Subspace Clustering Based on Self-organizing Map

Jin Tian(✉) and Mengyi Gu

College of Management and Economics, Tianjin University, Tianjin, China
jtian@tju.edu.cn

Abstract. Clustering in high-dimensional data space is a difficult task due to the interference from different dimensions. A dimension may be relevant for some clusters and irrelevant for other data. Subspace clustering aims at finding local cluster structures in certain related subspace. We propose a novel approach to finding subspace clusters based on the trained Self-Organizing Map neural network (SOM). The proposed method takes advantage of nonlinear mapping of SOM and search for subspace clusters on input neurons instead of the whole data space. Experiment results show that the proposed method performs better compared with original SOM and some traditional subspace clustering algorithms.

Keywords: Self-organizing map · Subspace clustering · High-dimensional clustering

1 Introduction

Clustering is an essential task for data mining and knowledge discovering characterized with unsupervised learning. It aims to explore intrinsic property and structure of data and provide auxiliary information for further analysis. Traditional clustering methods attempt to detect clusters composed of similar samples based on distance measurement. However, advances in information technology and Internet contribute to data with growing dimensionality, which pose great challenge for many traditional data mining methods due to the curse of dimensionality [1]. Large number of dimensions brings similarity of distances originated from sparsity and disturbs from irrelevant dimension, leading to great difficulty in finding meaningful clusters with high quality. Feature selection is a good choice to dimensionality reduction by searching optimum subspace with the most predictive information [2] and helps to improve performance in many models. But sometimes features only work for partial samples and appear as noise for others, which is a more common phenomenon in high dimensional data space. Interference to clustering will generate no matter whether to remove this kind of feature or not.

To deal with such problem, subspace clustering [3] is proposed to provide a new solution where each cluster is assigned with a subspace comprising dimensions relevant to data points in it and irrelevant to others [4]. Clustering determined in subspace reduces the computation cost and provide more targeted information of local structure in given dataset. This new topic addresses much attention from researchers and

G. Q. Huang et al. (eds.), *Proceeding of the 24th International Conference on Industrial Engineering and Engineering Management 2018*,
https://doi.org/10.1007/978-981-13-3402-3_17

different algorithms have been proposed [5–8]. Many applications of subspace clustering are found in the field of computer vision [9], bioinformatics [10], and marketing research [11]. But it still remains a big challenge to discover subspace clusters of high quality with the existence of noise and outliers.

Approaches of subspace clustering are generally divided into three major groups: cell-based, density-based and clustering-oriented [5]. Cell-based approaches, which compose an important branch of subspace clustering, divide data space into grid cells with a certain threshold and search clusters on the cells considering count of data points in these cells. Subspaces are composed of dimensions satisfying some restriction. Discretization of the data accelerates the searching process and makes contribution of efficiency, which is essential for subspace clustering. But the cell is usually in the shape regular square and may lead to information loss. The limitation may finally result in clustering with less accuracy.

Self-organizing map neural network (SOM) is an one kind of unsupervised neural networks with good approximation of the data domain [12]. High dimensional data can be mapped into a plane grid map by SOM. It is able to preserve topological structure in dataset through keeping points with high similarity in original input space close on the map. Each neuron on the map is assigned with a set of points. That is, the input space can be split into cells with arbitrarily shaped data space represented by neurons. Therefore, we proposed a novel cell-based approach called Subspace Clustering Based on Self-Organizing Map (SCBSOM). This method aims to find similar cells with related dimensions using the trained SOM and allows overlapping between clusters. Clustering will be conducted first in each dimension and then a merging procedure is followed. SCBSOM searches for clusters on cells instead of original data points, which is more efficient. And the topological preservation of SOM make contribution to higher accuracy compared with other cell-based methods.

The rest of the paper is organized as follows. Section 2 gives a brief introduction of original SOM algorithm and Sect. 3 describes the proposed SCBSOM model in details. Experimental results are presented in Sect. 4. And finally we conclude major findings in Sect. 5.

