FEM-DBSCAN: An Efficient Density-Based Clustering Approach

Uranus Kazemi & Reza Boostani

Iranian Journal of Science and Technology, Transactions of Electrical Engineering

ISSN 2228-6179

Iran J Sci Technol Trans Electr Eng
DOI 10.1007/s40998-020-00396-4

Springer

Iranian Journal of Science and Technology, Transactions of Electrical Engineering
https://doi.org/10.1007/s40998-020-00396-4

RESEARCH PAPER

FEM-DBSCAN: An Efficient Density-Based Clustering Approach

Uranus Kazemi[1] · Reza Boostani[2]

Received: 10 May 2020 / Accepted: 26 November 2020

Abstract
Due to the uncontrollable growth of data generation in various networks, rapid clustering of massive datasets is seriously demanded in order to reveal the hidden structure of data as well as discovering the relations among samples. Among the clustering approaches, density-based clustering methods showed an acceptable processing speed to encounter with big data. However, they have some fixed parameters, which are not certainly optimized for all parts of the feature space. Moreover, the complexity of these clustering methods is highly dependent on the number of samples. In this paper, we have deployed Fisher expectation maximization (FEM) to adaptively divide the feature space into some subspaces, where no cluster is shared between the adjacent subspaces. Afterward, we applied density-based spatial clustering of applications with noise (DBSCAN) to each partition yielding to decrease the computational complexity on each thread as well as better learning of its parameters on each subspace. The performance of the proposed method was assessed over three big-size and ten middle-size datasets. The achieved results implied the superiority of the proposed method to OPTICS, Den Clue and DBSCAN methods in terms of clustering accuracy (purity) and processing time.

Keywords Big data · Fisher expectation maximization (FEM) · Adaptive DBSCAN

1 Introduction

In the last decade, social networks like Facebook and Twitter have become an essential part of the people's life and numerous data were daily generated in the form of advertising message, email, tweets, comments, videos, audios and texts. To manage this huge volume of samples (big data), several methods were developed to arrange and categorize them. As far as labeling is a time consuming and costly process, research teams tended to categorize big data in an unsupervised manner, though the categorization accuracy of classification methods is higher than the clustering ones. Nonetheless, developing efficient clustering methods, to provide an acceptable performance and to execute in a short interval time, is still a challenge (Kazemi 2018). In fact, a clustering algorithm assigns samples to different clusters according to a similarity/dissimilarity measure and these process is iteratively learned under a criterion (objective function) like mean square error (MSE) or maximizing the distance between the clusters simultaneous with shrinking the samples within each cluster (Fisher value). Among the plenty of suggested clustering schemes, just a few can handle big data in a timely manner.

Clustering methods can be fundamentally divided into four categories of partition based, graph based, hierarchical and density based (Bhardwaj and Dash 2015). *K*-means (Zhou et al. 2019), Iso-Data (Sadeghi-Mobarakeh and Mohsenian-Rad 2018) and CURE (Cai and Liang 2018) are the known partition-based clustering methods in several fields (Soni and Ganatra 2012), whereas they are all order sensitive and successive running of them on a certain dataset yield into the same results. Moreover, partition-based methods are unable to remove noisy samples, which diminish the quality of the formed clusters. Graph-based methods (e.g., shared nearest neighbor (Ye and Sakurai 2015; Zhou et al. 2019) consider samples as nodes of a graph and try to determine

✉ Uranus Kazemi
uranuskazemi@yahoo.com

Reza Boostani
boostani@shirazu.ac.ir

[1] Department of Computer Engineering, Apadana Institute of Higher Education, Shiraz, Iran

[2] CSE & IT Department, Electrical and Computer Engineering, Shiraz University, Shiraz, Iran

Published online: 05 January 2021

trustable links among the nodes in order to create meaningful clusters. These methods are able to create clusters with different sizes and shapes and also capable of removing noisy samples. Nonetheless, defining a proper similarity function for these algorithms is problematic.

Hierarchical clustering methods like single and complete linkage (Nagpal et al. 2013) suffer from high computational complexity as well as the absence of a reliable termination criterion. Moreover, these methods are unable to remove noisy samples, though they can form clusters with different sizes and shapes.

Density-based clustering methods have recently attracted much attention of research teams due to their applicability in handling big datasets as well as providing acceptable clustering performance. Density-based methods can be categorized in two categories. A few methods like Den Clue (Kriegel et al. 2011) and its variants estimate a probability density function (PDF) using a nonparametric method like Parzen window. Next, by applying a threshold to the estimated PDF and running the hill-climbing method on the dense regions, the center and boundary of clusters are determined. Nonetheless, these methods suffer a high computational complexity when it is encountered with large number of samples. The second category of density-based methods are all based on density-based spatial clustering of applications with noise (DBSCAN) (Bhardwaj and Dash 2015) method, which is the most known and breakthrough clustering algorithm. DBSCAN provides low computational complexity, flexible enough to create clusters with arbitrary sizes/shapes as well as removing noisy samples, though it suffers from a high computational burden in confronting with high-dimensional datasets (e.g., text data), although these properties make DBSCAN as different densities. In addition, the original version of DBSCAN has no data-driven mechanism for estimating its parameters. A suitable method for big data clustering, it suffers from high sensitivity to its initial parameters such as Min-Pts and ε. DBSCAN cannot detect the gradient of distribution within a cluster and it may concatenate adjacent clusters.

In this paper, the researchers presented a clustering method that has the ability to cluster big data. In this work Fisher expectation maximum (FEM) used to divide a feature space to some subspaces. Then, in each subspace parameters of DBSCAN algorithm are tuned according to Gap-Statistics index. Finally, DBSCAN algorithm were executed in parallel on subspaces. The researchers evaluated the proposed method on standard datasets. The results showed that the proposed method is better than outperforms OPTICS and DBSCAN and Other clustering methods.

2 Related Works

Due to the undeniable positive points of DBSCAN, this method is the base of most density-based methods and research teams have been focused to modify this manner by addressing its weaknesses. Ordering points to identify the clustering structure (OPTICS) (Bhardwaj and Dash 2015) is introduced as the adaptive version of DBSCAN which is able detects clusters with different densities by gradually adapting the value of ε. Nevertheless, OPTICS needs a lot of time to incrementally adapt its parameters and cannot explicitly find the boundary of clusters. Local DBSCAN (Noticewala and Vaghela 2014) is designed to detect and create clusters, where no gradient change is happened within a cluster. However, it excludes a considerable number of samples as noisy samples, which diminishes its clustering accuracy and needs a plenty of time to cluster input samples; consequently, LDBSCAN is not suitable for big data clustering.

Fast DBSCAN (Duan et al. 2007) which is designed to keep core samples (seeds), is located in the center of local symmetric hyper-spheres in the feature space and removes a considerable number of samples that deteriorate the densities. This preprocessing leads to discard valuable information and therefore decreases the clustering accuracy at the cost of having faster execution. FDBSCAN is just able to provide acceptable performance on symmetric datasets while in practice we are encountered with variety of asymmetric (e.g., skewed) datasets.

