Partitioning Clustering Based on Support Vector Ranking

Qing Peng[1], Yan Wang[1(✉)], Ge Ou[1], Yuan Tian[1], Lan Huang[1], and Wei Pang[2(✉)]

[1] College of Computer Science and Technology, Jilin University, Changchun, China
wy6868@jlu.edu.cn
[2] School of Nature and Computing Sciences, University of Aberdeen, Aberdeen, Scotland, UK
pang.wei@abdn.ac.uk

Abstract. Support Vector Clustering (SVC) has become a significant boundary-based clustering algorithm. In this paper we propose a novel SVC algorithm named "Partitioning Clustering Based on Support Vector Ranking (PC-SVR)", which is aimed at improving the traditional SVC, which suffers the drawback of high computational cost during the process of cluster partition. PC-SVR is divided into two parts. For the first part, we sort the support vectors (SVs) based on their geometrical properties in the feature space. Based on this, the second part is to partition the samples by utilizing the clustering algorithm of similarity segmentation based point sorting (CASS-PS) and thus produce the clustering. Theoretically, PC-SVR inherits the advantages of both SVC and CASS-PS while avoids the downsides of these two algorithms at the same time. According to the experimental results, PC-SVR demonstrates good performance in clustering, and it outperforms several existing approaches in terms of Rand index, adjust Rand index, and accuracy index.

Keywords: Support vector clustering · Support vector ranking · Partitioning clustering

1 Introduction

Data Clustering has been an important task in data mining, and existing clustering algorithms can be classified into five categories [1]: partitioning methods [2–4], hierarchical methods [2, 5–7], density-based methods [8–10], grid-based methods [11, 12], and model-based methods [13, 14].

Among many clustering algorithms, support vector clustering (SVC) [15, 16] has become a significant boundary-based clustering algorithm in several applications such as community discovery, speech recognition and bioinformatics analysis [17]. SVC has the following features: first, it can be applied to various shapes of the clusters; second,

The original version of this chapter was revised: An acknowledgement has been added. The erratum to this chapter is available at DOI: 10.1007/978-3-319-49586-6_61

J. Li et al. (Eds.): ADMA 2016, LNAI 10086, pp. 726–737, 2016.
DOI: 10.1007/978-3-319-49586-6_52

the number of clusters is not needed in advance; third, it can deal with structured data by using kernel functions; fourth, it can reduce the impact of noise on the cluster partition.

However, there is still room for improvement for SVC. The algorithm is still inadequate due to two bottlenecks: expensive computational cost and poor labeling piece, and this degrades the popularity of SVC [17]. To address these limitations, some work has been done: Ben-Hur *et al.* [15] improved the original algorithm and proposed a method called support vector graph (SVG). The main idea of this method was that support vectors (SVs) were used to construct the adjacency matrix and derive connected component with an aim to reduce time complexity; Yang et al. [18] proposed the proximity graph (PG), and its time complexity was reduced to $O(N \log N)$ or $O(N)$; Lee et al. [19] devised gradient descent (GD) by looking for the stable equilibrium point (SEP); Jung et al. [20] proposed the fast support vector clustering (FSVC), which improved the speed of the algorithm as well as the quality of clustering; Sei-Hyung Lee [21] designed a cone-based cluster partition method to avoid random operations, and it was called Cone Cluster Labeling (CCL), which improved the quality of clustering but increased operation cost; Convex decomposition based cluster labeling (CDCL) [22] was proposed to improve both the efficiency and accuracy of clustering based on convex decomposition; L-CRITICAL was a novel SVC cluster labeling algorithm, and it solved the labeling phase of SVC within competitive processing time [23]; Proximity Multi-sphere Support Vector Clustering (PMS-SVC) was developed based on the multi-sphere approach to support vector data description [24]; Rough–Fuzzy Support Vector Clustering (RFSVC) can obtain rough fuzzy clusters using the support vectors as cluster representatives [25].

