Adapting K-Means Algorithm for Discovering Clusters in Subspaces

Yanchang Zhao[1], Chengqi Zhang[1], Shichao Zhang[1] and Lianwei Zhao[2]

[1] Faculty of Information Technology, University of Technology, Sydney, Australia
{yczhao, chengqi, zhangsc}@it.uts.edu.au
[2] Dept. of Computer Science, Beijing Jiaotong University, Beijing 100044, China
lw.zhao@163.com

Abstract. Subspace clustering is a challenging task in the field of data mining. Traditional distance measures fail to differentiate the furthest point from the nearest point in very high dimensional data space. To tackle the problem, we design minimal subspace distance which measures the similarity between two points in the subspace where they are nearest to each other. It can discover subspace clusters implicitly when measuring the similarities between points. We use the new similarity measure to improve traditional k-means algorithm for discovering clusters in subspaces. By clustering with low-dimensional minimal subspace distance first, the clusters in low-dimensional subspaces are detected. Then by gradually increasing the dimension of minimal subspace distance, the clusters get refined in higher dimensional subspaces. Our experiments on both synthetic data and real data show the effectiveness of the proposed similarity measure and algorithm.

1. Introduction

As a main technique for data mining, clustering is confronted with increasingly high dimensional data. The dimension of data can be hundreds or thousands in the fields of retail, bioinformatics, telecom, etc., which brings the "curse of dimensionality". It not only makes the index structure less efficient than linear scan, but also questions the meaningfulness of looking for the nearest neighbor [5], which in turn makes it ineffective to discover clusters in full dimensional space. The key point lies in that traditional distance measures fail to differentiate the nearest neighbor from the farthest point in very high-dimensional space. One solution is to measure the distance in subspaces, but it is not easy to select the appropriate subspaces. Fern et al. proposed random projection by choosing subspaces randomly and then the results of several random projections are combined in an ensemble way [3]. Procopiuc et al. chose the subspaces where a random group of points

```
Algorithm: k-means
Input: The number of clusters k and a dataset
Output: A set of clusters that minimizes the squared-
error criterion.
1. Select k objects as initial cluster centers;
2. Assign each data object to the nearest center;
3. Update the cluster center as the mean value of the
objects for each cluster;
4. Repeat steps 2 and 3 until centers do not change or
the criterion function converges.
```

Fig. 1. K-means algorithm

are in a ω-width hyper-rectangular box [7]. Agrawal et al. [2] proposed to discover the subspaces in an APRIORI-like way.

To tackle the above problem, we design a new similarity measure, *minimal subspace distance*, for measuring the similarities between points in high dimensional space and discovering subspace clusters. The new measure defines the minimal *l*-D distance between two points as the minimum of their distances in all *l*-D subspaces and thus discovers implicitly the subspace of clusters while computing similarities. Based on our new similarity measure, k-means algorithm is improved for discovering subspace clusters in high dimensional space. Our experiments on both synthetic data and real-life data show the effectiveness of the proposed similarity measure and algorithm.

2. K-Means Algorithm

K-means algorithm is one of the most well-known and widely used partitioning methods for clustering. It works in the following steps. First, it selects *k* objects from the dataset, each of which initially represents a cluster center. Each object is assigned to the cluster to which it is most similar, based on the distance between the object and the cluster center. Then the means of clusters are computed as the new cluster centers. The process iterates until the criterion function converges. A typical criterion function is the squared-error criterion, defined as

$$E = \sum_{i=1}^{k} \sum_{p \in C_i} | p - m_i |^2 \qquad \textbf{(1)}$$

where E is the sum of square-error, p is a point, and m_i is the center of cluster C_i. The k-means algorithm is given in Figure 1. For detailed description of k-means clustering, please refer to [4].

3. Adapting K-Means Algorithm for Subspace Clustering

In this section, a new similarity measure, *minimal subspace distance*, will be proposed to discover clusters in subspaces. Based on the new similarity measure, k-means algorithm will be adapted for discovering subspace clusters.

3.1. Motivation

Euclidean distance is the mostly used distance measure in the field of data mining. However, the difference between the nearest point and the farthest one becomes less discriminating with the increase of dimensionality [5]. It is the same case with Minkowski distance (L_p-norm, p=2,3,...), except the Manhattan distance (p=1). Aggarwal et al. suggested to use fractional distance metrics (i.e., L_p-norm with $0<p<1$) to measure the similarity between objects in high dimensional space [1].

