Efficient and Robust Clustering on Large-scale Data Sets Using Fuzzy Neighborhood Functions

Hao Liu, Satoshi Oyama, Masahito Kurihara, Haruhiko Sato

***Abstract*—Density-based clustering algorithms are applied for the detection of clusters in spatial data sets, but typical algorithms usually have difficulties in selecting appropriate parameters. Recently, the FN-DBSCAN algorithm extended the density-based clustering algorithms with fuzzy set theory and solved this problem. However, FN-DBSCAN has a time complexity of $O(n^2)$, which indicates that it is not suitable to deal with large-scale data sets. In this paper, we propose a novel clustering algorithm called landmark FN-DBSCAN which ensures linear time and space complexity with respect to the size of the input data set and empirically provides good clustering qualities.**

***Index Terms*—clustering, fuzzy neighborhood functions, FN-DBSCAN**

I. Introduction

CLUSTERING is an important tool for data analysis. It aims to divide a given data set into several clusters, where each pair of data in the same cluster has greater similarity than that in two different clusters. In the past decades, many clustering algorithms have been proposed. A rough but widely agreed framework [1] is to classify clustering techniques into partitional clustering [2], [3], hierarchical clustering [4], [5] and density-based clustering [6], [7]. Density-based clustering techniques have several advantages, e.g. the number of clusters need not be known beforehand, the detected clusters can be represented in arbitrary shapes and outliers can be detected and eliminated. These advantages make the density-based clustering algorithms suitable for dealing with spatial data sets. However, they usually have difficulties in selecting appropriate parameters. Recently, the Fuzzy Neighborhood DBSCAN (FN-DBSCAN) extended the density-based clustering algorithms with fuzzy set theory, which makes density-based clustering algorithms more robust [8]. However, FN-DBSCAN requires a time complexity of $O(n^2)$, where n is the number of data in the data set, implying that FN-DBSCAN is not suitable for applications with large scale data sets. In this paper, we propose a novel clustering algorithm called landmark FN-DBSCAN. Here, 'landmark' represents a subset of the input data set, which makes the algorithm efficient with large-scale data sets. We present a theoretical analysis on time and space complexities, which indicates that they are linearly dependent on the size of the data set. The experiments presented in this paper also show that landmark FN-DBSCAN is much faster than FN-DBSCAN and provides good clustering qualities.

Hao Liu, Satoshi Oyama, Masahito Kurihara and Haruhiko Sato are with Division of Synergetic Information Science in Graduate School of Information Science and Technology, Hokkaido University, Sappro, Japan, 060-0814. E-mail: liuhao@complex.ist.hokudai.ac.jp, oyama@ist.hokudai.ac.jp, kurihara@ist.hokudai.ac.jp, haru@complex.ist.hokudai.ac.jp.

II. Related Work

A. DBSCAN

DBSCAN is the first proposed and still widely used density-based clustering algorithm. It introduced two basic concepts, i.e. the ε-*neighborhood* of a data and the *core data* (*cardinality*) [9]. Based on these concepts, DBSCAN applies a distance-based strategy to estimate the local density. It assumes that the probability density of a small area in the feature space is uniform and the data density in each desired cluster is higher than that outside the cluster. The density of noisy data is expected to be lower than that of normal data. The key idea of DBSCAN is that for each data in a cluster, the number of data in its neighborhood (determined by the parameter ε) has to exceed some threshold (determined by the parameter $MinPts$).

However, the parameter ε is a globally fixed value, which indicates that the neighborhoods of all data have the same radius value. If we measure the degree of the neighborhood membership for each data pair, the membership function used in DBSCAN can be described by Equation (1).

$$N_d(d') = \begin{cases} 1, & \text{if } dis(d, d') \leq \varepsilon \\ 0, & \text{otherwise} \end{cases} \quad (1)$$

In this model, crisp neighborhood model, all data in one ε-neighborhood have the same value of membership degrees, which makes it difficult to calculate the *cardinality* of the neighborhood. considering three data, d, d_1 and d_2, also, we want to differentiate the membership degrees between two data pairs, i. e. (d, d_1) and (d, d_2)(Fig. 1). Unfortunately, the membership degrees for (d, d_1) and (d, d_2) are the same according to Equation (1).