Multi-density DBSCAN algorithm, upon the density levels partitioning (DBSCAN-DLP) (Xiong et al. 2012) is developed to segment input data into various density level sets, where statistical indexes are used to detect the density gradient. Afterward, it estimates ε for each density level set, and then applies DBSCAN to the corresponding set. The performance of DBSCAN-DLP is sensitive to Min-Pts as well as suffering high computational complexity when it faces with large datasets. Running the *K*-means clustering in the heart of this algorithm in accompany with running a sorting algorithm, increases the time complexity of this algorithm.

Varied DBSCAN (Liu et al. 2007) is a spatial clustering algorithm, which tries to detect the variation of density during the clustering. It adaptively estimates the value of ε by determining the distance of kth nearest neighbor of all samples and sort this distance (k-dist) in an ascending order, where k acts similar to Min-Pts. At each jump in the k-dist graph, the value of ε is determined for a cluster. In the next stage, DBSCAN is run for the previously determined ε values in order to detect clusters with different densities. The idea of VDBSCAN is similar to that of DBSCAN-DLP and they share similar advantages and

disadvantages. Partition-based DBSCAN (PDBSCAN) is proposed to divide the feature space into small partitions with the hope of having a uniform density at each partition. Then, a DBSCAN is applied to each partition and parallel executing of them leads to diminish the execution time. To improve the preprocessing of PDBSCAN, ant clustering algorithm (ACA) is first applied to inputs in order to improve the partitioning process in a way that the distribution of density became uniform and then PDBSCAN is applied to the tuned partitions. This method is termed as PACA-DBSCAN (Chaudhari Chaitali 2012), which is able to produce convincing results on small- and mid-size datasets while its complexity is intolerable when encounters with millions of samples.

DBSCAN for spatial-temporal data (ST-DBSCAN) (Birant and Kut 2007), is another extension of DBSCAN that added three conditions for better estimation of core points, noisy samples and adjacent clusters. This method is designed to be applied in big datasets, which are stored as temporal slices over the spatial bases (large spatial-temporal data). Nonetheless, applying these three conditions increases the complexity of algorithm at the cost of better clustering quality. In another modification, input samples are considered as graph nodes and minimum spanning tree (MST) is applied to this graph for optimizing the ε parameter in DBSCAN, which is termed as MDBSCAN (Chowdhury and Bhattacharjee 2014). Similar to the former extensions of DBSCAN, it suffers from high computational burden when running MST on a huge graph to adaptively estimate ε. A scalable Map Reduce version of DBSCAN, termed as MR-DBSCAN (He et al. 2014) is introduced for asymmetric datasets having either skewed- or long tails-distributions. Map Reduce is a known platform in parallel programming to speed up the execution time and include four stages. They proposed a rapid partitioning strategy to concatenate those partitions, which contains the same cluster, by some statistical metrics. On the other hand, due to the asymmetric form of clusters, one cell has several neighbors and therefore, the merging process can lead to decrease the clustering accuracy as well as imposing additional computational burden.

Analyzing the DBSCAN extensions over static data show that none of them is perfect and each one has its own pros and cons. However, all of the clustering methods use the Euclidean distance as their similarity measure and determining this distance in the following situations imposes a high computational burden to them: facing with high-dimensional inputs like text data (Jiang et al. 2018) and genomic sequences (Pratas et al. 2016; Soni and Ganatra 2012) encountering with large number of samples (Lin et al. 2017) increasing the number of iterations for learning the clusters.

To preserve the clustering performance as well as diminishing the complexity, the researchers proposed a two-stage clustering scheme which can be implemented via parallel processing techniques. In the first stage, feature space is segmented into some distinct subspaces by the FEM method in a way that each subspace captures only one cluster of data. In the second stage, the parameters DBSCAN include Min-Pts and ε, are tuned by the Gap-Statistics method (Tibshirani et al. 2001) in a way that the number of detected clusters by DBSCAN is confirmed by the Gap-Statistics index. Afterward, the tuned DBSCANs are executed in parallel over their corresponding subspaces.

The rest of this paper is organized as follows. Section 3 explains the proposed algorithm. Besides, Sect. 4 illustrates the evaluation criteria, results and discusses the advantages and limitations of the proposed method compared to the counterparts. Finally, the paper is concluded in Sect. 5 and an outline to the future work is presented.

3 Methods and Materials

In this part, the proposed distributed method termed as FEM-DBSCAN is explained and then the selected datasets are characterized in terms of number of samples and dimensions. Finally, the evaluation criteria are introduced.

3.1 The Proposed Method

In the proposed method, the feature space is partitioned into some subspaces by the FEM method, where we expect to have a few adjacent clusters.

There is no guarantee that the density of the neighbor clusters be uniform and equal. Then, we tuned the parameters (Min-Pts and ε) of DBSCAN using the Gap-Statistic method. As far as Gap-Statistic index determines the number of clusters in a dataset, the number of clusters produced by DBSCAN on each subspace should meet the Gap-Statistic index on the same subspace. Afterward, the DBSCANs are applied to their regions in parallel and categorize the whole data. The block diagram of the proposed method is sketched in Fig. 1.

3.1.1 FEM

The FEM is a partitioning method which acts a bit similar to Gaussian mixture model (Bouveyron and Brunet 2011). This algorithm is based on the EM algorithm to which an additional step is introduced, between the E and the M steps. This additional step is named F-step, which aims to compute the projection matrix U whose columns span the discriminative latent space. Before explaining the steps, some definitions are given as follows:

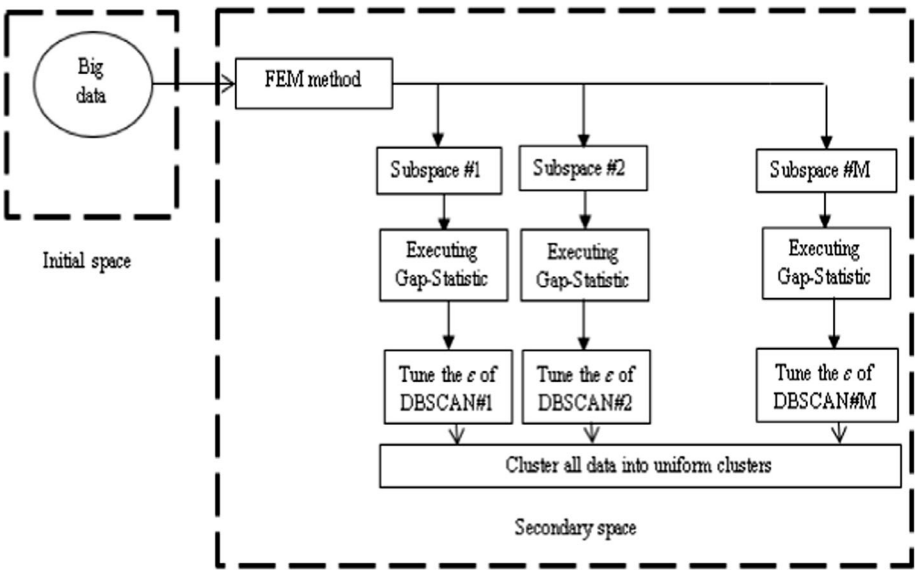

Fig. 1 An overview of the proposed method

Definition 1 Let $\{x_1, \ldots, x_n\} \in E$ are the independent input samples, which are going to be clustered into K homogeneous groups. The samples by the linear transformation of U is projected into a lower dimensional space $E \in \mathbb{R}^p$, called as latent space which is a more discriminative subspace with the dimension of $d \leq K-1$, where $K < p$. Let $\{y_1, \ldots, y_n\} \in \mathbb{R}^p$ are the projected samples, where each sample y_i is assigned by a value $z_i \in \{1, \ldots, K\}$, where $z_i = k$ demonstrates that the sample y_i belongs to the kth cluster. The input samples ($X \in E$) and the projected samples ($Y \in \mathbb{R}^p$) are linked by the linear transformation of U, as described as follows:

