KNN-DBSCAN: Using k-nearest Neighbor Information for Parameter-free Density Based Clustering

Ankush Sharma
M.Tech. Scholar
Vedant College of Engineering and Technology
Kota, India
ankushsharmajec@gmail.com

Amit Sharma
Associate Professor
Vedant College of Engineering and Technology
Kota, India
amittechnosoft@gmail.com

***Abstract*—Density based clustering is adopted in situations where clusters of arbitrary shape exist. DBSCAN is a popular density concept but suffers from the drawback of dependence on user-defined parameters like many other density based methods. In order to utilize the potential of this clustering method we propose a combination method. The information of k-nearest neighbors is used with DBSCAN to achieve a parameter-free clustering technique. The parameters are set according to information of the data as it gets accumulated in a cluster structure.**

Keywords—clustering, density-based, DBSCAN, knn, dominant sets, graph-based clustering

I. INTRODUCTION

The conventional algorithms work well under certain assumptions about the distribution and type of dataset being used. Complicated applications like bioinformatics, object detection, image segmentation and human activity analysis require clustering for recognition of irregular shaped clusters. Such arbitrary shapes cannot be easily recognized by traditional algorithms and some graph theory and other concepts like density or field intensity are more useful in these situations. Though several successful density based clustering algorithms exist and are being used, the main drawback is that they have many user defined parameters. The number of user defined parameters in density based clustering algorithms is very high. Situation is worsened because the quality of output depends heavily on values of these parameters. So the performance depends on how expert and experienced the user is while setting values of input parameters.

Even conventional algorithms like k-means[1] requires value of k to be input which denotes the number of clusters to be generated in output. Sometimes this may prove as good for certain applications where user is aware of the number of clusters require. But many applications like image segmentation do not have a predefined value of k. In such cases the user should have deep knowledge of the data set as well the application to determine the value of k. In more complex algorithms like DBSCAN [2], DDC [3] or DSets [4] the number of parameters to be defined by the user are many and the user may not have sufficient knowledge to decide their values appropriately. Bad values of the input parameters may not only affect the quality of output cluster structure but also make the algorithm futile for that particular application. In all such cases where the results of clustering are influenced by the expertise of user in selection of parameters, a careful tuning process is required to obtain satisfactory output. To overcome this drawback, two kinds of approach have been considered. First is to automatically learning the values of these parameters and second is to support this decision by mixing more than one clustering processes.

Hamerly and Elkan [5] have proposed GMeans in which the parameter k of Kmeans is learnt automatically in hierarchical fashion by increasing it till it reaches a saturated value. Muhr and Granitzer [6] has adopted an Split and Merge method to automatically find the number of clusters.

The number of user defined parameters in density based clustering algorithms is very high. In DBSCAN[2],DDC[3] there are three input parameters and each of them has a significant effect on the quality of output. Liang and Chen [7] have proposed a divide and conquer strategy to automatically decide the number of cluster while the clustering is done as per DDC. Hou et al [8] have shown how certain concepts can be merged and a parameter free algorithm can be designed. Specifically, their algorithm DSets-DBSCAN successively generates clusters. A dominant set is first extracted and then a random peak is chosen from it to grow cluster around it using DBSCAN. The purpose is to learn the parameters of DBSCAN from properties of the extracted dominant set and thus make the method parameter-free. Our proposal is inspired from this work. We have proposed the use of knn graphs instead of dominant sets because they are time consuming and dominant sets produce very small sized cores.

This paper proposes a clustering algorithm that combines two different methods together. First is graph-theoretic concept of finding strongly connected components in a penalized k-nearest neighbors graph. Second method is DBSCAN. Both methods work sequentially and hence can be combined. The combined method has advantage of both methods plus the established and time-tested practical implementation of both is available. This makes our proposal a feasible and practical solution for clustering where arbitrary shaped clusters exist.

