A Fuzzy Subspace Algorithm for Clustering High Dimensional Data

Guojun Gan[1], Jianhong Wu[1], and Zijiang Yang[2]

[1] Department of Mathematics and Statistics, York University, Toronto, Ontario, Canada M3J 1P3
{gjgan, wujh}@mathstat.yorku.ca
[2] School of Information Technology, Atkinson Faculty of Liberal and Professional Studies, York University, Toronto, Ontario, Canada, M3J 1P3
zyang@mathstat.yorku.ca

Abstract. In fuzzy clustering algorithms each object has a fuzzy membership associated with each cluster indicating the degree of association of the object to the cluster. Here we present a fuzzy subspace clustering algorithm, FSC, in which each dimension has a weight associated with each cluster indicating the degree of importance of the dimension to the cluster. Using fuzzy techniques for subspace clustering, our algorithm avoids the difficulty of choosing appropriate cluster dimensions for each cluster during the iterations. Our analysis and simulations strongly show that FSC is very efficient and the clustering results produced by FSC are very high in accuracy.

1 Introduction

Data clustering[1] is an unsupervised process that divides a given data set into groups or clusters such that the points within the same cluster are more similar than points across different clusters. Data clustering is a primary tool of data mining, a process of exploration and analysis of large amount of data in order to discover useful information, thus has found applications in many areas such as text mining, pattern recognition, gene expressions, customer segmentations, image processing, to name just a few.

For data sets in high dimensional spaces, most of the conventional clustering algorithms do not work well in terms of effectiveness and efficiency, because of the inherent sparsity of high dimensional data [2]. To cluster data in high dimensional spaces, we encounter several problems. First of all, the distance between any two points becomes almost the same [2], therefore it is difficult to differentiate similar data points from dissimilar ones. Secondly, clusters are embedded in the subspaces of the high dimensional space, and different clusters may exist in different subspaces of different dimensions [3]. Because of these problems, almost all conventional clustering algorithms fail to work well for high dimensional data sets. One possible solution is to use dimension reduction techniques such as PCA(Principal Component Analysis) and Karhunen-Loève Transformation, or feature selection techniques [3].

X. Li, O.R. Zaiane, and Z. Li (Eds.): ADMA 2006, LNAI 4093, pp. 271–278, 2006.

The idea behind dimension reduction approaches and feature selection approaches is to first reduce the dimensionality of the original data set by removing less important variables or by transforming the original data set into one in a low dimensional space, and then apply conventional clustering algorithms to cluster the new data set. In either dimension reduction approaches or feature selection approaches, it is necessary to prune off some variables, which may lead to a significant loss of information. This can be illustrated by considering a 3-dimensional data set that has 3 clusters: one is embedded in (x, y)-plane, another is embedded in (y, z)-plane and the third one is embedded in (z, x)-plane. For such a data set, an application of a dimension reduction or a feature selection method is unable to recover all the cluster structures, for the 3 clusters are formed in different subspaces. In general, clustering algorithms based on dimension reduction or feature selection techniques generate clusters that may not fully reflect the original cluster structures.

This difficulty that conventional clustering algorithms encounter in dealing with high dimensional data sets inspired the invention of subspace clustering algorithms or projected clustering algorithms [3] whose goal is to find clusters embedded in subspaces of the original data space with their own associated dimensions. Some subspace clustering algorithms are designed to identify arbitrarily oriented subspace clusters (e.g. ORCLUS and Projective k-Means) whose cluster dimensions are linear combinations of the original dimensions, while others are designed to discover regular subspace clusters (e.g. PART and SUBCAD) whose cluster dimensions are elements of the set of the original dimensions.

However, almost all of the subspace clustering algorithms give equal non-zero weights to cluster dimensions and zero weights to non-cluster dimensions. Consider a cluster embedded in a 50-dimensional subspace of a 100-dimensional data set, for example, the cluster dimensions (say 1,2,...,50) found by PROCLUS [4] are assumed to have equal contributions to the cluster, but other dimensions (51,52,...,100) are assumed to have zero contributions to the cluster. This practice leads to the problem of how to choose the cluster dimensions of a specific cluster.