Nevertheless, many researchers argued that most meaningful clusters only exist in subspaces for very high dimensional data, so they used the traditional L_p-norm (p=1,2,3,...) to discover clusters in subspaces [2, 3, 6, 7]. For subspace clustering in high-dimensional space, clusters are constrained to be axis-paralleled hyper-rectangles in subspaces by Agrawal et al. [2], and projective clusters are defined as axis-aligned box by Procopiuc et al. [7]. Therefore, it is reasonable to define a cluster to be the union of those objects which are within a subspace hyper-rectangle.

What if the subspace of clusters is unknown in advance? In which subspace should the objects be projected? To discover subspace clusters, a new similarity measure, *minimal subspace distance*, is defined in the following, which can improve traditional L_p-norm (p=1,2,3,...) for subspace clustering in high-dimensional space.

3.2. Minimal Subspace Distance

Definition 1. [Minimal subspace distance] Assume that X and Y are two points in a d-dimensional space, and the coordinates of them are $(x_1,x_2,...,x_d)$ and $(y_1,y_2,...,y_d)$, respectively. The minimal l-D subspace distance between X and Y is defined as the minimum of the distances between them in all l-dimensional subspaces, as given by the following formula.

$$MSD^{(l)}(X,Y) = \min_{S_l}\left(dist(X_{S_l}, Y_{S_l})\right) \quad (2)$$

where $S_l=(j_1,j_2,...,j_l)$ is a l-dimensional subspace, X_{S_l} and Y_{S_l} are respectively the projected vectors of X and Y in subspace S_l, and $dist(\cdot)$ is a traditional distance measure.

It is obvious that minimal subspace distance meets two of the requirements of distance metric, non-negativity and symmetry. However, it does not satisfy the triangle inequality. The reason lies in that it measures the similarity between two points in the subspace where they are nearest to each other and that the subspaces are usually different for different pairs of points. It discovers subspaces of clusters implicitly while measuring the similarities between points by using the subspace in which points are nearest. Therefore, it is effective to discover subspace clusters, although it is not really a distance metric.

When L_p-norm is used as the measure of distance, the minimal subspace distance is the L_p-norm distance calculated with the l minimal differences between each pair of x_i and y_i, as the following formula shows.

$$MSD_p^{(l)}(X,Y) = \left(\sum_{i=1}^{l} | x_{j_i} - y_{j_i} |^p\right)^{1/p} \quad (3)$$

where j_i (i=1..l) are the first l dimensions when sorting $| x_{j_i} - y_{j_i} |$ in ascending order. If maximum distance (L_∞-norm) is used as the measure of similarity, the minimal l-D distance is the maximum of the l minimal differences between each pair of x_i and y_i, that is, the l-th minimum of $|x_i - y_i|$ (i=1..d). Therefore, $MSD_\infty^{(l)}(X,Y) \le \varepsilon$ means that X and Y are in a hyper-rectangle with edge of ε in l dimensions and without any limits in other dimensions. Therefore, the above similarity measure provides an effective measure for hyper-rectangular clusters in subspaces.

With the help of the above minimal subspace distance, it will be easier to discover clusters in subspaces. For two objects, it finds the subspace in which they are the most similar or nearest to each other. Assume that L_∞-norm is used. For example, if the minimal 4-D subspace distance between two objects is 7, it means that the two objects are within a 4-D hyper-rectangle with edge length of 7.

Minimal subspace distance measures the similarity between objects in the subspace where objects are nearest to each other, so it is effective to find subspaces where clusters exist and then discovers clusters in these subspaces. With the new definition of similarity measure, our algorithm is capable of finding projected clusters and subspaces automatically when the average dimensionality of subspaces is given. The effectiveness of the new similarity measure will be shown in our experiments.

3.3. Adapted K-Means Algorithm

Based on the above minimal subspace distance, we adapt the well-known k-means algorithm for discovering clusters in subspaces. Traditional k-means algorithm cannot discover subspace clusters because it uses full-dimensional distance measure to compute the similarity between points. In most cases, different clusters usually exist in different subspaces and the dimensions of subspaces also vary from cluster to cluster. Therefore, subspaces should also be discovered while clustering data. Our idea is to use minimal subspace distance to discover the subspace implicitly when measuring the similarity between points. First, we run k-means algorithm with low-dimensional minimal subspace distance and the clusters in low-dimensional subspaces are discovered. Then, by clustering with increasingly higher-dimensional minimal subspace distance, the clustering gets refined. The algorithms for adapted k-means are shown in Figures 2 and 3.

