Journal of Computer Science 4 (3): 252-255, 2008
ISSN 1549-3636

An Efficient Approach for Computing Silhouette Coefficients

Moh'd Belal Al- Zoubi and Mohammad al Rawi
Department of Computer Information Systems, University of Jordan, Amman 11942, Jordan

Abstract: One popular approach for finding the best number of clusters (K) in a data set is through computing the silhouette coefficients. The silhouette coefficients for different values of K, are first found and then the maximum value of these coefficients is chosen. However, computing the silhouette coefficient for different Ks is a very time consuming process. This is due to the amount of CPU time spent on distance calculations. A proposed approach to compute the silhouette coefficient quickly had been presented. The approach was based on decreasing the number of addition operations when computing distances. The results were efficient and more than 50% of the CPU time was achieved when applied to different data sets.

Key words: Silhouette coefficients, clustering, data mining, pattern recognition

INTRODUCTION

Clustering, also called unsupervised learning, is defined as the process of grouping a set of objects into classes (groups) of similar objects, such that the objects in a group will be similar (or related) to each other and different from (or unrelated to) the objects in other groups[1].

However, in most clustering algorithms (e.g., K-means and PAM), usually one does not know the number of clusters (or groups), in a set[2-4]. These algorithms provide a fixed value of k in advance.

Finding the right number of clusters is a challenging issue in cluster analysis literature, for which no unique solution exists[4,5]. Therefore, different approaches have been proposed[4-7]. One of the most popular methods to select the right value of K is by means of the silhouette coefficients[4,8-11]. For a given point i in a cluster A, the silhouette of i, s(i) is defined as follows[4]:

$$s(i) = \frac{b(i) - a(i)}{\max\{a(i), b(i)\}}$$

where, a(i) is the average dissimilarity between point i and all other points in A (the cluster to which i belongs) and b(i) is the average dissimilarity between point i and the points in the closest cluster to A, which is B in this case.

The average of all silhouettes in the data set S'is called the average silhouettes width for all points in the data set. The value S' will be denoted by S'(K), which is used for the selection of the right value of the number of clusters, K, by choosing that k for which S'(K) is as high as possible. The Silhouette Coefficient (SC) is then defined as follows:

$$SC = \max\{S`(k)\}$$

where, the maximum is taken over all K for which the silhouettes can be constructed, which means K = 2, 3, …, n-1[4].

As illustrated above, it is clear that distance calculations (usually Euclidean distance) from each point, i, to the points in the current cluster (the cluster to which i belongs) and the points in the neighboring clusters to calculate the silhouette coefficients for one value (one run) of K. This process is repeated for many values (many runs) of K, causing a long CPU time to spend on distance calculations.

In this article, a new approach had been proposed to compute the silhouette coefficients quickly. The approach is based on omitting some of the addition operations.

PROPOSED APPRAOCH

The Euclidean distance between a query point q and a data point x, in a D-dimensional space R^D, is given as follows:

Applying Eq. 1, the calculation of one distance involves D multiplications, D additions and D subtraction, where D is the number of variables (dimensions). For computing N distances, then ND

Corresponding Author: Moh'd Belal Al- Zoubi, Department of Computer Information Systems, University of Jordan, Amman 11942, Jordan Tel: +962-6-5355000/22551 Fax: +962-5354070

multiplications, ND additions and ND subtraction are involved. For large data sets with a large number of dimensions, the computation of distances requires a long CPU time, which hinders the development of effective processes that involve distance computations, which is the case when computing the silhouette coefficients in this study:

$$d(q,x) = \sum_{j=1}^{D}(q_j - x_j)^2 \quad (1)$$

In the proposed approach, equation (1) can be written as follows[12]:

$$d(q, x) = \sum_{j=1}^{D} q_j^2 - 2\sum_{j=1}^{D} q_j x_j + \sum_{j=1}^{D} x_j^2 \quad (2)$$

or as:

$$d(q, x) = \sum_{j=1}^{D} q_j^2 - W\sum_{j=1}^{D} x_j + \sum_{j=1}^{D} x_j^2 \quad (3)$$

where, $W = 2\sum_{j=1}^{D} q_j$.