The clustering algorithm of similarity segmentation based point sorting (CASS-PS) [26] has a faster speed in clustering. However, the similarity measure of the algorithm is based on distance, which is likely to cause staggered sorting issue between different cluster elements, and this will reduce the accuracy of clustering results.

In this paper, we propose an improved SVC algorithm called partitioning clustering based on support vector ranking (PC-SVR). The algorithm's crucial components are (1) SV's sorting based on their geometric properties in the feature space and (2) cluster partition that uses the clustering algorithm of similarity segmentation based point sorting (CASS-PS). The proposed algorithm guarantees the quality of the clustering and improves the speed of clustering at the same time.

2 Partitioning Clustering Based on Support Vector Ranking

Our PC-SVR algorithm combines the first stage of SVC and CASS-PS, and the algorithm is composed of two stages: first, sort the support vectors (SVs) into an array; second, split the sorted array.

2.1 Support Vector Sorting

In the feature space, data are mapped to the minimal sphere. Assume this sphere is S, and the center is a. According to $K(x,x) = \exp\left(-q \cdot \|x - x\|^2\right) = 1$, we can get

$K(x,x) = <\Phi(x) \cdot \Phi(x) > = \|\Phi(x)\|^2 = 1$, which means all the data points are located on the surface of the unit ball. Assume this ball is B, and the center of B is O. So, the covering is the intersection, whose shape is like a cap. The center of this cap is denoted as a', as shown in Fig. 1. Since SVs are on the surface of S, they are also on the intersection hyper line of S and B.$\Phi(v_i)$ and $\Phi(v_j)$ are SVs in the feature space, and θ is the angle between the support vectors and two sphere center. The transverse section of the cap is illustrated in Fig. 2.

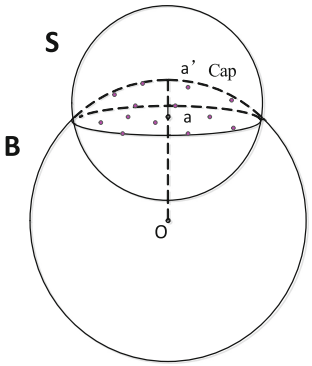


Fig. 1 Intersection between ball and sphere

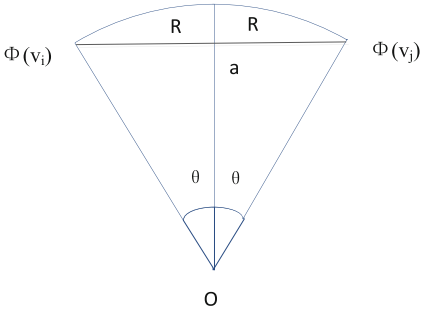


Fig. 2 Transverse section of the cap

Given a dataset containing N data points $\{x_i | x_i \subseteq x, 1 \leq i \leq N\}$, let $V = \{v_i | v_i$ *is a SV*$, 1 \leq i \leq N_{SV}\}$. In this research, we will use the geometric properties of samples in the feature space as follows [21]:

Lemma 1. $\angle(\Phi(v_i o a')) = \angle(\Phi(v_j o a'))\ \forall v_i, v_j \in V$

Lemma 2. $\forall x \in X, v \in V, \angle(\Phi(v) o \Phi(x)) < \theta \Leftrightarrow \ ||v - x|| < ||v - \Phi^{-1}(a')||$

Lemma 3. $x \in X, v \in V, ||v - x|| < ||v - \Phi^{-1}(a')|| \Leftrightarrow x, v$ belongs to the same cluster.

The following corollary can be proven by the above three properties, as in [17]:

Corollary. In the feature space of Gaussian Kernel, SVs are collected in terms of clusters on the intersection hyper line of S and B. This is illustrated in Fig. 3.

