

Clustering Ensembles Based on Normalized Edges^{*}

Yan Li¹, Jian Yu², Pengwei Hao^{1,3}, and Zhulin Li¹

¹ Center for Information Science, Peking University,
Beijing, 100871, China
{yanli, lizhulin}@cis.pku.edu.cn

² Inst. of Computer Science & Engineering, Beijing Jiaotong Univ.,
Beijing, 100044, China
jianyu@center.njtu.edu.cn

³ Dept. of Computer Science, Queen Mary, Univ. of London,
London, E1 4NS, UK
phao@dcs.qmul.ac.uk

Abstract. The co-association (CA) matrix was previously introduced to combine multiple partitions. In this paper, we analyze the CA matrix, and address its difference from the similarity matrix using Euclidean distance. We also explore how to find a proper and better algorithm to obtain the final partition using the CA matrix. To get more robust and reasonable clustering ensemble results, a new hierarchical clustering algorithm is proposed by developing a novel concept of normalized edges to measure the similarity between clusters. The experimental results of the proposed approach are compared with those of some single runs of well-known clustering algorithms and other ensemble methods and the comparison clearly demonstrates the effectiveness of our algorithm.

1 Introduction

Cluster analysis is an important tool for exploratory data analysis, aiming to find homogeneous groups in a data set of unlabeled objects. Numerous algorithms have been and are being developed [2], [9], [10], such as the K-means (KM), the single-linkage (SL) or the average-linkage (AL) and the spectral clustering algorithms [14], [15].

However, clustering is inherently an ill-posed problem. All the previous methods are designed with certain assumptions and favor some type of biases, and no single one is universally suitable for solving all the problems [19]. Hoping to exploit the strength of many individual clustering algorithms, people turn to

^{*} This work was partially supported by the National Natural Science Foundation under Grant No. 60303014, the Fok Ying Tung Education Foundation under Grant No. 101068, the Specialized Research Found of Doctoral Program of Higher Education of China under Grant No. 20050004008, and the Foundation for the Authors of National Excellent Doctoral Dissertation, China, under Grant 200038.

clustering ensembles, seeking improvement over a single clustering algorithm in such aspects as *robustness*, *novelty* and *scalability*, *et al.*

Besides formal arguments on the effectiveness of cluster ensembles [18], many combining algorithms have been proposed, and their good performance further justified the use of cluster ensembles. A few examples are: methods based on hypergraph (CSPA, HGPA, and MCLA) [16] or bipartite graph partitioning [3], evidence accumulation using the CA matrix (EAC-SL and EAC-AL) [6], mixture model using a unified representation for multiple partitions [17], bagged clustering [11], and combination by plurality voting [1], [4], [7], [20].

Despite the primary success achieved by those algorithms, they are far from ideal. To make the clustering ensembles practical and helpful for us, an effective algorithm is crucial, which is the focus of this paper. We first analyze the CA matrix, and discuss the problem of designing a proper algorithm for it in Sect. 2. Based on a novel concept of *normalized edges* as we have defined in this paper, we propose a hierarchical algorithm to find the final partition in Sect. 3. In Sect. 4, experiment results demonstrate the effectiveness of our proposed method.

2 Analysis of the Co-Association Matrix

2.1 The Co-Association (CA) Matrix

In order to combine the multiple partitions of the data, one can first map the data to a new feature space as a way to accumulate the information provided by each partition. The CA matrix¹ C [6] is a newly constructed similarity matrix from multiple partitions of the original data. It takes the co-occurrence of pairs of patterns in the same clusters as votes for their association, with elements

$$C(i, j) = \frac{n_{ij}}{N}, \quad (1)$$

where n_{ij} is the number of times the pair x_i and x_j is assigned to the same cluster among the N partitions. In fact, the CA matrix records the frequency that every pair of points is in the same cluster.

The CA matrix ($N = 30$, and the number of clusters is fixed to 60^2) of the *2-spirals* data³ (Fig. 1a) is shown in Fig. 1c. For comparison, the similarity matrix (normalized into the 0-1 scale) based on the Euclidean distance for the original patterns is plotted in Fig. 1b. Evidently, the similarity of pairs of points from different clusters is mostly much smaller in the CA matrix than that in the original similarity matrix, showing the CA matrix captures the global structure of the data. Thus, it is not surprising that the evidence accumulation method operating on the CA matrix [6] performs well and that is why we use the CA matrix to combine the multiple partitions in our method.

¹ This matrix is also used in the CSPA algorithm [16]; and in [12] but by the name of *consensus matrix*.

