http://ansinet.com/itj

ITJ

ISSN 1812-5638

INFORMATION TECHNOLOGY JOURNAL

ANSInet
Asian Network for Scientific Information
308 Lasani Town, Sargodha Road, Faisalabad - Pakistan

Information Technology Journal 6 (2): 255-258, 2007
ISSN 1812-5638

A Robust Algorithm for Subspace Clustering of High-Dimensional Data*

[1,2]Hongfang Zhou, [1]Boqin Feng, [2]Lintao Lv and [1]Yue Hui
[1]School of Electronics and Information Engineering,
Xian Jiaotong University, Xian, 710049, Shaanxi, China
[2]School of Computer Science and Engineering,
Xi'an University of Technology, Xi'an, 710048, Shaanxi, China

Abstract: Subspace clustering has been studied extensively and widely since traditional algorithms are ineffective in high-dimensional data spaces. Firstly, they were sensitive to noises, which are inevitable in high-dimensional data spaces; secondly, they were too severely dependent on some distance metrics, which cannot act as virtual indicators as in high-dimensional data spaces; thirdly, they often use a global threshold, but different groups of features behave differently in various dimensional subspaces. Accordingly, traditional clustering algorithms are not suitable in high-dimensional spaces. On the analysis of the advantages and disadvantages inherent to the traditional clustering algorithm, we propose a robust algorithm JPA (Joining-Pruning Algorithm). Our algorithm is based on an efficient two-phase architecture. The experiments show that our algorithm achieves a significant gain of runtime and quality in comparison to nowadays subspace clustering algorithms.

Key word: Subspace clustering, high-dimensional data, local feature relevance

INTRODUCTION

Clustering is an unsupervised learning process, in which data objects are partitioned into some unknown groups, such that all points within any such group are similar to each other, but dissimilar with respect to points in other groups. Many algorithms have already been proposed. All of these algorithms show acceptable performance on lower dimensional datasets, and they are based on the characteristics or features of the observed objects (Cheng *et al*, 1999). It is infeasible obviously for clustering objects in high-dimensional data. Firstly, if all of the features are equally coped with, the time required in clustering process will be exponential and even unacceptable; secondly, for some specific group of features, they may behave differently in different clusters; thirdly, the same features collection should be given different weight values because their importance in different clusters is different, which can be inferred from the second aspect; fourthly, most clustering approaches use a global density threshold for performance reasons (Procopiuc *et al*., 2002) and this is evidently unreasonable. Thus, clusters can no longer be found in the full- dimensional space.

Based on the analysis above, it is necessary to decide about the relevance of different features linked with the clustering process (Yang *et al*., 2002). In this research, we present a novel and robust subspace clustering algorithm JPA (Joining-Pruning Algorithm) which consists of joining and pruning two phases. It starts with 1D clusters that can be constructed with any previous clustering method and merges these 1D clusters to generate candidates of 2D clusters. Based on the 2D spaces and some prior knowledge, we prune some useless clusters, and make true 2D clusters. Like Analogously, we can cluster them in higher and higher dimensions step by step.

SUBSPACE CLUSTERING ALGORITHM

In the following, we assume that D is a database of n points in a d-dimensional feature space, i.e., $D \subseteq R^d$.

General idea: Clustering in high-dimensional spaces is more complicated than those in low-dimensional spaces due to its vast volume of the search and output space. The time consuming step is to identify the relevant attributes of the subspace cluster (Parsons *et al*, 2004). We claim that it is desirable to generate subspace cluster candidates quickly, and then to some clusters which is likely to be true clusters to comparatively degrees, we set about to cope with in details. This reduces the search space and especially the output space significantly. Our key idea is to use a joining-pruning architecture to speed up the whole process. Our algorithm JPA (Joining-Pruning Algorithm) consists of the following two steps:

Corresponding Author: Hongfang Zhou1, School of Electronics and Information Engineering, Xi'an University of Jiaotong, Xi'an, 710049, Shaanxi, China
* This work is supported by the National High Technology Research and Development (2001AA113182)

Joining: Firstly, we project the original data size(C_A)objects onto every dimension related to the d-dimensional spaces, and then compute all 1D clusters using some previous known clustering algorithm, such as k-means, k-medias, DBSCAN, CLIQUE, etc. The 1D clusters are then merged to find 2D clusters.

