Knowledge-Based Systems 23 (2010) 883–889

Contents lists available at ScienceDirect

Knowledge-Based Systems

journal homepage: www.elsevier.com/locate/knosys

Data clustering with size constraints

Shunzhi Zhu [a], Dingding Wang [b], Tao Li [a,b,*]

[a] Department of Computer Science & Technology, Xiamen University of Technology, Xiamen 361024, PR China
[b] School of Computer Science, Florida International University, Miami, FL 33199, USA

ARTICLE INFO

Article history:
Received 25 January 2010
Received in revised form 29 April 2010
Accepted 13 June 2010
Available online 13 July 2010

Keywords:
Constrained clustering
Size constraints
Linear programming
Data mining
Background knowledge

ABSTRACT

Data clustering is an important and frequently used unsupervised learning method. Recent research has demonstrated that incorporating instance-level background information to traditional clustering algorithms can increase the clustering performance. In this paper, we extend traditional clustering by introducing additional prior knowledge such as the size of each cluster. We propose a heuristic algorithm to transform size constrained clustering problems into integer linear programming problems. Experiments on both synthetic and UCI datasets demonstrate that our proposed approach can utilize cluster size constraints and lead to the improvement of clustering accuracy.

1. Introduction

The goal of cluster analysis is to divide data objects into groups so that objects within a group are similar to one another and different from objects in other groups. Traditionally, clustering is viewed as an unsupervised learning method which groups data objects based only on the information presented in the dataset without any external label information [28]. *K*-means [18] is one of the simplest and most famous clustering algorithms. It defines a centroid for each cluster, which is usually the mean of a group of objects. The algorithm starts by choosing *K* initial centroids, where *K* is a user-specified number of desired clusters, and then iteratively refines and updates the centroids until there is no further change with the centroids.

In real world applications such as image coding clustering, spatial clustering in geoinformatics, and document clustering [11,14,15,20,24,26,28,17], people usually obtain some background information of the data objects' relationships or the approximate size of each group before conducting clustering. This information supposes to be very helpful in clustering the data. However, traditional clustering algorithms do not provide effective mechanisms to make use of this information.

Recent research has looked at using instance-level background information, such as pairwise must-link and cannot-link constraints. If two objects are known to be in the same group, we say that they are must-linked. Or if they are known to be in different groups, we say that they are cannot-linked. Wagstaff et al. [29,30] incorporated this type of background information to *K*-means algorithm by ensuring that constraints are satisfied at each iteration during the clustering process. Basu et al. [4,5,7,13] also considered pairwise constrains to learn an underlying metric between points while clustering. Other work on learning distance metrics for constrained clustering can be found in [6,8,9,12,25]. In addition, many methods have been developed to incorporate domain knowledge for fuzzy clustering where the data objects can be assigned to multiple clusters to various degrees (membership values). In particular, many different types of knowledge hints have been used for fuzzy clustering, including partial supervision where some data points have been labeled [23], knowledge-based indicators and guidance including proximity hints where the resemblances between some pairs of data points are provided and uncertainty hints where the confidence or difficulty of the cluster membership function for a data point is characterized [21], and domain knowledge represented in the form of a collection of viewpoints (e.g., externally introduced prototypes/representatives by users) [22]. However, little work has been reported on using the size constraints for clustering.

There is another type of work focusing on balancing constraints, i.e., clusters are of approximately the same size or importance. Besides the demands of several applications, balanced clustering is also helpful in generating more meaningful initial clusters and avoiding outlier clusters. Banerjee and Ghosh [2,3] showed that a small sample was sufficient to obtain a core clustering and then allocated the rest of the data points to the core clusters while satisfying balancing constraints. Zhong and Ghosh [32,33] also took

[*] Corresponding author at: School of Computer Science, Florida International University, Miami, FL 33199, USA. Tel.: +1 305 348 6036; fax: +1 305 348 3549.
E-mail addresses: szzhu@xmut.edu.cn (S. Zhu), dwang003@cs.fiu.edu (D. Wang), taoli@cs.fiu.edu (T. Li).

0950-7051/$ - see front matter © 2010 Elsevier B.V. All rights reserved.
doi:10.1016/j.knosys.2010.06.003

balancing constraints into consideration and developed an iterative bipartitioning heuristic for sample assignment. All of the effort has illustrated that utilizing background knowledge can improve clustering performance in accuracy and scalability.