$$Y = UX + \varepsilon \tag{1}$$

where U is a $p \times d$ orthonormal matrix which satisfies the following condition: $U^tU = I_d$. The p-dimensional noise vector of ε models the non-discriminative part of this projection and obeys from a normal Gaussian function $\psi(\varepsilon \sim \mathcal{N}(0 \cdot p\psi))$. In addition, in the latent space, X is assumed, conditionally to $Z = k$, to be Gaussian:

$$X_{|Z} = K \sim \mathcal{N}(\mu_K \cdot p\Sigma_K) \tag{2}$$

where $\mu_K \in \mathbb{R}^d$ and $\Sigma_K \in \mathbb{R}^{d \times d}$ are, respectively, the mean vector and the covariance matrix of the kth cluster. Since linear projection does not change the input distribution, the distribution of projected samples y_i obeys a Gaussian function as described in Eq. (3),

$$Y_{X \cdot Z=K} \sim \mathcal{N}(U\ X \cdot \psi) \tag{3}$$

Thus the marginal distribution y_i can be written in the form of a Gaussian mixture model:

$$f(y) = \sum_{K=1}^{K} \pi_k \varnothing(y; m_k \cdot S_k) \tag{4}$$

where π_k is the coefficient of the kth Gaussian function $\varphi(\cdot; m_k, S_k)$ represented by the mean vector of $m_k = U_{\mu_k}$ and the covariance matrix $S_k = U\sum_K U^t + \psi$. In addition, the matrix $W = [U, V]$, $p \times p$ is defined as an orthogonal matrix ($W^tW = WW^t = I_p$), where the matrix $V_{(p-d)*p}$ is an orthogonal complement of U. The noise covariance matrix ψ satisfies the following conditions: $V^t\psi V = \beta I_{p-d}$ and $U^t\psi U = 0_d$.

As stated, the FEM algorithm has three steps:

- E-Step: Determining the conditional probabilities that observations belong to the K Gaussian.
- F-Step: Estimating the orientation matrix U corresponding to the conditional probabilities.
- M-Step: Estimating the parameters of *GMM* in the latent space by maximizing the likelihood function. What follows is the formulation of the FEM algorithm at iteration q:

The E-Step determines the posterior probabilities $t_{ik}^{(q)}$, in which the observations belong to the Kth function using the following update formulas (5 and 6):

$$t_{ik}^{(q)} = \hat{\pi}_k^{(q-1)} \varnothing\left(y_i \cdot \hat{\theta}_k^{(q-1)}\right) / \sum_{l=1}^{K} \hat{\pi}_l^{(q-1)} \varnothing\left(y_i \cdot \hat{\theta}_l^{(q-1)}\right) \tag{5}$$

$$\hat{\theta}_k = \left\{\hat{\mu}_k \cdot \hat{\Sigma}_k \cdot \hat{\beta}_k \cdot \hat{U}\right\} \tag{6}$$

As we mentioned, F-Step considers the estimated posterior probability and iteratively estimates the directions of linear transform $U^{(q)}$ into the latent space by maximizing the Fisher's criterion.

$$\hat{U}^{(q)} = \text{maxtrace}\left((U^t S U)^{-1} U^t S_{\text{B}}^{(q)} U\right) \tag{7}$$

$$U^t U = I_d \tag{8}$$

where S is the covariance matrix of the entire dataset and $S_{\text{B}}^{(q)}$ is the between-class matrix, determined as follows:

$$S_{\text{B}}^{(q)} = \frac{1}{n}\sum_{k=1}^{K} n_k^{(q)} \left(m_k^{(q)} - \overline{y}\right)\left(m_k^{(q)} - \overline{y}\right)^t \tag{9}$$

where

$$n_k^{(q)} = \sum_{i=1}^{n} t_{ik}^{(q)} \cdot m_k^{(q)} = 1/n_k^{(q)} \sum_{i=1}^{n} t_{ik}^{(q)} y_i \tag{10}$$

$$\overline{y} = 1/n \sum_{i=1}^{n} y_i \tag{11}$$

In the final step (M-Step), the parameters of GMM are estimated in the latent subspace by maximizing the conditional expectation of the log-likelihood function:

$$\begin{aligned} Q(\theta) = -\frac{1}{2}\sum_{k=1}^{K} n_k^{(q)} \Bigg[& -2\log\left(\pi_k + \text{trace}\left(\sum_{k}^{-1} \hat{U}^{(q)t} C_k^{(q)} \hat{U}^{(q)}\right)\right) \\ & + \log\left(\left|\sum k\right|\right) + (p-d)\log(\beta_k) \\ & + \frac{\text{trace}\left(C_k^{(q)}\right) - \sum_{j=1}^{d} \hat{u}_j^{(q)t} C_k^{(q)} \hat{u}_j^{(q)}}{\beta_k} + p\log(2\pi) \Bigg] \end{aligned} \tag{12}$$

where $C_k^{(q)} = \frac{1}{n_k^{(q)}} \sum_{i=1}^{n} t_{ik}^{(q)} \left(y_i - m_k^{(q)}\right)\left(y_i - m_k^{(q)}\right)^t$ is the covariance matrix of the kth function and $\hat{u}_j^{(q)}$ is the jth column vector of $\hat{U}^{(q)} \cdot n_k^{(q)} = \sum_{i=1}^{n} t_{ik}^{(q)}$. Maximizing Q conditionally to $\hat{U}^{(q)}$ leads to the following formula for updating the mixture parameters of the model $\text{DLM}_{[\sum_k \cdot \beta_k]}$.

$$\hat{\pi}_k^{(q)} = \frac{n_k^{(q)}}{n} \tag{13}$$

$$\hat{\mu}_k^{(q)} = \frac{1}{n_k^{(q)}} \sum_{i=1}^{n} t_{ik}^{(q)} \hat{U}^{(q)t} y_i \tag{14}$$

$$\sum_{k}^{(q)} \hat{U}^{(q)t} C_k \hat{U}^{(q)} \tag{15}$$

$$\hat{\beta}_k^{(q)} = \frac{\text{trace}(C_k) - \sum_{j=1}^{d} \hat{u}_j^{(q)t} C_k \hat{u}_j^{(q)}}{p-d} \tag{16}$$

In this section, the feature space is adaptively segmented into subspaces that no cluster is shared between adjacent regions. In this regard, FEM is applied at first to both and decease the dimensionality of input samples as well as locating the clusters as far as possible. Hence, the possibility of getting a better clustering performance is increased in the projected space. After applying FEM to input space, some subspaces are created in which we expect to have different groups of adjacent samples in each subspace. Actually, by applying FEM, input samples are projected into a more separable space, in which the clustering performance can be improved. According to the characteristics of data, the number of subfields is specified. Figure 2 shows a famous dataset that is projected into four subspaces by FEM.