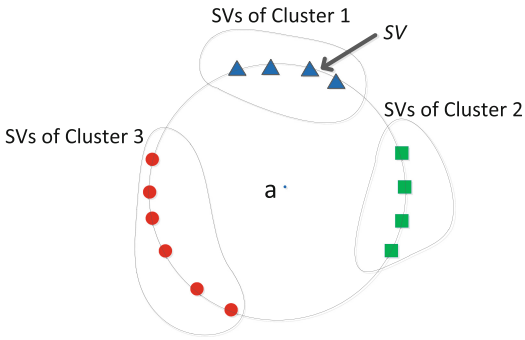


Fig. 3 The distribution of SVs in three clusters

From the above properties, for any $v_i, v_j \in V$, in the feature space, they have the same angle as Oa' in Fig. 1, and if the angle between the sample point and the SV is less than θ, the point and the SV belong to the same cluster. In the data space, the point to which the distance from the SV is less than $||v - \Phi^{-1}(a')||$ has the same cluster label as the SV, and thus the computation of the feature space can be converted to the computation of input space.

Therefore, we can use the angle between two SVs (v_i and v_j) and O to measure the distance between two SVs. The relation between the angle and the Gaussian kernel function is as follows: $\cos(\angle\Phi(v_i)O\Phi(v_j)) = <\Phi(v_i) \cdot \Phi(v_j)> = K(v_i \cdot v_j)$. Namely, the comparison of the distance of two SVs is transformed into the comparison of the kernel function, and the greater the distance between the two SVs is, the larger the angle is, and the smaller the kernel value is.

The similarity matrix for SVs is constructed according to the values of the kernel function. Then, according to the matrix, SVs are sorted as follows: first, the two SVs whose $\Phi(v_i)$, $\Phi(v_j)$ have the minimum distance are selected as the head and tail of an ordered array; Second, find the SV whose $\Phi(v_k)$ has the minimum distance from the head (or tail) of the sorted array as the head (or tail) of the array. Repeat this step until all data points are stored in an array and we can get the sequence of SVs in the feature space on the circumference, as shown in Fig. 4.

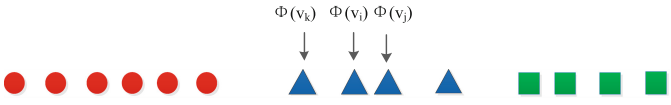


Fig. 4 The Sequence of SVs

2.2 Partition Clustering

This part mainly uses the CASS-PS algorithm to partition the cluster.

In the data space, we firstly calculate the distance between each sample point (except for SVs) and each SV, and find the nearest SV. Then we insert the point into the adjacent position of the array in which the SV is located. Repeat this process until all the sample points are stored in the array. In order to observe the transformation of the distance between adjacent elements more directly, we draw the distance curve between the adjacent sample points according to the distance between elements. The distance curve can show obvious changes in distance between adjacent elements, and especially it has a great wave between adjacent sample points of SVs.

At the same time, we can use the wavelet filter function to reduce the impact of noise points or isolated points, and thus we can find the best segmentation point more accurately. Then we set up a threshold whose value can be set as the mean amplitude, and we ignore the part below the threshold in order to simplify the determination of split points. So the continuous curve is divided into several discontinuous curve segments. Furthermore, we find the position which has the maximum distance between adjacent elements as the splitting point in each curve. Then we sort these splitting points after finding the splitting points of all (the whole) curve, and the position which has the maximum distance between adjacent splitting points is selected as the first splitting place. According to this procedure, it can be decided that the next step is re-segmentation or termination. After algorithm terminates, we output the number of clusters.

2.3 The Implementation of PC-SVR Algorithm

In this research we mainly use the geometric properties of sample points in the feature space and CASS-PS to improve the cluster partitioning, which is the second stage of the SVC algorithm. In the feature space, SVs are collected based on the clusters on the intersection hyper line of the minimal sphere and the unit ball. Sorting SVs is based on the similarity between two SVs, which is based on the value of the kernel function. Since SVs are already sorted, it is useful to avoid the limitation of the CASS-PS algorithm, that is, the sample points of different clusters tend to overlap.

