

An algorithm for lattice-structured subspace clusters

Haiyun Bian

bianh@ececs.uc.edu

University of Cincinnati, OH 45221

Raj Bhatnagar

raj@ececs.uc.edu

University of Cincinnati, OH 45221

Abstract

Most of the current subspace clustering methods find non-overlapping clusters of similar objects. We present a new subspace clustering algorithm that has two valuable capabilities not available in most of the current methods. First, it finds possibly overlapping subspace clusters; and second, it can discover clusters that preserve some user-specified interesting properties. Our methods build a lattice of these subspace clusters and it further enables discovery of meaningful linkage-chains in terms of objects or attributes among distant clusters. Our algorithm can be applied to graph data, social network data, and biological data to obtain a better understanding of the substructures inherent in the data space.

1 Introduction

Clustering seeks to identify homogeneous groups of objects based on the values of their attributes. We view a data space as a table in which rows correspond to individual data objects and columns correspond to attributes. When the number of attributes (dimensions) becomes large, similarity defined on the whole attribute-set becomes low for almost any subset of objects, which makes finding full-dimensional clusters difficult. Dimension reduction techniques such as principle component analysis make the final interpretation of the clusters very difficult. *Subspace clustering* is a good solution for finding clusters in such data spaces, by defining similarity only on a selected subset of attributes (regional similarity) for a set of clustered objects. Each subspace cluster is a group of similar objects within its own subset of dimensions. Several methods have been proposed recently for discovering interesting subspace-clusters [1, 2, 3, 4, 5, 6].

Most of the existing subspace clustering methods find non-overlapping clusters, that is, each object belongs to at most one subspace cluster. However, finding overlapping subspace clusters is useful and necessary in many applications for the following reasons.

Multi-domain functionality

Each subspace cluster signifies some functional

	a	b	c	d	e	f
a	1	1	1	0	0	1
b	1	1	1	0	0	0
c	1	1	1	1	1	1
d	0	0	1	1	1	1
e	0	0	1	1	1	1
f	1	0	1	1	1	1

Table 1: Pairwise multi-functionality

	a_1	a_2	a_3	a_4	a_5
a	1	1	1	0	0
b	1	1	1	0	0
c	1	1	1	1	1
d	0	0	1	1	1
e	0	0	1	1	1

Table 2: Object-attribute multi-functionality

group of objects in the domain. It is very common that some objects participate in multiple functional groups and hence should belong to multiple subspace clusters. For example, a particular gene may play active role in multiple biological processes. We show two different examples situations here. The first is in the form of pairwise similarity between a set of objects as shown in table 1. The same set of objects form the row and the column labels in the table, as is the case for a graph incidence matrix. An entry of ‘1’ in the table indicates the objects of the pair are similar to each other. The table shows two clearly defined clusters, one with objects $\{a, b, c\}$, and the other with objects $\{c, d, e, f\}$. c is a multi-functional object in the sense that it belongs to more than one subspace cluster.

The second example is in the form of a table of object-attribute pairs as shown in table 2. Clusters in this type of data are defined by the shared object-attribute values. The example shown in the table 2 has two clusters, one with objects $\{a, b, c\}$ in subspace $\{a_1, a_2, a_3\}$, and the other with object $\{c, d, e\}$ in subspace $\{a_3, a_4, a_5\}$. Object c occurs in both the clusters

which are within different subspaces, and attribute a_3 is common between the two subspaces.

Connection between distantly related objects

A good criterion for connection path between two objects can be defined at the cluster level instead of the single object level. The connections among the individual objects. Consider the example in table 1, if we are trying to find a good connection between b and f , two paths satisfy the shortest path criterion, path $b \rightarrow a \rightarrow f$ and path $b \rightarrow c \rightarrow f$. However, since $\{a, b, c\}$ and $\{c, d, e, f\}$ are clusters, the connection $a \rightarrow f$ is more likely to be a case of spurious connectivity, and the preference should go to $b \rightarrow c \rightarrow f$. defined. We show in this paper that the cluster level connections are best captured by the lattice built from all the dense regions found in the data.