² This strategy, initially splitting the data into a large number of small clusters and then combining them, is the so-called *split-and-merge* approach [5].

³ To make comparison easy, we reorder the data, so the first 100 points are from one cluster and the last 100 points from the other cluster.

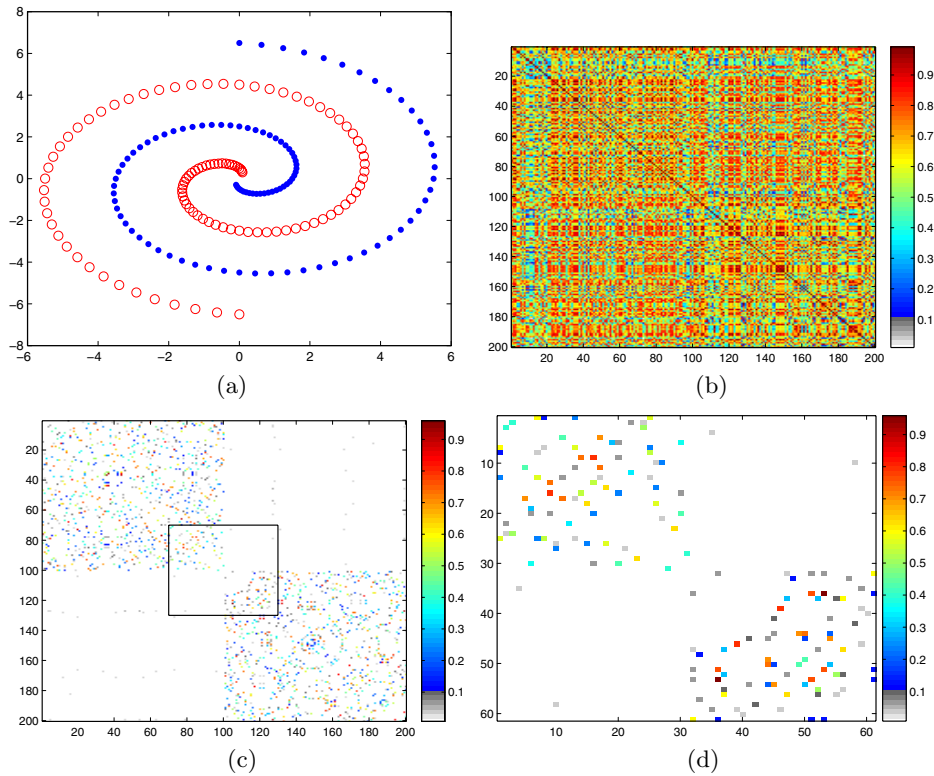


Fig. 1. The *2-spirals* data set and its similarity matrices. (a) *2-spirals* data. (b) Similarity matrix using Euclidean distance. (c) CA matrix: new similarity matrix using multiple partitions. (d) Enlarged part of (c).

2.2 A Proper Algorithm

Despite the good discrimination ability of the CA matrix, improper clustering algorithms can still lead to bad results. For example, in [6] (e.g. table 2) and [17] (e.g. Fig. 9 and 10b), the authors found that, based on the CA values, the results of the consensus functions (i.e., SL, AL and CL) differ significantly, and the choice of a good consensus function is sensitive to the choice of the data set. So, does there exist a better consensus function? How can we find it?

Compared with ordinary similarity matrix using Euclidean distance, the CA matrix has special characteristics. Without loss of generality, we take the CA matrix (Fig. 1c) of the *2-spirals* data as an example, part of which is highlighted in Fig. 1d. The similarity matrix of the data using Euclidean distance is shown in Fig. 1b. To construct a good algorithm, we believe that the following characteristics should be taken into account: 1) points from different clusters are always dissimilar, 2) a large percentage of pairs of points from the same cluster have very low similarity, and 3) if two points from the same cluster are dissimilar, then there always should be a path of some points (or just one point) between them

who are successively similar. Referring to these features, we can also explore the reasons why the SL, AL and CL algorithms would have such performance in [6].

3 Normalized Edges and the Algorithm

Based on the above analysis, we will customize a hierarchical clustering algorithm to operate on the CA matrix for combining multiple partitions, in hopes of discovering the true structure of the data more successfully and robustly.