Motivated by fuzzy clustering and LAC [5], we propose a fuzzy subspace clustering algorithm, FSC, to cluster high dimensional data sets. FSC finds regular subspace clusters with each dimension of the original data being associated with each cluster with a weight. The higher density of a cluster in a dimension, the more weight will be assigned to that dimension. In other words, all dimensions of the original data are associated with each cluster, but they have different degrees of association with that cluster.

2 Related Work

The recent subspace clustering algorithms can be roughly classified into three categories: Grid-based algorithms such as CLIQUE [3], MAFIA [6], Partitioning and/or hierarchical algorithms such as ORCLUS [7], FINDIT [8], and Neural Network-based algorithms such as PART [2].

CLIQUE [3] first partitions the whole data space into non-overlapping rectangular units, and then searches for dense units and merges them to form clusters. The subspace clustering is achieved due to the fact that if a k-dimension unit $(a_1, b_1) \times (a_2, b_2) \times \cdots \times (a_k, b_k)$ is dense, then any $(k-1)$-dimension unit $(a_{i_1}, b_{i_1}) \times (a_{i_2}, b_{i_2}) \times \cdots \times (a_{i_{k-1}}, b_{i_{k-1}})$ is also dense, where (a_i, b_i) is the interval of the unit in the i-th dimension, $1 \leq i_1 < i_2 < \cdots < i_{k-1} \leq k$. ENCLUS [9] and MAFIA [6] are also Grid-based subspace clustering algorithms.

PROCLUS [4] is a variation of k-Medoid algorithm [10] for subspace clustering. PROCLUS finds out the subspace dimensions of each cluster via a process of evaluating the locality of the space near it. FINDIT [8], ORCLUS [7], FLOC [11], DOC [12], SUBCAD [13] and projective k-Means [14] are also partitioning subspace clustering algorithms.

PART [2] is a new neural network architecture to find projected clusters for data sets in high dimensional spaces. In PART, a so-called selective output signaling mechanism is provided in order to deal with the inherent sparsity in the full space of the high dimensional data points. PART is very effective to find the subspace in which a cluster is embedded, but the difficulty of tuning some parameters in the algorithm and the sensitivity to data input order restrict its application. CLTree [15] is an algorithm for clustering numerical data based on a supervised learning technique called decision tree construction. The resulting clusters found by CLTree are described in terms of hyper-rectangle regions. The CLTree algorithm is able to separate outliers from real clusters effectively, since it naturally identifies sparse and dense regions.

LAC [5] defines subspace clusters as weighted clusters such that each cluster consists of a subset of data points together with a vector of weights. To be precise, let us consider a data set D of n points in the d-dimensional Euclidean space and a set of centers $\{\mathbf{z}_1, \mathbf{z}_2, ..., \mathbf{z}_k\} \subset \mathbb{R}^d$, couped with a set of corresponding weight vectors $\{\mathbf{w}_1, \mathbf{w}_2, ..., \mathbf{w}_k\} \subset \mathbb{R}^d$. LAC defines the jth $(1 \leq j \leq k)$ cluster as $C_j = \left\{ \mathbf{x} \in D : \left(\sum_{i=1}^{d} w_{ji}(x_i - z_{ji})^2 \right)^{\frac{1}{2}} < \left(\sum_{i=1}^{d} w_{li}(x_i - z_{li})^2 \right)^{\frac{1}{2}}, \forall l \neq j \right\}$, where x_i, z_{ji} and w_{ji} are the ith components of $\mathbf{x}$, $\mathbf{z}_j$ and $\mathbf{w}_j$, respectively. The centers and weights are chose such that the error measure, $E = \sum_{j=1}^{k} \sum_{i=1}^{d} w_{ji} e^{-X_{ji}}$, is minimized subject to the constraints $\sum_{i=1}^{d} w_{ji}^2 = 1$, $\forall j$, where $X_{ji} = \frac{1}{|C_j|} \sum_{\mathbf{x} \in C_j} (x_i - z_{ji})^2$.

