CMPUT695 Principles of Knowledge Discovery in Data

Finding Generalized Projected Clusters in High Dimensional Spaces

Charu C. Aggarwal and Philip S. Yu

Presented by: Yaling Pei

Instructor: Dr. Osmar R. Zaïane
Nov. 5, 2002

Outline

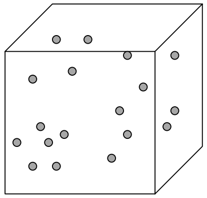
- Motivation
- Dimension reduction
- Algorithm overview
- Experimentation
- Conclusions

Data Clustering Analysis

- Partitioning a set of data into groups
  - Intra-class similarity is maximized
  - Inter-class similarity is minimized
- Applications in practical problems
- Clustering methods have been studied extensively
- Many well known clustering algorithms

Challenges

Most clustering algorithms do not work efficiently in higher dimensional space



- Inherent sparsity of the points
- Dimensionality curse

Objective

- Provide a general framework and algorithms in which clusters can be constructed in any arbitrarily projected space of lower dimensionality

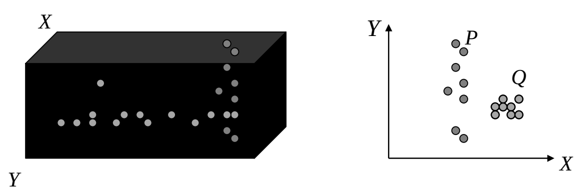
- Cluster high-dimensional data in a more meaningful way.

Outline

- Motivation
- <u>Dimension Reduction</u>
- Algorithm Overview
- Experimentation
- Conclusions

Feature selection method

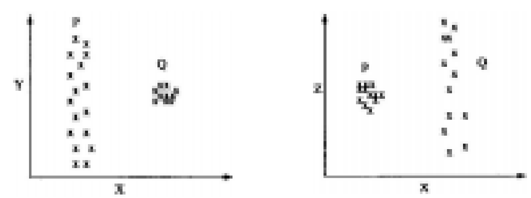
- Finding the particular dimensions on which the points in the data are correlated
- Pruning away remaining dimensions (noise)



- Problem →Loss of information

Axis Parallel Projection Methods

- Data points are projected to subspaces along axis
- Finding Locally dense subspace
  - each dimension is relevant to at least one of the clusters

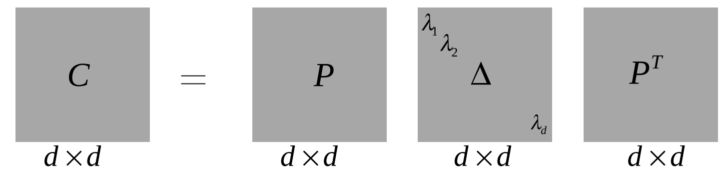


In reality, Clusters tend to exist in arbitrarily oriented subspace

Basic Idea

- Construct the covariance matrix (*C*) for the dataset
  - C is symmetric
  - Entry (i, j) = covariance between dimensions i and j
- Find eigenvalues and eigenvectors of *C*
  - Eigenvectors define an orthonormal system
  - Eigenvalues denote the spread along newly defined dimensions

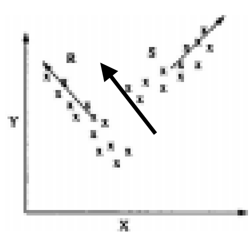
Singular Value Decomposition-SVD



- SVD: $C = P \Delta P^T$
- *C* - symmetric covariance matrix
- Δ - diagonal matrix
- λ_i - eigenvalues of C
- *P* - matrix with orthonormal eigenvectors

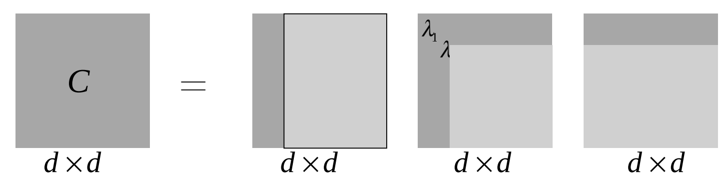
Subspace selection

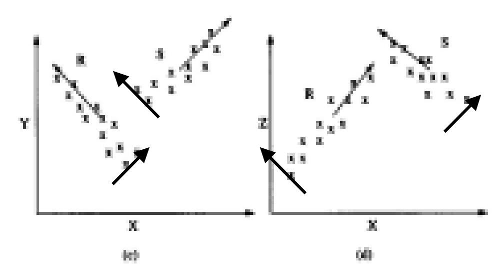
- Large eigenvalues correspond to eigenvectors with the maximum spread or variance
- We choose dimensions with the least spread to form subspace for each cluster



