www.ijecs.in
International Journal Of Engineering And Computer Science ISSN: 2319-7242
Volume 5 Issue 10 Oct. 2016, Page No. 18399-18403

An Efficient Method for Finding Closed Subspace Clusters for High Dimensional Data

S. Anuradha[1], K.B. Madhuri[2], B. Jaya Lakshmi[3]

[1]Gayatri Vidya Parishad College of Engineering,
Visakhapatnam, Andhra Pradhesh, India
anuradha.sanapala@gmail.com

[2]Gayatri Vidya Parishad College of Engineering,
Visakhapatnam, Andhra Pradhesh, India
kbmcst1@yahoo.com

[3]Gayatri Vidya Parishad College of Engineering,
Visakhapatnam, Andhra Pradhesh, India
mdjayadas@gmail.com

Abstract: ***Subspace clustering tries to find groups of similar objects from the given dataset such that the objects are projected on only a subset of the feature space. It finds meaningful clusters in all possible subspaces. However, when it comes to the quality of the resultant subspace clusters most of the subspace clusters are redundant. These redundant subspace clusters don't provide new information. Hence there is a need for eliminating such redundant subspace clusters and output only those subspace clusters which are non redundant and each of them contributing some unique information to the data miner. The set of non redundant subspace clusters is helpful for easy analysis. In order to accomplish this, the concept of closedness has been applied to the subspace clusters. An algorithm known as Finding Closed Subspace Clusters (FCSC) is presented which efficiently outputs the closed subspace clusters from a given set of subspace clusters produced from any subspace clustering algorithm. Based on the experimental study conducted, the number of clusters generated by FCSC has been greatly reduced when compared to the existing SUBCLU algorithm and the average purity of the clusters is marginally improved without loss of coverage.***

Keywords: Curse of dimensionality, Subspace Clustering, Redundant Clusters, Closed Clusters.

1. Introduction

Clustering is the process of finding similar object sets from the given dataset. Most of the traditional clustering algorithms try to find clusters in the whole feature space. These algorithms work well if the number of dimensions is fewer. However, as the number of dimensions increases the similarity measures such as distance between objects become meaningless. Usually the objects are sparsely distributed when the number of dimensions increases. This difficulty is referred to as the 'curse of dimensionality'. Hence the concept of subspace clustering has been evolved to resolve the dimensionality issue. Subspace clustering tries to find meaningful clusters in subsets of the whole feature space.

Despite the fact that the subspace clustering resolves the issues of the traditional clustering for high dimensional data, there is a drawback associated with it. Subspace clustering generates numerous results and most of them are redundant. This paper proposes a new concept called closed clusters. The closed clusters eliminate the problem of redundancy by producing only the non redundant subspace clusters from the set of results generated by any subspace clustering algorithm. From the results generated by any subspace clustering algorithm FCSC efficiently produces the non redundant closed subspace clusters.

2. Related Work

Subspace clustering finds clusters from a subset of attributes. There exist many algorithms which rely upon the density of objects to form clusters. The density-connected subspace clustering finds clusters based on the notion of density connectedness [1]. An object o is said to be core object if it contains at least minpts number of objects in its surrounding area of influence where minpts is specified by the user. An object o_1 is said to be directly density reachable from o_2 if o_1 is a neighbour of o_2 and o_2 is a core object. An object o_1 is said to be density reachable from o_2 if there exist a set of objects x_1 to x_n such that x_1 is o_2 and x_n is o_1 and x_{i+1} is directly density reachable from x_i. Two objects o_1 and o_2 are said to be density connected if there exist a third object o_3 such that both o_1 and o_2 are density reachable from o_3. The density connected subspace clustering algorithm connects all density connected points and thus forms clusters [1]. However most of the subspace clusters resulted from this algorithm are redundant.

There are some other algorithms which eliminate the redundant clusters both locally and globally [2]. These approaches suffer from the high computational cost of generating subspace clusters. Instead of generating all the subspace clusters, from the set of lower dimensional projections the clusters in higher dimensions are approximated [3]. Initially the set of one dimensional subspace clusters are formed by using any traditional clustering algorithm. Then the set of two dimensional subspace clusters are computed by taking the intersection of elements in the clusters of two different one dimensional spaces. These lower dimensional clusters are organized into k groups. Finally k representatives are selected one from each group. There is a drawback with this approach. There may be some loss in the coverage of objects. For finding maximal clusters in high dimensional data

SUBSCALE algorithm gives the density reachable sets by maintaining the signature of all elements in each dimension [4].