The first and third terms of Eq. 3 can be calculated only once (for all runs) for the whole data set in a pre-processing step and stored as dimensionless quantities. This is also valid for computing W. Therefore, only the second term of Eq. 3 is calculated in the run time and hence, ND additions can be saved and the performance of the distance computations is expected to increase.

RESULTS AND DISCUSSION

We had investigated the efficiency of the new proposed approach, compared to the conventional (exhaustive) approach, when applied on different data sets to compute the silhouette coefficients. The proposed approach had generated outputs that are identical to the outputs of the conventional approach. The performance of the proposed approach had been reported in terms of the CPU time and the percentage of savings compared to the conventional approach.

In our tests, six data sets had been tested to compute the silhouette coefficients. The first two sets had randomly been generated while the other four sets had been obtained from the UCI Repository of Machine Learning Databases[13]. These are Breast, Letter, Pima and Segmentation data sets. The description of these data sets is shown in Table 1, where N is the number of points and D represents the dimensionality (number of dimensions) of data.

Table 1: Description of datasets

DataSet	N	D
Rnd1	5000	2
Rnd2	10000	4
Breast	699	10
Letter	20000	16
Pima	768	8
Segmentation	2310	19

Table 2: The CPU run time (in seconds) of the proposed and conventional approaches when applied on the Rnd1 data set for a different number of runs

Runs	Conventional	Proposed	(%) Savings
1	43	28	35
2	85	57	33
3	128	85	34
4	168	113	33
5	210	141	33

Table 3: The CPU run time (in seconds) of the proposed and conventional approaches when applied on the Rnd2 data set for a different number of runs

Runs	Old	New	(%) Savings
1	285	162	43
2	586	339	42
3	882	501	43
4	1158	665	43
5	1439	826	43

Table 4: The CPU run time (in seconds) of the proposed and conventional approaches when applied on the Pima data set for a different number of runs

Runs	Old	New	(%) Savings
1	4	2	50
2	7	3	57
3	10	5	50
4	14	6	57
5	17	8	53

Table 2 shows the CPU run time (in seconds) for the proposed approach and conventional (exhaustive) approach when applied on the Rnd1 data set. It shows that the performance of the proposed approach has a good speed improvement over the conventional approach in all cases. The Table shows that up to 35% of the CPU time savings had been achieved.

Table 3 shows the CPU run time (in seconds) for the proposed and conventional approaches when applied on the Rnd2 data set. It shows that the performance of the proposed approach has a very good speed improvement over the conventional approach in all cases. The Table shows that up to 43% of the CPU time savings had been achieved.

Table 4 shows the CPU run time (in seconds) for the proposed and conventional approaches when applied on the Pima data set. It shows that the performance of the proposed approach has a significant speed improvement over the conventional approach in

Table 5: The CPU run time (in seconds) of the proposed and conventional approaches when applied on the Breast data set for a different number of runs

Runs	Old	New	(%) Savings
1	3	1.5	50
2	6	3	50
3	9	4	56
4	12	6	50
5	15	7	53

Table 6: The CPU run time (in seconds) of the proposed and conventional approaches when applied on the Letter data set for a different number of runs

Runs	Old	New	(%) Savings
1	3770	1859	51
2	7683	3743	51
3	11440	5604	51
4	15251	7324	52
5	19425	9352	52

Table 7: The CPU run time (in seconds) of the proposed and conventional approaches when applied on the Segmentation data set for a different number of runs

Runs	Conventional	Proposed	(%) Savings
1	60	29	52
2	122	58	52
3	183	87	52
4	254	113	56
5	305	143	53

all cases. The Table shows that up to 57% of the CPU time savings had been achieved.