3.1.2 Gap-Statistic

Estimating the proper number of clusters in a dataset is a challenging issue. The Gap statistic is a standard method for determining the number of clusters in a set of data. Gap statistics is proposed (Tibshirani et al. 2001) to determine the proper number of clusters in a multidimensional dataset $\{x_{ij}\}$, $i = 1, 2, ..., n$, $j = 1, 2, ..., p$, where n is the number of samples and p in the number of attributes. The aim is to find the number of clusters (k) $C_1, C_2, ..., C_k$, where C_r is the indexes of samples in the rth cluster ($n_r = |C_r|$). Let d_{ij} be the Euclidean distance between two samples i and j. Summation of the pairwise distances D_r for all points in the cluster r is:

$$D_r = \sum_{ij' \in C_r} d_{ij} \tag{17}$$

And

$$W_k = \sum_{r=1}^{k} \frac{1}{2n_r} D_r \tag{18}$$

where W_k is the summation of within-cluster of all clusters. To calculate the Gap index, the difference between the expected value of $\log(W_{k^*})$, under the reference distribution, and the $\log(W_k)$ is determined as follows:

$$\text{Gap}_n(k) = E_n^* \log\left(W_k^*\right) - \log(W_k) \tag{19}$$

The suitable number of clusters is the smallest k that satisfies the following relation.

$$\text{Gap}_n(k) \geq \text{Gap}_n(k+1) - s_{k+1} \tag{20}$$

where s_k is the simulation error determined by the standard deviation $\text{sd}_{(k)}$. The expected value $E_n^* \log\left(W_k^*\right)$ of within-dispersion is determined by the following equation:

$$E_n^* \log\left(W_k^*\right) = \frac{1}{B}\sum \log\left(W_{kb}^*\right) \tag{21}$$

where W_{kb}^* is determined by clustering the B reference samples. If the data has just one dimension, the reference

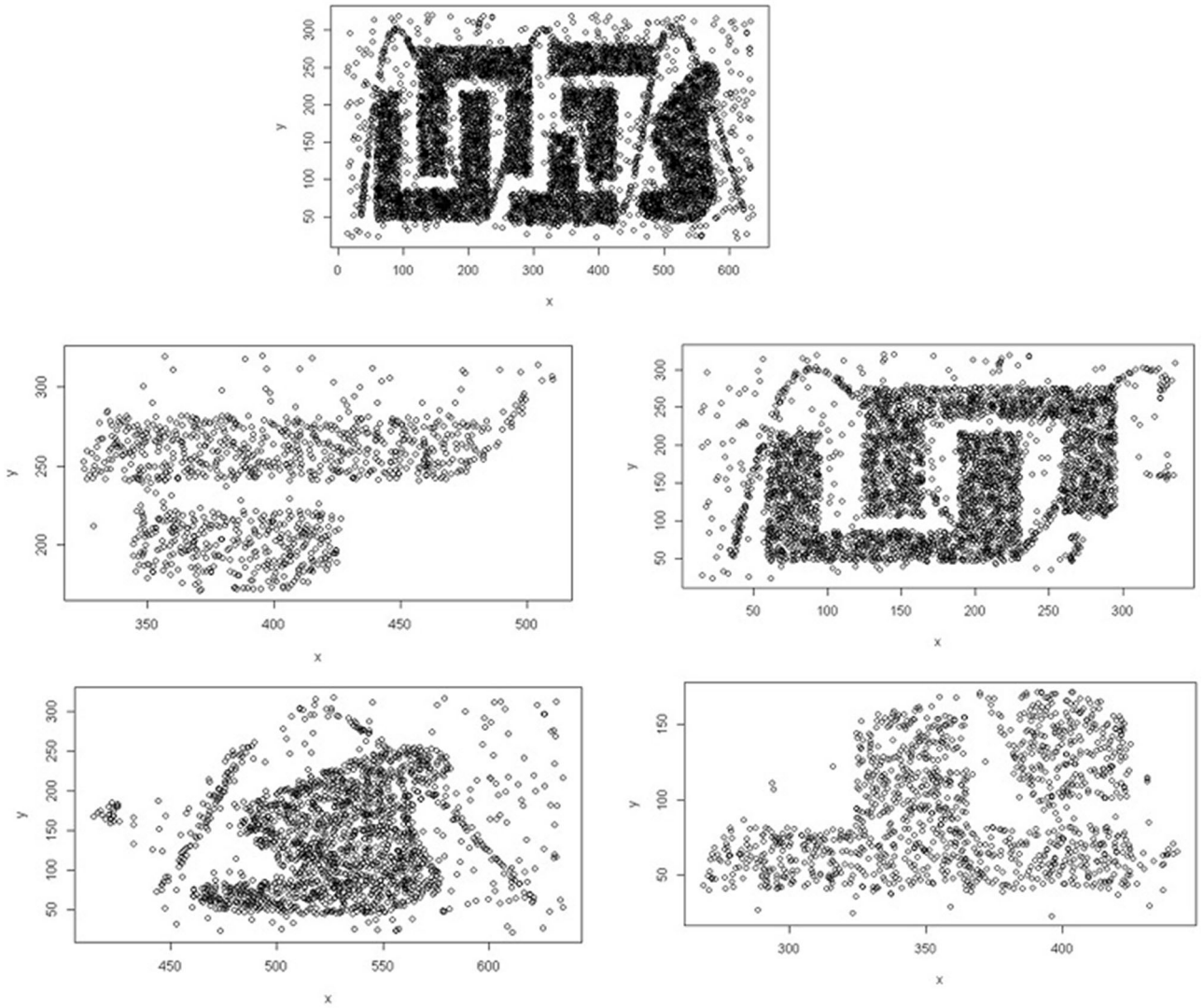

Fig. 2 Applying FEM to t4.8k dataset is illustrated

distribution is selected as a uniform distribution. For multidimensional data, p separate uniform distributions are generated and for each dimension, the corresponding generated attribute is drawn from the corresponding distribution.

3.1.3 Estimation of ε in Each Subspace

Considering the Gap statistics results, we want to tune the parameters of *Min-Pts* and ε for each subspace. For this purpose, we adopt the method proposed in Rahmah and Sitanggang (2016), in which the distance from each point to its kth nearest neighbor (call it as *K-dist*) is determined and then sort all points according to their *K-dist* values. In the *K*-dist graph, the sudden gradient changes determine the value of ε. Figure 12 shows graph for tune of ε of DBSCAN.