Similar to the subspace clustering, in pattern mining also there are quite a few approaches for reducing the number of itemsets. One of such approaches discovers frequent itemsets based on closed frequent itesmset framework [5]. With the discovered frequent closed frequent itemsets, association rules are generated. Hence generating association rules from a database reduces to generating closed frequent itemsets. Given a dataset with n transactions and d items, the set of items that repeat a minimum number of times in the dataset are called frequent. If the number of items in an itemset is k, the itemset is called k-itemset. The support of an itemset is defined as the percentage of transactions in which the itemset is present in the dataset. From a given dataset first all the frequent itemsets are determined along with their supports. For the obtained frequent itmesets the corresponding association rules are generated. By mining the association rules from the closed frequent itemsets, the number of rules generated reduces greatly.

Another work related to reducing the set of frequent itemsets is approximating a set of frequent sets [6]. From the given set of frequent itemsets, a set of k itemsets are selected such that this set best approximates the given set of frequent itemsets. Out of these k sets the approximated set of frequent itemsets is coverd as one of the k sets. The other sets are also covered which do not contribute the output. This is called the bound constraint. From a collection of frequent itemsets, a set of k frequent itemsets that approximate the entire collection can give a valuable knowledge to the data miner without loss of much information. The frequency threshold values of frequent itemsets are arbitrary. Hence an approximate collection of itemsets can better describe the given set of frequent itemsets.

The problem of finding k sets is similar to the Max k-Cover problem. It is as follows. From a given number of sets, the k sets must be selected such that they cover maximum elements from the collection. It follows a greedy approach. It first marks all the elements as unprocessed. It proceeds iteratively. It adds a set after iteration and removes the same from unprocessed list. After k iterations the k sets will be obtained.

The other method for reducing the number of frequent itemsets is compressing the frequent itemsets [7]. In this method the set of frequent itemsets are clustered based on a closeness measure. Then a representative pattern is selected from each cluster. The distance between two closed frequent patterns is calculated as the ratio of intersection of two itemsets to the union of the sets subtracted from 1. When can one say a frequent pattern can represent another frequent pattern? A frequent pattern fp_1 is said to be represented by another frequent pattern fp_2 if all the items in fp_1 are covered in fp_2. If a cluster consists of n frequent items a representative frequent itemset is selected such that the items of all other frequent itemsets in the cluster are covered in the representative frequent itemset. Once the frequent itemsets are clustered, a representative frequent pattern is selected from each cluster. To find the representative frequent pattern from a given set of frequent itemsets there are two methods; one is RPglobal method and another one is RPlocal method [7].

The first method follows a greedy approach in selecting the items for the representative pattern. From the given set of frequent patterns the frequent patterns that satisfy the minimum support are placed in a set E. For each pattern in the original set, if it covers another pattern the latter is inserted into the former. The process of adding the representatives to the output and removing them from the set E continues. Finally it returns the k frequent patterns that represent the entire collection of patterns. The second method for finding representative patterns is based on a closed pruning criterion.

There is another way of summarizing the itemsets by presenting only the itemsets for which the support counts cannot be derived directly [8]. Apart from finding closed frequent patterns, there is another method for mining closed sequential patterns which uses occurrence checking as the pruning criterion [9]. Further, an algorithm finds maximal sequential patterns by defining the equivalent neighbours approximately with the help of edit distance [10].

These methods suffer from the loss of coverage of objects. Hence the concept of closedness has been adapted to subspace clustering so that it finds the non redundant closed clusters without loss of coverage.

3. Problem Statement

Given a dataset with n transactions Tr_1, Tr_2... Tr_n where each transaction Tr_i consists of items It_1... It_k. Here $k \subseteq I$ (the total number of items in the dataset). A set of items that occur concurrently in the transactions are called itemsets. The support of an itemset is defined as the number of transactions in which the itemset occurs within the whole dataset. An itemset is called frequent if the support of that itemset is greater than or equal to minimum support.