978-1-5090-6106-8/17/$31.00 ©2017 IEEE

II. PROPOSAL

A. Outline of Proposal

The proposed method is a simple agglomerative approach. Each cluster is build one by one. A cluster is grown around its peak and then removed from dataset. The remaining dataset is again processed to search a peak or kernel. Each time a peak or kernel is discovered, concepts of density are used to grow the cluster around it. A related work to this proposal is DSets-DBSCAN [8], which works around a randomly selected peak out of the kernel identified through Dominant set process. We replace the kernel finding process by finding a strongly connected component in k-nearest neighbors' graph. Also, the clusters growing phase is accelerated by growing it in parallel around every point of core. The next sections discuss these steps in details and then the formal algorithm is present.

B. Finding Kernels

Clustering refers to task of partitioning a dataset into non-overlapping group or subsets such that members in a group are very similar to each other. If the data objects are viewed as points in m-dimensional space, the clusters can be viewed as regions separated from each other and having higher density than surroundings. The centre of a cluster is expected to be highly dense region. Hence, it is better to find such highly dense regions and grow clusters around them. These regions of high density are called cores or kernels of clusters. The proposed method uses the concept of k-nearest neighbour (knn) to discover cluster kernels.

The knn method for finding kernels assumes the entire dataset to be a directed graph. Every data object is a node in a graph. The edges is graph exist between every pair of vertices and have weight equal to distance among the objects. Thus, a pairwise distance matrix can be treated as an input to this method, graph is constructed from the data graph as follows:

i. For every node only keep edges to k nearest neighbors, and delete next.

ii. If for any two nodes $v_i$and v_j the edge exists in only one direction then delete it.

Let the data graph $G_D\left(V_D, E_D\right)$ be such that $V_D = \{V_1, v_2 \ldots\ldots v_n\}$ where v_i corresponds to i^{th} data object, and $E_D = \{(v_i, v_j | v_i, v_j) \in V_D\}$ and there exists a weighing function $w(v_i,v_j)$ = distance between i^{th} and j^{th} data objects.

The knn graph is constructed as $G_K(V_K < E_K)$ where $V_K = V_D$ and $E_k = \{(v_i, v_j | v_j \; is \; the \; j^{th}$ nearest neighbor of $v_i \; and \; j \leq k \; and \; (v_i, v_j) \in E_k)\}$.

The knn graph G_k now consists of only strongly connected components of the data graph G_D. These can be viewed analogous to the cores of clusters.

C. Expanding kernels into clusters

The kernels obtained from knn graph are expanded to form clusters by including all those points in the cluster which are density reachable from the points in the kernel. This concept of density reachability is borrowed from DBSCAN[2]. The following quantities are first computed for this purpose:

i. MinPts – It is the minimum number of neighbor points that a point must have to be called a core point. It actually sets the minimum size of a cluster. Usually, in 2-D data a value of 3 or 4 is sufficient. In our proposal the value is not much critical for performance, hence it is taken 3.

ii. Eps – A point p is neighbor of point q if it lies within eps distance. Originally, DBSCAN has fixed value of eps defined by user and output depends much on it. We propose to keep it according to working dataset as proposed in[8]. It is computed as the maximum of all 3^{rd} nearest neighbor distances of all points.

$$Eps = \max_{p \in C} ||p,p_3||_2 \quad (1)$$

Where C is the kernel of a cluster as extracted from knn graph, p3 is 3^{rd} nearest neighbor of p. In case the kernel itself has fewer points than 3(when many clusters have already been formed and very few datapoints remains) there Eps is computed as

$$Eps = \max_{p \in C} || p,p_1||_2 \quad (2)$$

Where p_1, is nearest neighbor of p.

iii. Now, density reachability can be explained as a point p is density reachable from point q if there exists a path between p and q of t points $p_1,p_2 \ldots p_t$ such that all points in the path are density reachable in succession. Any pair of points p and q are directly reachable if q lies within *Eps* distance of p. A directly reachable point is density reachable too.