Our Contribution We present in this paper a new subspace clustering algorithm that can find overlapping subspace clusters satisfying different constraints. A lattice structure is built from the subspace clusters, and connective relationships among the objects and attributes of a cluster can be revealed by the lattice. Our algorithm is capable of dealing with similarity measures as well as some kinds of discrepancy as the metric for clustering objects together. Also, our algorithm simultaneously clusters on both the objects space and the attributes space.

2 Related Research

Subspace clustering in *CLIQUE* [1] is a density-based method which partitions the data space into non-overlapping rectangular units. A search is performed to find dense units within all possible subspaces. The pruning strategy is based on the fact that “if a k -dimensional region is dense, then all the $k-1$ dimensional regions that it contains must also be dense”. *ENCLUS* [4] follows the density based idea, but uses entropy as a heuristic to prune away uninteresting subspaces. *PROCLUS* [2] is a variation of the k -medoid algorithm, and it returns a partition of the data points into clusters together with the set of dimensions on which each cluster is correlated. Bi-clustering algorithms [5, 6] are proposed to meet the requirements in bioinformatics field where we need to find sets of genes showing similarity under a set of conditions. Greedy search is used in [5] to find the subsets of genes and subsets of conditions for which a measure called *mean square residue score* is minimized, and the algorithm returns non-overlapping bi-clusters. Work presented in *FLOC*[6] is a further improvement on the bi-clustering idea by allowing the finding of multiple bi-clusters simultaneously.

3 Our Approach

3.1 Problem Description A data space \mathcal{DS} is given by a set of attributes $\mathcal{A} = \{a_1, a_2, \dots, a_n\}$ and a population of objects $\mathcal{O} = \{o_1, o_2, \dots, o_m\}$. Each o_i has a value for each of the attributes in \mathcal{A} . A subspace cluster c of the data space \mathcal{DS} is defined as $\langle \mathcal{O}_i, \mathcal{A}_i \rangle$, where $\mathcal{O}_i \subseteq \mathcal{O}$ and $\mathcal{A}_i \subseteq \mathcal{A}$. We call \mathcal{O}_i and \mathcal{A}_i the object set and the attribute set of the subspace cluster respectively. The symbol d_{ij} denotes the j^{th} attribute of the i^{th} object. The following definitions, used in our algorithm, are defined on 0/1 binary valued data spaces. **DEFINITION 3.1.** A subspace cluster $c = \langle \mathcal{O}_i, \mathcal{A}_i \rangle$ on 0/1 valued data space is a **dense region** if o_{ij} are ‘1’ for all $o_i \in \mathcal{O}_i$ and $a_j \in \mathcal{A}_i$.

DEFINITION 3.2. A dense region $c = \langle \mathcal{O}_i, \mathcal{A}_i \rangle$ is a **maximal dense region** if all regions that are proper supersets of c are not dense.

A region c_i is said to be proper superset of c_j if either $\mathcal{O}_j \subset \mathcal{O}_i$, or $\mathcal{A}_j \subset \mathcal{A}_i$, or both. That is, a dense region is maximal if and only if adding any object (attribute) will force the enlarged region to contain ‘0’ entries. We have proved that our definition of maximal dense region is equivalent to the **concept’s** definition in formal concept analysis [9]. (The proof is omitted due to space limitation.) So all the dense regions form a lattice structure based on the following order: for two dense regions $c_1 = \langle \mathcal{O}_1, \mathcal{A}_1 \rangle$ and $c_2 = \langle \mathcal{O}_2, \mathcal{A}_2 \rangle$, we define $c_2 \prec c_1$ if $\mathcal{O}_2 \subset \mathcal{O}_1$. c_2 is called a **child** of c_1 , and c_1 is called a **parent** of c_2 . c_1 is called the **cover (direct parent)** of c_2 if $c_2 \prec c_1$, and for all $c_i: c_2 \prec c_i$ ($c_i \neq c_1$ and $c_i \neq c_2$), we have $c_1 \prec c_i$; correspondingly, c_2 is called the **direct child** of c_1 .