Pruning: As for the conventional clustering algorithms, there are some inherent disadvantages Therefore, it is inevitable there exist some meamingless or fault subclusters in the joining step. Accordingly, we adopt some pruning measures to detect and remove these clusters in this step.

Repeat the two steps above mentioned literatively, it is assured that the optimal results could be obtained until the clusters are stable.

Preprocessing: It is well founded that the information which is useful to cluster in high-dimensional data spaces is hidden in clusters of lower dimensional data spaces. Hence, our proposed algorithm JPA first generates cluster candidates in each dimension of the data space. In the preprocessing step, we can choose any known clustering method such as k-means, DBSCAN, grid-based and model-based clustering algorithm.

It is well known that density-based clustering satisfies downward closure property. In more details, if a group of data objects cannot compose a cluster in lower dimensional spaces, they are impossible to compose clusters in higher dimensional spaces. So in the preprocessing step, it is necessary to drop irrelevant clusters, which is impossible to be devoted as clusters later and affect the whole processing speed.

Joining: Candidates of higher dimensional clusters are determined by properly merging the lower dimensional clusters. If we cope with any cluster equally (search exhaustively all impossible combinations), the required time will increase crazily. Thus, we have to find the most potential candidates by searching for all clusters in lower dimensional data spaces. The similarity between two clusters is defined as: $C_A, C_B \in C^1$

Definition 1 (Inner-Similarity):

$$\text{Inner}-\text{Similarity}(C_A, C_B) = \frac{\sum \text{Cos}(c_A, c_B)}{\text{size}(C_A) * \text{size}(C_B)}$$

where C_A and C_B are two random cluster in one subspace. C_A and C_B denote the cluster center vectors of cluster C_A and C_B, respectively. size (C_A) and size (C_B) correspond to the size of cluster C_A and C_B.

Because the subspace generates dynamically, we need compute the similarity between every two clusters not only in the same subspace, but also in the different subspace. So besides the inner-similarity needs computed, the outer-similarity requires measured too. We define outer-similarity in different subspace as:

Definition 2 (Outer-Similarity):

$$\text{Outer}-\text{similarity}(C_A, C_B) = \frac{\sum \text{Cos}(c_A, c_B) \times \text{Cos}(s_A, s_B)}{\text{size}(C_A)/\text{size}(S_A) \times \text{size}(C_B)/\text{size}(S_B)}$$

Here, C_A and C_B are center vectors of cluster C_A and C_B, S_A and S_B are subspace center vectors of subspace S_A and S_B, respectively. Size (C_A) and size (C_B) are size of cluster C_A and C_B, size (S_A) and size (S_B) are size of subspace S_A and S_B.

Considering synthetically inner-similarity and outer-similarity, we will get the final score value.

Definition 3 (Score):

$$\text{Score}(C_A) = \sqrt{\text{Inner}-\text{Similarity}(C_A, \forall C_B \in S_A) \times \text{Outer}-\text{Similarity}(C_A, \forall C_C \in S_B)}$$

In contrast to functions commonly used to measure the similarity between sets, such as the Euclidean distance, our similarity function is insensitive to noise. This is necessary because the clusters usually contain a lot of noises on Web.

