ASSIGNMENT - 1
Overview : Density-Connected Subspace Clustering for High-Dimensional Data

Aditya Rajesh Patil
dataHacks
aditya18@iitg.ac.in
180101004

Bhasker Goel
dataHacks
bhasker18@iitg.ac.in
180101015

Kartikeya Saxena
dataHacks
kartikey18a@iitg.ac.in
180101034

V Anirudh
dataHacks
anirudh18@iitg.ac.in
180101084

1 WHAT BOTTLENECK THIS CLUSTERING ALGORITHM ATTACKS?

The algorithm SUBCLU attacks the following bottlenecks -

(1) Most real life datasets are characterized by high-dimensional and sparse dataspaces which in turn make it very difficult to find clusters in the original data space. (The Curse Of Dimensionality)
The Algorithm attacks this by finding interesting clusters in subspaces which would have been hidden/impossible to detect in the original data space.
(2) Earlier clustering algorithms were primarily based on Grid-based approaches (E.g CLIQUE) which made it difficult for them to detect clusters that were shaped or positioned arbitrarily on the grid.
By not relying on any Grid-based approach and using DBSCAN, SUBCLU is perfectly capable of detecting arbitrary shaped and positioned clusters in the subspace.
(3) Subspace clustering algorithms often check for clusters in all Subspaces (the number of which is exponential). SUBCLU however,
i) Using Monotonicity of Density-Connectivity and
ii) Pruning irrelevant subspaces avoids doing this.
(4) Algorithms (E.g - Dimensionality Reduction or Projected Clustering) more or less, find one final clustering of the Data. But it is perfectly possible that different clustering is possible based on different attributes of data. SUBCLU allows objects to be clustered varyingly in different subspaces.
(5) Dimensionality Reduction Clustering Algorithms create new features from existing features to force-eradicate the Curse of Dimensionality. But in doing this, the features lose their intuitive meaning and the clustering becomes very hard to interpret. SUBCLU, even the considering subspaces, keeps the attributes the same as the original dataset.

Conference'17, July 2017, Washington, DC, USA

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

2 WHAT IS THE OVERALL IDEA OF THE ALGORITHM? DO INCLUDE A FLOWCHART OR A FIGURE THAT EXPLAIN THE WHOLE ALGORITHM.

The algorithm is designed to detect the density- connected sets in all subspaces of high dimensional data in a bottom up, greedy fashion. First, the clusters present in 1-dimension are identified applying DBSCAN as a subroutine to each 1-dimensional subspace. Then k-dimensional clusters are used to generate k+1-dimensional clusters by using DBSCAN recursively and the parameters for DBSCAN are predetermined. The algorithm is built on two properties of clusters.

(1) If two points are not in a cluster in a particular dimension, then they will not be in the same cluster in any higher dimension (Monotonicity of Density-Connected Sets).
(2) A cluster in a lower dimension might not be a cluster in higher dimension. It might lose a few points from the cluster of lower dimension, but never gain other points.

From the above two properties of clusters the following idea can be concluded:
Pruning Irrelevant Subspaces: Given two subspaces S and T, such that T is a subset of S. If T has no cluster present in it, then S will also have no clusters and computation on S can be ignored.
A naive implementation of SUBCLU would result in running DBSCAN 2^d times, that is in all possible subspaces of a d dimension space. But using Pruning of Irrelevant Subspaces reduces the computational overhead of the algorithm. This leads to an efficient implementation of SUBCLU.
SUBCLU uses inverted files data structure to process range queries encountered in the algorithm.
Flowcharts explaining the whole algorithm is shown in Figure 1 and Figure 2.