To grow a cluster, given its kernel, following steps are taken:

Repeat until no more points are added to C.

i. For every point p in current cluster c, find set P of points which are density reachable from every point in C.

ii. C= C ∪ P

D. Concept of Working Set

The proposed algorithm identifies one cluster at a time. Thus, it needs to be repeated after removing that cluster from dataset. At any instant of execution, the current dataset which is to be processed is called working set. Initially entire dataset is working set. At each round a cluster is removed from it. The reduced dataset is now current working set. At every round of algorithm the processing takes place on the working set.

E. Formal Algorithm of Proposed Clustering

At any instant of execution the working set of datapoints is X_T. It is used to construct data graphs G_D which is then converted to knn graph Gk as per the neighborhood parameter k. The largest strongly connected component is G_k is extracted as a new cluster. This cluster is expanded using density reachability concepts of DBSCAN by computing Eps of current working dataset. Output this cluster. Reduce the

current working set by removing the points of cluster. The process is repeated until current working set is empty.

Algorithm KNN-DBSCAN

Step 1: Compute pairwise Euclidean distance of all points in dataset. This is initial working set.

Step 2: Construct data graphs G_D.

Step 3: Construct k-nearest neighbor graph G_k with $k = \alpha * n$, where n is number of data points in working set.

Step 4: Extract the largest strongly connected component from Gk, call it C.

Step 5: With Minpts = 3, compute Eps for C as per (1) and (2) according to size of C.

Step 6: Expand cluster of C by adding all density reachable points of working set.

Step 7: Remove C from working set.

Step 8: Repeat from Step2 until an empty working set is obtained.

F. Note on value of k

To construct a knn graph, the value of k needs to be defined. Contrary to the original knn concept, we cannot have a fixed value of k. Since the number of points in working dataset reduces as algorithm progresses, the value of k should be reduced. Hence, we keep value of k to be computed according to size of working set. Let at any point of time, size of working set be n, then k is calculated as a multiple of n.

III. RESULTS AND IMPLEMENTATION

Both the DSets-DBSCAN and KNN-DBSCAN are implemented as MATLAB programs. The datasets chosen for the experiments are picked so that their ground truth cluster structure has some unique characteristic that can be utilized to verify the effect of clustering method used. Total 8 datasets having synthetically generated two-dimensional points, are used, as A1 [9], Compound [10], D31[11], Flame [12], Jain [13], Path Based [14], R15[11] and Touching.

A. Visualizing Results

The results of experiments are recorded visually by marking points in different clusters by different markers. Since all datasets are two-dimensional, the visualization of points is direct. The output clusters are shown successively in Fig 1 till 16 for the DSets-DBSCAN and KNN-DBSCAN for the eight datasets.

In A1 dataset, the DSets-DBSCAN is able to separate out only 14 out of 20 clusters properly, while the proposed KNN-DBSCAN algorithm identifies 17 clusters properly. In Compound dataset where 3 pairs of clusters exist, DSets-DBSCAN fairs better in rightmost pair, and fails for other 2 pairs. The proposed KNN-DBSCAN has good output for upper left pair, and for right pair it is satisfactory. Both the algorithms are unable to separate the lower left pair. For D31 clusters, DSets-DBSCAN has separated only 19 clusters, whereas the proposed algorithm has identified 21 clusters completely, and 4 clusters with very less noise. The performance of DSets-DBSCAN is very poor for the Flame dataset as it gets trapped in the position where the two clusters overlap. The proposed KNN-DBSCAN algorithm is able to separate the two clusters to more extent. Visually, the performance of both algorithms for Jain dataset shows no winner. One cluster is more properly recognized and the other is segmented into parts. Same is for the path-based dataset. As expected, R15 is easier than other datasets for both algorithms. Very little noise is indicated in output of DSets-DBSCAN for clusters in outer ring region, but the clusters in inner region are not separated. KNN-DBSCAN is able to separate the closely lying clusters of inner ring also and the clusters of outer ring region are totally free from noise. DSets-DBSCAN gives very poor quality output for the small dataset of touching clusters. None of the clusters are recognized in proper shape or parts. No separation among the two clusters is clear. While the KNN-DBSCAN identifies one cluster as whole and well separated. The other cluster has only proper core, but noise at borders.