The **prime operator** ‘ \prime ’ is defined as: $\mathcal{O}'_i := \{a \in \mathcal{A} | d_{oa} = 1 \text{ for all } o \in \mathcal{O}_i\}$, and $\mathcal{A}'_i := \{o \in \mathcal{O} | d_{oa} = 1 \text{ for all } a \in \mathcal{A}_i\}$. Then a maximal dense region $c = \langle \mathcal{O}_i, \mathcal{A}_i \rangle$ is a pair where $\mathcal{O}'_i = \mathcal{A}_i$ and $\mathcal{A}'_i = \mathcal{O}_i$. We call maximal dense region as dense region in the remaining discussion. The lattice built from dense regions is constructed using the following rules:

- Each node in the lattice is a dense region;
- If node c_1 is a cover for node c_2 , put c_1 above c_2 and add an edge between them;
- The lattice top is $c_{top} = \langle \mathcal{O}'', \mathcal{O}' \rangle$, and lattice bottom is $c_{bottom} = \langle \mathcal{A}', \mathcal{A}'' \rangle$.

We define a **path** in a lattice to be a contiguous sequence of edges between nodes. Two nodes are said to be **connected** if there exists a path between them.

3.2 Algorithm Outline Our method has three main phases. Phase-0 transforms a multi-valued data space to a binary data space. In phase-1, dense regions are found

and the lattice is built. In phase-2, connections are searched through the lattice. Algorithms for building concept lattices find all concepts without any pruning and therefore have a high complexity. Our definition of dense region is mathematically equivalent to a formal concept [9], the semantics of **dense regions** can be used as pruning tool to discover subspaces with different semantics for the clusters. That is, a combination of the binary transformation (phase-0), pruning strategy (phase-1) and search strategy (phase-2) produce clusters satisfying different \mathcal{P} -property. Formally, \mathcal{P} -cluster is defined as the follows:

DEFINITION 3.3. Let $\mathcal{B}(\mathcal{DS})$ denote set of all dense regions in data space \mathcal{DS} . A subspace cluster $c = \langle \mathcal{O}_i, \mathcal{A}_i \rangle$ on data space \mathcal{DS} is called a \mathcal{P} -cluster if it satisfies some property \mathcal{P} . Here \mathcal{P} is defined as a function: $\mathcal{P}: \mathcal{B}(\mathcal{DS}) \rightarrow \{0, 1\}$

3.3 Phase-0: To Binary Data For multiple valued data sets, a function F is applied to the original data to convert it into binary valued data. The following are some example F functions and are similar to the ideas of *scale* defined in the formal concept analysis [9], but here they have different semantics interpretations. Choice of F significantly determines the \mathcal{P} property of the clusters found by the algorithm. For similarity-based clusters, we define F as a function that maps each attribute value o_{ij} into a 0/1 vector: $F: o_{ij} \rightarrow \{(0|1)^*\}$

Choice of F from *Equality functions* can find clusters in which objects have same values for the attributes, and when F is from *range functions* we find clusters in which objects have values in certain ranges.

We have used another F' function which can be used to find clusters of objects with discrepant attribute values. We define discrepant F' as functions that map each attribute value pair into 0/1: $F': o_{ij} \times o_{kj} \rightarrow \{0, 1\}$. An example where this may be useful is the situation where negatively correlated genes provide insights into the structures of some biochemical pathways [10].

We define $F'(o_{ij}, o_{kj}) = 1$ if $o_{ij} \neq o_{kj}$ or they are in different ranges for real valued attributes. Clusters found satisfy the \mathcal{P} property that all objects in every cluster are different for all the attributes of the clusters.

3.4 Phase-1: Pruned Lattices Objective of this phase is to build the lattice of dense regions using the binary data space (from phase-0). Many exhaustive algorithms are available to find all concepts (=clusters) in formal contexts [8, 9]. These algorithms have high complexity because the number of possible concepts increases exponentially with the number of objects. Our algorithm is motivated by the concept search algorithms, but uses an additional **Pruning** feature.