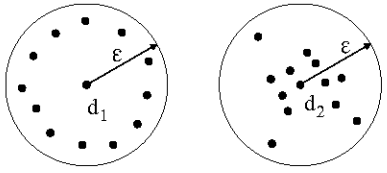

Fig. 1. Example of using crisp neighborhood in DBSCAN. Data d_1 and d_2 have the same value of *cardinality*, 12, but different values of density.

B. Fuzzy Neighborhood DBSCAN

Fuzzy Neighborhood DBSCAN (FN-DBSCAN) extended the original DBSCAN algorithm with fuzzy set theory to provide a better solution than the crisp neighborhood model used in DBSCAN. The key idea of FN-DBSCAN is to use fuzzy neighborhood functions to define a new fuzzy neighborhood instead of the old crisp neighborhood. Given two data d and d' $(d, d' \in D)$, an example of fuzzy

ISBN: 978-988-19251-8-3
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

neighborhood functions as an exponential function is given by Equation (2).

$$f_d(d') = \exp\left(-\left(k\frac{dis(d,d')}{d^{max}}\right)^2\right) \quad (2)$$

where k is a positive real number ($k > 0$) affecting the neighborhood radius and d^{max} is the maximum distance of all data pairs in D.

As expected, different membership degrees can be distinguished by applying this neighborhood function. FN-DBSCAN introduced a new definition for describing a soft neighborhood based on fuzzy neighborhood functions [8]. By defining the fuzzy neighborhood, the original, distance-based, DBSCAN can be transformed into the level-based FN-DBSCAN. Thus, in the previous example shown in Fig 1, the $fuzzy\ cardinality$ of d_1 and d_2 are not the same according to the definition of $fuzzy\ core\ data$ (more discussions in [8]).

In fact, the FN-DBSCAN alogrithm is very similar to the original DBSCAN algorithm. More precisely, if FN-DBSCAN uses the same techniques of defining neighborhoods and calculating the cardinalities, FN-DBSCAN will become DBSCAN. Furthermore, we know that the time complexity of FN-DBSCAN is $O(n^2)$, which is the same as that of DBSCAN.

III. Landmark Fuzzy Neighborhood DBSCAN

In this section, we propose a novel clustering algorithm, landmark FN-DBSCAN. This algorithm can provide a similar clustering quality as that provided by FN-DBSCAN, but only requires a time complexity linearly depended on the size of input data set.

A. Algorithm

The landmark FN-DBSCAN algorithm consists of three steps:

1) Divide a data set into several subsets represented by the generated 'landmarks'.
2) Execute a modified version of FN-DBSCAN on the generated landmark set and output the landmark index.
3) Label data according to the landmark index.

To reduce the expensive cost of directly processing the data set, the input data set is divided into several subsets of smaller sizes. In this procedure, some data in the data set are selected and further processed as 'landmarks' and each landmark is used to represent one subset. Here we present several concepts related to this procedure.

Definition 1 (landmark): Given a data set D as an $n \times m$ matrix, where n is the number of data and m is dimensionality of data, a landmark (l), which is a triplet is defined as

$$l = \langle \mathbf{V}, N_f^L(l), \mu \rangle \quad (3)$$

where $\mathbf{V}$ is an m-dimensional vector equaling to a data in D (determined by Algorithm 1), $N_f^L(l)$ is a subset of D, containing all the data in the fuzzy neighborhood of l (Definition 2) and μ is a positive real number called the membership level of l (Equation (7)).

With the property $\mathbf{V}$, a landmark can compare the membership degree with a data or another landmark. To measure the membership degree in such cases, two variants of exponential fuzzy neighborhood functions are used.