The detailed steps of our algorithm are finally given as follows:

(1) Given a sample set $S = \{x_i | x_i \subseteq X\}$, its sample size is N, and set parameters q and C. Initialize a one-dimensional array based on the sample size;
(2) Calculate the kernel matrix of the sample set;
(3) Calculate the radius R of the minimal sphere and SVs according to Lagrange polynomial;
(4) Calculate the kernel matrix of SVs, and construct a similarity matrix of support vectors;
(5) Sort SVs according to the similarity matrix and get a sorted array of SVs. At this point, the first stage of the algorithm is completed;
(6) Calculate the distance from other sample points to all SVs, and find the closest SV to the sample point to be sorted. Insert the sample point into the back of the SV;

(7) Repeat Step 6, until all other sample points are completed with the interpolation, and we get a new sample point array;
(8) Draw the curve of the distance between adjacent sample points. Apply the wavelet filter function to the sample point array to reduce noise. Set a certain threshold and retain the portion above the threshold as the split segment;
(9) Find the points that have the maximum distance (the peak of the distance curve) in various segments, and sort these points. According to the number of clusters, select the corresponding points as the splitting points to split the array of sample points;
(10) Label cluster labels on the sample points.

3 Experiment Analysis

3.1 Evaluation Criteria of Experimental Results

In this paper, Rand index [27], Adjust Rand index [28] and Accuracy index [29] are used to evaluate the clustering results.

The Rand index is an external evaluation metric and it evaluates the effectiveness of clustering by comparing the actual results and the results obtained by the clustering algorithms. Given a dataset that contains n elements and its known partition result P, we run the algorithm to be evaluated to get another partition result Q. Suppose r is the number of data which belong to the same cluster in P and Q, s is the number of data which belong to different clusters in P and Q, t is the number of data belong to the same cluster in P but belong to the different cluster in Q, and v is the number of data belong to the same cluster in Q but belong to a different cluster in P. On the base of the above, r and s can determine the similarity of clustering results, while t and v can describe the inconsistency of the results. Rand index is given as follows:

$$RI = \frac{r+s}{r+s+t+v} \tag{1}$$

The values of Rand index range in [0, 1], and the greater the value of RI is, the better the clustering results are.

The adjust Rand index will standardize the clustering results in addition to the comparison of the known clustering results and the results obtained by an algorithm. The formula is as follows:

$$s_1 = \sum_{i=1}^{K_P} C_{N_i}^2, \quad s_2 = \sum_{j=1}^{K_Q} C_{N_j}^2, \quad s_3 = \frac{2s_1 s_2}{N(N-1)}, \qquad ARI = \frac{\sum_{i=1}^{K_P}\sum_{j=1}^{K_Q} C_{N_{ij}}^2 - s_3}{(s_1+s_2)/2 - s_3} \tag{2}$$

In the above, P and Q represent the two clustering results of a sample set consisting of n elements, and K_P and K_Q are the numbers of clusters in P and Q, respectively. N_i and N_j represent the numbers of elements in clusters i and j in P and Q, respectively,

and N_{ij} represents the number of elements in both cluster i in P and cluster j in Q. Adjust Rand index ranges in [-1, 1], and the greater the index value is, the more similar the results of the two clustering results are. Adjust Rand index can also be used as a method for determining whether the algorithm is applicable to certain datasets.

Accuracy is one of the most commonly used external evaluation indices. The formula is as follows:

$$AC = \frac{\sum_{i=1}^{m} c_i}{N} \tag{3}$$

In the above, m represents the number of clusters, and N represents the number of elements in the sample set. The above formula is based on the principle of similarity comparison between the correct results and the results obtained by the clustering algorithm.