3 HOW IS THE EVALUATION OF THE ALGORITHM CARRIED OUT?

The performance evaluation of the algorithm was carried out with (minPts = 8, epsilon = 2 as DBSCAN parameters) using synthetic data sets and real world Gene Expression Data of CDC15 mutant

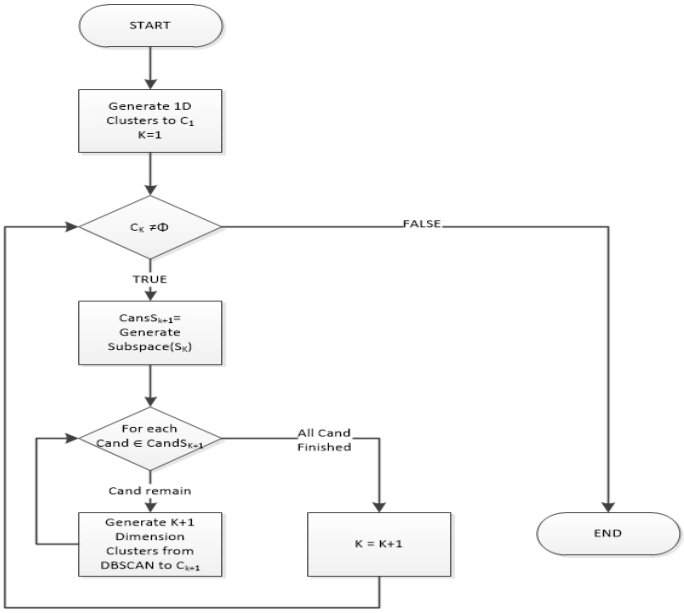

Figure 1: SUBCLU Algorithm

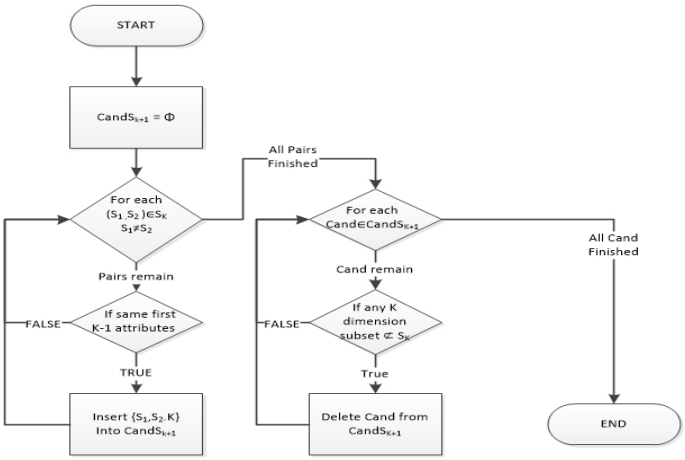

Figure 2: GenerateSubspace Subroutine

[SSZ+98]. Synthetic data sets of varying size and structure were created with varying dimensionality of subspace clusters and feature space. It was found that the runtime SUBCLU grows with at least a quadratic factor against data set size, dimensionality of the data set and dimensionality of subspace clusters.

For Accuracy evaluation, the results on the synthetic datasets were compared to an efficient implementation of CLIQUE (CLustering In QUEst) Algorithm and it was shown that SUBCLU outperforms the latter.

4 ARE ABLE TO FIND OUT THE CODE AND DATASETS FOR THE EXPERIMENTS IN THE PAPER? IF NOT, HAVE YOU CONTACTED THE AUTHORS?

The Paper provides the Pseudocode for what the authors did, but not the actual code. One of the Datasets used was real world Gene Expression Data [SSZ+98][1] which we were able to find.

The others they synthetically generated to match the application at hand. Although they provided a vague description of how they generated the data, we decided to contact the authors for the Original datasets they generated and used. We have also requested them for the code they prepared.