3.1 Normalized Edges

Our proposed hierarchical clustering algorithm is based on a novel concept of *normalized edges* for measuring the similarity between clusters. Treating all points of the data as a set of vertices, we can define an undirected and unweighted graph. An edge exists between two points (or vertices), x_i and x_j , if and only if their similarity is larger than a threshold θ . In fact, this defines a *threshold graph*. For simplicity, we define a function *edge* between two points x_i and x_j ,

$$edge(x_i, x_j) = \begin{cases} 1 & \text{if } \text{sim}(x_i, x_j) > \theta, \\ 0 & \text{otherwise.} \end{cases} \quad (2)$$

The notion of *edges* between two clusters C_i and C_j , $edges(C_i, C_j)$, is just the number of distinct edges connecting these two groups. Essentially, this can be used as a measure of the *goodness* of merging them in an agglomerative hierarchical clustering algorithm. However, this naive approach may work well only for well-separated and approximately equal-sized clusters.

A proper way to fix this problem is to normalize the number of edges between two clusters $edges(C_i, C_j)$, by dividing it by the (estimated) expected number of edges between them, which is inspired by *goodness measure* used in ROCK [8]. Hence, the number of *normalized edges (NE)* between two clusters, C_i and C_j , is

$$NE(C_i, C_j) = \frac{edges(C_i, C_j)}{(n_i + n_j)^{1+f(\theta)} - n_i^{1+f(\theta)} - n_j^{1+f(\theta)}}, \quad (3)$$

where n_i and n_j are the number of points, $n_i^{1+f(\theta)}$ and $n_j^{1+f(\theta)}$ are the estimated expected number of edges, in the clusters C_i and C_j respectively.

As in [8], we assume that every point in C_i has $n_i^{f(\theta)}$ edges with other points in the cluster, then the total number of edges between points in the cluster is $n_i^{1+f(\theta)}$ (each edge is counted twice). Thus the expected number of edges between pairs of points (each point from a different cluster) becomes $(n_i + n_j)^{1+f(\theta)} - n_i^{1+f(\theta)} - n_j^{1+f(\theta)}$. Intuitively, the function $f(\theta)$ is introduced to measure the influence of θ on the number of edges. Based on the analysis of Guha et al. [8], it is also defined as $(1 - \theta)/(1 + \theta)$ in this paper.

3.2 Our Clustering Algorithm

With the definition of *normalized edges* to measure the similarity between two clusters, we can use this measure to construct a new agglomerative hierarchical algorithm. To reduce the workload of calculation, we need not re-start calculating the similarity between clusters after each merging step. We just have to update the similarity between the merged and the other clusters. That is, if clusters C_i and C_j are merged into a new cluster C_k , then we have (for $l \neq i, j$ and k) $edges(C_l, C_k) = edges(C_l, C_i) + edges(C_l, C_j)$, and $n_k = n_i + n_j$.

Thus we can calculate the *normalized edges* $NE(C_l, C_k)$ by definition.

The proposed algorithm can be summarized as follows:

The Algorithm

Input:

The similarity matrix (the CA matrix in this paper), the threshold θ , and the number of clusters k .

Initialization:

To calculate *edge* between all pairs of points based on the similarity matrix.

Repeat:

1. To merge two clusters, among all possible pairs of clusters, with the largest *normalized edges*;
2. To update the *edges* and *normalized edges* between the merged clusters and the other clusters.

Until:

Only k clusters left or the number of edges between every pair of the remaining clusters becomes zero.

Certainly, we can speed up the algorithm by many methods used in the traditional agglomerative clustering algorithms. The *edge* between every pair of points can be computed in $O(n^2)$ time, and the worst time complexity of the clustering algorithm is $O(n^2 \log n)$, just as the ROCK algorithm.

For the threshold θ in our algorithm, we are still seeking a general way to determine it. Empirically, the algorithm worked well with θ in the interval $[0.1, 0.4]$ for many data sets, and was not very sensitive to it (e.g., for all the data sets in the experiments of this paper, we fixed θ to 0.30).

4 Experiments

We compared our algorithm with single runs of some well-known clustering algorithms and other ensemble ones, and the experiment results demonstrate the effectiveness of our method.

4.1 Data Sets, Algorithms and Parameters Selection

We summarized the details of the data sets in Table 1, which had been adopted by other authors to test their ensemble algorithms [4], [6], [13], [17].