Balancing constraints can be viewed as a special case of size constraints where all the clusters have the same size. Several real life clustering applications require the clusters that have fixed size, but not necessarily the equal size for all the clusters. For example, a typical task in marketing study is customer segmentation where customers are divided into different groups where a particular sales team or a specific amount of marketing dollars is allocated to each group. If each sales team is of different size or the allocation of marketing dollars is of different amount, then the customer segmentation problem becomes a data clustering problem with size constraints. Similarly, a job scheduling problem, where a number of jobs are assigned to different machines/processes, can be modeled as a data clustering problem with size constraints if different machines/processes have different capacities. Many other problems such as document clustering where each cluster has a fixed storage space and spatial clustering where each cluster has a specific number of spatial objects can be naturally formulated as data clustering problems with size constraints.

In this paper, we extend balancing constraints to size constraints, i.e., based on the prior knowledge of the distribution of the data, we assign the size of each cluster and try to find a partition which satisfies the size constraints. We also present some case studies of considering size constraints and instance-level cannot-link constraints simultaneously. We propose a heuristic procedure to solve these constrained clustering problems by transforming them into integer linear programming optimization problems. Experiments on synthetic and UCI datasets demonstrate the improvement of clustering accuracy using our proposed methods.

The rest of the paper is organized as follows. In Section 2, we present the problem formulation. In Section 3, we describe our heuristic procedure to produce a near-optimal solution. In Section 4, we present the experimental results. Finally, in Section 5, we conclude our work and discuss the future work.

2. Problem formulation

In the problem of clustering with size constraints, we have the prior knowledge of the number of objects in each cluster. And we can also obtain the partition result of any traditional clustering algorithm, such as K-means. Then the problem is formulated as follows.

Given a data set of n objects, let $A = (A_1, A_2, \dots, A_p)$ be a known partition with p clusters, and $NumA = (na_1, na_2, \dots, na_p)$ be the number of objects in each cluster in A. We look for another partition $B = (B_1, B_2, \dots, B_p)$ which maximizes the agreement between A and B, and $NumB = (nb_1, nb_2, \dots, nb_p)$ represents the size constraints, i.e., the number of objects in each cluster in B.

A and B can be represented as $n \times p$ partition matrices. Each row of the matrix represents an object, and each column is for a cluster. $a_{ij} = 1$ or $b_{ij} = 1$ when object i belongs to cluster j in partition A or B. A can be represented as

$$A = \begin{bmatrix} 1 & 0 & 0 & \dots & 0 & 0 & 0 \\ & & & \dots & & & \\ & & & \dots & & & \\ 1 & 0 & 0 & \dots & 0 & 0 & 0 \\ 0 & 1 & 0 & \dots & 0 & 0 & 0 \\ 0 & 1 & 0 & \dots & 0 & 0 & 0 \\ & & & \dots & & & \\ & & & \dots & & & \\ 0 & 0 & 0 & \dots & 0 & 0 & 1 \\ 0 & 0 & 0 & \dots & 0 & 0 & 1 \end{bmatrix}, \tag{1}$$

where

$$\sum_{i=1}^{n} a_{ij} = na_j, \quad j = 1, \dots, p,$$

and

$$\sum_{j=1}^{p} a_{ij} = 1, \quad i = 1, \dots, n.$$

It is easy to see that AA^T is an $n \times n$ matrix with the values

$$(AA^T)_{ij} = \begin{cases} 1, & i \text{ and } j \text{ are in the same group in } A, \\ 0, & \text{otherwise}. \end{cases} \tag{2}$$

The problem is to find another partition B, which minimizes

$$\|AA^T - BB^T\|,$$

satisfying $\sum_{i=1}^{n} b_{ij} = nb_j,\ j = 1, \dots, p$, and $\sum_{j=1}^{p} b_{ij} = 1,\ i = 1, \dots, n$.

The problem is similar to finding a partition which maximizes its agreement with another known partition [19].

3. Solution of size constrained clustering

Now, the original size constrained clustering problem becomes an optimization problem. Here, we propose a heuristic algorithm to efficiently find the solution.