We show that the proposed pruning methods can restrict the search space effectively.

3.4.1 Size Pruning We consider the following two types of size constraints: minimum number of objects and minimum number of attributes in a cluster. Algorithm 1 given below finds all dense regions that have at least s objects. It is based on the fact that the object

Algorithm 1

Let L be the list of candidate dense regions, and initially empty;

```

foreach  $a_i \in A$ 
foreach  $S_i = \langle \mathcal{O}_i, \mathcal{A}_i \rangle \in L$ 
   $O \leftarrow a_i' \cap \mathcal{O}_i$ 
  if  $|O| > s$  and  $O \notin L$ 
    add  $L \leftarrow L \cup \langle O, \mathcal{A}_i \cup a_i \rangle$ 
  else if  $|O| > n$  and  $\langle O, \mathcal{A}_j \rangle \in L$ 
     $\langle O, \mathcal{A}_j \rangle \leftarrow \langle O, \mathcal{A}_j \cup \mathcal{A}_i \cup a_i \rangle$ 

```

set \mathcal{O}_i of any dense region c must be intersection of all $(a_i)' \in \mathcal{A}_i$, where $\mathcal{A}_i \subset A$. It finds all the dense regions that have at least s objects, and the lattice structure is built after the dense regions are found.

3.4.2 Effectiveness of Size Pruning We tested the size constraint on congressional voting data set from the UCI repository [11]. This data set includes votes for 435 congressmen on 16 issues. Each attribute has three possible values: 'y', 'n', and '?'. Using algorithm 1 without any size constraint, there are 227032 maximal dense regions (concepts) in this data. Figure 1 shows the number of dense regions found and pruned for different size constraints on the object set. With the increasing threshold on the number of objects, the number of dense regions found decreases significantly and the number of pruned dense regions increases. The most effective region for the size constraint pruning is [0,20].

3.4.3 Semantic Pruning We can also set semantic constraints for pruning, and only those dense regions that satisfy the constraints will be found. Dense regions in binary valued data spaces have simpler semantics: regions of all 1's or regions of all 0's. For multiple valued data spaces, dense regions can have much richer semantics. All combinations of values of different attributes can constitute dense regions, while not all of them maybe semantically meaningful.

3.4.4 Effectiveness of Semantic Pruning We tested the semantic constraints on the same data set. The 16 issues are grouped into five categories, with each category having 4-8 issues. The semantics constraint is set as: a dense region is interesting when all the con-

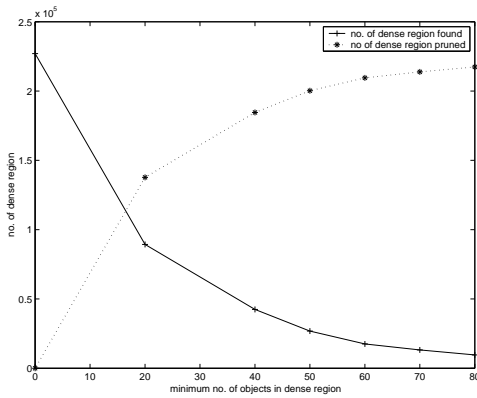


Figure 1: Size Constraint pruning: number of dense regions found and pruned

gressmen in this region agree with each other on at least $\alpha\%$ of the issues in any of the five categories. The results show that the number of dense regions found decreases quadratically with the increase of α , and the number of pruned regions increases quadratically.

3.5 Phase-2: Search for Connections and Combinations All dense regions found are not equally interesting. Paths among nodes of the lattice and mergers of some dense regions (leading to non-dense clusters) may have valuable semantic content. We consider the following two types of cluster shapes. The first type is a symmetric matrix, such as the graph incidence matrix. The second type is a non-symmetric matrix, resulting in a rectangular shape in the dataspace.