First, considering a landmark l and a data d ($d \in D$), the membership degree between l and d can be measured by Equation (4).

$$f_l(d) = \exp\left(-\left(r \cdot k \cdot \frac{dis(\mathbf{V},d)}{\Delta d^{max}}\right)^2\right) \quad (4)$$

where r, k are positive real numbers and Δd^{max} is the maximum distance between $\mathbf{V}$ and all the other data in D.

Second, for measuring the membership degree between two landmarks l_1 and l_2, the membership function is given by Equation (5).

$$f_{l_1}(l_2) = \exp\left(-\left(k \cdot \frac{dis(\mathbf{V_1},\mathbf{V_2})}{\Delta d^{max}}\right)^2\right) \quad (5)$$

where k is a positive real number and Δd^{max} is the maximum distance between $\mathbf{V_1}$ and all other landmarks.

Definition 2 (fuzzy-neighborhood of a landmark): Given a set D where D can be a set of data or a set of landmarks, and a positive real number ε_1, the $fuzzy\text{-}neighborhood$ of a landmark l, denoted as $N_f^L(l)$, is a set of data or a set of landmarks defined by

$$N_f^L(l) = \{d \in D \mid f_l(d) \geq \varepsilon_1\} \quad (6)$$

where $f_l(d)$ can be obtained by Equation (4) or Equation (5). We say d is in the fuzzy neighborhood of landmark l.

The last property of a landmark, the membership level, μ, can be calculated by the following equation:

$$\mu = \sum_{d \in N_f^L(l)} f_l(d) \quad (7)$$

With the above concepts we present a technique of generating landmarks using Algorithm 1.

Algorithm 1 LandmarkGeneration

Input: D, r, k, ε_1
Output: L

```
L ← φ;
for all d in D do
    find a landmark l = (V, u, s) ∈ L, such that l.V = min{dis(l.V, d)}.
    u ← exp(−(r · k · dis(l.V,d)/Δd^max)^2);
    if L = φ or u < ε1 then
        V ← d; N ← φ; u ← 0;
        l ← (V, N, u);
        L ← L ∪ {l};
    else
        l.N ← l.N ∪ {d};
        l.u ← l.u + u;
    end if
end for
```

From Algorithm 1, we observe that landmarks are generated dynamically during the algorithm execution and all data in the data set must belong to one landmark. Finally, all data can be labeled according to each corresponding landmark.

Since the output of Algorithm 1 is a set of $landmarks$, which is not acceptable by the FN-DBSCAN algorithm, we

ISBN: 978-988-19251-8-3
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

make a variation on the standard FN-DBSCAN algorithm so that the modified version can process them.

Here, we present a method to calculate the *cardinality* of the *landmark* neighborhood. Considering a set of *landmarks*, L, where $l = \left\langle \mathbf{V}, N_f^L(l), \mu \right\rangle \in L$, the *cardinality* of the neighborhood of l can be calculated by

$$card(l) = \sum_{l' \in N_f^L(l)} l'.\mu \tag{8}$$

B. Complexity Analysis

Theorem 1: The time complexity of landmark FN-DBSCAN is $O(kn + k^2)$, where n is the number of data and k is the number of generated landmarks.

In Step (1), the algorithm scans the data set once, which takes $O(n)$ as the time time complexity. In each loop, to find the landmark with the minimum distance to the current data, it is expected to take a O(1/2*kn) as the total time complexity. For calculating Δd^{max}, $O(kn)$ is taken as the time complexity. So, the time complexity of Step (1) is $O(kn)$. In Step (2) has the same time complexity of FN-DBSCAN, $O(k^2)$. Obviously, Step (3) has $O(n)$ as the time complexity. Therefore, the time complexity of landmark FN-DBSCAN is $O(kn + k^2 + n) = O(kn + k^2)$

However, in practice the number of generated landmarks is much lesser than the number of data in the data set, i.e. $k \ll n$. In this case, the time complexity of landmark FN-DBSCAN reduces to $O(n)$, which indicates that it is suitable for large-scale data sets.