3.1. The heuristic procedure

To solve the problem stated in the previous section, first of all, we define

$$D_a = diag(na_1, na_2, \dots, na_p), \tag{3}$$

and

$$D_b = diag(nb_1, nb_2, \dots, nb_p). \tag{4}$$

Let

$$U_j = \frac{1}{\sqrt{na_j}} \begin{bmatrix} a_{1j} \\ a_{2j} \\ \dots \\ \dots \\ \dots \\ a_{nj} \end{bmatrix}, \quad j = 1, \dots, p, \tag{5}$$

where each $a_{ij} \in \{0, 1\}$, and

$$\sum_i a_{ij} = na_j, \quad j = 1, \dots, p,$$

$$\sum_j a_{ij} = 1, \quad i = 1, \dots, n.$$

Then, we can see that actually $U = A(D_a)^{-1/2}$. In the same way, let

$$V_j = \frac{1}{\sqrt{nb_j}} \begin{bmatrix} b_{1j} \\ b_{2j} \\ \dots \\ \dots \\ \dots \\ b_{nj} \end{bmatrix}, \quad j = 1, \dots, p, \tag{6}$$

where each $b_{ij} \in \{0, 1\}$, and

$$\sum_i b_{ij} = nb_j, \quad j = 1, \dots, p,$$

$$\sum_j b_{ij} = 1, \quad i = 1, \dots, n.$$

Similarly, $V = B(D_b)^{-1/2}$.

Then, we can write

$$AA^T = UD_aTU, \tag{7}$$

and

$$BB^T = UD_bTU. \tag{8}$$

Thus

$$\|AA^T - BB^T\|^2 = \|UD_aU^T - VD_bV^T\|^2 = \|D_a - U^TVD_b(U^TV)^T\|^2. \tag{9}$$

If there exists V, so that

$$U^TV = J = \begin{bmatrix} 1 & 0 & 0 & \ldots & 0 & 0 & 0 \\ 0 & 1 & 0 & \ldots & 0 & 0 & 0 \\ 0 & 0 & 1 & \ldots & 0 & 0 & 0 \\ & & & \ldots & & & \\ & & & \ldots & & & \\ 0 & 0 & 0 & \ldots & 1 & 0 & 0 \\ 0 & 0 & 0 & \ldots & 0 & 1 & 0 \\ 0 & 0 & 0 & \ldots & 0 & 0 & 1 \end{bmatrix}, \tag{10}$$

then we can get

$$\|AA^T - BB^T\|^2 = \|D_a - D_b\|^2. \tag{11}$$

B would be the best approximation of A, and at the same time satisfies the size constraints. However, it is difficult to find such a V so that $U^TV = J$. Therefore, we would like to choose V to minimize $\|U^TV - J\|$. Note that

$$\|U^TV - J\| = 2p - 2\sum_{i=1}^{n}\sum_{j=1}^{p} u_{ij}v_{ij}, \tag{12}$$

where u_{ij} and v_{ij} are the components of the vectors U_j and V_j, respectively. Thus, this size constrained clustering can be transformed to the following linear programming problem:

$$minimize - \left[\sum_{j=1}^{p} \frac{1}{\sqrt{na_j nb_j}} \sum_{i=1}^{n} a_{ij}b_{ij}\right], \tag{13}$$

where

$$\sum_i b_{ij} = nb_j, \quad j = 1,\ldots,p,$$
$$\sum_j b_{ij} = 1, \quad i = 1,\ldots,n,$$

and

$$b_{ij} \in \{0,1\}.$$

This is a typical binary integer linear programming problem which can easily be solved by any existing solver.

In summary, the heuristic procedure has two steps:

- Apply an efficient and effective traditional or instance-level constrained clustering algorithm to the data and
- Create size constraints based on the prior knowledge of the data, and then transform the size constrained clustering to a binary integer linear programming problem by the approach described above.

3.2. Discussion

One thing worth mentioning is that if we use any other instance-level constrained clustering algorithm as the base algorithm in the first step, some constraints may be violated in the final results. Actually, instance-level constraints such as cannot-link constraints can be incorporated into our transformed linear programming as an inequality constraint. For example, if the object k and object l cannot be in the same cluster, the constraint can be presented as:

$$b_{kj} + b_{lj} \leqslant 1, \quad j = 1,\ldots,p. \tag{14}$$

Therefore, our approach can either (1) take advantage of other instance-level constrained clustering algorithms and add the size prior knowledge to them or (2) directly solve the constrained clustering problem based on the traditional clustering methods.