The algorithm begins with calculating the Euclidean distance of all pairs of samples by the *dist* function. Therefore, a distance matrix is constructed, which is then normalized in order to facilitate the search for the kth nearest neighbors on each line of this matrix. For each sample, the distances are arranged in ascending order to find the kth nearest neighbor. The *K*-dist values are plotted and for each sudden change in the slope of the curve, the corresponding value of ε is determined. Afterward, we execute DBSCAN on each subspace with respect to the corresponding Min-Pts and ε. The following is the suggested pseudo-code:

Pseudo-code of the proposed algorithm

Input: D, K

Output: set of clusters

1: **begin**

2: **for** each dataset D **do**

3: call library (FEM)

4: data(D)

5: r←fem (D [… …], K)

6: // r shows numbers of subspace

7: **end for**

8: **while** there is subspace **do**

9: // Determine the number of clusters in each subspace with Gap Statistic

10: Clustering of any space taking into account the value of k in the range of 1 to k_{MAX}

11: Calculate the amount of in-cluster differences for each performance and place it in the W_K variable

12: Generate B data sets from each subspace, somehow having a uniform random distribution

13: Clustering of each of these B datasets for different values of k (in the range of 1 to k_{MAX}).

14: Calculate the amount of in-cluster differences for each performance and place it in the W_{kb} variable

15: Calculate the gap statistic as the difference of the observed values of W_K from their expected values under the assumption of zero W_{kb}

16: Calculate the standard deviation of the statistic: $Gap(k) = 1/B(sum(log(W_{kb}) - log(W_K)))$

17: Select the optimal number of clusters with a minimum value of k such that $Gap(k) >= Gap(k + 1) - s_{K+1}$

18: **end while**

19: find suitable ε for each level of density in subspace

20: **for** any subspace have size "n" **do**

21: **for** i

22: **for** $j = 1$ to n

23: $D(i, j) \leftarrow$ find the distance to (x_i, x_j)

24: Find minimum values of distances to nearest k

25: **end for**

26: **end for**

27: **end for**

28: Sort distances ascending and plot to find each value

29: ε corresponds to critical change in curves

30: **end**

As mentioned in the proposed method, FEM is first used to determine each subspace. The feature space is adaptively segmented into subspaces that no cluster is shared between adjacent regions. In this method, FEM is applied at first to both decease the dimensionality of input samples as well as locating the clusters as far as possible. Hence, the possibility of getting a better clustering performance is increased in the projected space. After applying FEM to input space, some subspaces are created in which we expect to have a different groups of adjacent samples in each subspace. Then, the Gap Statistic is used to determine the number of optimal clusters in each subspace. In Gap statistics, the amount of in-cluster differences is done for each of the performances and this value is placed in the variable W_K. Number B datasets are generated from each subspace in such a way that they have a uniform random distribution. Each of these B datasets is clustered for different values of k (in the range of 1 to $K_{\max}$). Then, the amount of in-cluster differences for each of the performances is calculated and the W_{kb} variable is placed. The gap statistic is then calculated as the difference of the observed values of W_K from their expected values under the assumption of zero W_{kb}, and then the standard deviation of the statistic is obtained. The optimal number of clusters is obtained as the minimum value of k so $Gap(k) >= Gap(k + 1) - s_{K+1}$. Considering the Gap statistics results, we want to tune the parameters of Min-Pts and ε for each subspace. For this purpose, in which the distance from each point to its kth nearest neighbor (call it as K-dist) is determined and then sort all points according to their K-dist values. In the K-dist graph, the sudden gradient changes determine the value of ε. Figure 10 shows graph for tune of ε of DBSCAN. The algorithm begins with calculating the Euclidean distance of all pairs of samples by the dist function. Therefore, a distance matrix is constructed, which is then normalized in order to facilitate the search for the kth nearest neighbors on each line of this matrix. For each sample, the distances are arranged in ascending order to find the kth nearest neighbor. The K-dist values are plotted and for each sudden change in the slope of the curve, the corresponding value of ε is determined. Afterward, we execute DBSCAN on each subspace with respect to the corresponding Min-Pts and ε.

3.2 Datasets

To evaluate the proposed method, several datasets were adopted, some of which have label and some of others have no label. The datasets are expressed in terms of number of samples and number of dimensions. Since labeling is a costly and time-consuming process, most of big datasets are unlabeled, though for medium size datasets, labeled datasets are used to measure. Table 1 shows the characteristics of the deployed datasets (Fränti and Sieranoja 2018) in this research.

4 Evaluation

Labeled datasets are used to assess the proposed clustering method in terms of purity, which measures how many samples in a cluster have the same label. The clustering

Table 1 Characteristics of the deployed datasets

Dataset name	Number of data	Number of dimensions	Number of clusters
Unlabeled datasets			
T4.8k	8000	2	6
Brich2	11,000	2	100
Flame	240	2	2
Labeled datasets			
R15	600	3	15
D31	3100	2	31
Unbalance	6500	2	8
JSI	600	3	8
Yeast	1484	8	10
Breast	699	9	2
Iris	150	4	3
Wine	178	13	3
Glass	240	9	7
Diabetes	768	8	2
Bupa	345	5	2

performance for unlabeled datasets is assessed under Fisher value (Duda and Hart 1973), Davies–Bouldin (DB) (Davies and Bouldin 1979) and Silhouette value (Rousseeuw 1987). What follows is the explanation of these criteria.

One of the simplest external evaluation indicators in clustering is the Purity criterion, which measures the percentage of correspondence between clustering tags and actual tags. In this case, the label of each cluster matches the actual label of the category that has the most subscriptions, and the number of points in the cluster that are classified in the correct category are counted. The ratio of this number to the total number of points makes up the purity criterion. In other words, purity criterion is the percent of the total number of samples that are correctly classified in the unit range [0,1]. A value close to 1 indicates a higher accuracy of data clustering and is determined as follows:

$$\text{Purity} = \sum_{r=1}^{k} \frac{n_r}{n} P(S_r) \tag{22}$$

where $P(S_r)$ examines the accuracy of the rth cluster and n_r denotes the number of samples in the rth cluster and n the total number of samples. In this respect, we consider the maximum distribution of samples for a cluster.

Depending on how the purity criterion is calculated, it is clear that the maximum value for it will be 1, and this happens when the labels obtained from the clustering match exactly with the actual labels. Also, if no cluster tag matches the actual tag, this index becomes zero.