Theorem 2: The space complexity of landmark FN-DBSCAN is $O(n + k)$, where n is the number of data and k is the number of generated landmarks.

In Step (1), the algorithm requires the space complexity at $O(n)$ and $O(k)$ to store the data set and the generated landmarks, respectively. Furthermore, it needs to store the index of all data in landmark neighborhoods, which is $O(n)$ in total. In Step (2), the algorithm requires the same space complexity as that of FN-DBSCAN, $O(k)$. Step (3) needs no more extra space. Therefore, the space complexity is $O(n + k + n + k) = O(n + k)$.

Similar to the time complexity, the space complexity will reduce to $O(n)$ if $k \ll n$.

IV. Experiments

In this section, both clustering quality and clustering efficiency of the landmark FN-DBSCAN algorithm were evaluated in comparison with FN-DBCAN.

The clustering quality results was evaluated by comparing with the true partition (gold standard). A well-known method, Rand-Index [10], was used to evaluate the clustering quality. A high value obtained from Rand-index indicated that the evaluated method provided a high clustering quality (accuracy) on the input data set, and vice versa.

Two synthetic spatial data sets and two real world data sets were used as experimental objects. Their details are shown in Table I. Here data sets of different sizes were prepared, but they shared the same data distributions. Correct answers for each data set (including one data set of different sizes) were pre-prepared and then used in the Rand-Index calculations.

The first two data sets, Anchor and Banana, were used to test the capability of the algorithm to detect clusters with arbitrary shapes and that of the data sets to deal with noisy data.

TABLE I
Data Sets Used in Experiments.

name	Size	♯D	♯C	noise
Anchor	20000	2	2	yes
Banana	12000	2	2	yes
Letter	20000	16	26	no
Pendigits	10992	16	10	no

The symbol ♯D means the 'number of dimensions' and ♯C means the 'number of clusters'.

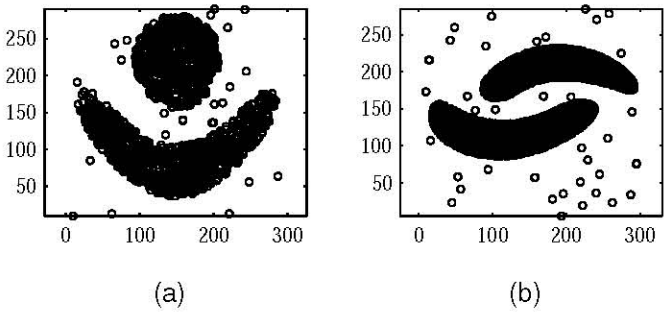

Fig. 2. Samples of used synthetic data sets. (a) Anchor data set (2500 data). (b) Banana data set (3000 data).

Anchor, illustrated in Fig 2(a), consists of two clusters with a shape like an anchor. We used eight groups of Anchor data sets with different scales. Generally speaking, both landmark FN-DBSCAN and FN-DBSCAN can provide good clustering quality, i.e two clusters can be found and noisy data can be detected. The detailed results of clustering quality with different sizes are shown in Fig. 3(a). We observe that the landmark FN-DBSCAN algorithm and the FN-DBSCAN algorithm achieved similar results, and both obtained Rand-Index values of approximately 0.99. However, there were substantial differences in their efficiencies. The time cost of the FN-DBSCAN algorithm increased very rapidly, while that of the landmark FN-DBSCAN algorithm increased slowly (Fig. 3(b), $r = 3$). For example, when the size of the data set was 2500, the landmark FN-DBSCAN algorithm saved approximately 85% of the time of FN-DBSCAN and provided almost the same quality ($r = 3$). On increasing the number of data to 20000, it saved 95.5% of the time of FN-DBSCAN and even provided a slightly better quality ($r = 3$).