As described in Section 1, in many real life clustering applications, the sizes of each cluster are known in advance. Using the size constraints would help improve the clustering performance. In addition to the application requirements, clustering with size constraints is also helpful to decrease the sensitivity of random initialization in clustering process and to avoid the generation of outlier clusters or highly imbalanced clusters [33]. Note that in many cases, however, the prior knowledge about the exact size of the clusters might be difficult to acquire. In these cases, we can consider the relaxation of the size constraints. Instead of specifying the exact size of each cluster, our method also allows for the approximate size range of a cluster. For example, in some applications, the cluster size cannot exceed a threshold t, then this constraint can be written as:

$$\sum_i b_{ij} \leqslant t, \quad j = 1,\ldots,p,$$

and

$$\sum_j b_{ij} = 1, \quad i = 1,\ldots,n.$$

Some experiments on the relaxation of size constraints are presented in Section 4.4.3. Note that a special case is the balancing constraint where we can require the size of each cluster is roughly the same.

4. Experimental evaluation

In this section, we conduct experiments to evaluate the performance of our proposed approach. First of all, we look at a case

Table 1
Data sets from UCI.

Dataset	Number of objects	Number of classes	Cluster size
Iris	150	3	50, 50, 50
Wine	178	3	71, 59, 48
Balance scale	625	3	288, 288, 49
Ionosphere	351	2	226, 125

Table 2
Comparisons with *K*-means algorithm. Remark: *KM* denotes the *K*-means algorithm, *SC* represents our heuristic size constrained clustering approach, *Acc* stands for accuracy, and *Ent* is for entropy.

Data	Algo	*Acc*	*Ent*	ARI	NMI
Iris	*KM*	0.8133	0.2852	0.5913	0.6402
	SC	0.8467	0.2559	0.6416	0.6750
Wine	*KM*	0.7022	0.4477	0.3711	0.4288
	SC	0.7079	0.4400	0.3863	0.4383
Balance scale	*KM*	0.4976	0.7182	0.1186	0.0742
	SC	0.5248	0.7020	0.1389	0.0931
Ionosphere	*KM*	0.7123	0.3321	0.3634	0.2599
	SC	0.8034	0.3534	0.4056	0.2926

study to test the feasibility of our method. We then introduce the datasets and evaluation metrics used in our experiments. Finally, the experimental results on different datasets are presented and discussed.

4.1. A case study on feasibility

First of all, we use a very simple synthetic data set to illustrate the feasibility of our approach. In particular, the simple example demonstrates the importance of the size constrained clustering problem.

Given data X which contains six objects and each object has six attributes. We can write X as the following 6 × 6 matrix, in which each row represents an object and each column represents an attribute. Formally,

$$X = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

The class labels for the data are

$$ClassLabel = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 2 \\ 2 \\ 2 \end{bmatrix},$$

which means that there are two groups. The first three objects are in one group and the rest three are in another. The traditional K-means algorithm groups the first two objects together and the rest four in another group. If we know that the size of each cluster should be three, then using our heuristic approach, the clustering results would be the same as the ground truth.

This simple case study demonstrates that if we take advantage of size prior knowledge, the performance of clustering can be optimized, especially when in the case that there exist some marginal objects.

4.2. Experimental datasets

In order to evaluate the effectiveness of our heuristic approach, we use four UCI datasets to conduct size constrained clustering. The characteristics of the datasets are summarized in Table 1 and more description of the datasets can be found in [1].

4.3. Evaluation measures

To measure the clustering performance, we consider four measures including accuracy, adjusted rand index (ARI), normalized mutual information (NMI), and entropy. These measures are briefly described below.

- Accuracy discovers the relationship between each cluster and class which is the ground truth. It sums up the total matching degree between all pairs of clusters and classes [16]. Accuracy can be represented as:

$$Accuracy = Max\left(\sum_{C_k, L_m} T(C_k, L_m)\right)\Big/ N, \tag{15}$$

where C_k denotes the kth cluster, and L_m is the mth class. $T(C_k, L_m)$ is the number of entities which belong to class m are assigned to cluster k. Accuracy computes the maximum sum of

