, d_{i+n}\}$ will include documents not belonging to the vicinity of d_i. To overcome this problem, it should be assumed that document d_h does not belong to the vicinity of d_i when $\cos(\mathbf{d}_i, \mathbf{d}_h)$ is very small, which leads to the removal of $\cos(\mathbf{d}_i, \mathbf{d}_h)$ from $\omega_i(n)$. Therefore, by defining $s'(i,h)$ such that

$$s'(i,h) = \begin{cases} \cos(\mathbf{d}_i, \mathbf{d}_h), & \text{if } \cos(\mathbf{d}_i, \mathbf{d}_h) > \epsilon \\ 0, & \text{otherwise} \end{cases},$$

based on a small number ϵ (e.g., $\epsilon = 0.01$), $\omega_i(n)$ is reformulated as $\omega'_i(n) \equiv m^{-1} \sum_{h=1}^{n} s'(i, i+h)$ where m indicates the number of documents for which $\cos(\mathbf{d}_i, \mathbf{d}_{i+h}) > \epsilon$.

Finally, in a situation with no special information on the vicinity of each document included in the target collection, it would be reasonable to simply average $\omega'_i(n)$ over documents for determining the threshold θ, namely,

$$\theta = \frac{1}{N-n} \sum_{i=1}^{N-n} \omega'_i(n). \quad (2)$$

Note that the average of Equation (2) must be computed by removing d_i where $\omega'_i(n) = 0$ in practice.

3.3 Triple-pass Leader-follower Clustering Algorithm

Consequently, the double-pass algorithm shown in Figure 1 can be extended to a 'triple-pass' algorithm as follows.

- First scan: Threshold θ is automatically determined by Equation (2).
- Second scan: Cluster vectors are generated by step 2) in Figure 1.
- Third scan: Every document is allocated to a cluster by step 3) in Figure 1.

The triple-pass leader-follower clustering is very efficient because a set of clusters will be generated after scanning the target file only three times. If cluster vector generation and document allocation are concurrently executed at the second scan, then the processing time becomes shorter at the cost of cluster quality.

4. Experiment on Performance of Triple-pass Clustering

This experiment empirically examined the validity of clustering results from the triple-pass leader-follower algorithm by comparing effectiveness measured by nMI [12] between the algorithm and other techniques with automatic estimation of the number of clusters.

4.1 Clustering Methods for Comparison

The following clustering techniques were used for comparing effectiveness in this experiment:

- The spherical k-means algorithm in which the number of clusters derived from the cover-coefficient is used.
- The HDP mixture model.

The k-means algorithm based on the cosine similarity is often called 'spherical k-means' because $\mathbf{d}_i/\|\mathbf{d}_i\|$ is a unit vector distributed on a hyper-sphere (note that cluster vector $\mathbf{c}_k/\|\mathbf{c}_k\|$ is also

a unit vector, and that the cosine measure is computed as an inner product of the two unit vectors). Although there are several versions of the spherical k-means algorithm (e.g., see [5], [9]), this experiment modified the Hartigan-Wong algorithm [7] by replacing the Euclidean distance with the inner product of two unit vectors (see Appendix A.1 for the details) because the Hartigan-Wong algorithm is known to be usually more effective than other k-means algorithms.

On the other hand, for the HDP mixture model, a technique of Gibbs sampling based on Chinese restaurant franchise (CRF) model [14] was employed (see Appendix A.2). In this experiment, after the Gibbs sampling was executed 3,000 times iteratively (note that this is a single chain), 10 samples extracted from the 2010th iteration to 3000th iteration with 10 intervals were used for generating sets of clusters, respectively, and the nMI score was averaged over the ten samples.

4.2 Dataset for Experiment and Evaluation Metric

The Reuter corpus RCV1 [11] created as a test collection for text categorization was used to compare the effectiveness of document clustering techniques. Since one or more topic codes are assigned to each record of the corpus, which can be considered as 'answers' of clustering, the validity of clusters generated by the techniques can be assessed based on the topic codes (note that the topic codes were used only for evaluation). Particularly, as a test dataset for this experiment, a set of 6,374 records to which just a single topic code is assigned was extracted from news articles published during August 1996 (i.e., $N = 6374$) because evaluation of clustering results including multi-topic documents becomes too complicated. In total, 68 different topic codes appear in the 6,374 records (i.e., the 'true' number of clusters is 68).