Table 1. Characteristics of the Data Sets

Data set	No. of features	No. of classes	No. of points/class(noise)	Total no. of points
2-spirals	2	2	100-100	200
Complex image	2	7	200-200-100-100-50-50-33-(10)	743
3-paths	2	3	200-200-200-(200)	800
Wisconsin Breast-cancer	9	2	239-444	683
Std Yeast	17	5	67-135-75-52-55	384

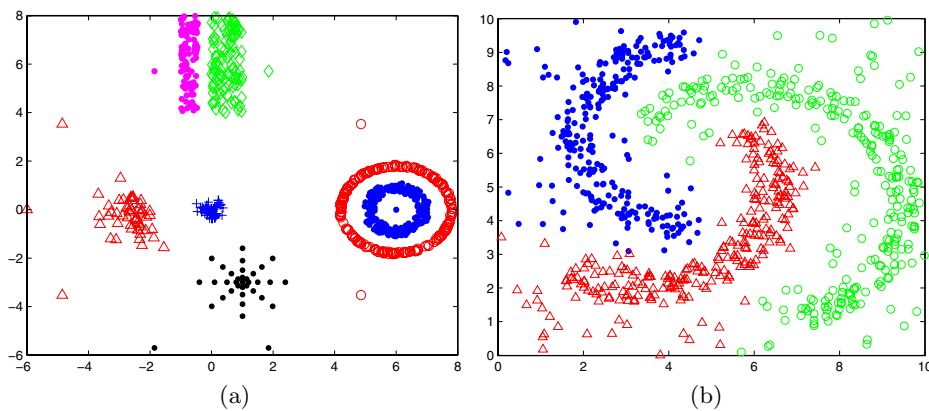


Fig. 2. The *complex image* and *3-paths* data sets. Clusters are indicated by different colors/patterns. Here show the clustering results of our ensemble algorithm. (a) *Complex image*, the 10 points of the outer circle are considered as noise. (b) *3-paths*, 3 path-like clusters (200 points for each), corrupted by 200 noise points.

We compared the experiment results of our ensemble algorithm (denoted by CA-HNE) with single runs of following algorithms: KM, SL, AL, and spectral clustering algorithm (SC) [14], and following ensemble methods: CSPA, HGPA and MCLA [16], Boost-KM [7], EAC-SL and EAC-AL [6], and Latent-EM [17].

For the algorithms proposed by other authors, the parameters selection and other settings are the same as suggested in their papers. The multiple partitions of the data were obtained by running KM algorithm with random initialization of cluster centers.

We found that, using only 10 or 20 component partitions, our algorithm worked well for many data sets, while many other authors used a (much) larger number for their algorithm, e.g. 50 [6], 100 [4], 500 [12]. For simplicity, we generated 30 partitions for our algorithm and 50 for all other ensemble ones. For our algorithm, k was chosen as a constant, usually larger than the true number of clusters of the data. The ‘true’ number of clusters of the data, was assumed to be known, as in [6], [17].

4.2 Results, Comparison and Analysis

Table 2 summarizes the mean error rates and standard deviations from 20 independent runs of the different methods on the data sets. The error rates are obtained by matching the clustering results with the ground-truth information, taken as the known labeling of the real-world data sets or the perceptual grouping of the artificial ones. Since all the clustering algorithms considered here do not detect outliers in the data, we ignore the noise points when calculating the error rates. Notice that the SL and AL algorithms give unvaried clustering results for each data set.

Table 2. Mean Error Rates and Standard Deviations of Different Algorithms

Data set	KM	SL	AL	SC	CSPA	HGPA
2-spirals	.399±.011	0	.480	0±0	.418±.059	.394±.078
Complex image	.550±.073	.523	.478	.250±.074	.404±.076	.389±.048
3-paths	.327±.005	.667	.350	.046±.001	.212±.058	.251±.068
Breast-cancer	.039±.001	.349	.057	.029±0	.167±.020	.147±.017
Std Yeast	.358±.057	.638	.341	.320±0	.442±.011	.431±.014
Data set	MCLA	Boost -KM	EAC -SL	EAC -AL	Latent -EM	CA -HNE
2-spirals	.365±.075	.429±.007	0±0	.325±.051	.418±.030	0±0
Complex image	.397±.078	.533±.032	.136±.125	.546±.043	.540±.055	.013±.023
3-paths	.214±.138	.328±.000	.617±.122	.375±.046	.349±.049	.006±.017
Breast-cancer	.131±.030	.039±.001	.319±.093	.047±.007	.039±.001	.030±.004
Std Yeast	.422±.022	.332±.021	.594±.062	.349±.023	.390±.072	.333±.030

We can see that the evidence accumulation clustering (EAC-SL or EAC-AL) can sometimes discover the structure of the data successfully. However, which method will succeed depends heavily on the choice of the data sets. In general, the AL (SL) consensus function based on CA matrix is appropriate if standard AL (SL) agglomerative clustering method works well for the data, and vice versa [17]. This may be problematic since sometimes the characteristic of the data is difficult to know, or is complex for standard agglomerative clustering (e.g. *3-paths*). However, our algorithm tackles this problem well. For our method, the mean error rates presented in Table 2 are all the best or comparable to the best, and the partitions of the *complex image* and *3-paths* data sets are as good as we expected, see Fig. 2a and 2b. The experiments show the effectiveness of our algorithm: it gives the best (or comparable to the best) overall performance for all the data sets. It clusters all the chosen data sets reasonably, though they have hybrid or complex characteristic or are corrupted by noise. The CSPA algorithm does not perform well for these chosen data sets, though it is also based on the CA matrix. Again, this demonstrates the importance of the choice of algorithms for the final partition based on the CA matrix. Other ensemble methods HGPA, MCLA, Boost-KM, latent-EM still favor some type of biases, and do not perform well for other kinds of data sets.