3.2 Experimental datasets

In this research, the experiments are carried out by both artificial data and real data. All the datasets are described in Table 1. The two artificial datasets: Example 1 and Example 2.

Example 1 is a set of two-dimensional datasets with the size of 150 × 2. In order to verify the feasibility of the algorithm, the dataset is relatively easy to separate.

Table 1. Description of datasets

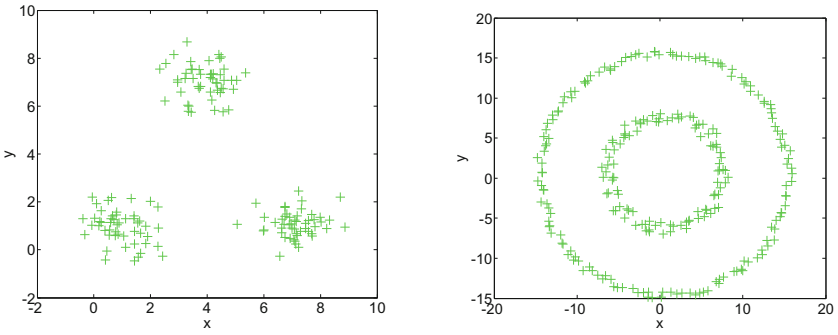
Dataset	Dims	Size	Clusters
Example 1	2	150	3
Example 2	2	250	2
Iris	4	150	3
Wine	13	178	3
Wisconsin	9	683	2
Balance Scale	4	625	3

Example 2 is a set of concentric ring datasets with the size of ... This type of dataset is difficult to cluster, and the purpose is to verify whether the algorithm can deal with the linearly inseparable situations.

The four real datasets used in this research are taken from UCI [30], and they are frequently used in clustering analysis: Iris dataset, Wine dataset, Wisconsin dataset and Balance Scale dataset.

3.3 Experimental Results and Analysis

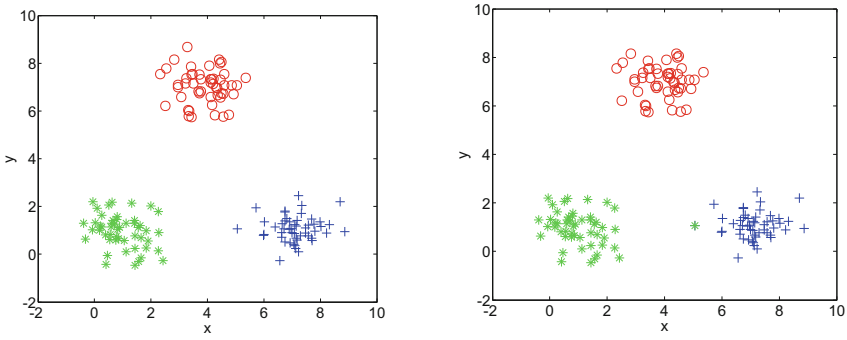
(1) We make a comparison between the experimental results on two artificial datasets based on the PC-SVR algorithm and the original clustering algorithm of similarity segmentation based point sorting algorithm (CASS-PS). The two artificial datasets are shown in Fig. 5.