Fisher value for a clustering method indicates that how much the clusters are concentrated around their means simultaneous with how much the clusters are far from each other. In this regard, each cluster is considered as a class and by taking the trace of between-class scatter matrix (trace(S_b)) divides by the trace of the within class scatter matrix (trace(S_w)), the Fisher value is determined. The between and within class scatter matrixes are defined below:

$$S_b = \frac{1}{N} \sum_{i=1}^{c} L_i S_{bi} \tag{23}$$

$$S_{bi} = (x - \mu_i)(x - \mu_i)^T \tag{24}$$

where L_i and μ_i are the number of samples and the mean of the ith cluster, respectively. In addition, S_w is the summation of all within class scatter matrixes and S_{wi} is the within class scatter matrix of the ith cluster.

$$S_w = \frac{1}{N} \sum_{i=1}^{c} S_{wi} \tag{25}$$

$$S_{wi} = \sum_{j \in l_i} (x_j - \mu_i)(x_j - \mu_i)^T \tag{26}$$

$$F\_\text{value} = \frac{\text{trace}\ (S_B)}{\text{trace}(S_W)} \tag{27}$$

DB measures the status of two by two clusters and for each cluster, the worst value is selected. In fact, it does not depend on the number of clusters or the clustering algorithm and this index calculates the average of the maximum internal dispersion to dispersion ratio between clusters. The final value of this index is an average over the worst values of all clusters. The similarity measure R_{ij} for the ith, jth clusters is determined as follows:

$$R_{ij} = \frac{s_i + s_j}{D_{ij}} \tag{28}$$

where S_i and S_j are the variances of the ith, jth clusters, respectively, and D_{ij} is the distance between their means. The worst case for the ith cluster maximizes the R_{ij}over the clusters. The DB index is determined as follows:

$$DB_i = {}_j^{\text{Max}} R_{ij}, DB = \frac{1}{C}\sum_{i=1}^{C} DB_i \tag{29}$$

where C is the number of clusters. Silhouette value is determined for each sample and measures its belongingness to its cluster compared to other clusters. This index is defined below:

$$S(i) = \frac{b(i) - a(i)}{max[a(i), b\ (i)]} \tag{30}$$

where $a(i)$ is average distance of simple i with the other samples in the same cluster and $b(i)$ is the minimum distance of sample i with all samples in other clusters. $S(i)$ can be in the interval of $[-1, 1]$, where negative value implies that this sample is not belonged to its cluster and vice versa. Positive value in the summation of silhouette values of samples within each cluster shows its validity while if this summation be negative, it shows that this cluster should be removed and its samples should be assigned to its neighbor clusters.

F-measure is an external evaluation criterion for measuring accuracy of clustering methods. This criterion depends on two factors: precision and recall. F-score is computed by weighted mean of recall and precision.

$$\text{Recall}(x, y) = N_{xy}/N_x \tag{31}$$

$$\text{Precision}(x, y) = N_{xy}/N_y \tag{32}$$

where N_y is element of cluster y and N_x is number of element of class x for class x and cluster y, N_{xy} is numbers of elements of class x in cluster y. Following equation calculates F-measure for class x and cluster y as follows:

$$F(x, y) = \frac{2 \times \text{Recall}(x, y) \times \text{Precision}(x, y))}{(\text{Precision}(x, y) + \text{Recall}(x, y))} \tag{33}$$

Calculated F-measure is a result of weighted average of Precision and Recall for each class x, as shown in Eq. (34).

$$\text{FC} = \frac{\sum_i (|i| \times F(i))}{\sum_i |i|} \tag{34}$$

where $|i|$ is the size of class i. This criterion is one at best and zero at worst.

Another method of evaluating clustering is the accuracy criterion, which measures the degree of accuracy. In this criterion, given that $C = \{C_1, \ldots, C_k\}$ is a set of clusters, and a_i is equal to number of data that are in their correct clustering (C_i) and the dataset has n objects and k is the number of clusters, then accuracy is measured by Eq. (35):

$$\text{AC} = \frac{\sum_{i=1}^{k} a_i}{n} \tag{35}$$

When labeled data presented through clustering methods, two methods of accuracy and F-measure lend themselves to use. Note that, data cannot be used if they are unlabeled.

5 Experimental Results and Discussions

In this section, results of our method are presented and compared to state-of-the-art methods (OPTICS and STDBSCAN and PDBSCAN), over the described datasets in terms of the mentioned metrics. Afterward, the robustness against noise of the proposed method is determined and finally the computational complexity of the compared methods is compared.

5.1 Clustering Results

In this part, Datasets name and the number of subspaces created by FEM, the number of clusters estimated by Gap statistics, the proper neighborhood radius (ε), and Minpts in each subspace are presented in Table 2.

The results of applying the proposed method and the compared ones over the datasets considered in terms of purity and F-measure and accuracy (for the labeled

Table 2 Estimated the number of subspaces, number of clusters and the hyper-parameters of DBSCAN for different datasets

Dataset name	The number of subspaces created by FEM	# of clusters by Gap statistics	Neighborhood radius (ε)	Minpts
T4.8k	4	8	10	15
Brich2	4	5	1400	15
Flame	3	4	3200	15
R15	4	9	0.5	15
D31	3	14	1400	15
Unbalance	4	8	1	15
JSI	4	3	0.5	15
Yeast	4	1	0.5	15
Breast	1	2	0.5	15
Iris	1	2	0.5	15
Wine	1	2	0.5	15
Glass	1	2	0.5	15
Diabetes	1	3	0.5	15
Bupa	1	3	0.5	15

datasets) and the three other metrics (for the unlabeled datasets) are shown in Figs. 3, 4, 5, 6, 7, and 8. The criterion of "accuracy" is calculated on the basis of percentage. Note that the best parameter is set through the cross-validation for each dataset.

As it can be seen in Figs. 3, 4, and 5 the proposed method provides a higher purity and *F*-measure than the other compared methods on the labeled datasets. Similarly, in Figs. 6 and 7, the proposed method has outperformed the rivals in terms of Fisher value and Davies–Bouldin index, while the results in terms of Silhouette value in Fig. 8 do not show any significant superiority.