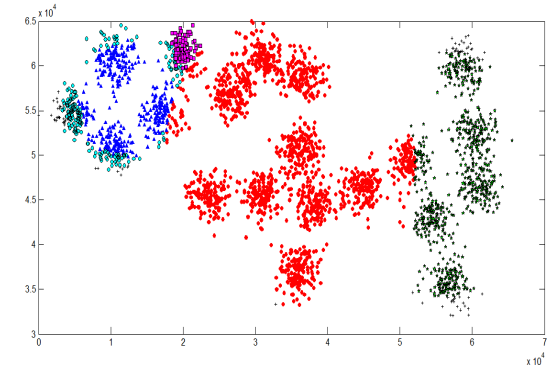

Fig. 1. Clusters obtained by DSets-DBSCAN over A1 dataset

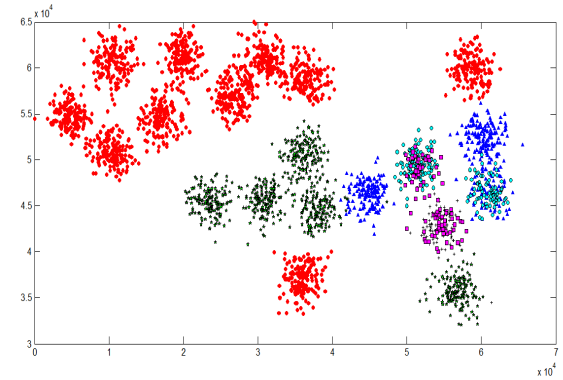

Fig. 2. Clusters obtained by KNN-DBSCAN over A1 dataset

978-1-5090-6106-8/17/$31.00 ©2017 IEEE

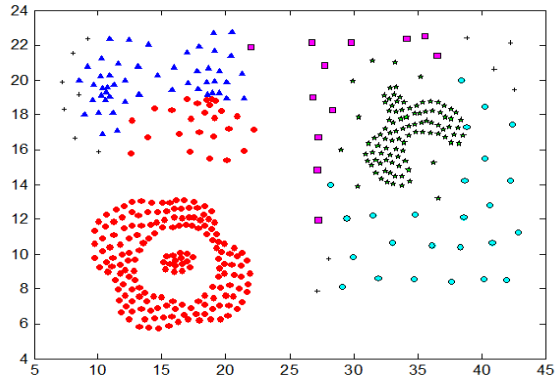

Fig. 3. Clusters obtained by DSets-DBSCAN over Compound dataset

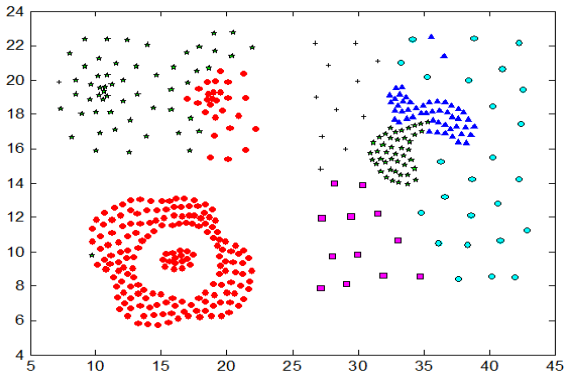

Fig. 4. Clusters obtained by KNN-DBSCAN over Compound dataset

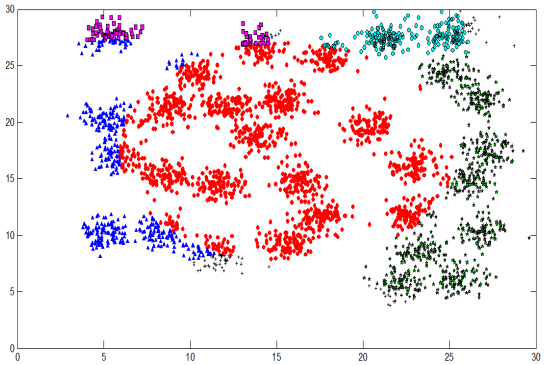

Fig. 5. Clusters obtained by DSets-DBSCAN over D31 dataset

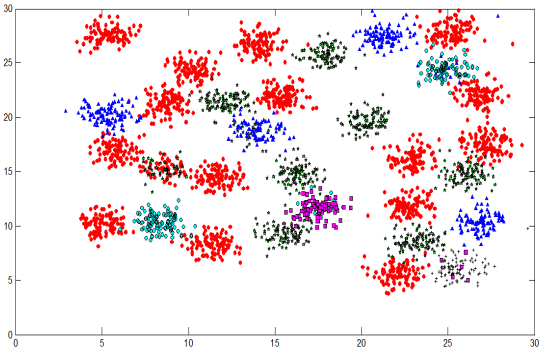

Fig. 6. Clusters obtained by KNN-DBSCAN over D31 dataset

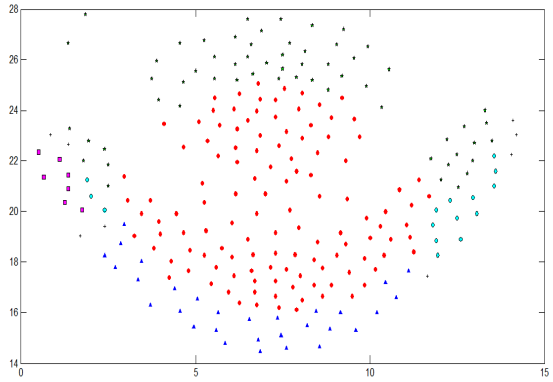

Fig. 7. Clusters obtained by DSets-DBSCAN over Flame dataset

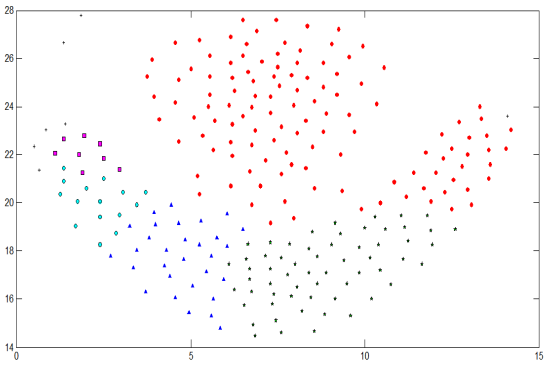

Fig. 8. Clusters obtained by KNN-DBSCAN over Flame dataset

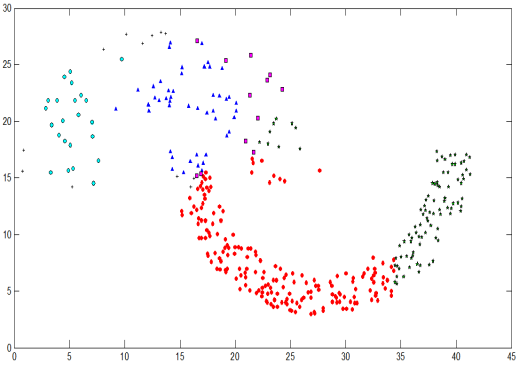

Fig. 9. Clusters obtained by DSets-DBSCAN over Jain dataset

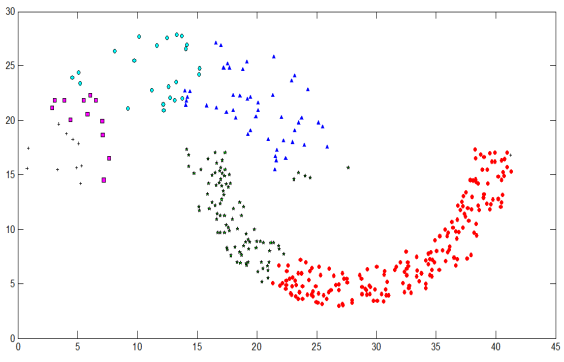