Definition 1. **(Frequent Pattern):** A frequent pattern is defined as a pattern such as an itemset or a subsequence that appear frequently in the transactions of a dataset.

Definition 2. **(Closed Frequent Pattern):** A frequent pattern FP is said to be closed if there does not exist another frequent pattern FP' within the same dataset such that FP'⊃FP and *sup* (FP') = *sup* (FP).

There is a close association between pattern mining and subspace clustering [11], [12].

Given a dataset with object set O = {o_1, o_2... o_n} and dimension set D = {a_1, a_2... a_d}, subspace clustering aims at finding set of similar objects which are projected only on a subset of attributes.

Definition 3. **(Subspace Cluster):** A subspace cluster <M, N> is defined as a set of objects M with a set of attributes N such that $M \subseteq O$ and $N \subseteq D$ and any two objects o_i, $o_j \in M \; \forall$ i, j are similar with respect to the similarity measure used for clustering.

Subspace clustering yields numerous results out of which only few clusters give unique information and most of the other subspace clusters share almost equivalent information as the unique ones shares. It is termed as the notion of redundancy [13].

Definition 4. **(Redundant Subspace Cluster):** A subspace cluster <M, N> is said to be redundant if there exist another subspace cluster <M', N'> such that M = M' and N' ⊃ N.

There is an important aspect in Definition 4 that is to be looked at. When both M and M' are equal, the set of objects from a lower dimension (N) are considered as redundant because a cluster from the high dimensional subspace provides more information.

The concept of closedness of frequent patterns has been applied to the subspace clustering and hence a closed subspace cluster can be defined as follows:

Definition 5. **(Closed Subspace Clusters):** A subspace cluster <M, N> is said to be closed if there does not exist a subspace cluster <M', N'> such that M = M' and N' ⊃ N.

Hence the problem of finding closed frequent patterns can be transformed into the problem of mining closed subspace clusters. From the given set of subspace clusters S = {<M, N>}, the problem is to find the set of closed subspace clusters where each of resultant cluster satisfies the condition in Definition 5.

4. Mining Closed Clusters

4.1. Finding Closed Subspace Clusters (FCSC) Algorithm

Figure 1 depicts the algorithm for efficiently finding closed subspace clusters. Step 1 marks all the subspace clusters as unvisited. Step 2 marks all subspace clusters as closed. It then repeatedly checks for each subspace cluster whether it is closed or not. The algorithm searches for a match in any of the higher dimensional spaces.

Algorithm: Finding Closed Subspace Clusters (FCSC)
Input: Set of subspace clusters S = {<M, N>}
Output: Set of closed subspace clusters S' = {<M', N'>}
Method:
(1) mark the **vStatus** of all subspace clusters to **false**;
(2) mark the **closedStatus** of all subspace clusters to **true**;
(3) **do**
(4) **for each** subspace cluster <M, N> ∈ S
(5) **if** (<M, N>. **vStatus** is **false**)
(6) change the vStatus of <M, N> to true;
(7) **for each** subspace clusters {<M', N'> ∈ S
(8) **if** (N' ⊃ N and M = M')
(9) place all subspace clusters <M_i, N> into a set T and <M_j', N'> into a set U;
(10) flag = **checkIfClosed (T, U);**
(11) **if** (flag = true)
(12) **for each** <M, N> in T
(13) <M, N>. **closedStatus = false;**
(14) **break;**
(15) **end for**
(16) **end if**
(17) **end if**
(18) **end for**
(19) **for each** <M, N> in T
(20) **if** (<M, N>. **closedStatus = false)**
(21) <M_i, N>. **vStatus = true;**
(22) **end if**
(23) **end for**
(24) **end if**
(25) **end for**
(26) **until** the **vStatus** of all subspace clusters becomes true;
(27) **for each** <M, N> in S
(28) **if** (<M, N>. **closedStatus=true)**
(29) place <M, N> in S'
(30) **end if**
(31) **end for**
(32) output S'

Figure 1: Algorithm FCSC

Procedure **checkIfClosed (T, U)**

(1) initialise clusterCount=0;
(2) **if** (T.count = U.count)
(3) **for each** <M', N'> in T
(4) **for each** <M', N'> in U
(5) **if** (M = M')
(6) increment clusterCount;
(7) **end if**
(8) **end for**
(9) **end for**
(10) **if** (clusterCount = T.count)
(11) **return true**;
(12) **else**
(13) **return false**;