3 Fuzzy Subspace Clustering Algorithm

The main idea behind our algorithm is to impose weights to the distance measure of the k-Means algorithm[16] in order to capture appropriate subspace information. Given a data set $D = \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_n\}$ in the d-dimensional Euclidean space and k centers $Z = \{\mathbf{z}_1, \mathbf{z}_2, ..., \mathbf{z}_k\}$, then the objective function of the k-Means

algorithm is formulated as $E = \sum_{j=1}^{k} \sum_{\mathbf{x} \in C_j} \|\mathbf{x} - \mathbf{z}_j\|^2$, where $\|\cdot\|$ is the Euclidean norm and C_j is the jth cluster.

Similar to LAC [5], we associate with each cluster a weight vector in order to capture the subspace information of that cluster. To be more precise, let W be a $k \times d$ real matrix satisfying the following conditions:

$$0 \leq w_{jh} \leq 1, \quad 1 \leq j \leq k, \quad 1 \leq h \leq d, \tag{1a}$$

$$\sum_{h=1}^{d} w_{jh} = 1, \quad 1 \leq j \leq k. \tag{1b}$$

Then the hth dimension is associated with the jth cluster to a degree of w_{jh} or the jth cluster has dimension weights specified by $w_{j1}, w_{j2}, ..., w_{jd}$. We call the weight matrix W the fuzzy dimension weight matrix. Mathematically, the objective function of our algorithm is formated as

$$E_f(W, Z) = \sum_{j=1}^{k} \sum_{\mathbf{x} \in C_j} \sum_{h=1}^{d} w_{jh}^{\alpha} (x_h - z_{jh})^2, \tag{2}$$

where $\alpha \in (1, \infty)$ is a weighting component or fuzzier. Given the estimates of Z and W, the jth cluster are formulated as

$$C_j = \{\mathbf{x} \in D : \sum_{h=1}^{d} w_{jh}^{\alpha} (x_h - z_{jh})^2 = \min_{1 \leq l \leq k} \sum_{h=1}^{d} w_{lh}^{\alpha} (x_h - z_{lh})^2\}, \tag{3}$$

together with the fuzzy dimension weights $\mathbf{w}_j = (w_{j1}, w_{j2}, ..., w_{jd})$.

To find the cluster centers Z given the estimate of W such that the objective function $E_f(W, Z)$ defined in Equation (2) is minimized, we take partial derivatives of $E_f(W, Z)$ with respect to z_{jh}s, set them to zeros and solve the resulting equation system. That is,

$$\frac{\partial E_f(W, Z)}{\partial z_{jh}} = \sum_{\mathbf{x} \in C_j} -2 w_{jh}^{\alpha} (x_h - z_{jh}) = 0, \quad 1 \leq j \leq k, \, 1 \leq h \leq d,$$

which give

$$z_{jh} = \frac{\sum\limits_{\mathbf{x} \in C_j} w_{jh}^{\alpha} x_h}{\sum\limits_{\mathbf{x} \in C_j} w_{jh}^{\alpha}} = \frac{\sum\limits_{\mathbf{x} \in C_j} x_h}{|C_j|}, \quad 1 \leq j \leq k, \, 1 \leq h \leq d, \tag{4}$$

where $|C_j|$ denotes the number of points in C_j.