Fig. 10. Clusters obtained by KNN-DBSCAN over Jain dataset

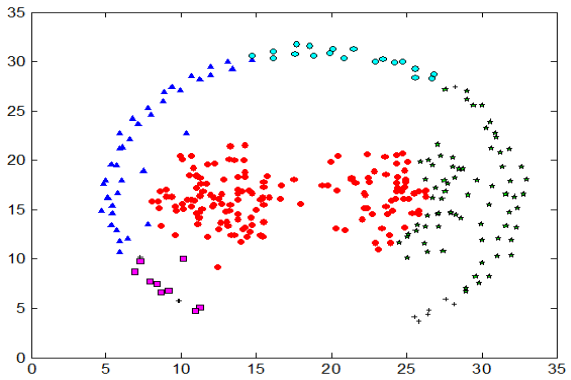

Fig. 11. Clusters obtained by DSets-DBSCAN over Path-based dataset

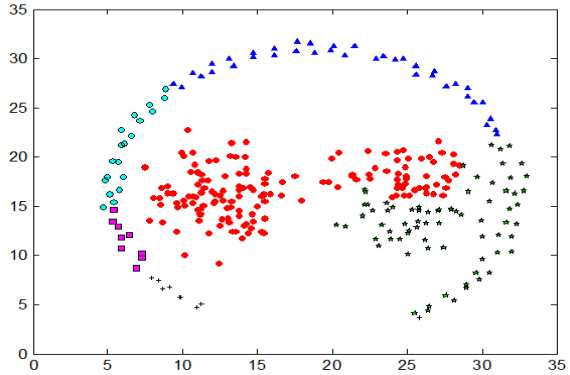

Fig 12 Clusters obtained by KNN-DBSCAN over Path-based dataset

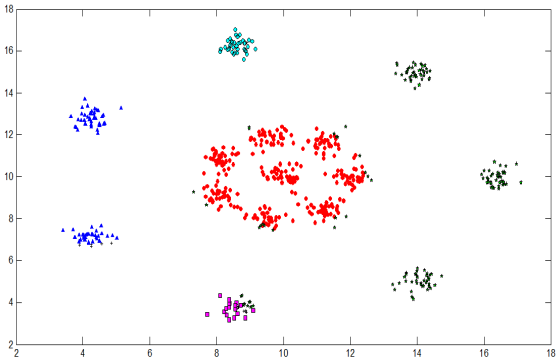

Fig. 13. Clusters obtained by DSets-DBSCAN over R15 dataset

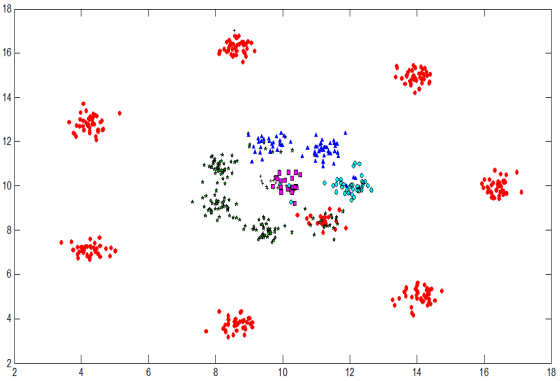

Fig. 14. Clusters obtained by KNN-DBSCAN over R15 dataset

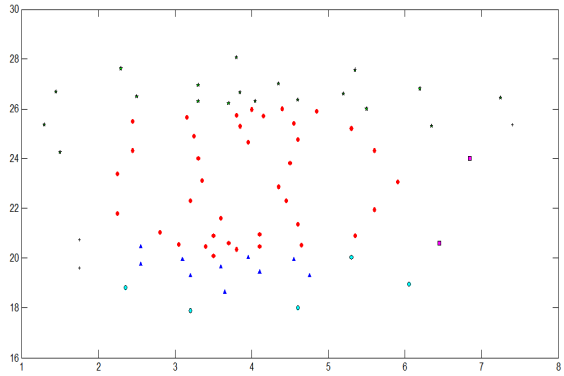

Fig. 15. Clusters obtained by DSets-DBSCAN over Touching dataset

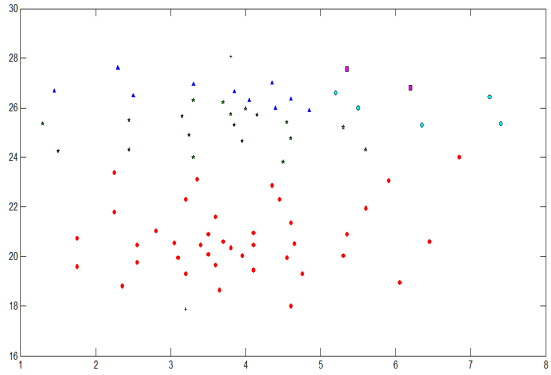

Fig. 16. Clusters obtained by KNN-DBSCAN over Touching dataset

B. *Comparisons*

The results of DSets-DBSCAN and KNN-DBSCAN are compared with each other on the basis of purity of clusters generated as output. The purity of output is computed as:

$$Purity(\Omega, C) = \frac{1}{N}\sum_k \max_j \left|\omega_k \cap c_j\right| \quad (3)$$

Where $\Omega = \{\omega_1, \omega_2, \ldots, \omega_k\}$ is set of output clusters $C = \{c_1, c_2, \ldots, c_j\}$ and is set of ground-truth clusters. A purity value of 0 is very poor and of 1 is perfect clustering. The values of purity obtained in the experiments over 8 datasets by the two algorithms are shown in Fig 17 and clearly indicate that the proposed algorithm performs better than DSets-DBSCAN.

CONCLUSION

Density-based clustering and graph-theoretic clustering are better suited to identify any arbitrary shape of cluster and has more widespread applicability. But majority of these algorithms have a great drawback that they are dependent on user-defined parameters for good performance. Thus, performance is dependent on user expertise, experience and knowledge of data. It is always desirable that the algorithms should be free from user-defined parameters if theses affect the quality of output significantly. To accomplish this, we have proposed a combination algorithm. The compatible processes