Before starting with the merge phase, we have to detect promising merge candidates, possibly hidden in the set of one-dimensional clusters. Different clusters may overlap in one or more dimensions, thus one lower-dimensional cluster can include objects of several higher-dimensional clusters. In our approach, we try to merge similar lower-dimensional clusters. In order to avoid that we merge all possible combinations, we have to split some one-dimensional cluster into several separate instances, such that one instance mainly contains some objects and the other contains the remaining objects. More generally, each one-dimensional cluster which includes more than one overlapping subspace cluster, should be split into multiple one-dimensional clusters in such a way, that each of them contains at most one of the higher-dimensional clusters.

Pruning: There is a true fact that vast volumes of noises exist in the data spaces. Therefore, there may be clusters which do not contain relevant information of the corresponding subspace cluster in the joining procedure

above. We must take measures to removes noises or irrelevant features for our specific tasks in the every step. And even in the later stages, we should also adopt some suitable measures to identify and remove meaningless or fault clusters. Therefore, we require some additional steps in order to achieve optimal results. Here, we present some preliminaries at first.

Definition 4 (Core object):

Let $\varepsilon \epsilon R^+$ and $\varepsilon \epsilon MinPts N^+$. If an object satisfies the condition $|N\varepsilon(o)| \geq$, we call the object ^{o} is a core object.

To some degrees, we think if there more core objects in a dataset, there exists a larger number of clusters, there exists the larger size of clusters, or there exists high density of clusters in a dataset. But they are not enough for us to determine the interestingness of a subspace. For example, if a dense region contains many objects in the -ε-neighborhood, but these objects are not core objects, this dense region is still an interested region. Hence, not only the number of core objects, but also the number of non-core objects are important indicators related to the interestingness of a cluster (Kailing *et al*, 2003). So we give a more reasonable definition about score of a subspace in definition 5.

Definition 5 (Score of a subspace):

The score of a subspace S_i, measuring the interestingness or preferences of a subspace is defined as

$$Score(S_i) = \frac{size(S_i)}{n \cdot \frac{Vol_e^{dim[S_i]}}{attrRange^{dim[S_i]}}} \cdot dim[S_i]$$

where $size(S_i)$ denotes the number of all data objects in the ε-neighborhood of all core-objects in the subspace S_i; $dim[S_i]$ denotes the dimensionality of subspace S; $Vol_e^{dim[si]}$ shows the volume of the hypersphere with radius ε and didimensions. In general, $attrRange^{dim[si]}$ is the Euclidian distance, and the Vol_e^{d} can be computed as:

$$Vol_e^d = \frac{\sqrt{\pi^d}}{\Gamma(d/2+1)} \cdot \varepsilon^d$$

where $\Gamma(x+1) = x \cdot \Gamma(x)$, $\Gamma(1) = 1$ and $\Gamma(0.5) = \sqrt{\pi}$.

The pruning removes a cluster candidate s resulting from the joining step above if the following condition holds:

$$score(S - \{s\}) > score(S) \wedge \forall s_p \in S:$$
$$s_p \neq s \Rightarrow score(S - \{s_p\}) \leq score(S - \{s\})$$

After removing the cluster candidate s, we continue the top-down pruning with the remaining merge candidates, until we achieve the best score. The score function avoids that large high-dimensional subspace clusters degenerate to low-dimensional clusters.

We achieve better subspace cluster results when applying an additional pruning step. First, we use the union of all clusters in lower-dimension in order to avoid that potential cluster candidates are lost or become uncompleted. Then, we cluster again the merged set of objects in the corresponding subspace. Thereby, we can apply any clustering algorithm, e.g. DBSCAN, SNN clustering, *k*-means, grid-based clustering or even hierarchical clustering. We propose the use of SNN to detect clusters of different density, or the use of DBSCAN. Thereby the density threshold is adjusted to the subspace dimensionality, i.e. the minPts-value is kept constant and we adjust the |Å-range as follows:

$$\varepsilon = \varepsilon_1 n / \sqrt[d]{n}$$

where ε_1 denotes the ε-value in the 1D subspace (Kailing *et al*., 2004).