(a) Example 1 (b) Example 2

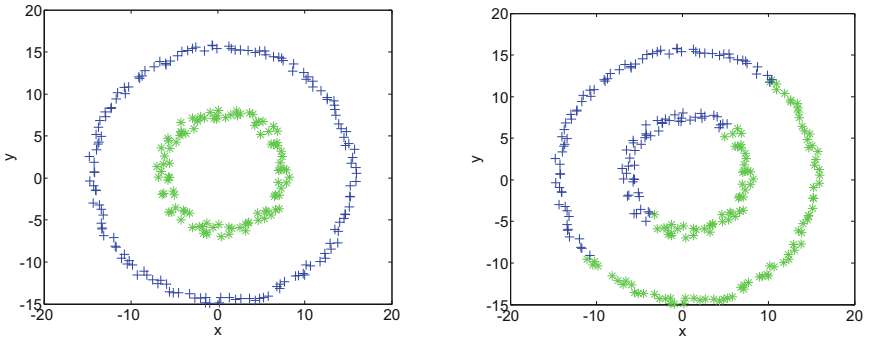
Fig. 5 Two artificial datasets

As shown in Fig. 6, the clustering results of the two algorithms on Example 1 are both satisfactory. But the PC-SVR algorithm is more accurate than the CASS-PS algorithm, and it does not have wrong clustering points. Figure 6 shows that PC-SVR which uses the sorting and clustering algorithm after sorting SVs makes the sorting process more accurate, and it does not tend to assign the data points of the same cluster to the wrong ones.



(a) The clustering result for PC-SVR (b) The clustering result for CASS-PS

Fig. 6 A comparison of Example 1 clustering effect



(e) Final clustering result for PC-SVR (f) Final clustering result for CASS-PS

Fig. 7 A comparison of clustering effect on Example 2

Figure 7 shows the results on Example 2 dataset obtained by using the two algorithms.

As shown in Fig. 7, PC-SVR inherits the advantages of SVC to deal with the linear inseparable datasets when clustering, and it is more accurate than SVC. For this type of datasets, the clustering result of CASS-PS algorithm is not satisfactory.

(2) The comparison of time cost between PC-SVR and SVC is presented in Table 2.

Table 2. Time comparison between SVC and PC-SVR (in second)

Datasets	SVC	PC-SVR
Example 1	215.0942	5.6316
Example 2	47.1435	9.3601
Iris	39.8895	4.9608
Wine	637.2953	5.6940
Balance scale	50.1387	20.787
Wisconsin	37.3594	18.326

Table 2 presents the comparison between SVC and PC-SVR on the two artificial datasets and four sets of classical data in terms of the running time. From this table, we can see the efficiency of the PC-SVR algorithm has been greatly improved compared with the original SVC algorithm.

(3) We use Rand Index to make a comparison of experimental results between the PC-SVR algorithm and other four existing algorithms on three sets of real datasets. The other four clustering algorithms are Support Vector Clustering, Cluster Algorithm of Similarity Segment based Point Sorting, Convex Decomposition based Cluster Labeling and Cone Cluster Labeling. The above experimental results about CDCL and CCL is from reference [22]. The results are shown in Fig. 8.

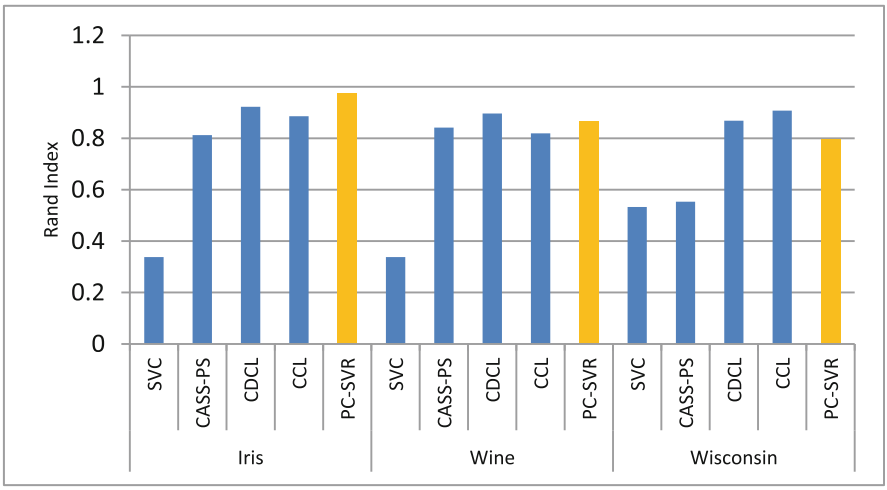


Fig. 8 A comparison of Rand Index for all five algorithms

Figure 8 reports the results for the Rand Index on the three datasets with five algorithms. We can see that PC-SVR performs the best on Iris dataset. Although does not getting the highest Rand Index values on other two datasets, PC-SVR is only 3.46 % lower than CDCL (the best one on Wine dataset) and 12.15 % lower than CCL (the best one on Wisconsin dataset).