To find the fuzzy dimension weight matrix W given the estimate of Z such that the objective function $E_f(W, Z)$ is minimized, we use the method of Lagrange multipliers. To do this, we first write the Lagrangian function as

$$F(W, Z, \Lambda) = E_f(W, Z) - \sum_{j=1}^{k} \lambda_j \left(\sum_{h=1}^{d} w_{jh} - 1 \right).$$

By taking partial derivatives, we have

$$\frac{\partial F(W,Z,\Lambda)}{\partial w_{jh}} = \sum_{\mathbf{x}\in C_j} \alpha w_{jh}^{\alpha-1}(x_h - z_{jh})^2 - \lambda_j = 0, \quad 1 \le j \le k,\ 1 \le h \le d,$$

and

$$\frac{\partial F(W,Z,\Lambda)}{\partial \lambda_j} = \sum_{h=1}^{d} w_{jh} - 1 = 0, \quad 1 \le j \le k,$$

which, with some simple manipulations, leads to

$$w_{jh} = \frac{1}{\sum\limits_{l=1}^{d} \left[\frac{\sum\limits_{\mathbf{x}\in C_j}(x_h - z_{jh})^2}{\sum\limits_{\mathbf{x}\in C_j}(x_l - z_{jl})^2}\right]^{\frac{1}{\alpha-1}}}, \quad 1 \le j \le k,\ 1 \le h \le d. \tag{5}$$

To avoid divide-by-zero error, we introduce a small bias ϵ (say $\epsilon = 0.0001$) in Equation (5). That is, we update W given the estimate of Z as follows:

$$w_{jh} = \frac{1}{\sum\limits_{l=1}^{d} \left[\frac{V_{jh}+\epsilon}{V_{jl}+\epsilon}\right]^{\frac{1}{\alpha-1}}}, \quad 1 \le j \le k,\ 1 \le h \le d, \tag{6}$$

where $V_{jh} = \sum\limits_{\mathbf{x}\in C_j}(x_h - z_{jh})^2$ for $1 \le j \le k$ and $1 \le h \le d$.

We see from Equation (4) and Equation (6) that FSC is very similar to the fuzzy k-Means algorithm [17] in terms of the way they update centers and fuzzy weights. FSC starts with initial centers Z, and then repeats estimating the fuzzy dimension weight matrix W given the estimate of Z and estimating the centers Z given the estimate of W until it converges.

4 Experimental Evaluations

FSC is coded in C++ programming language. Synthetic data sets are generated by a Matlab program using the method introduced by Aggarwal et al. [4]. In our experiments, we specify $\alpha = 2.1$.

Our first data set contains 300 3-dimensional points with 3 clusters embedded in different planes. We run FSC 100 times on this data with $k = 3$ and get the

Table 1. FSC: The input clusters (left) and the output fuzzy dimension weights together with the cluster dimensions (right) for the first data set

Input	Dimensions	Points
A	1,2	100
B	2,3	100
C	3,1	100

Found	w_{i1}	w_{i2}	w_{i3}	Dimensions	Points
1	0.040034	0.444520	0.515446	2, 3	100
2	0.573635	0.391231	0.035134	2, 1	100
3	0.427178	0.036284	0.536538	1, 3	100

same result. The best $E_f(W, Z)$ and average $E_f(W, Z)$ of the 100 runs are identical to 10.691746. Table 1 summarizes the clustering results. We see from Table 1 that FSC is capable of clustering each object correctly and at the same time identifying the true subspaces for each cluster. Note that the cluster dimensions of a cluster are arranged in ascending order according to their weights and the cutting point is obtained by clustering the fuzzy dimension weights of the cluster into 2 groups by k-Means.

Table 2. FSC: Dimensions of input clusters (left) and output clusters (right) for the second data set

Input	Dimensions	Points
A	6,7,8,10,11	387
B	5,7,8,10,11,12,13,16	87
C	3,5,6,10,12,13	317
D	1,3,6,9,10,11,13,18	80
E	1,10,13,20	129

Found	Dimensions	Points
1	20,13,10,1	129
2	9,3,1,6,10,13,11,18	80
3	3,5,12,6,13,10	317
4	11,6,10,7,8	387
5	7,16,11	87

Our second data set contains 1,000 20-dimensional points with 5 clusters embedded in different subspaces of different dimensions (See Table 2). We also run FSC 100 times on this data set with $k = 5$. The best $E_f(W, Z)$ and the average $E_f(W, Z)$ are 1102.126302 and 1396.434035, respectively. In particular, the number of correct clusterings is 49 out of 100. The best output is given in Table 2 from which we see that in the best case all subspace clusters are recovered by FSC except for cluster B where k-Means gives only 3 cluster dimensions.