3.5.1 Symmetric-reflective Matrix Many of the symmetric-reflective matrices, such as graph incidence matrix and pairwise similarity data, have a commonality that the object set is the same as the attribute set. The lattice of dense regions built from this type of data has a lot of special characteristics which can be used to facilitate the mining process. For clarity purpose, we call graph vertex as vertex, and dense region in the lattice as node.

DEFINITION 3.4. A dense region $c = \langle \mathcal{O}_i, \mathcal{A}_i \rangle$ is called **square dense region** if $|\mathcal{O}_i| = |\mathcal{A}_i|$.

DEFINITION 3.5. A dense region $c = \langle \mathcal{O}_i, \mathcal{A}_i \rangle$ is called **core** if $\mathcal{O}_i = \mathcal{A}_i$.

A core in graph data corresponds to a clique, an important subgraph. They are used to define closely connected objects, such as human communities, gene functional groups.

Observations: All square dense regions $c = \langle \mathcal{O}_i, \mathcal{A}_i \rangle$ in lattice of symmetric and reflective matrices data

that satisfy $\mathcal{O}_i \cap \mathcal{A}_i \neq \phi$ are cores. The lattice of dense regions for symmetric and reflective matrices is symmetric w.r.t the cores. That is, for any path in the lattice starting from a core to the lattice top, there is a corresponding path starting from the same core to the lattice bottom, with every node $c = \langle \mathcal{O}_i, \mathcal{A}_i \rangle$ in the first path having a corresponding node $c = \langle \mathcal{A}_i, \mathcal{O}_i \rangle$ in the second path. So, for a lattice of symmetric and reflective matrix data, only half of the lattice is needed to encode all the connectivity information. This can help us design more efficient algorithms for building the lattices.

Find dense regions of various p -properties:

DEFINITION 3.6. A region is called α -dense if the percentage of 1 entries in the submatrix exceeds some threshold α .

DEFINITION 3.7. For a dense region c , we define $\mathcal{L}_c = \langle \mathcal{O}_{L_c}, \mathcal{A}_{L_c} \rangle$, where \mathcal{O}_{L_c} is the union of the object sets of direct parents of c , and \mathcal{A}_{L_c} is the union of the attribute sets of all the direct parents of c .

Observation For any dense region c , any two objects in \mathcal{O}_{L_c} are at most 1-hop from each other. Here, one hop means one intermediate vertex.

More generally, if we go more than one level up the direct-parent relationships, the maximum distance between any two objects in the resulting combined cluster is at most $(n + m - 1)$ -hops from each other, where n and m are the number of cover operations taken at two branches.

It is inefficient to search for α -dense regions starting from every node. A better way is to start from the cores, and form larger clusters by taking the union of the object sets of the covers of the cores and the resultant α -dense clusters have an upper bound on the maximum hops between any pair of objects within each cluster, and also density maximality.

Connection between cores: The lattice structure reveals important information about connections between cores. We consider the following three types of relationships for a pair of cores, assuming the lattice top $c_{top} = \langle \mathcal{O}, \phi \rangle$.

Type I: If all paths between two cores pass the lattice top, then these cores are disconnected.

Type II: If two cores have at least one common parent, there is at least one multi-functional object for the two cores.

Type III: If two cores are connected by at least one path, but share no parent except c_{top} , then there are some objects outside both the cores that provide the connection between the cores. These

objects can be easily identified by following the *cover* and *direct child* links.

3.5.2 Non-symmetric Matrix Most of the clusters in object-attribute datasets are non-symmetric, that is, they are rectangular shaped submatrices of the data space. So cores do not exist but the square dense regions still exist. A size constraint on both the object space and the attribute space will turn in a much smaller set of dense regions that may act as the cores, which we call q -cores. After the q -cores are identified, similar strategies can be used to find α -dense regions between the q -cores. Connections between the q -cores can also be categorized into similar three types but the search needs to be performed within both halves of the lattice.