978-1-5090-6106-8/17/$31.00 ©2017 IEEE

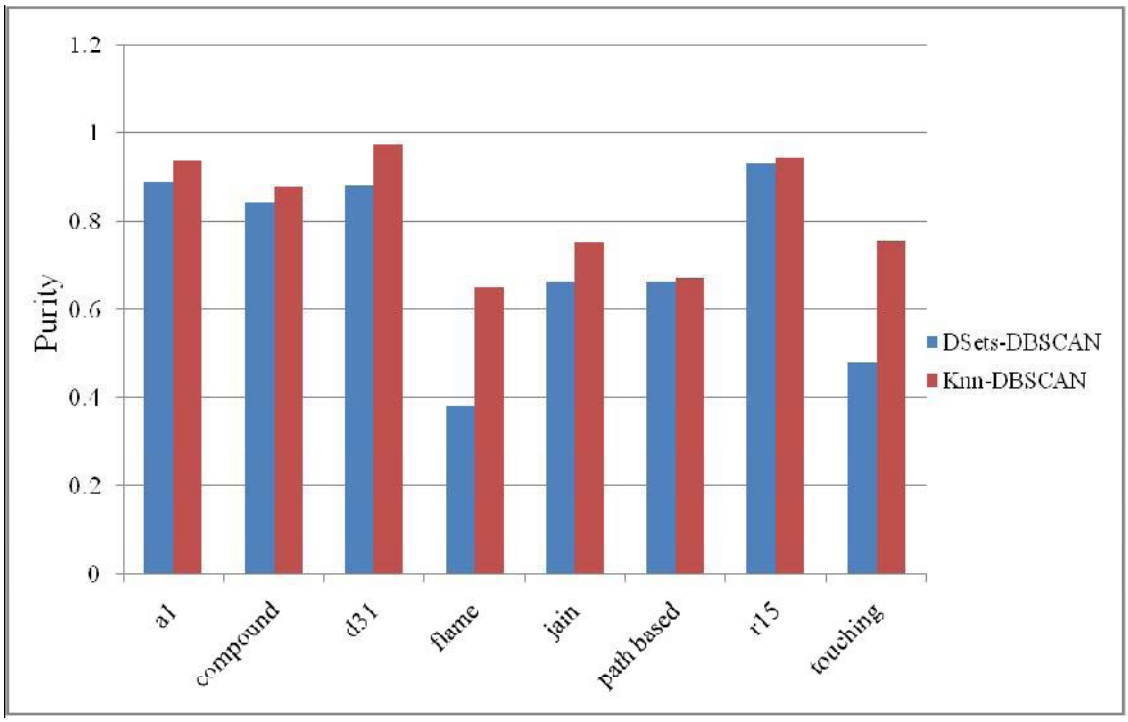

Fig. 17. Comparison of Cluster Purity achieved in datasets by the two algorithms

of k-nearest neighbour graphs and DBSCAN are mixed together to design a clustering method. The proposed method successively generates clusters, one at each round. Every round has one step of knn graph formation, followed by extraction of a core and growing of that core into cluster using DBSCAN. This gives a density-based clustering method with advantage of both the algorithms and frees the user from deciding values of parameters. The parameters required by knn graph formation and DBSCAN are now computed using some rules in the proposed method.

The current work can be extended to assisted choice of the connected component being extracted from knn graph formation phase. Currently, we have proposed to extract the largest component only. Also, other graph-based or connectivity based methods can be combined in the manner similar to proposal to see if quality of output can further be improved.

REFERENCES

[1] E.W.Forgy, "Cluster analysis of multivariate data: efficiency v/s interpretability of classifications", *Biometrics, Vol. 21, pp. 768–769*, 1965.

[2] M.Ester , H.P. Kriegel, J.Sander and X.Xu, "A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise", *KDD-96 Proceedings. Copyright* , 1996.

[3] A. Rodriguez and A. Laio, "Clustering by fast search and find of density peaks," *Science*, vol. 344, no. 6191, pp. 1492–1496, Jun. 2014.

[4] M. Pavan and M. Pelillo, "Dominant sets and pairwise clustering," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 29, no. 1, pp. 167–172, Jan. 2007.