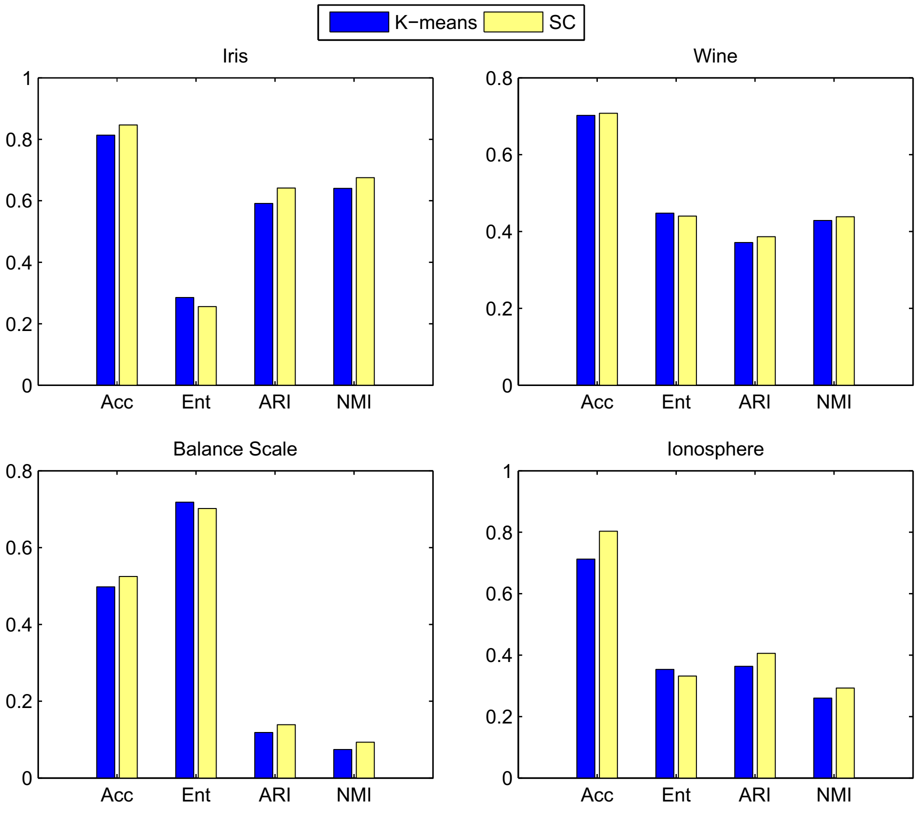

Fig. 1. Comparisons with K-means. Remark: *SC* represents our heuristic size constrained clustering approach, *Acc* stands for accuracy, and *Ent* is for entropy.

$T(C_k, L_m)$ for all pairs of clusters and classes, and there is no overlap among these pairs. It is obvious that greater the accuracy, better the clustering performance.

- ARI [10] measures the agreements between the clustering and the partition by class labels. It is defined as the number of pairs of objects which are both placed in the same cluster and the same class, or both in different clusters and different classes, divided by the total number of objects. ARI is set between [0,1], and the higher the ARI, the more the resemblance between the clustering results and the labels.
- NMI [27] measures the amount of statistical information shared by two random variables representing cluster assignment and underlying class label. Suppose entry n_{ij} denotes the amount of data items belonging to cluster i and class j. NMI is computed as:

$$NMI = \frac{\sum_{i=1}^{c}\sum_{j=1}^{k}\frac{n_{ij}}{n}log\frac{n_{ij}n}{n_{i\cdot}n_{\cdot j}}}{\sqrt{\left(\sum_{i=1}^{c}\frac{-n_{i\cdot}}{n}log\frac{n_{i\cdot}}{n}\right)\left(\sum_{j=1}^{k}\frac{-n_{\cdot j}}{n}log\frac{n_{\cdot j}}{n}\right)}}, \tag{16}$$

where $n_{i\cdot} = \sum_{j=1}^{k} n_{ij}$, $n_{\cdot j} = \sum_{i=1}^{c} n_{ij}$, n, c, k denote the total number of data objects, the number of clusters, and the number of classes, respectively. Based on our prior knowledge on the number of classes, usually, we set the number of clusters equal to the true number of classes, i.e., $c = k$.

- Entropy measures how classes are distributed on various clusters. In general, the smaller the entropy value, the better the clustering quality is. More details on the purity and entropy measures can be found in [31].

Table 3
Comparisons with *MPCK*-means algorithm. Remark: *MPC* is *MPCK*-means algorithm, *ISC*1 represents our first way to incorporate both size and instance-level constraints, *ISC*2 represents our second way to do so, *Acc* stands for accuracy, and *Ent* is for entropy.

Data	Algo	*Acc*	*Ent*	ARI	NMI
Iris	*MPC*	0.8600	0.2399	0.6648	0.6933
	*ISC*1	0.8933	0.1830	0.7302	0.7582
	*ISC*2	0.8867	0.1971	0.7163	0.7419
Wine	*MPC*	0.9101	0.2199	0.7498	0.7087
	*ISC*1	0.9494	0.1401	0.8516	0.8267
	*ISC*2	0.9157	0.2282	0.7631	0.7205
Balance scale	*MPC*	0.5680	0.6243	0.0760	0.0527
	*ISC*1	0.6112	0.5823	0.1414	0.1165
	*ISC*2	0.6240	0.5973	0.1439	0.0895
Ionosphere	*MPC*	0.8148	0.3380	0.3910	0.2800
	*ISC*1	0.8177	0.3358	0.3984	0.2868
	*ISC*2	0.8205	0.3321	0.4056	0.2926

4.4. Results and discussions

Here, we conduct three sets of experiments to compare our proposed approach with both traditional *K*-means algorithm and instance-level constrained *MPCK*-means algorithm [7]. We also evaluate the flexibility of our approach to deal with the approximate size range constraints instead of exact cluster size constraints.