By standard text processing which consists of tokenization, removing stopwords and stemming by Porter's algorithm, document vectors for clustering were generated from the records. As described above, term frequency was simply used as the element of document vectors, and instead of incorporating the idf factor into the element, 'non-specific' terms appearing in more than 10% of all documents (i.e., over 647 documents) were removed from all document vectors. Also, terms appearing in only one document were not used as features for clustering. As a result, in total, 19,610 different terms were used in document vectors (i.e., $M = 19610$) and the average document length amounted to 99.99.

At the evaluation stage, nMI was used for measuring cluster validity. As the normalizing factor, $\max[E(C), E(A)]$ was adopted where $E(C)$ and $E(A)$ are entropy of cluster set C and of topic code set A, respectively [12].

4.3 Comparison of Clustering Results

The experimental results are shown in Tables 1 and 2. As indicated in Table 1, the number of clusters estimated by using the cover-coefficient was 247.3 (i.e., $L = 247$), which is substantially larger than the number of topic codes (i.e., = 68). In the HDP mixture model, the number of clusters was estimated to be 54, 74 and 84 for the parameters $\gamma = 0.1$, $\gamma = 0.5$ and $\gamma = 1.0$, respectively, where γ is a hyper-parameter governing the possibility that a new topic is selected (see Appendix A.2). Although the values estimated by the HDP mixture model were closer to the number of topic codes, the estimation appears to be highly dependent on the hyper-parameter (note that the other hyper-parameters also affect the final results). However, it may be possible that 3,000 iterations were insufficient for attaining sufficient convergence, and two values may become closer by iterating the sampling many more times. Because the HDP mixture is not the main target of this study, the problem of convergence when applying the HDP mixture to document clustering was left for future research.

In the case of triple-pass leader-follower clustering, the resultant number of clusters and the value of threshold θ varied largely with different sample size of n. It seems that a small sample (e.g., $n = 1$ or 2) does not provide good results, which is consistent with intuition derived from Equation (2). However, larger-size samples did not always improve the estimation. For example, 21

Table 1 No. of clusters estimated from the data

Methods	Parameters	No. of clusters	θ
Cover-coefficient		247.3	-
HDP	$\gamma = 0.1^{+}$	54.1*	-
	$\gamma = 0.5^{+}$	73.7*	-
	$\gamma = 1.0^{+}$	84.3*	-
leader-follower	$n = 1, \epsilon = 0.001$	209	.113
	$n = 2, \epsilon = 0.001$	152	.096
	$n = 3, \epsilon = 0.001$	126	.088
	$n = 5, \epsilon = 0.001$	91	.078
	$n = 10, \epsilon = 0.001$	68	.066
	$n = 20, \epsilon = 0.001$	37	.055
	$n = 30, \epsilon = 0.001$	27	.049
	$n = 50, \epsilon = 0.001$	21	.043
	$n = 1, \epsilon = 0.01$	279	.130
	$n = 2, \epsilon = 0.01$	205	.111
	$n = 3, \epsilon = 0.01$	174	.101
	$n = 5, \epsilon = 0.01$	130	.090
	$n = 10, \epsilon = 0.01$	93	.077
	$n = 20, \epsilon = 0.01$	66	.065
	$n = 30, \epsilon = 0.01$	47	.059
	$n = 50, \epsilon = 0.01$	36	.053

Note: + Other hyper-parameters are $\alpha = 0.1$ and $\beta = 0.01$.
* An average over 10 samples.

Table 2 nMI scores of clustering results

Methods	Parameters	nMI scores	No. of scans
k-means	$L = 247$	.4008	36
	$L = 74$	.4315	43
	$L = 68$	.4313	30
	$L = 66$	.4297	45
	$L = 54$	.4454	39
HDP	$\gamma = 0.1^{+}$	.4715*	3000
	$\gamma = 0.5^{+}$	.4725*	3000
	$\gamma = 1.0^{+}$	.4486*	3000
leader-follower	$n = 1, \epsilon = 0.001$	.4392	3
	$n = 2, \epsilon = 0.001$	.4423	3
	$n = 3, \epsilon = 0.001$	.4393	3
	$n = 5, \epsilon = 0.001$	.4553	3
	$n = 10, \epsilon = 0.001$	.4770	3
	$n = 20, \epsilon = 0.001$	.4643	3
	$n = 30, \epsilon = 0.001$	.4584	3
	$n = 50, \epsilon = 0.001$	.3910	3
	$n = 1, \epsilon = 0.01$	.4306	3
	$n = 2, \epsilon = 0.01$	.4353	3
	$n = 3, \epsilon = 0.01$	.4420	3
	$n = 5, \epsilon = 0.01$	.4366	3
	$n = 10, \epsilon = 0.01$	.4506	3
	$n = 20, \epsilon = 0.01$	.4767	3
	$n = 30, \epsilon = 0.01$	.4718	3
	$n = 50, \epsilon = 0.01$	.4563	3

Note: + Other hyper-parameters are $\alpha = 0.1$ and $\beta = 0.01$.
* An average over 10 samples.

clusters with $n = 50$ and $\epsilon = 0.001$ is clearly too few for the test dataset. In this experiment, a valid result was obtained in the run with $n = 10$ and $\epsilon = 0.001$, which generated 68 clusters.