Subspace with maximum point distribution is complementary to subspace with the least spread

Applying SVD





Outline

- Motivation
- Dimension reduction
- Algorithm overview
- Experimentation
- Conclusions

ORCLUS

- Arbitrarily ORiented projected CLUSter generation
- SVD is used for dimension reduction
- Clustering by combining partitioning method and hierarchical method
- Extended CF-vector (ECF-vector) is used to ensure scalability for very large databases
- Two input parameters
  - Number of clusters, ***k***
  - Dimensionality of subspace for clustering, ***l***

Concepts

- Centroid of a cluster
  - Algebraic average of all the points in the cluster
- Distance between two points x_1 and x_2
  - Euclidean distance metric
- Dimensionality of the dataset $|D|$
- Initial seeds - a set of points $\{s_i \mid i=1, 2, \ldots k_0\}$
  - $k_0 > k$

Generalized projected cluster

- Subspace ε – a set of vectors
- Cluster ***C*** – a set of data points in subspace ε
- Points in ***C*** are closely clustered in the subspace defined by the vectors in ε
- Projected energy of ***C*** in ε

$$R(C,\varepsilon) = \frac{\sum_{i=1}^{t} \{Pdist(x_i, X(C), \varepsilon)\}^2}{t}$$

Data Clustering

- Basic idea

  Select a set of initial points as seeds, iteratively find each cluster in reduced dimensions and merge closest clusters till *k* clusters are found.

  Dimensionality is reduced gradually.

Data clustering – Procedure

1. Initially, partition the dataset into k_0 clusters by assigning each point to its closest seed
2. Each seed is replaced by the centroid of the newly created cluster
3. Find subspace for each cluster (SVD)
4. Merge clusters by a factor of $\alpha < 1$ and reduce dimensionality of current cluster by a factor of $\beta < 1$

Same number of iterations to reduce $k_0 \rightarrow k$ and $|D| \rightarrow l$

Terminate

Merging

- Goal – find clusters with least projected energy
- Each cluster is associated with its own subspace
- Merge clusters $\boldsymbol{C_i}$ and $\boldsymbol{C_j}$ when projected energy of $C_i \cup C_j$ is the smallest
  - Find least spread subspace for points in $C_i \cup C_j$
  - Find the centroid of $C_i \cup C_j$ and compute projected energy
- $C_2^{k_i}$ times pair-wise comparison

Problems with merging

- Not feasible with very large database
  - work explicitly with the set of current clusters
  - Covariance matrix calculation is I/O intensive
- Solution

Extended cluster feature vectors

Scalability for very large databases

- Cluster Feature vector (CF-vector)
- Extended CF-vector (ECF-vector)
  - Specific to a given cluster ***C***
  - Containing ($d^2 + d + 1$) entries

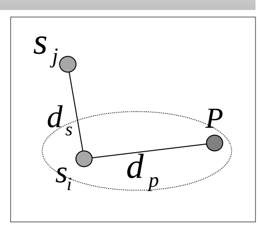
> ➢ECF1^C is set of d^2 entries → $\sum_C x_i \cdot x_j$
> ➢ECF2^C is set of d entries → $\sum_C x_i$
> ➢ECF3^C is the number of points in the cluster
> ➢ECF^C = (ECF1^C , ECF2^C , ECF1^C)

How Does Extended CF-vector work?

- Covariance matrix can be derived directly from the ECF-vector
- Satisfying additive property
  - The ECF-vector for $C_i \cup C_j$ is equal to the sum of the corresponding ECF-vectors of $\boldsymbol{C_1}$ and $\boldsymbol{C_2}$

> ECF-vectors are maintained for each cluster
> *instead of*
> the current clusters associated with each seed.