4.4.1. Comparisons with K-means

The first set of experiments evaluates the effectiveness of our proposed heuristic approach by using *K*-means as the base algorithm in the first step of our approach, and comparing our results

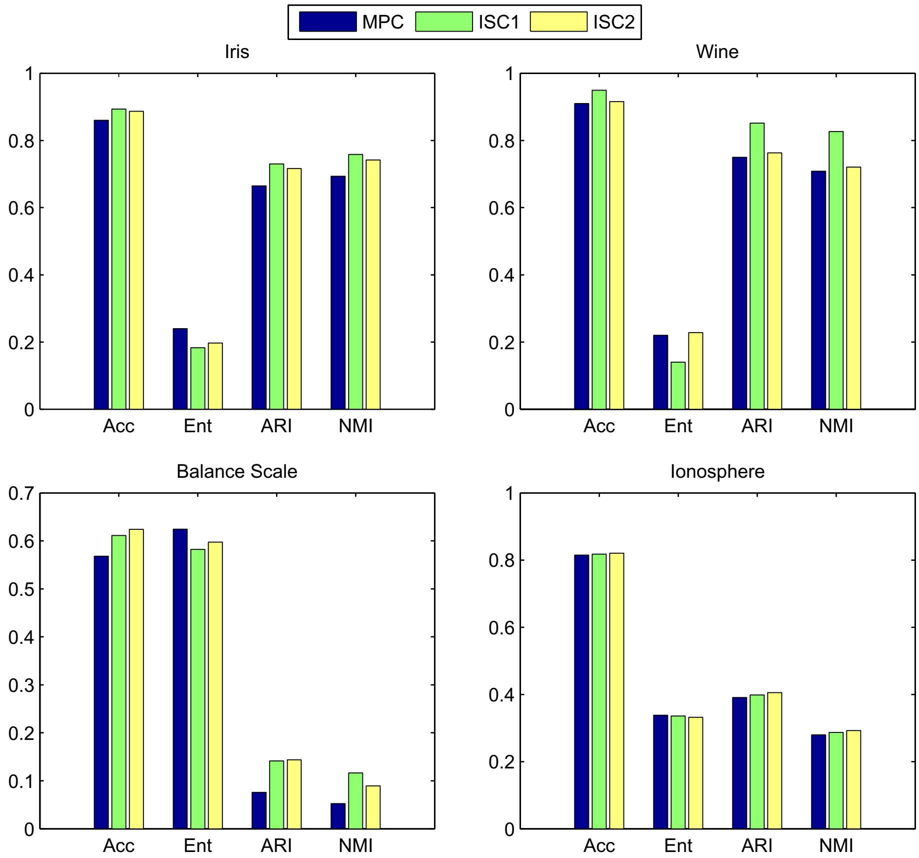

Fig. 2. Comparisons with *MPCK*-means. Remark: *MPC* is *MPCK*-means algorithm, *ISC*1 represents our first way to incorporate both size and instance-level constraints, *ISC*2 represents our second way to do so, *Acc* stands for accuracy, and *Ent* is for entropy.

with the results of traditional *K*-means algorithm. Table 2 and Fig. 1 illustrate the evaluation results of our heuristic size constrained clustering using the four measures which are introduced in Section 4.3.

From Table 2 and Fig. 1, we can clearly observe that the clustering performance is enhanced by incorporating size constraints into *K*-means using our approach.

4.4.2. Comparisons with MPCK-means

The second set of experiments compares our approach with *MPCK*-means algorithm [7], which considers instance-level constraints. We incorporate both size constraints and cannot-link constraints into our approach in the following two ways: (1) use *MPCK*-means as the base algorithm in the first step of our approach, and in the second step, size constraints are taken into consideration; (2) use *K*-means as the base algorithm, and write cannot-link constraints in the form of inequality constraints of the linear programming in the second step of our approach. Fifty pairs of cannot-link constraints are generated randomly for each datasets, and the experimental results are shown in Table 3 and Fig. 2.