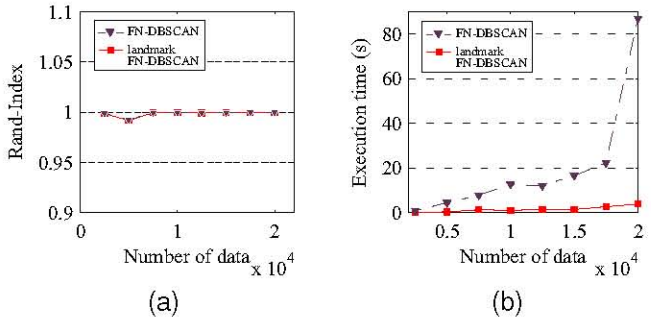

Fig. 3. Results of Anchor data set ($r = 3$). Comparison of (a) Clustering quality and (b) Clustering Efficiency.

ISBN: 978-988-19251-8-3
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

TABLE II
LETTER DATA SET.

Size	Landmark FN-DBSCAN Rand-Index r value					*FNR	Landmark FN-DBSCAN execution time(s) r value					*FNt (s)
	0.5	0.6	0.7	0.8	0.9		0.5	0.6	0.7	0.8	0.9	
2000	0.9612	0.9623	0.9623	0.9622	0.9621	0.9620	0.45	0.84	0.98	1.27	3.23	1.69
4000	0.9611	0.9620	0.9623	0.9624	0.9623	0.9528	0.83	1.42	2.52	3.19	4.16	7.03
8000	0.9616	0.9623	0.9623	0.9622	0.9621	0.9239	8.80	16.94	28.06	40.48	53.61	437.02
16000	0.9617	0.9623	0.9622	0.9621	0.9621	0.8083	19.16	103.13	131.94	212.63	495.86	1494.45
20000	0.9613	0.9621	0.9622	0.9621	0.9620	0.9615	25.16	107.95	220.83	257.94	723.47	2722.22

*FNR means the 'FN-DBSCAN´s Rand-Index value' and FNt means the 'FN-DBSCAN´s execution time'

TABLE III
PENDIGITS DATA SET.

Size	Landmark FN-DBSCAN Rand-Index r value					*FNR	Landmark FN-DBSCAN execution time(s) r value					*FNt (s)
	0.2	0.3	0.4	0.5	0.6		0.2	0.3	0.4	0.5	0.6	
1000	0.9260	0.9220	0.9090	0.9037	0.9018	0.9003	0.06	0.06	0.09	0.17	0.23	0.42
2000	0.9193	0.9151	0.9086	0.9054	0.9030	0.9068	0.05	0.11	0.22	0.38	0.58	1.39
4000	0.9300	0.9185	0.9083	0.9044	0.9025	0.9180	0.11	0.25	0.56	0.98	1.63	5.52
8000	0.9190	0.9154	0.9074	0.9042	0.9020	0.9263	0.55	1.36	2.53	4.09	6.98	66.44
10992	0.9242	0.9160	0.9075	0.9036	0.9021	0.9336	0.45	1.41	3.16	6.45	10.83	120.41

*FNR means the 'FN-DBSCAN´s Rand-Index value' and FNt means the 'FN-DBSCAN´s execution'

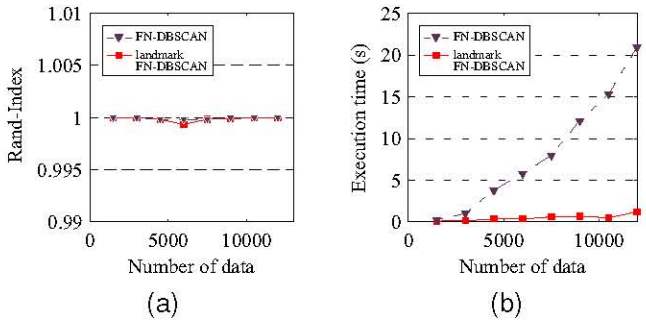

Fig. 4. Results of Banana data set ($r = 3$). Comparison of (a) Clustering quality and (b) Clustering Efficiency.