Table 2 shows the nMI score of each run. The difference in scores between algorithms is not so large, which suggests that a specific algorithm does not provide particularly 'poor' clustering results in this experiment. Among them, the triple-pass leader-follower clustering algorithm with $n = 10$ and $\epsilon = 0.001$ showed the highest effectiveness measured by nMI (= 0.4770) in this experiment, which was followed by the HDP mixture with $\gamma = 0.5$ (the score of nMI was 0.4725 on average for 10 samples).

Unfortunately, combination of the cover-coefficient and the spherical k-means algorithm did not yield a better set of clusters in this experiment (nMI was 0.4008). To confirm the effectiveness of the spherical k-means algorithm, it was also executed with $L = 68$ (the number of topic codes), $L = 66$ (the number of clusters estimated by the triple-pass leader-follower clustering algorithm when $n = 20$ and $\epsilon = 0.01$), and $L = 54$ and 74 (the numbers of clusters estimated by the HDP mixture model). The results are also shown in Table 2, which indicates that the spherical k-means algorithm provided sufficiently comparable results, but did not outperform the leader-follower clustering algorithm. It should be noted that this result does not necessarily mean that the leader-follower clustering algorithm is superior because there are various k-means algorithms and the difference of nMI scores was slight.

Similarly, in order to determine empirically the superiority of the leader-follower clustering algorithm over the HDP mixture model, it is necessary to repeatedly execute more times the Gibbs sampling for the HDP with various sets of parameters, which is not the aim of this experiment. For the purpose of this article, it is sufficient to obtain evidence that the triple-pass leader-follower clustering algorithm can create clusters of 'comparable' quality with those generated by the HDP mixture model or the k-means algorithm requiring many more scans of the dataset.

Table 2 also indicates the number of times that the document file was scanned by each algorithm. In the case of the spherical k-means based on the Hartigan-Wong methods, scans from 30 to 45 were needed until the result converged, which is about ten times more than that in the triple-pass leader-follower clustering algorithm. Therefore, in terms of 'cost-effectiveness', it can be concluded that the triple-pass leader-follower clustering algorithm clearly outperforms the HDP mixture model or the spherical k-means algorithm with automatic estimation of the number of clusters.

5. Discussion

Although the experiment showed that the triple-pass leader-follower clustering algorithm generated sufficiently valid clusters of documents after only three scans of the dataset, it was also clarified that the validity is largely dependent on the parameter n, which is the sample size for estimating threshold θ , and ϵ, which is the threshold for identifying the vicinity of each document. Because the actual vicinity of each document exemplified in Figure 2 is unknown in an unsupervised situation, it might be unreasonable to estimate $\tau_i(V_i)$ from the data. In addition, it is not theoretically guaranteed that this value provides a 'local' threshold for merging documents around the vicinity of d_i, and also that its average can be used as threshold θ which is a 'global' parameter for applying the entire document space constructed by the set.

However, as long as the values of these parameters are appropriately selected, this 'heuristic' rule for estimating the threshold worked well in this experiment in spite of its theoretical weakness. As described above, since the leader-follower clustering algorithm is very efficient, it would be worth exploring methods that can automatically estimate an optimal value of the threshold, which is one of the findings of this experiment.

It would be natural to consider that the combination of $n = 10$ and $\epsilon = 0.001$ will not always yield good results for other document collections. Nevertheless, it is clearly better to depend on the heuristic rule proposed by this article than on arbitrarily direct allocation of a value to the threshold in actual situations because the heuristic rule tries to probe the actual content of the target dataset based on samples from it before applying the clustering algorithm. If much experience of using the rule for various document collections is accumulated, then a 'standard' sample size for a given value of ϵ may be identified empirically.

6. Concluding Remarks

In this article, the triple-pass leader-follower clustering algorithm was proposed as a method for grouping documents without prior knowledge of the number of clusters, and the experiment showed that the algorithm can yield valid clusters through comparisons of effectiveness with other techniques, the HDP mixture model and the spherical k-means algorithm with automatic estimation of the number of clusters based on the cover-coefficient. However, the experimental result was obtained from only a single document collection, which was specially created by extracting a portion of the RCV1 corpus. Therefore, future research is required to test the proposed algorithm on various document collections, which would be useful to obtain knowledge on an optimal sample size that has to be specified a priori as a parameter for executing the algorithm.