In addition, in order to fully verify the clustering performance of the PC-SVR algorithm, we use the other two indices of clustering results to evaluate and compare the clustering performance of PC-SVR algorithm and the other four classical algorithms which are K-means, Hierarchical Clustering, Support Vector Clustering and Cluster Algorithm of Similarity Segment based Point Sorting on the four real datasets, that is, Adjust Rand Index and Accuracy. The results are listed in Tables 3 and 4.

The above results show that the PC-SVR algorithm can ensure the quality of the clustering and improve the speed of clustering, and the clustering performance is excellent.

Table 3. Ajust Rand Index

Dataset	Iris	Wine	Balance scale	Wisconsin
K-Means	0.7302	0.3711	**0.1335**	0.4914
HC	0.5621	−0.0054	0.0854	0.0073
SVC	0.00018143	0	0	0.0024
CASS-PS	0.5621	**0.6569**	0.131	0.0073
PC-SVR	**0.941**	0.6162	0.1298	**0.5921**

Table 4. Accuracy

Dataset	Iris	Wine	Balance scale	Wisconsin
K-Means	0.8933	0.7022	0.5264	0.8514
HC	0.6867	0.3876	0.5216	0.6327
SVC	0.34	0.3989	0.4608	0.6292
CASS-PS	0.7688	**0.882**	0.5696	0.6643
PC-SVR	**0.9975**	0.8384	**0.6416**	**0.8858**

4 Conclusions and Future Work

In this paper, the PC-SVR algorithm based on support vector sorting has been proposed, and it is divided into two parts: support vector sorting and segmentation. In the first part, we sort the SVs on the basis of their geometrical properties of the feature space. In the second part, we partition the samples by using the point sorting-based partition cluster algorithm and generate the clustering. Experimental results demonstrate the effect of PC-SVR for improving the performance of SVC, and better clustering performance has been achieved compared with existing approaches.

In the future work, we would explore the potential application fields of our approach, for instance, in the field of bioinformatics and social media analysis.

Acknowledgement. This work is supported by the National Natural Science Foundation of China (Grant Nos. 61472159, 61572227), Development Project of Jilin Province of China (20140101180JC,20160204022GX).

References

1. Tung, A.K., Hou, J., Han, J.: Spatial clustering in the presence of obstacles. In: 17th IEEE International Conference on Data Engineering, pp. 359–367. IEEE, Heidelberg (2001)
2. Kaufman, L.R., Rousseeuw, P.: Finding Groups in Data: An Introduction to Cluster Analysis. Hoboken NJ John Wiley & Sons Inc., New York (1990)
3. Ng, R.T., Han, J.: Efficient and Effective Clustering Methods for Spatial Data Mining. University of British Columbia, Vancouver (1994)
4. Bradley, P.S., Fayyad, U.M., Reina., C.: Scaling Clustering Algorithms to Large Databases. In: KDD, New York, pp. 9–15 (1998)
5. Zhang, T., Ramakrishnan, R., Livny, M.: BIRCH: an efficient data clustering method for very large databases. In: ACM SIGMOD International Conference on Management of Data, pp. 103–114. ACM, New York (1996)
6. Guha, S., Rastogi, R., Shim, K.: CURE: an efficient clustering algorithm for large databases. In: ACM SIGMOD Record, vol. 27, No. 2, pp. 73–84. ACM, Seattle (1998)
7. Karypis, G., Han, E.-H., Kumar, V.: Chameleon: Hierarchical clustering using dynamic modeling. Computer **32**(8), 68–75 (1999)
8. Ester, M., et al.: A density-based algorithm for discovering clusters in large spatial databases with noise. In: KDD, vol. 96, No. 34, Portland, pp. 226–231 (1996)