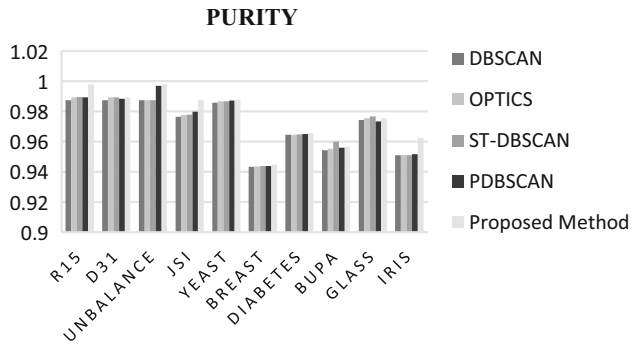

Fig. 3 Purity of the proposed method compares to the counterparts

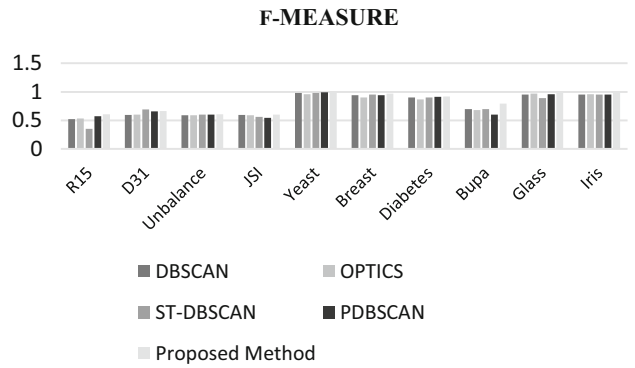

Fig. 4 *F*-measure of the proposed method compares to the counterparts

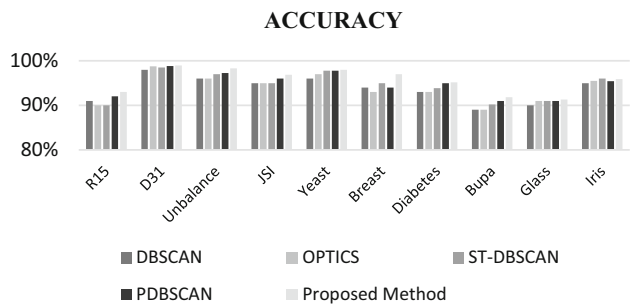

Fig. 5 Accuracy of the proposed method compares to the counterparts

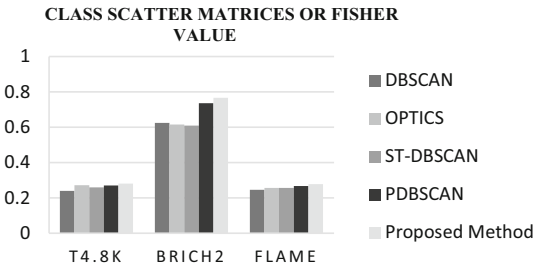

Fig. 6 Clustering results of the compared methods in terms of Fisher value

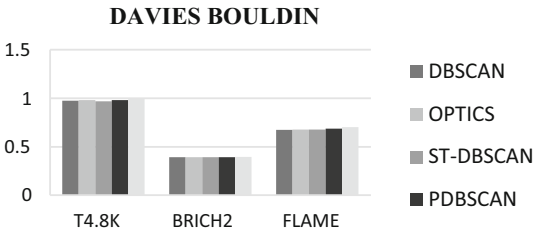

Fig. 7 Clustering results of the compared methods in terms of DB index

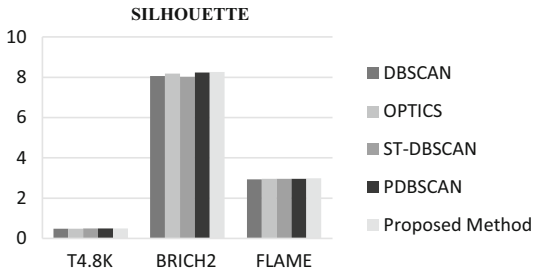

Fig. 8 Clustering results of the compared methods in terms of Silhouette value

5.2 Robustness Against Noise

It is known that during each data recording, some undesired signals are added due to several reasons like the noise of environment, power line and noise of the apparatus. Noises can be added, multiplied or nonlinearly invade the measured signals and corrupt its information. It is obvious that noise can adversely affect the results of data analysis. Since additive noise is the most common noise, here the researchers added noise to the samples and evaluate the results under the noisy condition. In the labeled datasets, noise can be simulated by randomly change the label of a certain ratio of the whole population. Here, a dataset selected and after inserting the noise, the samples are depicted in Fig. 9 (left figure shows the original samples and right figure shows the noisy samples).

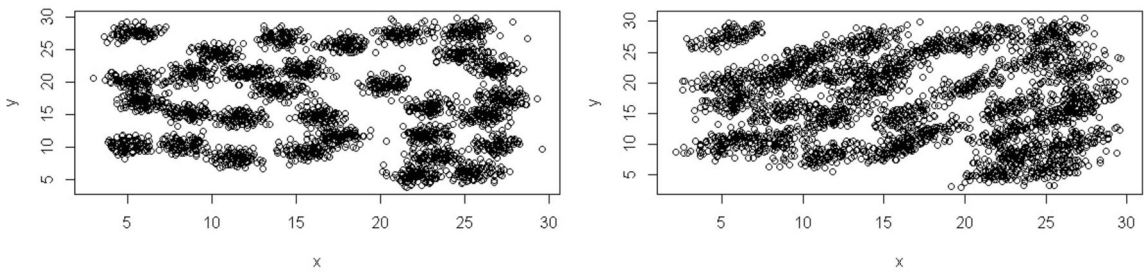

Fig. 9 Selected data to add noise in the proposed method

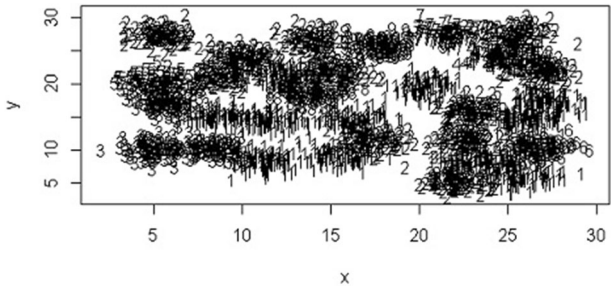

Fig. 10 FEM methods in data with noise in the proposed method

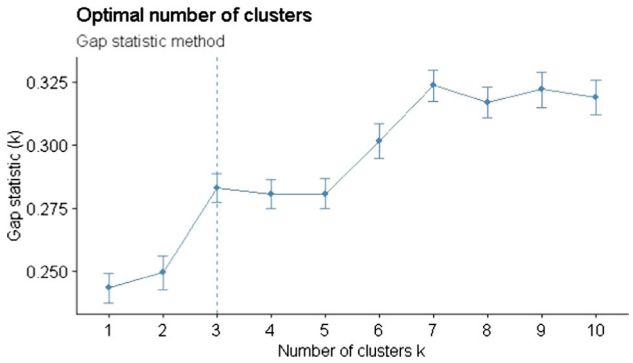

Fig. 11 Instead of number of subspaces

Applying FEM to the noisy samples in Fig. 9 is illustrated in Fig. 10. We have estimated the number of subspaces by the Gap statistics as shown in Fig. 11.

The proper value of ε is determined by the maximum slope change in Fig. 12. According to this diagram, this

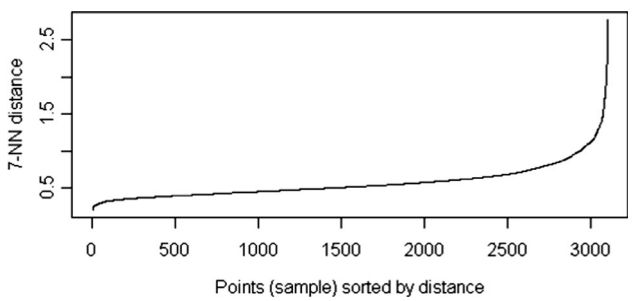

Fig. 12 Determination of ε per subspace in noise dataset

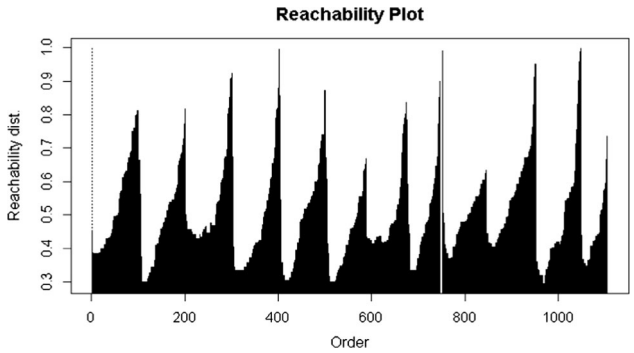

Fig. 13 Reachability plot in the proposed method with noise (ε = 0.8, MinPts = 15)

parameter is set to $\varepsilon = 0.8$ with MinPts = 15, for each subspaces. We show the reachability plot of the proposed method in presence of noise in Fig. 13.