References

[1] Blei, D. M., Ng, A. Y. and Jordan, M. I.: Latent Dirichlet allocation, *J. Mach. Learn. Res.*, Vol. 3, pp. 993–1022 (2003).

[2] Can, F. and Ozkarahan, E.A.: Two partitioning type clustering algorithms, *J. Ame. Soc. Inf. Sci. and Technol.*, Vol. 35, pp. 268–276 (1984).

[3] Can, F. and Ozkarahan, E.A.: Similarity and stability analysis of the two partitioning type clustering algorithms, *J. Ame. Soc. Inf. Sci. and Technol.*, Vol. 36, pp. 3–14 (1985).

[4] Crouch, D. B.: A file organization and maintenance procedure for dynamic document collections, *Inf. Process. Manage.*, Vol. 11, pp. 11–21 (1975).

[5] Dhillon, I. S. and Modha, D. S.: Concept decompositions for large sparse text data using clustering, *Mach. Learn.*, Vol. 42, pp. 143–175 (2001).

[6] Duda, R. O., Hart, P. E. and Stork, D. G.: *Pattern Classification*, Wiley, New York, 2nd edition (2001).

[7] Hartigan, J. A. and Wong, A.: A k-means clustering algorithm, *Appl. Stat.*, Vol. 28, pp. 100–108 (1979).

[8] Hofmann, T.: Probabilistic latent semantic indexing, *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, New York, ACM Press, pp.50–57 (1999).

[9] Kogan, J.: *Introduction to Clustering Large and High-dimensional Data*, Cambridge, Cambridge University Press (2007).
[10] Kishida, K.: High-speed rough clustering for very large document collections, *J. Ame. Soc. Inf. Sci. and Technol.*, Vol. 61, pp. 1092–1104 (2010).
[11] Lewis, D. D., Yang, Y., Rose, T. G. and Li, F.: RCV1: a new benchmark collection for text categorization research, *J. Mach. Learn. Res.*, Vol. 5, pp. 361–397 (2004).
[12] Liu, X., Gong, Y., Xu, W. and Zhu, S.: Document clustering with cluster refinement and model selection capabilities, *Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, New York, ACM Press, pp.191–198 (2002).
[13] McLachlan, G. and Peel, D.: *Finite Mixture Models*, New York, John Wiley & Sons (2000).
[14] Teh, Y. W., Jordan, I., Beal, M. J. and Blei, D. M.: Hierarchical Dirichlet processes, *J. Ame. Stat. Assoc.*, Vol. 101, pp. 1566–1581 (2006).
[15] Willett, P.: Document clustering using an inverted file approach, *J. Inf. Sci.*, Vol. 2, pp. 223–231 (1980).
[16] Xu, R. and Wunsch II, D. C.: *Clustering*, New York, Wiley (2009).

Appendix

A.1 Spherical K-means Algorithm

In the Hartigan-Wong algorithm [7], the density of generated clusters is used as an objective criterion for clustering. If the density increases by moving a document to another cluster, then the document is reallocated to the cluster in an iterative procedure. For the case of cosine measure, the 'density' of cluster C_k is reasonably defined such that $J_k = \sum_{i:d_i \in C_k} \mathbf{v}_i^T \mathbf{c}_k / \|\mathbf{c}_k\|$ where $\mathbf{v}_i \equiv \mathbf{d}_i / \|\mathbf{d}_i\|$ and $\mathbf{c}_k = \sum_{i:d_i \in C_k} \mathbf{v}_i$. From simple manipulation, the increase of J_k by moving document d_* to cluster C_k can be represented by

$$\Delta^+ J_k = \left(\frac{\|\mathbf{c}_k\|}{\|\mathbf{c}_k + \mathbf{v}_*\|} - 1 \right) J_k + \frac{2\mathbf{v}_*^T \mathbf{c}_k + 1}{\|\mathbf{c}_k + \mathbf{v}_*\|},$$

where $\mathbf{v}_* \equiv \mathbf{d}_* / \|\mathbf{d}_*\|$. On the other hand, when document d_* is removed from cluster C_k, the decrease of J_k becomes

$$\Delta^- J_k = \left(1 - \frac{\|\mathbf{c}_k\|}{\|\mathbf{c}_k - \mathbf{v}_*\|} \right) J_k + \frac{2\mathbf{v}_*^T \mathbf{c}_k - 1}{\|\mathbf{c}_k - \mathbf{v}_*\|}.$$

By changing the Euclidean distance to the cosine measure and incorporating straightforwardly $\Delta^+ J_k$ and $\Delta^- J_k$ into the original Hartigan-Wong algorithm, it is possible to execute an effective spherical k-means algorithm (first L documents were automatically selected as initial seeds).