9. Ankerst, M., et al.: OPTICS: ordering points to identify the clustering structure. In: ACM Sigmod Record. vol. 28, pp. 49–60. ACM, Philadelphia (1999)
10. Hinneburg, A., Keim, D.A.: An efficient approach to clustering in large multimedia databases with noise. In: KDD, vol. 98, New York, pp. 58–65 (1998)
11. Wang, W., Yang, J., Muntz, R.: STING: A statistical information grid approach to spatial data mining. In: VLDB, vol. 97, Athens, pp. 186–195 (1997)
12. Sheikholeslami, G., Chatterjee, S., Zhang, A.: Wavecluster: a multi-resolution clustering approach for very large spatial databases. In: VLDB, vol. 98, New York, pp. 428–439 (1998)
13. Shavlik, J.W., Dietterich, T.G.: Readings in machine learning. Morgan Kaufmann (1990)
14. Kohonen, T.: Self-organized formation of topologically correct feature maps. Biol. Cybern. **43**(1), 59–69 (1982)
15. Ben-Hur, A., et al.: Support vector clustering. J. Mach. Learn. Res. **2**(12), 125–137 (2001)
16. Schölkopf, B., et al.: Estimating the support of a high-dimensional distribution. Neural Comput. **13**(7), 1443–1471 (2001)
17. Ping, L., Chun-Guang, Z., Xu, Z.: Improved support vector clustering. Eng. Appl. Artif. Intell. **23**(4), 552–559 (2010)
18. Yang, J., Estivill-Castro, V., Chalup, S.K.: Support vector clustering through proximity graph modelling. In: 9th International Conference on Neural Information Processing, pp. 898–903. IEEE, Singapore (2002)
19. Lee, J., Lee, D.: An improved cluster labeling method for support vector clustering. IEEE Trans. Pattern Anal. Mach. Intell. **27**(3), 461–464 (2005)
20. Jung, K.H., Lee, D., Lee, J.: Fast support-based clustering method for large-scale problems. Pattern Recogn. **43**(5), 1975–1983 (2010)
21. Lee, S.H., Daniels, K.M.: Gaussian kernel width exploration and cone cluster labeling for support vector clustering. Pattern Anal. Appl. **15**(3), 327–344 (2012)
22. Ping, Y., et al.: Convex decomposition based cluster labeling method for support vector clustering. J. Comput. Sci. Technol. **27**(2), 428–442 (2012)
23. D'Orangeville, V., et al.: Efficient cluster labeling for support vector clustering. IEEE Trans. Knowl. Data Eng. **25**(11), 2494–2506 (2013)
24. Le, T., et al.: Proximity multi-sphere support vector clustering. Neural Comput. Appl. **22**(7–8), 1309–1319 (2013)
25. Saltos, R., Weber, R.: A rough-fuzzy approach for support vector clustering. Inf. Sci. **339**, 353–368 (2016)
26. Li, H.B., Wang, Y., Huang, L., et al.: Clustering algorithm of similarity segmentation based on point sorting. In: The International Conference on Logistics Engineering, Management and Computer Science (2015)
27. Rand, W.M.: Objective criteria for the evaluation of clustering methods. J. Am. Stat. Assoc. **66**(336), 846–850 (1971)
28. Naganathan, G.K., et al.: A prototype on-line AOTF hyperspectral image acquisition system for tenderness assessment of beef carcasses. J. Food Eng. **154**, 1–9 (2015)
29. Shao-Hong, Z., Yang, L., Dong-Qing, X.: Unsupervised evaluation of cluster ensemble solutions. In: 7th International Conference on Advanced Computational Intelligence, pp. 101–106. IEEE, Wuyi (2015)
30. UCI machine learning repository. http://archive.ics.uci.edu/ml