Figure 2: Procedure CheckIfClosed

If a match occurs, in step 10 the algorithm calls another procedure checkIfClosed. This procedure checks whether all the clusters in both subspaces are same. In order to accomplish this, it first checks whether the number of clusters in both subspaces are same. If it is same, it then proceeds further by checking if all the clusters are same in both projections. If this routine returns true, it means that for the subspace cluster which is under process there exist a higher dimensional space with the same set of clusters. By the definition of closed subspace cluster there should not exist any higher dimensional space such that the set of clusters are same. Hence the set of subspace clusters which belong to the subspace which is in process are all marked as not closed. And all the remaining subspace clusters in the higher spaces need not be checked and hence the break statement is called. This process repeats until all the subspace clusters have been processed. Steps 3 through 26 explain this. After all the clusters are processed, the closed clusters are outputted which is shown in steps 27 through 32. By the time the algorithm terminates all the non closed clusters would have been identified. The clusters for which the closed status is not altered will be the closed clusters and hence would contribute to the output.

Finally the FCSC algorithm efficiently outputs the set of non redundant and closed clusters.

4.2. Architectural Flow Diagram

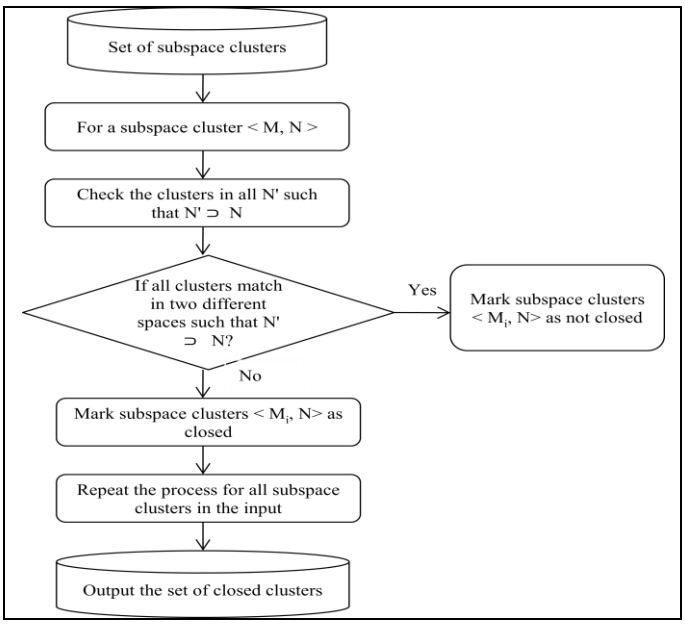

Figure 3: Architectural diagram of FCSC

Figure 3 gives a broad view of the architecture of the FCSC algorithm. It takes the set of subspace clusters as input and

checks for each subspace cluster whether it is closed or not. If it is closed, places it in the output stack otherwise proceeds with the next item in the list. Finally it outputs the set of closed clusters.

5. Result Analysis

The experiments are conducted on HP G42 Notebook PC with Windows 7 Home Basic, 3 GB RAM and Intel(R) Core(TM) i3 CPU M370 @ 2.40GHz. The algorithms are tested on various benchmark datasets taken from UCI Repository [14]. The measures, number of clusters and purity have been used to compare FCSC algorithm with the existing SUBCLU algorithm. Purity of clusters is computed by taking the percentage of total number of elements that are classified correctly [15].

Table 1: Number of clusters generated by SUBCLU & FCSC

Comparison of no. of clusters between SUBCLU & FCSC		
DATASET NAME	**SUBCLU**	**FCSC**
Bank Note	15	6
Vertebralc2w	71	44
User Knowledge Modelling	120	115
Seeds	199	143
Wholesale Customers	309	271
yeast	967	772
Wine Quality	1363	1258
Image Segmentation	1538	1361
Steel Plates	1827	1639

Table 1 shows the number of clusters generated by both SUBCLU and FCSC algorithms. It is quite obvious that the number of clusters have been greatly reduced. Figure 4 depicts the graphical representation of comparison between number of clusters generated by SUBCLU and FCSC.