The results of this set of experiments are interesting. When we incorporate the size constraints into *MPCK*-means algorithm, we cannot guarantee that all the original instance-level constraints are satisfied. In fact, some of them are violated in order to satisfy the size constraints, which means that size constraints have higher priority in our approach. From the results shown in Table 3 and Fig. 2, we notice that the performance is improved by our approach in either of the two methods mentioned before.

4.4.3. Relaxation of size constraints

In this set of experiments, we use Wine dataset and study the flexibility of our approach by relaxing the exact size constraints to range constraints, which means instead of specifying the exact size of each cluster, the approximate range is also acceptable. This is very important in some real world applications. We specify that the number of objects in each cluster cannot exceed 75.

Table 4
Case studies on size relaxation. Remark: *RC* represents our range constrained clustering approach, *Acc* stands for accuracy, and *Ent* is for entropy.

Data	Algo	*Acc*	*Ent*	ARI	NMI
Iris	*KM*	0.8133	0.2852	0.5913	0.6402
	RC	0.8467	0.2559	0.6416	0.6750

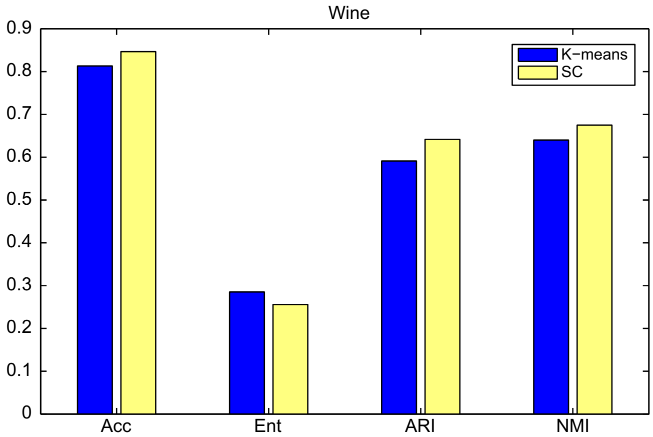

Fig. 3. Case studies on size relaxation. Remark: *RC* represents our range constrained clustering approach, *Acc* stands for accuracy, and *Ent* is for entropy.

Table 4 and Fig. 3 show the ability of our approach to handle the size range problem. From the results, we observe that size range is also a very important information to improve the clustering performance.

5. Conclusion and future work

In this paper, we develop a heuristic approach to incorporate size constraints into traditional or existing constrained clustering algorithms. Instance-level cannot-link constraints can also be incorporated into our proposed size constrained clustering. Instead of specifying exact cluster size, we can relax the size constraints as a rough size range for each cluster. Experimental results on UCI datasets show the improvement in the clustering performance.

There are several things we can consider in our future work. First of all, we would like to collect more complex data sets from real applications, and conduct comprehensive experiments. Additionally, we are interested in extending our approach to include various types of constraints that can be required in the application domain. Finally, it is also interesting to make our approach more flexible to handle "softer" constraints.

Acknowledgements

The work is partially supported by the Natural Science Foundation of Fujian Province under Grant No. 2010J01353 and by the Open Research Fund of the Lab of Spatial Data Mining and Information Sharing of Ministry of Education of China at Fuzhou University under Grant No. 201001.