Single runs of KM, SL, and AL algorithms result in good partitions when the data sets are suitable for them, but fail drastically otherwise (e.g. noise, hybrid structure). The SC algorithm gives reasonable partitions for some of the data sets, but fails for *complex image* and *3-paths*.

References

1. Dudoit, S., Fridlyand, J.: Bagging to Improve the Accuracy of a clustering procedure. *Bioinformatics*, 19 (2003) 1090–1099
2. Everitt, B.S., Landau, S., Leese, M.: *Cluster Analysis* (4e). Hodder Arnold (2001)
3. Fern, X., Brodley, C.: Solving cluster ensemble problems by bipartite graph partitioning. *Proc. 21st Int'l Conf. Machine Learning* (2004) 281–288
4. Fischer, B., Buhmann, J.M.: Bagging for Path-Based Clustering. *IEEE Trans. Patt. Anal. Machine Intell.*, 25(11) (2003) 1411–1415
5. Fred, A.L.N., Jain, A.K.: Data Clustering Using Evidence Accumulation. *Proc. 16th Int'l Conf. Pattern Recognition*, 280 (2002) 276–280
6. Fred, A.L.N., Jain, A.K.: Combining Multiple Clusterings Using Evidence Accumulation. *IEEE Trans. Patt. Anal. Machine Intell.*, 27(6) (2005) 835–850
7. Frossyniotis, D., Likas, A., Stafylopatis, A.: A Clustering Method Based on Boosting. *Pattern Recognition Letters*, 25 (2004) 641–654
8. Guha, S., Rastogi, R., Shim, K.: ROCK: A Robust Clustering Algorithm for Categorical Attributes. *Information systems*, 25(5) (2000) 345–366
9. Jain, A.K., Dubes, R.C.: *Algorithms for Clustering Data*. Prentice Hall (1988)
10. Kaufman, L., Rousseeuw, P.J.: *Finding Groups in Data: An Introduction to Cluster Analysis*. John Wiley & Sons (1990)
11. Leisch, F.: Bagged Clustering. Working Papers SFB Adaptive Information Systems and Modeling in Economics and Management Science, Institut für Information, Abt. Produktionsmanagement, Wien, Wirtschaftsuniv, 51 (1999)
12. Monti, S., Tamayo, P., Mesirov, J.P., Golub, T.R.: Consensus Clustering: A Resampling-Based Method for Class Discovery and Visualization of Gene Expression Microarray Data. *Machine Learning*, 52(1-2) (2003) 91–118
13. Newman, D.J., Hettich, S., Blake, C.L., Merz, C.J.: UCI Repository of machine learning databases, Univ. of California, Dept. of Info. and Computer Science (1998)
14. Ng, A.Y., Jordan, M.I., Weiss, Y.: On Spectral Clustering: Analysis and an Algorithm. *Advances in Neural Information Processing Systems*, 14 (2002)
15. Shi, J., Malik, J.: Normalized Cuts and Image Segmentation. *IEEE Trans. Patt. Anal. Machine Intell.*, 22(8) (2000) 888–905
16. Strehl, A., Ghosh, J.: Cluster Ensembles — A Knowledge Reuse Framework for Combining Partitionings. *J. Machine Learning Research*, 3 (2002) 583–617
17. Topchy, A., Jain, A.K., Punch, W.: Clustering Ensembles: Models of Consensus and Weak Partitions. *IEEE Trans. PAMI.*, 27(12) (2005) 1866–1881
18. Topchy, A.P., Law, M.H.C., Jain, A.K., Fred, A.L.: Analysis of Consensus Partition in Cluster Ensemble. *Proc. 4th IEEE Int'l Conf. Data Mining* (2004) 225–232
19. Xu, R., Wunsch, D.: Survey of clustering algorithms. *IEEE Trans. Neural Networks*, 16(3) (2005) 545–678
20. Zhou, Z.H., Tang, W.: Clusterer Ensemble. *Knowledge-Based Systems*, 19(1) (2006) 77–83