A.2 Gibbs Sampling for HDP Mixture Model

Gibbs sampling based on the CRF model [14] allows the topic assignment to be estimated for each token in given textual data. In this framework, each document is considered as a single restaurant of the franchise serving a common menu of dishes, and a customer of the restaurant (corresponding to a token) selects a table at which only a single dish is ordered where a dish is regarded as a topic. In the Gibbs sampling, random selection of a table at which each token is sitting and of a dish ordered by each table, is iteratively repeated based on the full conditional probabilities that are derived from the assumption of the HDP. The conditional probabilities are mathematically complicated and contain many notations listed in Table A·1.

First, the probability that the u-th table was chosen for given the h-th token (denoted by w_h) in each restaurant becomes

Table A·1 Mathematical notations

Notations	Definitions
$F_{j\|k}$	The number of times that the j-th term appears as a token at the table ordering the k-th dish in all documents (i.e., restaurants).
$F^{\neg h}_{j\|k[h]}$	The number of times that the j-th term appears as a token at the table ordering the $k[h]$-th dish in all documents where $k[h]$ indicates the index of a dish ordered at the table of the h-th token, but the h-th token itself is not counted.
$F^{\neg u}_{j\|k}$	The number of times that the j-th term appears as a token at the table ordering the k-th dish in all documents except for the u-th table.
$F^{\neg h}_{u\|i[h]}$	The number of tokens at the u-th table to which the h-th token belongs, in the i-th document. Note that the h-th token itself is not counted.
m_k	The number of tables ordering the k-th dish in all documents.
$m^{\neg u}_k$	The number of tables ordering the k-th dish in all documents except for the u-th table.
$\sigma[u]$	A set of indexes of tokens sitting at the u-th table and $\|\sigma[u]\|$ indicates its number.

$$P_h(u) \propto \begin{cases} F^{\neg h}_{u|i[h]} f^{\neg h}_k(w_h = t_j), & \text{if the table exists} \\ \alpha P(w_h = t_j | u^*), & \text{if the table is new} \end{cases}, \quad \text{(A.1)}$$

where t_j denotes the j-th index term, and other quantities are defined such that

$$f^{\neg h}_k(w_h = t_j) \equiv \frac{F^{\neg h}_{j|k[h]} + \beta}{\sum_{j'=1}^{M} F^{\neg h}_{j'|k[h]} + M\beta},$$

$$P(w_h = t_j | u^*) \equiv \sum_{k=1}^{L} \frac{m_k}{m' + \gamma} f^{\neg h}_k(w_h = t_j) + \frac{\gamma}{m' + \gamma} \frac{1}{M},$$

and $m' \equiv \sum_{k=1}^{L} m_k$ (α, β and γ are hyper-parameters of the Dirichlet distributions). If a new table (indicated by u^*) is chosen, then a dish (i.e., topic) eaten on it has to be selected according to the conditional probability represented by

$$P_u(k) \propto \begin{cases} m_k / (m' + \gamma), & \text{if the dish exists} \\ \gamma / (m' + \gamma), & \text{if the dish is new} \end{cases}. \quad \text{(A.2)}$$

Second, whether the dish for each table is changed or not is determined by sampling a dish based on the conditional probability represented by

$$P_u(k) \propto \begin{cases} m^{\neg u}_k g^{\neg u}_k, & \text{if the dish exists} \\ \gamma g^{\neg u}_{k*}, & \text{if the dish is new} \end{cases}, \quad \text{(A.3)}$$

where

$$g^{\neg u}_k \equiv \frac{\prod_{j \in \sigma[u]} \Gamma(F_{j|k} + \gamma)}{\prod_{j \in \sigma[u]} \Gamma(F^{\neg u}_{j|k} + \gamma)} \times \frac{\Gamma(\sum_{j=1}^{M} F^{\neg u}_{j|k} + L\gamma)}{\Gamma(\sum_{j=1}^{M} F_{j|k} + L\gamma)},$$

and $g^{\neg u}_{k*} \equiv (1/M)^{|\sigma[u]|}$ (Γ is a gamma function).

If the Gibbs sampling based on Equations (A.1), (A.2) and (A.3) is repeatedly executed, then a set of clusters can be obtained in each sample by allocating every document to the dish (i.e., topic) of a table at which the most tokens of the document are sitting. Note that other Gibbs sampling methods for estimating the HDP mixture model were given in [14].