If running k-means with minimal 1-D distance at first, and increasing the dimension of subspace by one at each step, it will be very costly when the dimension of data is high. Moreover, it is usually meaningless to discover a very low dimensional (say 3D) cluster in high dimensional (say 500D) data. Therefore, the minimal dimension of clusters *minl* is set as a start point and users can set the value of *minl* according to the specific dataset and application. In addition, *maxl*, the maximal dimension of clusters, is also provided to set a limit to the

```
Algorithm: Adapted k-means
Input: dataset X, cluster number k
Output: k centroids C={c_i} and cluster IDs

Decide minl, maxl, and stepl for clustering;
l = minl;
prevSumDist = Infinity;
Randomly select k points from X as C;
WHILE TRUE
    FOR each pair of point p_i and centroid c_j
        dist(p_i,c_j) = MSD(p_i,c_j, l); /*minimal l-D
            distance, see Figure 3 for detail*/
    ENDFOR
    FOR each point p_i
        clusterId(i) = t if dist(p_i,c_t)=min_j{dist(p_i,c_j)};
    ENDFOR
    sumDist =  sum of point to centroid distances;
    IF sumDist < prevSumDist
        prevSumDist = sumDist;
        prevClusterId = clusterId;
        FOR i=1 TO k
            c_i=the mean of those points in cluster i;
        ENDFOR
    ELSE
        l = l + stepl;
        If l > maxl
            break;
        ENDIF
    ENDIF
ENDWHILE
RETURN C and prevClusterId;
```

Fig. 2. Adapted k-means algorithm

```
Algorithm: MSD, which computes the minimal l-D
distance between two points p and q
Input: point p and q, dimension l
Output: the minimal l-D distance between p and q

Set diff_i (i=1..l) to be the minimal l values of
|p_i-q_i| (i=1,2…d);
dist = sqrt(sum_{i=1}^{l} diff_i^2) ;
RETURN dist;
```

Fig. 3. MSD algorithm

maximal dimension of subspaces. The default values of *minl* and *maxl* are respectively 1 and *d* (the dimension of dataset), if it is difficult to set appropriate values for them. For very high-dimensional data, it is

computing expensive to increase l by one at each step. Furthermore, the clustering at the next step gets little refined when increasing l to l+1. Hence, another parameter, *stepl*, is used as the increasing stride of l and the default value of *stepl* is $\lceil (maxl - minl)/10 \rceil$. That is, the traditional k-means algorithm will run for 10 times. With the introduction of *stepl*, the algorithm runs less iterations and the efficiency gets improved.

4. Experiments

The algorithm is implemented with Matlab and our experiments are performed on a PC with 256MB RAM and an Intel Pentium IV 1.6GHz CPU. These experiments show the superiority of our algorithm over traditional k-means algorithm for discovering clusters in subspaces.

4.1. Synthetic Data Generator

We use the *nngenc(X,C,N,D)*[1] function from Matlab[2] to generate clusters of data points, where X is a $R \times 2$ matrix of cluster bounds, C is the number of clusters, N is the number of data points in each cluster, and D is the standard deviation of clusters. The function returns a matrix containing $C \times N$ R-element vectors arranged in C clusters with centers inside bounds set by X, with N elements each, randomly around the centers with standard deviation of D. The range is set to [0, 100]. To generate subspace clusters, we set the values in some dimensions for some clusters to be of uniform distribution, and the subspaces vary from cluster to cluster. Since k-means algorithm partitions the whole dataset into k clusters and cannot eliminate noises, no noise is generated in the datasets.

4.2. Evaluation Criterion

Conditional Entropy (CE) and *Normalized Mutual Information (NMI)* are employed to measure the quality of clustering, since the clusters are known before hand and can be used to judge the clustering quality. *Compactness* [9] is also widely used to measure the quality of

[1] Detailed information can be found in ".\toolbox\nnet\nndemos\nngenc.m" in Matlab v7.0.1.
[2] http://www.mathworks.com/

clustering, but it favors sphere-shaped clusters since the diameter is used. *CE* and *NMI* have been used to measure the quality of clustering by Strehl et al [8] and Fern et al [3], and detailed description of the two measures can be found in the above two papers. Conditional entropy measures the uncertainty of the class labels given a clustering solution. For one clustering with m clusters and another with k clusters, the conditional entropy is defined as $CE = \sum_{j=1}^{k} \frac{n_j \times E_j}{n}$, where entropy $E_j = -\sum_{i=1}^{m} p_{ij} \log(p_{ij})$, n_i is the size of cluster i in the first clustering solution, n_j is the size of cluster j in the second clustering solution, p_{ij} is the probability that a member of cluster i in the first clustering belongs to cluster j in the second clustering, and n is the size of dataset. The value of *CE* is a non-negative real number. The less *CE* is, the more the tested result approaches the standard result. The two results become the same when *CE* is zero.