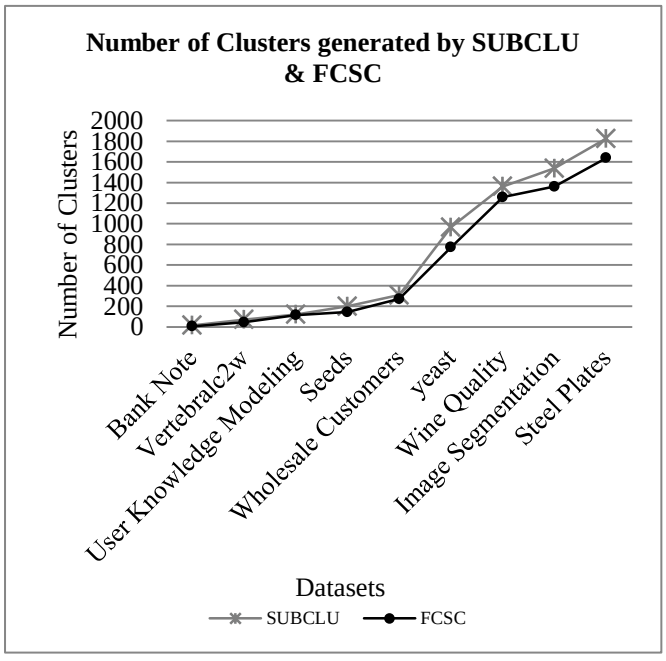

Figure 4: Number of cluster by SUBCLU & FCSC

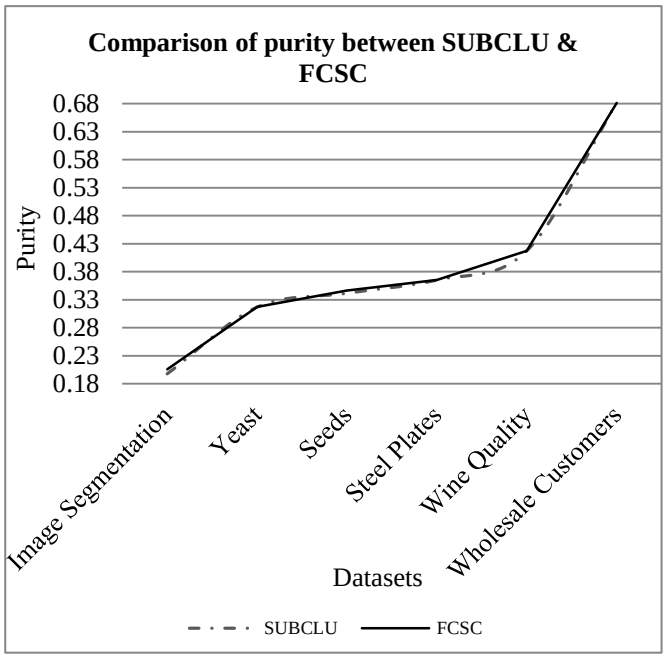

Figure 5: Purity of SUBCLU & FCSC

Figure 5 illustrates the comparison of purity of clusters generated by both the algorithms. From the Figure 5 it is clear that there is a marginal improvement in the quality of clusters. The FCSC algorithm works on the set of subspace clusters generated by any subspace clustering algorithm and hence it can be said that it is a post mining algorithm. It only removes all the redundant subspace clusters. As it does not alter the original clusters, 100% coverage of objects is guaranteed.

6. Conclusion

As the very use of subspace clustering has been restricted because of the reason that it produces tremendous number of results out of which most of them are redundant. By eliminating the redundant subspace clusters, the algorithm FCSC presents only the closed clusters which are less in number and hence easier for interpretation. Based on thorough experiments the FCSC algorithm is proved to be reducing the number of clusters on an average by 12% without loss of coverage of objects. The average purity of clusters is also marginally improved.

References

[1] K. Kailing, H.P. Kriegel and P. Kroger, "Density-connected subspace clustering for high dimensional data," In Proceedings of the 4th SIAM International Conference on Data Mining, Orlando, FL, pp. 46-257, 2004.