RESULTS AND DISCUSSION

We evaluated JPA algorithm in comparison with CLIQUE from two aspects-the relation of run time with the number of observed objects and the relation of run time with the dimensionality. During evaluating we use datasets in a graduate school in the last three years. Obviously, to the same scale of data objects, JPA algorithm need less time to cluster compared to CLIQUE, which reveals our proposed algorithm use fully the ordered function of merge candidate to speed up the cluster procedure. It is also noted that JPA scales slightly ṣuperlinear w.r.t. the number of points in the dataset, more than CLIQUE (Fig. 1), which is caused by the decision of relevance of attributes. How to speed up pruning procedure is our future research focus.

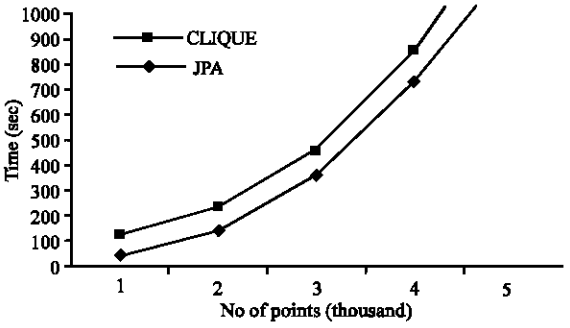

Fig. 1: Relation of number of points and their run time

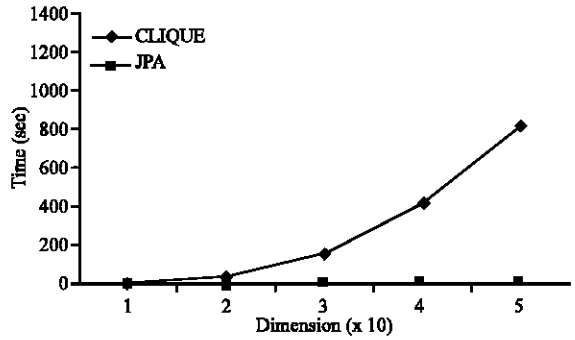

Fig. 2: Relation of data dimensionality and their run time

We investigate the scalability w.r.t. the data dimensionality using the same datasets. As it can be seen from Fig. 2, the runtimes of CLIQUE IN increase clearly superlinear, whereas the runtime of JPA increase only linear. This experiments show that JPA is the efficient algorithm that can be applied to a database in high dimensional spaces.

CONCLUSION

In this research, we proposed the new joining-pruning subspace clustering algorithm JPA, which overcomes problems of existing subspace clustering methods. JPA efficiently computes maximum dimensional cluster candidates from lower-dimensional spaces that can be pruned to obtain the true clusters in higher-dimensional spaces. Any clustering notion can be incorporated into JPA in order to account for clusters of varying density. A thorough experimental evaluation has shown that the effectiveness of JPA is significantly better than that of well-known algorithms such as CLIQUE.

REFERENCES

Cheng, C.H. *et al.*, 1999. Entropy-based subspace clustering for mining numerical data. In: Proc. ACM SIGKDD, pp: 84-93.

Kailing, K. *et al.*, 2003. Ranking interesting subspaces for clustering high dimensional data. In: Proc. 7th European Conf. On Principles and Practice of Knowledge Discovery in Databases (PKDD'03), pp: 241-252.

Kailing, K. *et al.*, 2004. Density- connected subspace clustering for high-dimensional data. In: Proc. SIAM Data Mining, pp: 246-256.

Procopiuc, C.M *et al.*, 2002. A monte carlo algorithm for fast projective clustering. In: Proc. ACM SIGMOD, pp: 418-427.

Parsons, L. *et al.*, 2004. Subspace clustering for high dimensional data: A review. SIGKDD Explorations, 6: pp: 90-105.

Yang, J. *et al.*, 2002, delta- clusters: capturing subspace correlation in a large data set. In: Proc. ICDE, pp: 517-528.