2 Self-organizing Map

SOM algorithm is first proposed by Kohonen in 1982 as an unsupervised neutral network [13]. Its good ability of dealing with high dimensional dataset and potential in visualization draw much attention from researchers. SOM has been widely used in many filed such as speech recognition, gene detection, document retrieval and satellite image classification.

An SOM is composed of input layer, output layer and connection weights. The input layer accept training data $\mathrm{x} \in \mathrm{R}^d$ with d neurons while the output layer is often laid out as a plane grid map with $M = m \times m$ neurons. And the weights keep connection from each input neuron to each output neuron and can be denoted as $W = \{w_i | w_i \in \mathrm{R}^d, i = 1, .., M\}$. SOM is trained in an iterative process including competition and convergence. In the tth iterative step, SOM finds the winner in the competition, that is, the closest neuron c to the input sample $\mathrm{x}(t)$:

$$c = \arg\min_i \{\|\mathrm{x}(t) - w_i(t)\|\} \tag{1}$$

Then the convergence procedure leads the SOM model adjusting towards expected order by updating the weight vectors based on the neighborhood relationships with the winner neuron:

$$w_i(t+1) = w_i(t) + h_{ci}(t)(x(t) - w_i(t)) \tag{2}$$

where $h_{ci}(t)$ is the neighborhood function that determines the neighbor update scheme for topology-preserving nature of SOM. The function $h_{ci}(t)$ is usually in the form of Gaussian function:

$$h_{ci}(t) = \alpha(t)\exp\left(\frac{-\mathrm{sqdist}(c,i)}{2\sigma^2(t)}\right) \tag{3}$$

where $\alpha(t)$ is the learning rate that monotonically decreases with step t, $\mathrm{sqdist}(c,i)$ is the square of distance between neuron c and neuron i on the plane grid map and $\sigma(t)$ is the kernel radius that determines the range of neighborhood relationships.

During the training process, the weights adjust according to input until maximum iterations is reached. Each input sample find its winner neuron on the map. Thus every neuron on the output layer contains a set of input samples.

3 Subspace Clustering Based on Self-organizing Map (SCBSOM)

The proposed SCBSOM is a cell-based method that yields valid subspace structure based on the output map instead of on the whole dataset directly. Firstly, with the weight connections in the learned SOM, the proposed algorithm generates possible clusters of each dimension. Then a merging process is conducted to combine the neuron clusters and the corresponding subspaces with related dimensions. The final clustering result is deduced from the neuron clusters by replacing each neuron with data points in it.

3.1 Find One-Dimensional Clusters

Given a trained SOM, we can determine clusters of neurons in each dimension. We link adjacent neurons if their difference (i.e. the distance between their connection weights) is smaller than a certain threshold given in advance. Each cluster is composed of neurons that have links with other neuron in the cluster.

This clustering process is to find all connected component of undirected graph and can be solved with DBSCAN algorithm.

3.2 Merging Procedure

After finding all one-dimensional clusters, a merge process is conducted to obtain subspace clusters of neurons. Here a general method is proposed to merge two subspace clusters.

First, similar clusters with different subspace should be merged. We use Jaccard coefficient to measure the similarity of two clusters, which is computed by

$$J(E,F) = \frac{|E \cap F|}{|E \cup F|} \tag{4}$$

where E and F are two clusters of neuron. Since the two clusters to be merged can hardly be exactly the same, a rule to determine which neuron to remain in the merged cluster is necessary. Here we define the cluster after merging with neurons that appears more times than half of the size of merging subspaces in one-dimensional clusters in the two subspaces. And the cluster is assigned with a new subspace as a union of these dimensions. Then the new merged cluster is added into the result set of subspace clusters and the similar two subspace clusters are removed.