[2] I. Assent, E. Muller, S. Gunnemann, R. Krieger and T. Seidl, "Less is more: Non-redundant subspace clustering," In MultiClust: 1st International Workshop on Discovering, Summarizing and Using Multiple Clusterings Held in Conjunction with KDD 2010,Washington, DC, 2010.

[3] G. Chen, X. Ma, D. Yang and S. Tang, "Efficient approaches for summarizing subspace clusters into k representatives," In Soft Computing, 15, pp. 845-853, 2011.

[4] Jingbo Shang, Jian Peng, and Jiawei Han, "MACFP: Maximal Approximate Consecutive Frequent Pattern Mining under Edit Distance," In Proceedings of 2016

SIAM International Conference on Data Mining (SDM), 2016.
[5] Pasquier N, Bastide Y, Taouil R and Lakhal L, "Discovering frequent closed itemsets for association rules," In Lecture notes in computer science: database theory, ICDT, pp. 398–416, 1999.
[6] Afrati F, Gionis A and Mannila H, "Approximating a collection of frequent set," In Proceedings of the 10th ACM SIGKDD international conference on knowledge discovery and data mining (KDD'04), pp. 12–19, 2004.
[7] Xin D, Han J, Yan X and Cheng H, "Mining compressed frequent pattern sets," In Proceedings of the 31st international conference on very large data bases (VLDB'05), pp. 709–720, 2005.
[8] T. Calders and B. Goethals, "Non-derivable itemset mining," In Data Mining Knowledge Discovery, 14 (1), pp. 171_206, 2007.
[9] V. Purushothama Raju and G.P. Saradhi Varma, "Mining closed sequential patterns in large sequence databases," International Journal of Database Management Systems (IJDMS), 7 (1), February 2015.
[10] Amardeep K and Amitava D, "A novel algorithm for fast and scalable subspace clustering of high-dimensional data," Journal of Big Data, August 2015.
[11] J. Vreeken and A. Zimek, "When Pattern met Subspace Cluster a Relationship Story", 2011.
[12] A. Zimek, I. Assent and J. Vreeken, "Frequent Pattern Mining Algorithms for Data Clustering" In Frequent Pattern Mining. C. C. Aggarwal and J. Han, Eds., Switzerland: Springer International Publishing, pp. 403-423, 2014.
[13] E. Muller, I. Assent, S. Gunnemann, R. Krieger and T. Seidl, "Relevant Subspace Clustering: mining the most interesting non-redundant concepts in high dimensional data," In ICDM, pp. 377-386, 2009.
[14] M. Lichman,. "UCI Machine Learning Repository." [Online]. Available: http://archive.ics.uci.edu/ml. Irvine, CA: University of California, School of Information and Computer Science, 2003. [Accessed: August 2016].
[15] Evaluation of clustering [Online]. Available: http://nlp.stanford.edu/IRbook/html/htmledition/evaluation-of-clustering-1.html, Cambridge University Press, 2009. [Accessed: September, 2016].

Authors' Profiles

S. Anuradha received B.Tech. degree in Computer Science and Engineering from Gayatri Vidya Parishad College of Engineering, Visakhapatnam, India in 2010. From 2010-2014 she worked as a project engineer with Wipro Technologies, Chennai, India. Currently she is pursuing M.Tech. in Software Engineering in Gayatri Vidya Parishad College of Engineering (Autonomous), Visakhapatnam, India.

K.B. Madhuri received M.Tech. degree in Computer Science and Technology from Andhra University in 1999. She obtained Ph.D from JNTU, Hyderabad in 2009. Presently she is working as Professor in department of Information Technology at Gayatri Vidya Parishad College of Engineering (A), Visakhapatnam, Andhra Pradesh, India. Her research interests include Data Mining, Pattern Recognition, Data warehousing and RDBMS. She published research papers in National and International Journals. She is a member of IEEE and associate member of Institute of Engineers (India).

B. Jaya Lakshmi received M.Tech. degree in Computer Science and Technology (AI&R) from Andhra University in 2009. She is pursuing Ph.D in JNTUK, Kakinada. Presently she is working as Assistant Professor in department of Information Technology at Gayatri Vidya Parishad College of Engineering (A), Visakhapatnam, Andhra Pradesh, India. Her research interests include Data Mining and Pattern Recognition. She published research papers in International Journals. She obtained UGC minor research project in 2014.