Table 5 shows the CPU run time (in seconds) for the proposed and conventional approaches when applied on the Breast data set. It shows that the performance of the proposed approach has a significant speed improvement over the conventional approach in all cases. The Table shows that up to 56% of the CPU time savings had been achieved.

Table 6 shows the CPU run time (in seconds) for the proposed and conventional approaches when applied on the Letter data set. It shows that the performance of the proposed approach has a significant speed improvement over the conventional approach in all cases. The Table shows that up to 52% of the CPU time savings had been achieved.

Table 7 shows the CPU run time (in seconds) for the proposed and conventional approaches when applied on the Segmentation data set. It shows that the performance of the proposed approach had a significant speed improvement over the conventional approach in all cases. The Table shows that up to 56% of the CPU time savings had been achieved.

It can be noticed from the results presented in the tables above that the performance of the proposed approach increases when the dimensionality increases. This is expected since the CPU time savings is based on decreasing the addition operations mainly spent on operations regarding dimensions.

CONCLUSION

A proposed approach to compute the silhouette coefficient quickly had been presented. The approach is based on decreasing the number of addition operations when computing distances. The results were efficient and more than 50% of the CPU time had been achieved when applied to different data sets. However, some extra memory is needed to store the data from the pre-processing step discussed earlier in this and. This will be handled in future works.

REFERENCES

1. Han, J. and M. Kumar, 2006. Data Mining-Concepts and Techniques, 2nd ed. Morgan Kaufman. ISBN: 13: 9781558609013.
2. Xu, R. and D. Wunsch , 2005. Survey of clustering algorithms. IEEE Trans. Neural Networks, 16: 645-678. DOI: 10.1109/TNN.2005.845141.
3. MacQueen, J., 1967. Some methods for classification and analysis of multivariate observations. In: Proc. 5th Berkeley Symp. Math. Stat. and Prob., pp: 281-97.
4. Kaufman, L. and P. Rousseeuw, 1990. Finding Groups in Data: An Introduction to Cluster Analysis. John Wiley and Sons, London. ISBN: 10: 0471878766.
5. Milligan, G. and M. Cooper, 1985. An examination of procedures for determining the number of clusters in a data set. Psychometrika, 50: 159-179. DOI: 10.1007/BF02294245.
6. Massart, D., F. Plastria and L. Kaufman, 1983. Non-hierarchical clustering with MASLOC. Patt. Recog., 16: 507-516.
7. Klastorin, T., 1985. The p-median problem for cluster analysis: A comparative test using the mixture model approach. Manage Sci., 31: 84-95. ISSN: 00251909.
8. Raskutti, B. and C. Leckie, 1999. An evaluation of criteria for measuring the quality of clusters. In: Proceedings of the 16th International Joint Conference on Artificial Intelligence, pp: 905-910. ISBN:1-55860-613-0.
9. Miller, H. and J. Han, 2001. Geographic data mining and knowledge discovery, CRC Press, New York. ISBN: 0415233690, 9780415233699.

10. Iomini, C., V. Babaev-Khaimov, M. Sassaroli and G. Piperno, 2001. Protein particles in chlamydomonas flagella undergo a transport cycle consisting of four phases. J. Cell Biol., 153: 13-24.
11. Adam, N., V. Janeja and V. Atluri, 2004. Neighborhood based detection of anomalies in high dimensional spatio-temporal sensor datasets. In: Proceedings of the 2004 ACM Symposium on Applied Computing, pp: 576-583 DOI: 10.1145/967900.968020.
12. Venkateswarlu, N. and P. Raju, 1993. A new fast classifier for remotely sensed imaginary. Int. J. Remote Sens., 14: 383-389. DOI: 10.1080/01431169308904343.
13. Blake, C.L. and C.J. Merz, 1998. UCI Repository of Machine Learning Databases. http://archive.ics.uci.edu/ml/datasets.html, University of California, Irvine, Department of Information and Computer Sciences.