For two clustering solutions X and Y, the normalized mutual information is defined as $NMI = \frac{MI}{\sqrt{H(X)H(Y)}}$, where mutual information $MI = \sum_{i,j} p_{ij} \log\left(\frac{p_{ij}}{p_i p_j}\right)$, and $H(X)$ and $H(Y)$ denote the entropies of X and Y. The value of *NMI* lies between zero and one. Contrary to *CE*, the larger the value of *NMI* is, the better is the clustering. If *NMI* is one, then the two clustering solutions are exactly the same. The values of *CE* and *NMI* between the actual clustering and the discovered clusters are used to judge the clustering accuracy. Therefore, a better clustering solution is of greater *NMI* and less *CE*.

4.3. Experimental Results

A 20D dataset with four clusters in 16D subspaces is used in the first experiment. There are 1000 points in the dataset, with 250 points in each cluster. The four actual clusters in generated dataset are show with parallel coordinates in Figure 4, and the number above each subfigure is the count of points in the cluster. *Minl* and *maxl* are respectively set to 1 and 16, and *stepl* is set to 1. The clusters discovered by our

algorithm and traditional k-means algorithm are shown in Figure 5 and 6, respectively. From Figure 4 and 5, the four clusters discovered by our algorithm are nearly the same as those actual clusters, except for only one point in cluster 2 which is wrongly assigned to cluster 4. From Figure 4 and 6, cluster 3 in Figure 4 is wrongly split into two clusters (clusters 1 and 4 in Figure 6), while clusters 2 and 4 in Figure 4 are wrongly merged into one cluster (cluster 2 in Figure 6) by traditional k-means algorithm. The confusion matrixes for the above clustering results are shown respectively in Table 1 and 2. In the two tables, columns C1-C4 stand for actual clusters, while rows D1-D4 stand for the clusters discovered. The numbers in the table are the counts of points in the joint part of the two corresponding clusters. The *CE* values of traditional and adapted algorithms are respectively 0.3466 and 0.0065, while the *NMI* values of them are respectively 0.8022 and 0.9953, which also validates that our algorithm is superior to traditional k-means algorithm for discovering clusters in subspaces.

The second experiment is conducted on synthetic data of 1000 points and the dimensions range from 20 to 800. The standard deviation is set to 0.12 for generating all these datasets. There are four equal-sized clusters in each dataset, and the clusters exist in different subspaces. The dimensions of the clusters are set to be 0.3 times the dimensions of datasets. There is no noise generated in the datasets. *Minl* is set to 1, *maxl* is set to the dimensionality of subspace clusters, and *stepl* is set to the default value, $\lceil (maxl - minl)/10 \rceil$. Ten datasets are generated for each dimensionality and each algorithm runs for 20 times on each dataset. The average experimental result is shown in Figure 7. There are nine groups in the figure, and groups 1-9 stand for the results on 20D, 40D, 60D, 80D, 100D, 200D, 400D, 600D and 800D data, respectively. The four bars in each group from left to right denote *tradCE* (*CE* of traditional k-means clustering), *tradNMI* (*NMI* of traditional k-means clustering), *adptCE* (*CE* of adapted k-means clustering) and *adptNMI* (*NMI* of adapted k-means clustering).

From Figure 7, it is clear that *adptCE* is less than *tradCE* and *adptNMI* is greater than *tradNMI* in most conditions, especially for datasets of higher dimensions. Therefore, our adapted k-means algorithm is more effective than traditional k-means algorithm for discovering clusters in subspaces and our adapted algorithm performs better than the traditional k-means algorithm for high-dimensional datasets. We can also see from the figure that, with the increase of dimension, *tradCE*

and *adptCE* decrease, while *tradNMI* and *adptNMI* increase. It seems that both traditional and adapted k-means algorithm performs better with the increase of dimension. The reason lies in that the points in a cluster tend to become more compact with the increase of dimension when the standard deviation remains unchanged.