References

[1] A. Asuncion, D.J. Newman, UCI Machine Learning Repository, University of California, School of Information and Computer Science, Irvine, CA, 2007, <http://www.ics.uci.edu/mlearn/MLRepository.html>.
[2] A. Banerjee, J. Ghosh, On scaling up balanced clustering algorithms, in: Proceedings of SIAM Data Mining, 2002, pp. 333–349.
[3] A. Banerjee, J. Ghosh, Scalable clustering algorithms with balancing constraints, Data Mining Knowledge Discovery 13 (3) (2006) 365–395.
[4] S. Basu, A. Banerjee, R.J. Mooney, Semi-supervised clustering by seeding, in: Proceedings of ICML, 2002, pp. 27–34.
[5] S. Basu, A. Banerjee, R.J. Mooney, Active semi-supervision for pairwise constrained clustering, in: Proceedings of SIAM Data Mining, 2004, pp. 333–344.
[6] T.D. Bie, M. Momma, N. Cristianini, Efficiently learning the metric using side-information, in: Proceedings of ALT, 2003, pp. 175–189.
[7] M. Bilenko, S. Basu, R.J. Mooney, Integrating constraints and metric learning in semi-supervised clustering, in: Proceedings of ICML, 2004, pp. 81–88.
[8] P.S. Bradley, K.P. Bennett, A. Demiriz, Constrained *K*-means clustering, Technical Report MSR-TR-2000-65, Microsoft Research, 2000.
[9] R. Ge, E. Martin, J. Wen, I. Davidson, Constraint-driven clustering, in: Proceedings of ACM SIGKDD, 2007, pp. 320–329.
[10] L. Hubert, P. Arabie, Comparing partitions, Journal of Classification 2 (1) (1985) 183–218.
[11] F. Jacquenet, C. Largeron, Discovering unexpected documents in corpora, Knowledge Based System 22 (6) (2009) 421–429.
[12] D. Klein, S.D. Kamvar, C.D. Manning, From instance-level constraints to space-level constraints: making the most of prior knowledge in data clustering, in: Proceedings of ICML, 2002, pp. 307–314.
[13] B. Kulis, S. Basu, I. Dhilon, R.J. Mooney, Semi-supervised graph clustering: a kernel approach, in: Proceedings of ICML, 2005, pp. 457–464.
[14] M. Laguia, J.L. Castro, Local distance-based classification, Knowledge Based System 21 (7) (2008) 692–703.
[15] M. Li, L. Zhang, Multinomial mixture model with feature selection for text clustering, Knowledge Based System 21 (7) (2008) 704–708.
[16] T. Li, C. Ding, M. Jordan, Solving consensus and semi-supervised clustering problems using nonnegative matrix factorization, in: Proceedings of IEEE ICDM, 2007, pp. 577–582.
[17] T. Li, S. Zhu, M. Ogihara, Algorithms for clustering high dimensional and distributed data, Intelligent Data Analysis 7 (4) (2003) 305–326.
[18] J.B. MacQueen, Some methods for classification and analysis of multivariate observations, in: Proceedings of the Fifth Symposium on Math, Statistics, and Probability, 1967, pp. 281–297.
[19] H. Massatfa, An algorithm to maximize the agreement between partitions, Journal of Classification 9 (1) (1992) 5–15.

[20] G.P. Patil, R. Acharya, R. Modarres, W.L. Myers, S.L. Rathbun, Hot-spot geoinformatics for digital government, in: Encyclopedia of Digital Government, vol. II, 2007.

[21] W. Pedrycz, Fuzzy clustering with a knowledge-based guidance, Pattern Recognition Letter 25 (4) (2004) 469–480.

[22] W. Pedrycz, V. Loia, S. Senatore, Fuzzy clustering with viewpoints, IEEE Transactions on Fuzzy Systems 18 (2) (2010) 274–284.

[23] W. Pedrycz, J. Waletzky, Fuzzy clustering with partial supervision, IEEE Transactions on Systems, Man, and Cybernetics – Part B: Cybernetics 27 (5) (1997) 787–795.

[24] S. Rogers, P. Langley, C. Wilson, Mining GPS data to augment road models, in: Proceedings of ACM SIGKDD, 1999, pp. 104–113.

[25] M. Schultz, T. Joachims, Learning a distance metric from relative comparisons, in: Proceedings of NIPS, 2003.

[26] J. Shi, J. Malik, Normalized cuts and image segmentation, IEEE Transactions on Pattern Analysis and Machine Intelligence 22 (8) (2000) 888–905.

[27] C. Studholme, D. Hill, D.J. Hawkes, An overlap invariant entropy measure of 3D medical image alignment, Pattern Recognition 32 (1) (1999) 71–86.

[28] P. Tan, M. Steinbach, V. Kumar, Introduction to Data Mining, Addison Wesley, 2005.

[29] K. Wagstaff, C. Cardie, Clustering with instance-level constraints, in: Proceedings of ICML, 2000, pp. 1103–1110.

[30] K. Wagstaff, C. Cardie, S. Rogers, S. Schroedl, Constrained *K*-means clustering with background knowledge, in: Proceedings of ICML, 2001, pp. 577–584.

[31] Y. Zhao, G. Karypis, Soft criterion functions for partitional document clustering: summary of results, in: Proceedings of CIKM, pp. 246–247, 2004.

[32] S. Zhong, J. Ghosh, Scalable, balanced model-based clustering, in: Proceedings of SIAM Data Mining, 2003, pp. 71–82.

[33] S. Zhong, J. Ghosh, A unified framework for model-based clustering, Journal of Machine Learning Research 4 (2003) 1001–1037.