Besides, clusters with the containment relationship also need to merge. The containment relationship of cluster E and F can be measured by

$$\begin{aligned} C_1(E,F) &= \frac{|E \cap F|}{|E|} \\ C_2(E,F) &= \frac{|E \cap F|}{|F|} \end{aligned} \tag{5}$$

If $C_1(E,F)$ is close to 1 and $C_2(E,F)$ gets a smaller value, E is contained in F. The merging procedure is similar to that of similar clusters. The containing subspace cluster is maintained in the result set while the other is removed.

The merging process is executed iteratively by adding subspace clusters of one-dimension each time, which can be summarized as follows:

Step 1: set an empty set.
Step 2: choose one dimension that has not been merged and determine its one-dimensional subspace clusters.
Step 3: choose one cluster from current set of one-dimensional subspace clusters.
Step 4: compare the selected cluster with all subspace clusters in the result set to find if there is a similar cluster to the selected one. If no similar cluster exists, go to Step 7.
Step 5: merge the chosen cluster with its similar one.
Step 6: compare the selected cluster with all subspace clusters in the result set to find if there is one cluster that is contained in it. If there is, merge them. Go to Step 8.
Step 7: add the selected cluster to result set and compare the selected cluster with all subspace clusters in the result set to find if there is one that contains it. If there is, merge them.

Step 8: if all clusters of current dimension is selected, go to Step 9. Otherwise go to Step 3.
Step 9: if all dimensions is merged, return the result set. Otherwise go to Step 2.

3.3 The Clustering Result

After the merging process, we determine all subspace clusters of neurons. As we know, each neuron covers a set of data point according to the mapping of SOM.

As a result, we can obtain the final subspace clustering result through an easy transformation from neurons to data points.

4 Experiments

In our experiments, SCBSOM is compared with original SOM and some state-of-art subspace clustering methods both on the synthetic datasets from OpenSubspace framework [14] and real world datasets from UCI Machine learning Repository [15]. We adjusted parameters for each algorithm separately to obtain a good performance as much as possible. Each experiment is conducted 30 times. F1 measure is used to evaluate the performances of these clustering algorithms.

4.1 Synthetic Datasets

There are three groups in the synthetic datasets to testify the performance of SCBSOM from different aspects.

The first group includes 4 datasets with noise of different proportion. Figure 1 shows the average F1 values on 4 datasets obtained by five algorithms with different noise.

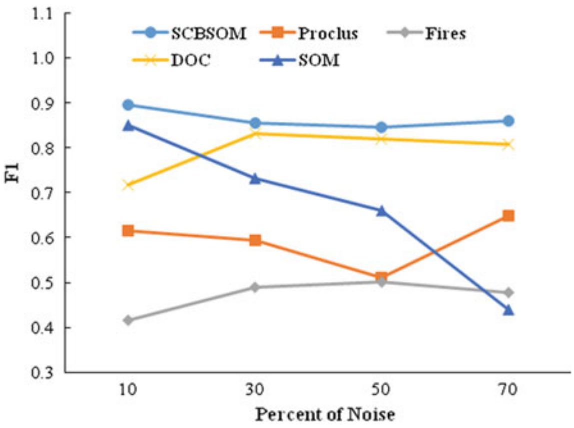

Fig. 1. Average F1 values for SCBSOM and compared algorithms on 4 datasets with different noise

As shown in Fig. 1, SCBSOM outperforms all the other four algorithms and is robust to noise while the original SOM shows poor performance when the amount of noise increases. This advantage of SCBSOM is important since in real world we can hardly obtain a dataset without noise, especially for high dimensional one.

The second group comprises 7 datasets with dimension number ranging from 5 to 75. Figure 2 illustrates the average F1 values on 7 datasets with different dimension numbers.