Experiments are also conducted on datasets with the dimensions of clusters respectively 0.2, 0.4, and 0.5 times the dimensions of datasets. Each algorithm runs for 20 times with each dataset and the average experimental results are shown in Figure 8-10. The same conclusion can be drawn as that from the second experiment. In addition, by comparing Figures 7-10, we can see that, when the dimension of data space remains unchanged, our algorithm performs better if the dimension of subspace clusters is higher.

4.4. Experiment on Real Data

The dataset of Wisconsin Diagnostic Breast Cancer from UCI machine learning repository (http://www.cs.uci.edu/~mlearn) is used in the experiment. The dataset is of 569 distances and 32 attributes. The first attribute is the ID of instance. The second attribute is the diagnosis class and there are two classes, "malignant" and "benign", in the dataset. The following 30 attributes are real-valued features. The first two attributes are removed before clustering and the diagnosis class is used only to check the accuracy of clustering. By setting the *minl*, *maxl* and *stepl* to 20, 30 and 1 respectively, two clusters are always discovered effectively. We run the algorithm for 30 times and the average accuracy is 97.3%, which shows the effectiveness of our algorithm.

Table 1. Confusion matrix of the clustering result of adapted k-means. The four clusters discovered are the same as those in the original dataset, except for only one point from C4.

	C1	C2	C3	C4
D1	250	0	0	0
D2	0	0	0	249
D3	0	0	250	0
D4	0	250	0	1

Table 2. Confusion matrix of the clustering result of traditional k-means. C2 and C4 are clustered into one group D2, while C3 is split into two groups, D1 and D4.

	C1	C2	C3	C4
D1	0	0	137	0
D2	0	250	0	250
D3	250	0	0	0
D4	0	0	113	0

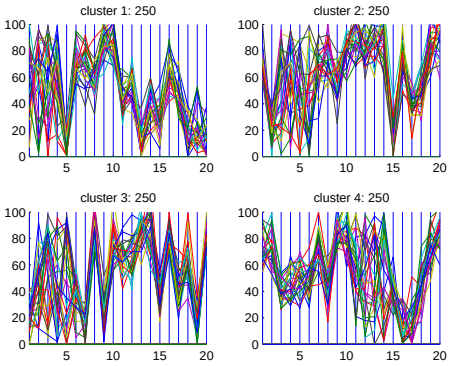

Fig. 4. Actual clusters. The four subfigures show the actual clusters in the dataset.

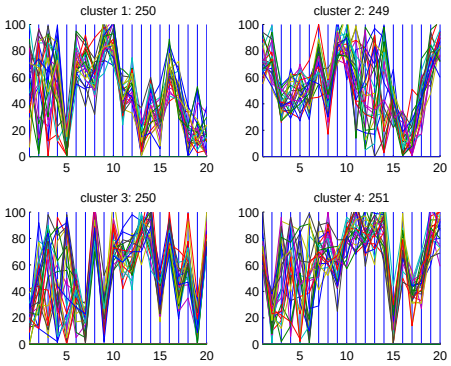

Fig. 5. Four clusters discovered by adapted k-means. The above clusters 1-4 are corresponding to clusters 1, 4, 3, 2 in Fig. 4, respectively.

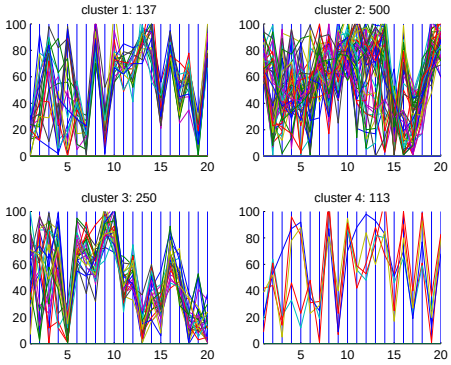

Fig. 6. Four clusters discovered by traditional k-means. The above cluster 3 is corresponding to cluster 1 in Fig. 4. Cluster 2 is the union of clusters 2 and 4 in Fig. 4, while clusters 1 and 4 are two parts of cluster 3 in Fig. 4.