Outlier Handling



- Point P is in the cluster having seed s_i
- S_j is the nearest other seed to seed s_i in subspace ε_i
- P is an outlier if its projected distance to s_i, $d_p > d_s$
- Discard a certain percentage of the seeds in each iteration, for which the clusters contain very few points

Time and Space Complexity

- Time
  - Depends on the initial number of seeds k_0
  - Total run time $O(k_0^3 + k_0 \cdot N \cdot d + k_0^2 \cdot d^3)$
- Space
  - ECF-vector cuts down the space needs considerably
  - Overall space requirement $O(k_0 \cdot d^2)$
  - Independent database size

Improving running speed

- Progressive sampling techniques
- Assign each seed only a randomly sampled subset of the points in each iteration
- CPU time is saved considerably in the first few iterations if k_0 is much larger than k
- Not much information loss due to increased sample size
  - Sample size is increased by a factor of α

Outline

- Motivation
- Dimension reduction
- Algorithm overview
- Experimentation
- Conclusions

Confusion Matrix

- contains information about actual and predicted classifications done by a classification system
- One entry is significantly larger than the others in each row and column →Clustering well

Input cluster / Output cluster	C_A	C_B
C_1	35	2
C_2	0	28

Experiment Results (1)

- Data – generated based on heuristics
- Failure of axis-parallel projections
  - Reducing ***l*** worsened cluster quality

Input Clusters / Output Clusters	A	B	C	D	E
1	639	178	0	0	131
2	0	931	549	465	29
3	121	35	851	265	90
4	50	238	135	3889	408
5	138	133	200	128	1412

Table 1: Axis parallel projections, *l=14, N=10,000*

Input Clusters / Output Clusters	A	B	C	D	E
1	367	302	361	308	363
2	70	835	831	1331	241
3	169	101	834	103	73
4	108	180	96	3001	128
5	331	184	141	104	1365

Table 2: Axis parallel projections, *l=6, N=10,000*

Experiment Results (2)

- ORCLUS
  - Initial seeds: $\boldsymbol{k_0} = 15 * \boldsymbol{k}$
  - Good confusion matrix when $\boldsymbol{l} = 2$ to $\boldsymbol{l} = 8$
  - $\boldsymbol{l} = 6$ → Best performance achieved

Input Clusters / Output Clusters	A	B	C	D	E
1	998	5	0	0	3
2	0	1401	8	134	0
3	33	0	1703	0	0
4	0	40	0	3633	99
5	34	61	33	0	2009

Table 3: ORCLUS, *l=6, N=10,000*

Input Clust. / Output Clust.	A	B	C	D	E
1	9301	523	12	0	0
2	122	18144	7	24	0
3	0	0	29821	0	22
4	3	0	0	20453	24
5	15	101	0	0	23950

Table 4: ORCLUS, *l=6, N=100,000*

Running Time

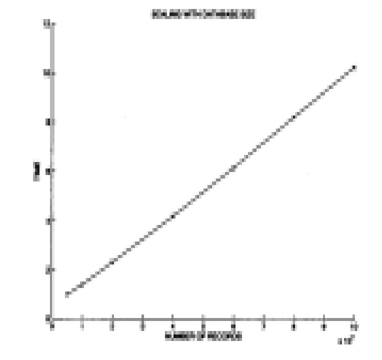


Figure 6: Scaling of running time with database size

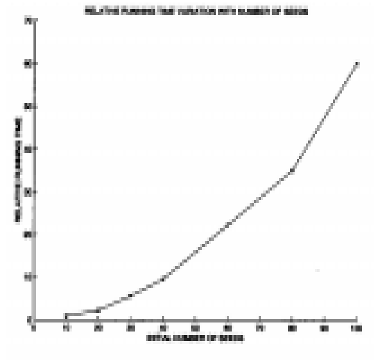


Figure 7: Scaling of running time with k_0

Outline

- Motivation
- Dimension reduction
- Algorithm overview
- Experimentation
- Conclusions

Comments

- Make use of inter-attribute correlations
- Cluster results are sensitive to input parameters
- No convenient method for the selection of *l*
- Tradeoff between accuracy and efficiency – k_0
- Future work
  - Apply it for effective high dimensional data visualization.

Extended CF-vector: ($d^2 + d + 1$) entries

($d^2 + d$)/2