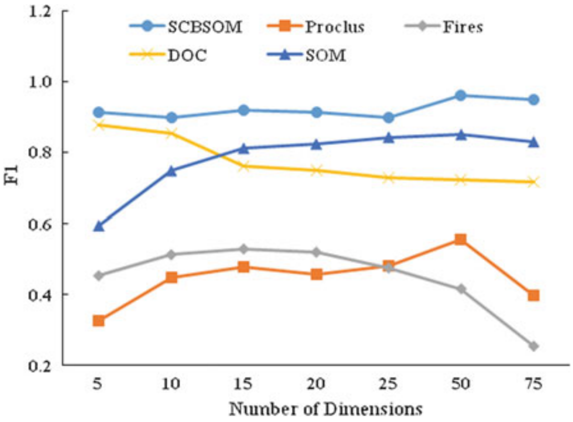

Fig. 2. Average F1 values for SCBSOM and compared algorithms on 7 datasets with different dimension number

According to Fig. 2, SCBSOM performs best among the compared algorithms. We notice that both SOM and SCBSOM get higher values of F1 with the increasing of dimensionality while the other three subspace clustering methods have decreased performance. The good ability of SOM to preserve topological structure of input data ensures the stability of SCBSOM in dimension, which enables the proposed algorithm to deal with high dimensional data with high accuracy.

The third group is 5 datasets of different size that differs from 1500 to 5500. Figure 3 presents the average F1 performances on 5 datasets with different data size.

Shown form Fig. 3, SCBSOM always achieves best clustering performances with both small size of data and large size of data, while other three subspace clustering algorithms are sensitive to the data size.

4.2 Real-World Datasets

In this subsection, the performance of SCBSOM is evaluated on 7 real-world datasets selected from the UCI Machine Learning Repository. Different from synthetic datasets with regular subspace clustering, these datasets are from real word with arbitrary structure. Table 1 shows the F1 performances of SCBSOM and other four compared algorithms. The highest F1 value in each row is outlined in bold.

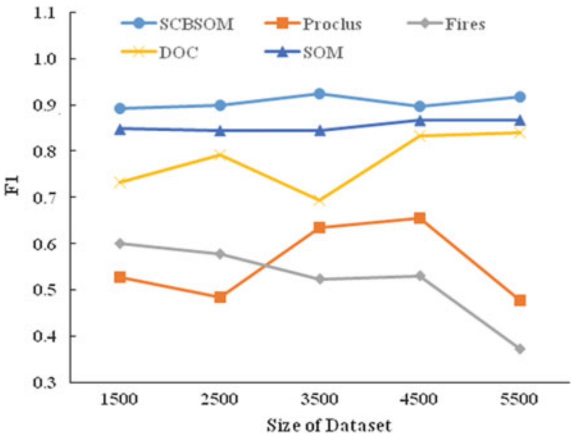

Fig. 3. Average F1 values for SCBSOM and compared algorithms on 5 datasets with different data size

Table 1. F1 values of SCBSOM and other compared algorithms on UCI datasets

Dataset	SCBSOM	SOM	Proclus	Fires	DOC
Breast	**0.7809**	0.4073	0.6639	0.4240	0.5080
Diabetes	**0.7682**	0.4739	0.4988	0.3941	0.4019
Glass	0.4729	0.4973	**0.6351**	0.0874	0.1635
Liver	**0.7125**	0.4003	0.5321	0.3489	0.3875
Shape	0.7162	**0.7211**	0.8005	0.0796	0.3156
Sonar	0.6477	0.4285	**0.8113**	0.3480	0.3480
Vowel	**0.3161**	0.3054	0.4674	0.0152	0.0816
Avg	**0.6306**	0.4620	0.6299	0.2425	0.3152

Shown form Table 1, SCBSOM outperforms other algorithms on Breast, Diabetes and Liver and produces the best clustering results on average. The superiority of SCBSOM on them makes it possible to be applied in finding complex clustering structure.