Table 3. FSC: Dimensions of input clusters for the third data set

Input	Dimensions	Points
A	8,17,27,46,48,52,56,57,68,71,76,80,89,93	1462
B	5,8,17,26,27,37,40,46,48,53,56,71,84,86,89,95,97	4406
C	7,9,17,26,41,46,65,73,84,86,97	1415
D	4,17,25,26,45,65,75,83,84,97	556
E	2,6,17,18,26,29,32,39,45,49,75,83,84,97	1661
Outlier		500

Our third data set has 10,000 100-dimensional points with 5 clusters embedded in different subspaces of different dimensions and contains 500 outliers. We run FSC on this data set 5 times with $k = 5$ and 5 times with $k = 6$. The results are given in Table 4 from which we see that all objects are clustered correctly in both cases and all outliers are differentiated from real clusters in the case of $k = 6$. In the case of $k = 5$, the best $E_f(W, Z)$ and the average $E_f(W, Z)$ are 3328.104712 and 4460.156128, respectively, and the number of correct clusterings is 2 out of 5, while in the case of $k = 6$, the best $E_f(W, Z)$ and the average $E_f(W, Z)$ are 6102.280185 and 7703.287459, respectively, and the number of correct clusterings is 3 out of 5. We also see from Table 4 that cluster 1 has the

Table 4. FSC: The misclassification matrices when $k = 5$ (top left) and $k = 6$ (top right), output clusters when $k = 5$ (middle) and $k = 6$ (bottom) for the third data set

	1	2	3	4	5
A	0	0	0	1462	0
B	4406	0	0	0	0
C	0	1415	0	0	0
D	0	0	0	0	556
E	0	0	1661	0	0
Outlier	0	0	0	0	500

	1	2	3	4	5	6
A	0	1462	0	0	0	0
B	0	0	0	4406	0	0
C	0	0	0	0	1415	0
D	0	0	0	0	0	556
E	0	0	1661	0	0	0
Ourlier	500	0	0	0	0	0

Found	Dimensions	Points
1	48,56,53,17,5,46,95,86,26,84,40,97	4460
2	73,46,41,86,84,26,97,17,7,65	1415
3	26,84,17,32,39,49,6,18,83,29,75,45,2,97	1661
4	76,89,71,27,56,52,68,8,46,17,57,93,80,48	1462
5	83,25,45,4,65,97	1056

Found	Dimensions	Points
1	30,22,35,7,16,11,73,100,2,33,39,10,53,62,34,12,45,9,76,54,85, 61,47,82,65,20,14,43,94,77,99,41,70,96,74,23,68,59,19,50,71,92, 57,26,32, 3,15,51,98,37,80,79,84,49	500
2	76,89,71,27,56,52,68,8,46,17,57,93,80,48	1462
3	26,84,17,32,39,49,6,18,83,29,75,45,2,97	1661
4	48,56,53,17,5,46,95,86,26,84,40,97	4406
5	73,46,41,86,84,26,97,17,7,65	1415
6	65,26,17,83,4,75,84,25	556

number of cluster dimensions significantly greater than other clusters do. This indicates that cluster 1 may be an outlier cluster.

The experiments presented above show that FSC is very powerful in recovering clusters embedded in subspaces of high dimensional spaces. FSC is simple and natural in terms of the presentation of the algorithm, and it is much easier to use than other subspace clustering algorithms such as PART and PROCLUS.