[5] G. Hamerly, C. Elkan, "Learning the k in k –means. Advances in neural information processing systems", 16,2003.

[6] M. Muhr and M.l Granitzer "Automatic Cluster Number Selection using a Split and Merge K-Means Approach",*IEEE Conference Publication, 20th International Workshop on Database and Expert Systems Application,pp. 363-367*, 2009.

[7] Z. Liang and P.Chen, "Delta-Density based clustering with a Divide-and-Conquer strategy: 3DC clustering", *Pattern Recognition Letters, Vol. 73, Issue C, pp. 52-59*, 2016.

[8] J.Hou , H.Gao and X. Li," DSets-DBSCAN: A Parameter-Free Clustering Algorithm",*IEEE Transaction of image processing,Vol. 25,No. 7*,July 2016.

[9] Kärkkäinen and P. Fränti, "Dynamic local search algorithm for the clustering problem", Research Report A-2002-6, 2002.

[10] C. T. Zahn, "Graph-theoretical methods for detecting and describing gestalt clusters". *IEEE Transactions on Computers*, Vol. 100, No. 1, pp. 68-86, 1971.

[11] C.J. Veenman, M.J.T. Reinders, and E. Backer, "A maximum variance cluster algorithm", IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 24, No. 9, pp. 1273-1280, 2002.

[12] L. Fu and E. Medico, "FLAME, a novel fuzzy clustering method for the analysis of DNA microarray data", BMC bioinformatics, Vol. 8, No. 1, pp. 3, 2007.

[13] A. Jain and M. Law, "Data clustering: A user's dilemma", Volume 3776 of Lecture Notes in Computer Science, pp. 1-10, 2005.

[14] H. Chang and D.Y. Yeung, Robust path-based spectral clustering. Pattern Recognition, 41(1): pp. 191-203, 2008.