4 Algorithm Evaluation

We have designed a synthetic data generator which is an updated version of the one presented in [7] by allowing one object to be in multiple clusters. Our first test used the number of objects as the control variable. Each attribute has a real value in $[0, 100]$ interval and is transformed into a binary equivalent by using a range function (the F function in phase-0) into ten equal length intervals. For a data set with 20 dimensions and five non-overlapping dense regions of five dimensions each, the running time increases linearly with the number of objects. Addition of 10% objects as random noise still discovers all the dense regions with the same time complexity. The same test was repeated with overlapping dense regions. Subspaces overlapped in both, the objects and the attributes. There were four dense regions with two overlapping pairs, each pair sharing two attributes and 40% objects. The running time is larger than that for the non-overlapping dense regions but still increases only linearly.

We tested the scalability of the algorithm as the number of attributes is increased from 20 to 100. All the datasets had 10,000 objects and two overlapping dense regions in 5-dimensional subspaces. The results show that the algorithm time grows quadratically with the number of attributes. We also tested the effect of size constraint on the running time of the algorithm. As intuitively expected, the smaller the size constraint, the karger is the number of clusters found and more time is taken by the algorithm.

5 Conclusions

We have presented a new subspace clustering algorithm that can find possibly overlapping subspace clusters in the data. A lattice is built from all the dense regions found, from which other clusters of interesting properties can be constructed and paths among clusters

can be identified. Our algorithm provides a very good interpretation of the relationships among overlapping subspace clusters. It is the first attempt that defines subspace clusters as combinations of cores in a lattice according to some partial ordering. This provides tremendous amount of control and information about the structural properties of the subspaces that may be constructed with the lattice of subspace cores. Many extensions of this research are possible. Algorithms to find dense regions that satisfy both the object set size constraint and attribute set size constraint can be designed, which may help in making the algorithm in phase-1 more efficient.

References

- [1] Rakesh Agrawal and Johannes Gehrke and Dimitrios Gunopulos and Prabhakar Raghavan, *Automatic subspace clustering of high dimensional data for data mining applications*, ACM SIGMOD Proceedings, 1998, pp. 94–105.
- [2] Charu C. Aggarwal and Joel L. Wolf and Philip S. Yu and Cecilia Procopiuc and Jong Soo Park, *Fast algorithms for projected clustering*, SIGMOD Conference Proceedings, 1999, pp. 61–72.
- [3] C. Aggarwal and C. Procopiuc and J. Wolf and P. Yu and J. Park, *A Framework for Finding Projected Clusters in High Dimensional Spaces*, ACM SIGMOD International Conference on Management of Data Proceedings, 1999.
- [4] Chun Hung Cheng and Ada Wai-Chee Fu and Yi Zhang, *Entropy-based Subspace Clustering for Mining Numerical Data*, Knowledge Discovery and Data Mining, 1999, pp. 84–93.
- [5] Yizong Cheng and George M. Church, *Biclustering of Expression Data*, ISMB Proceedings, 2000, pp. 93–103
- [6] Jiong Yang and Wei Wang and Haixun Wang and Philip S. Yu, *delta-cluster: Capturing Subspace Correlation in a Large Data Set*, ICDE Proceedings, 2002.
- [7] Mphamed Zait and Hammou Messatfa, *A comparative study of clustering methods*, Future Generation Computer Systems, 13 (1997), pp. 149–159.
- [8] Christian Lindig, *Fast concept analysis*, Working with Conceptual Structures - Contributions to ICCS, 2000.
- [9] B. Ganter and R. Wille, *Formal concept analysis: mathematical foundations*, Springer, Heidelberg, 1999.
- [10] Dhillon IS, Marcotte EM, Roshan U, *Diametrical clustering for identifying anti-correlated gene clusters*, Bioinformatics, 19(2003), pp. 1612-1619.
- [11] C.L. Blake and C.J. Merz, = *UCI Repository of machine learning databases* <http://www.ics.uci.edu/~mllearn/MLRepository.html>, University of California, Irvine, Department of ICS, 1998.