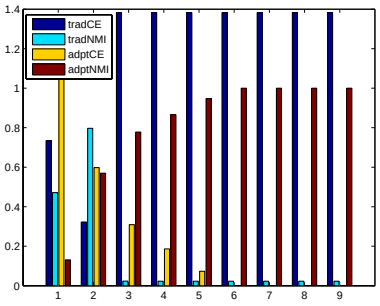

Fig. 7. Experimental result I. Groups 1-9 stand for 9 datasets with increasing dimensions and the four bars in each group from left to right denote *tradCE*, *tradNMI*, *adptCE* and *adptNMI*. The dimensions of clusters are 0.3 times the dimensions of datasets.

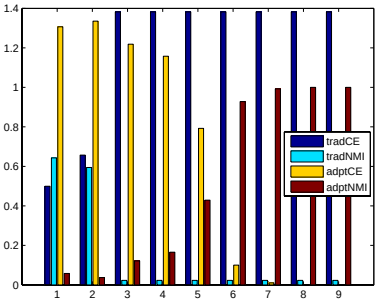

Fig. 8. Experimental result II. The dimensions of clusters are 0.2 times the dimensions of datasets.

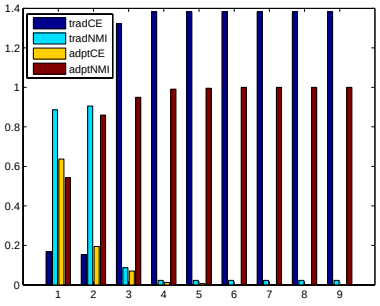

Fig. 9. Experimental result III. The dimensions of clusters are 0.4 times the dimensions of datasets.

5. Conclusion

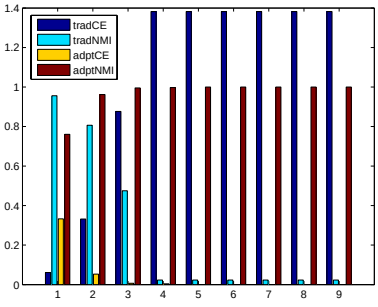

Fig. 10. Experimental result IV. The dimensions of clusters are 0.5 times the dimensions of datasets.

We designed a new similarity measure to discover clusters in subspaces and our proposed algorithm runs k-means clustering with increasing subspace dimensions. The experiments show that our algorithm performs better than traditional k-means algorithm for discovering clusters in subspaces and that the superiority of our algorithm becomes greater with the increase of dimension of data.
Our future work includes analyzing the characteristics and distribution of minimal subspace distance and applying it to other existing clustering algorithms for discovering subspace clusters.

References

[1] Charu C. Aggarwal, Alexander Hinneburg, and Daniel A. Keim, “On the Surprising Behavior of Distance Metrics in High Dimensional Space”. *In Proc. of the 8th International Conference on Database Theory*, 2001.

[2] R. Agrawal, J. Gehrke, et al., “Automatic subspace clustering of high dimensional data for data mining applications”. *In Proc. 1998 ACM-SIGMOD Int. Conf. Management of Data (SIGMOD'98)*, Seattle, WA, June 1998, pp.94-105.

[3] Xiaoli Zhang Fern, and Carla E. Brodley, “Random Projection for High Dimensional Data Clustering: A Clustering Ensemble Approach”, *In Proc. 20th Int. Conf. On Machine Learning (ICML'03)*, Washington DC, 2003.

[4] Jiawei Han, Micheline Kamber, *Data Mining: Concepts and Techniques*, Higher Education Press, Morgan Kaufmann Publishers, 2001.

[5] Alexander Hinneburg, Charu C. Aggarwal, and Daniel A. Keim, "What is the nearest neighbor in high dimensional spaces?", *In Proc. of the 26th International Conference on Very Large Data Bases*, Cairo, Egypt, 2000, pp 506-515.

[6] H. Nagesh, S. Goil, and A. Choudhary, “MAFIA: Efficient and Scalable Subspace Clustering for Very Large Data Sets”. *Technical Report 9906-010, Northwestern University*, June 1999.

[7] Magda Procopiuc, Michael Jones, Pankaj Agarwal, and T. M. Murali, “A Monte-Carlo Algorithm for Fast Projective Clustering”. *In Proc. of the 2002 International Conference on Management of Data*, 2002.

[8] A. Strehl, and J. Ghosh, “Cluster ensembles - a knowledge reuse framework for combining multiple partitions”. *Machine Learning Research*, 3, 2002, pp. 583-417.

[9] Mohamed Zait, and Hammou Messatfa, “A comparative study of clustering methods”. *Future Generation Computer Systems*, Vol. 13, 1997, pp. 149-159.