Banana data set contained two banana shaped clusters with 12000 data including noisy data (Fig. 2(b)). The clustering result of quality and efficiency are summarized in Fig. 4(a) and Fig. 4(b), respectively ($r = 3$). We observe that both landmark FN-DBSCAN and FN-DBSCAN could can achieve good results on this data set, although the former was significantly more efficient. When the data set size was 1500, landmark FN-DBSCAN was over 5 times faster than FN-DBSCAN, but provided the same quality ($r = 1.8$). On increasing the number of data, the landmark FN-DBSCAN algorithm was over 41 times faster than FN-DBSCAN and achieved a Rand-Index value of 0.994 (6000 data, $r = 1.5$) which is the same as FN-DBSCAN. Furthermore, it was over 47 times faster than FN-DBSCAN, whereas it achieved a Rand-Index value of 1.0 (12000 data, $r = 1.5$).

The other two data sets, Letter and Pendigits, were selected from UCI Machine Learning Repository (http://archive.ics.uci.edu/ml). The results of landmark FN-DBSCAN and FN-DBSCAN are summarized in Table II and Table III. respectively. Landmark FN-DBSCAN was over 12 times faster than FN-DBSCAN for 20000 size Letter data set ($r = 0.7$) and over 267 times faster for 10992 Pendigits data set ($r = 0.2$). However, we noticed that FN-DBSCAN provided a slightly better quality than landmark FN-DBSCAN on these two data sets, but the difference was marginal.

V. CONCLUSION

In this paper, we propose a novel clustering algorithm called landmark fuzzy neighborhood DBSCAN (landmark FN-DBSCAN). The presented concept, landmark, was used to represent a subset of the input data set which made the algorithm efficient for large-scale data sets. We presented a theoretical analysis on the time and space complexities of the algorithm, which showed that both were linearly dependent on the size of data set. The experiments presented in this paper also showed that landmark FN-DBSCAN was much faster than FN-DBSCAN and was able to provide a very similar clustering quality.

REFERENCES

[1] R. Xu and D. C. W. II, "Survey of clustering algorithms." *IEEE Trans. Neural Netw.*, vol. 16, no. 3, pp. 645–678, 2005.
[2] H. Zhexue, "Extensions to the k-means algorithm for clustering large data sets with categorical values," *Data Mining and Knowledge Discovery*, vol. 304, no. 3, pp. 283–304, 1999.
[3] C. H. Q. Ding and X. He, "K-nearest-neighbor consistency in data clustering: incorporating local information into global optimization." in *Proc. ACM Symp. on Applied Computing*, 2004, pp. 584–589.
[4] S. Theodoridis and K. Koutroumbas, *Pattern Recognition, Fourth Edition*. Academic Press, 2009.
[5] Y. Song, S. Jin, and J. Shen, "A unique property of single-link distance and its application in data clustering." *Data & Knowledge Engineering*, vol. 70, no. 11, pp. 984–1003, 2011.
[6] J. Sander, "Density-based clustering," in *Encyclopedia of Machine Learning*, 2010, pp. 270–273.
[7] H.-P. Kriegel, P. Kröger, J. Sander, and A. Zimek, "Density-based clustering," *Wiley Interdisc. Rew.: Data Mining and Knowledge Discovery*, vol. 1, no. 3, pp. 231–240, 2011.
[8] E. N. Nasibov and G. Ulutagay, "Robustness of density-based clustering methods with various neighborhood relations," *Fuzzy Sets and Systems*, vol. 160, no. 24, pp. 3601 – 3615, 2009.
[9] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *Proc. 2nd Int. Conf. on Knowledge Discovery and Data Mining (KDD'96)*, 1996, pp. 226–231.
[10] M. Halkidi, Y. Batistakis, and M. Vazirgiannis, "Cluster validity methods: Part i." *SIGMOD Record*, vol. 31, no. 2, pp. 40–45, 2002.

ISBN: 978-988-19251-8-3
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)