5 Conclusions and Remarks

We presented the fuzzy subspace clustering algorithm FSC for clustering high dimensional data sets. The novel contribution is the adoption of some fuzzy techniques for subspace clustering in a way that each dimension has a fuzzy dimension weight associated with each cluster. The experimental results have shown that FSC is very effective in recovering the subspace cluster structures embedded in high dimensional data. It is certainly of great interest to us if we can adopt fuzzy techniques for identifying arbitrarily oriented subspace clusters in high dimensional data.

References

[1] Jain, A., Murty, M., Flynn, P.: Data clustering: A review. ACM Computing Surveys **31** (1999) 264–323

[2] Cao, Y., Wu, J.: Projective ART for clustering data sets in high dimensional spaces. Neural Networks **15** (2002) 105–120

[3] Agrawal, R., Gehrke, J., Gunopulos, D., Raghavan, P.: Automatic subspace clustering of high dimensional data for data mining applications. In: SIGMOD Record ACM Special Interest Group on Management of Data. (1998) 94–105

[4] Aggarwal, C., Wolf, J., Yu, P., Procopiuc, C., Park, J.: Fast algorithms for projected clustering. In: Proceedings of the 1999 ACM SIGMOD international conference on Management of data, ACM Press (1999) 61–72

[5] Domeniconi, C., Papadopoulos, D., Gunopulos, D., Ma, S.: Subspace clustering of high dimensonal data. In: Proceedings of the SIAM International Conference on Data Mining, Lake Buena Vista, Florida (2004)

[6] Goil, S., Nagesh, H., Choudhary, A.: MAFIA: Efficient and scalable subspace clustering for very large data sets. Technical Report CPDC-TR-9906-010, Center for Parallel and Distributed Computing, Department of Electrical & Computer Engineering, Northwestern University (1999)

[7] Aggarwal, C., Yu, P.: Finding generalized projected clusters in high dimensional spaces. In Chen, W., Naughton, J.F., Bernstein, P.A., eds.: Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data, May 16-18, 2000, Dallas, Texas, USA. Volume 29., ACM (2000) 70–81

[8] Woo, K., Lee, J.: FINDIT: a fast and intelligent subspace clustering algorithm using dimension voting. PhD thesis, Korea Advanced Institue of Science and Technology, Department of Electrical Engineering and Computer Science (2002)

[9] Cheng, C., Fu, A., Zhang, Y.: Entropy-based subspace clustering for mining numerical data. In: Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining, ACM Press (1999) 84–93

[10] Kaufman, L., Rousseeuw, P.: Finding Groups in Data–An Introduction to Cluster Analysis. Wiley series in probability and mathematical statistics. John Wiley & Sons, Inc., New York (1990)

[11] Yang, J., Wang, W., Wang, H., Yu, P.: δ-clusters: capturing subspace correlation in a large data set. Data Engineering, 2002. Proceedings. 18th International Conference on (2002) 517 –528

[12] Procopiuc, C., Jones, M., Agarwal, P., Murali, T.: A monte carlo algorithm for fast projective clustering. In: Proceedings of the 2002 ACM SIGMOD international conference on Management of data, ACM Press (2002) 418–427

[13] Gan, G., Wu, J.: Subspace clustering for high dimensional categorical data. ACM SIGKDD Explorations Newsletter **6** (2004) 87–94

[14] Agarwal, P., Mustafa, N.: k-means projective clustering. In: Proceedings of the Twenty-third ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems(PODS), Paris, France, ACM (2004) 155–165

[15] Liu, B., Xia, Y., Yu, P.: Clustering through decision tree construction. In: Proceedings of the ninth international conference on Information and knowledge management, McLean, Virginia, USA, ACM Press (2000) 20–29

[16] Hartigan, J.: Clustering Algorithms. John Wiley & Sons, Toronto (1975)

[17] Huang, Z., Ng, M.: A fuzzy k-modes algorithm for clustering categorical data. IEEE Transactions on Fuzzy Systems **7** (1999) 446–452