5 Conclusion

This paper presents a novel method to finding subspace clustering based on SOM called SCBSOM. It first searches for one-dimensional clusters of neurons based on the trained SOM and then a merging procedure is conduct to generated subspace clusters. Finally the neurons in clusters are replaced by corresponding data points determined by SOM map. Since clusters are discovered on neurons of SOM map with smaller size compared with whole input data, SCBSOM can be classified into cell-based approach and find subspace clusters efficiently. The SCBSOM is an extend version of SOM so that it can preserve the properties SOM, such as good ability in clustering. The experimental results show that SCBSOM can perform well with noises, high dimensionality and different size of datasets.

Many feasible extensions can be pursued relative to our work in this paper. We plan to improve iterative process since the results may be affected by the order of selected dimension. Also, a more flexible way to integrate SOM and subspace discovering is expected.

Acknowledgements. The work was supported by the General Program of the National Science Foundation of China (Grant No. 71471127, 71371135).

References

1. M. Köppen, The curse of dimensionality, in *Fifth Online World Conference on Soft Computing in Industrial Applications* (2000)
2. S. Tabakhi, P. Moradi, Relevance–redundancy feature selection based on ant colony optimization. Pattern Recogn. **48**(9), 2798–2811 (2015)
3. R. Agrawal, J.E. Gehrke, D. Gunopulos, P. Raghavan, Automatic subspace clustering of high dimensional data for data mining applications, in *Proceedings of the 1998 ACM SIGMOD* (Seattle, WA, USA), pp. 94–105
4. H.F. Bassani, A.F.R. Araujo, Dimension selective self-organizing maps with time-varying structure for subspace and projected clustering. IEEE Trans. Neural Netw. Learn. Syst. **26** (3), 458–471 (2015)
5. E. Ller, S. Nnemann, I. Assent, T. Seidl, Evaluating clustering in subspace projections of high dimensional data. Proc. VLDB Endow. **2**(1), 1270–1281 (2009)
6. C.M. Procopiuc, M. Jones, P.K. Agarwal, T.M. Murali, A Monte Carlo algorithm for fast projective clustering, in *Proceedings of the 2002 ACM SIGMOD* (Madison, WI, USA), pp. 418–427
7. H.P. Kriegel, P. Kröger, M. Renz, S. Wurst, A generic framework for efficient subspace clustering of high-dimensional data, in *Fifth IEEE International Conference on Data Mining* (Houston, TX, USA, 2005)
8. C.C. Aggarwal, J.L. Wolf, P.S. Yu, C. Procopiuc, J.S. Park, Fast algorithms for projected clustering, in *Proceedings of the 1999 ACM SIGMOD* (Philadelphia, PA, USA), pp. 61–72
9. A.Y. Yang, J. Wright, Y. Ma, S.S. Sastry, Unsupervised segmentation of natural images via lossy data compression. Comput. Vis. Image Underst. **110**(2), 212–225 (2008)
10. D. Jiang, C. Tang, A. Zhang, Cluster analysis for gene expression data: a survey. IEEE Trans. Knowl. Data Eng. **16**(11), 1370–1386 (2004)
11. P.B. Chou, E. Grossman, D. Gunopulos, P. Kamesam, Identifying prospective customers, in *Proceedings of the 2000 ACM SIGKDD* (Boston, MA, USA), pp. 447–456
12. T. Kohonen, Essentials of the self-organizing map. Neural Netw. **37**(1), 52–65 (2013)
13. T. Kohonen, Self-organized formation of topologically correct feature maps. Biol. Cybern. **43**(1), 59–69 (1982)

14. E. Müller, I. Assent, S. Günnemann, T. Seidl, OpenSubspace: an open source framework for evaluation and exploration of subspace clustering algorithms in WEKA, in *Proceedings of 1st Open Source in Data Mining Workshop*, *OSDM'09* (Bangkok, Thailand), pp. 2–13
15. M. Lichman, UCI Machine Learning Repository, http://archive.ics.uci.edu/ml (University of California, School of Information and Computer Science, Irvine, CA, 2013)