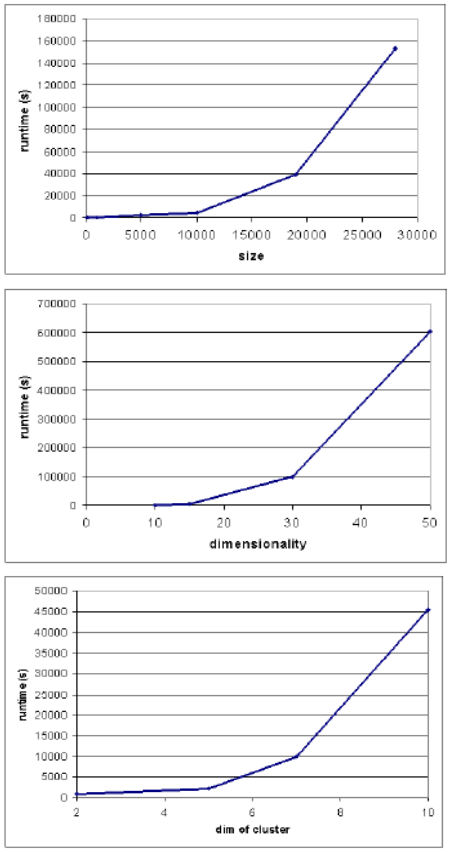

Figure 3: Performance Evaluation

Data set	d	dim. of subspace cluster	N	# generated clusters	true clusters found by SUBCLU	CLIQUE
DS01	10	4	18999	1	1	1
DS02	10	4	27704	1	1	1
DS03	15	5,5	3802	3	3	1
DS04	15	3,5,7	4325	3	2	1
DS05	15	5,5,5	4057	3	3	1
DS06	15	4,4,6,7,10	2671	6	5	2

Figure 4: Accuracy Evaluation

5 COULD YOU FIND ANY REAL-WORLD APPLICATIONS OF THIS ALGORITHM? DOES ANY ML/DATA ANALYSIS PACKAGE INCLUDE THIS ALGORITHM?

Subspace Clustering could potentially be used in the following real-world scenarios:

(1) **Information Integration Systems:** These systems are motivated by the fact that future needs will be satisfied by autonomous, heterogeneous distributed information sources

accessible through the internet. Such a system often maintains coverage statistics for each source based on a log of previous user queries. With such information, it can rank the sources for a given query for faster retrieval. The subspace clustering algorithm can be applied to the query list with the queries being instances and the sources corresponding to dimensions of the dataset. The result is a rapid grouping of queries where a group represents queries coming from the same set of sources. Conceptually, each group can be considered a query class where the classes are generated in a one-step process using subspace clustering.

(2) **Web Text Mining:** A fundamental problem with organizing web sources is that web pages are not machine readable. In addition, semantic heterogeneity is a major challenge. Recently, there has been strong interest in developing ontologies to serve as a semantic, conceptual, hierarchical model representing a domain or a web page. If the web pages are presented in the form of a document-term matrix where the instances correspond to the pages and the features correspond to the keywords in the page, the result of subspace clustering will be the identification of a set of keywords (subspaces) for a given group of pages. These keyword sets can be considered to be the main concepts connecting the corresponding groups. Ultimately, the clusters would represent a domain and their corresponding subspaces would indicate the key concepts of the domain.

(3) **DNA Microarray Analysis:** DNA microarray datasets provide information on the expression levels of thousands of genes under hundreds of conditions. Currently, microarray data must be preprocessed to reduce the number of attributes before meaningful clusters can be uncovered. In addition, individual gene products have many different roles under different circumstances. Subspace clustering is a promising technique that extends the power of traditional feature selection by searching for unique subspaces for each cluster.[2]

We found an implementation of SUBCLU in R in its “subspace” library The SUBCLU algorithm uses the DBSCAN algorithm to actually cluster the dataset. "mlpack" library of C++ also includes an implementation of DBSCAN accelerated with dual-tree range search technique. “scikit-learn” includes a Python implementation of DBSCAN for arbitrary Minkowski Metrics, which can be accelerated using k-d trees and ball trees but which uses worst-case quadratic memory.[3][4]

REFERENCES

[1] Gene Expression Dataset, https://www.molbiolcell.org/doi/suppl/10.1091/mbc.9.12.3273

[2] Example reference, https://www.kdd.org/exploration_files/parsons.pdf

[3] Density-based spatial clustering of applications with noise(DBSCAN) algorithm, https://en.wikipedia.org/wiki/DBSCAN

[4] SUBCLU R data analysis package, https://rdrr.io/cran/subspace/man/SubClu.html