The researchers evaluated the proposed method in two labeled datasets in presence of noise in terms of purity and *F*-measure and accuracy and the results are illustrated in Table 3.

5.3 Computational Complexity

In this proposed method, sub-space splitting is performed by FEM, which is optimized by the EM algorithm. EM, is not very easy. Objects are not assigned to a single space. First, the complexity of FEM is calculated in each subspace:

$$\mathrm{E}[z_n] = M^{-1}W^T(x_n - \overline{x}) \tag{36}$$

$$\mathrm{E}\left[z_n z_n^T\right] = \sigma^2 M^{-1} + W^T \mathrm{E}[z_n](x_n - \overline{x}) \tag{37}$$

$$W_{\text{new}} = \left[\sum_{n=1}^{N}(x_n - \overline{x})E[z_n]^T\right]\left[\sum_{n=1}^{N}E\left[z_n z_n^T\right]\right]^{-1} \tag{38}$$

$$\partial^2_{\text{new}} = \frac{1}{ND}\sum_{n=1}^{N}\{x_n - \overline{x}^2 - 2E[z_n]^T W_{\text{new}}^T(x_n - \overline{x}) + \mathrm{Tr}\left(E\left[z_n z_n^T\right]W_{\text{new}}^T W_{\text{new}}\right)\} \tag{39}$$

Here M is an $M \times M$ matrix, W is a $D \times M$ matrix and $(x_n - x)$ is vector with $D \times 1$ dimensions. Given the

Table 3 Compare the results of noisy datasets

Dataset name	D31		R15	
Number of data	3100		600	
Data status	Labeled		Labeled	
Subspace number without noise	4		4	
Subspace number with noise	7		9	
Estimating the number of clusters in any subspace with noise	Subspace#1	14	Subspace#1	2
	Subspace#2	12	Subspace#2	9
	Subspace#3	7	Subspace#3	2
	Subspace#4	4	Subspace#4	8
	Subspace#5	13	Subspace#5	2
	Subspace#6	10	Subspace#6	3
			Subspace#7	7
	Subspace#7	7	Subspace#8	9
			Subspace#9	2
Determine the optimal neighborhood radius (ε)	1		0.5	
Determine the minimum points in a cluster (minpts)	15		15	
Determine the minimum distance to merge two adjacent cluster (eps_cl)	Subspace#1	0.65	Subspace#1	0.5
	Subspace#2	0.66	Subspace#2	0.5
	Subspace#3	0.67	Subspace#3	0.5
			Subspace#4	0.5
	Subspace#4	0.71	Subspace#5	0.5
	Subspace#5	0.70	Subspace#6	0.5
	Subspace#6	0.66	Subspace#7	0.5
			Subspace#8	0.5
	Subspace#7	0.65	Subspace#9	0.5
Purity in without noise	Subspace#1	0.989154	Subspace#1	1
	Subspace#2	0.9817792	Subspace#2	0.9914286
	Subspace#3	0.9845475	Subspace#3	1
	Subspace#4	0.9778393	Subspace#4	1
Purity in with noise	Subspace#1	0.979344	Subspace#1	1
	Subspace#2	0.9584	Subspace#2	0.96951
	Subspace#3	0.987434	Subspace#3	1
	Subspace#4	1	Subspace#4	1
			Subspace#5	1
	Subspace#5	0.98213	Subspace#6	1
	Subspace#6	1	Subspace#7	0.9452
	Subspace#7	1	Subspace#8	1
			Subspace#9	0.96547
F-measure in without noise	Subspace#1	0.0951	Subspace#1	0.067
	Subspace#2	0.0932	Subspace#2	0.069
	Subspace#3	0.0921	Subspace#3	0.062
	Subspace#4	0.0951	Subspace#4	0.070
			Subspace#5	0.073
	Subspace#5	0.0958	Subspace#6	0.067
	Subspace#6	0.0961	Subspace#7	0.084
	Subspace#7	0.0902	Subspace#8	0.069
			Subspace#9	0.078

Table 3 (continued)

Dataset name	D31		R15	
F-measure in with noise	Subspace#1	0.0876	Subspace#1	0.060
	Subspace#2	0.0865	Subspace#2	0.062
	Subspace#3	0.0914	Subspace#3	0.051
	Subspace#4	0.0950	Subspace#4	0.059
			Subspace#5	0.072
	Subspace#5	0.0958	Subspace#6	0.060
	Subspace#6	0.0987	Subspace#7	0.081
	Subspace#7	0.0901	Subspace#8	0.071
			Subspace#9	0.053
Accuracy in without noise	Subspace#1	93%	Subspace#1	97%
	Subspace#2	92%	Subspace#2	94%
	Subspace#3	94%	Subspace#3	93%
	Subspace#4	90%	Subspace#4	92%
			Subspace#5	91%
	Subspace#5	98%	Subspace#6	90%
	Subspace#6	92%	Subspace#7	94%
	Subspace#7	91%	Subspace#8	92%
			Subspace#9	96%
Accuracy in with noise	Subspace#1	88%	Subspace#1	93%
	Subspace#2	90%	Subspace#2	90%
	Subspace#3	91%	Subspace#3	90%
	Subspace#4	85%	Subspace#4	87%
			Subspace#5	84%
	Subspace#5	84%	Subspace#6	89%
	Subspace#6	86.6%	Subspace#7	94%
	Subspace#7	90.1%	Subspace#8	89%
			Subspace#9	90%

number of dimensions in FEM, there is a matrix inversion step that we must invert the covariance matrix. Therefore, the researchers bring the number of dimensions into the complexity of the algorithm. If we presume that the EM algorithm uses linear algebra, which it does, then its complexity should be $O(p \times i \times n \times D)$, where p is number of points and i is the number of iterations and n is the number of parameters and D is number of dimensions. Therefore, the complexity of FEM is determined as $O(p \times i \times n \times D)$. The complexity of Gap statistics is determined by $O(p \times k \times I)$, where p is number of points, k is number of clusters and I is number of iterations. After calculating the number of clusters per subspace, we calculate the ε for each subspace. The complexity of this part is determined $O(n\log n)$. The algorithm DBSCAN on each subspace is $O(n\log n)$, where n is the number of samples in that subspace. Hence, the total number of operations in the proposed methods is determined by:

$$\begin{aligned} &O(p \times i \times n \times D) + O(p \times k \times I) + O(n \log n) \\ &\quad + O(K \times n \log n) \\ &\quad = O(n \log n) \end{aligned} \tag{40}$$

Table 4 Comparing the methods

Name of methods	DBSCAN	OPTICS	DENCLUE	Proposed method
Complexity	$O(n^2)$	$O(n\log n)$	$O(\log \lvert D \rvert)$	$O(n\log n)$
Varied density type	No	No	Yes	Yes
Type of data	Spatial data with noise	Spatial data with noise	Low data	Big data
Shape of cluster	Arbitrary	Arbitrary	Arbitrary	Arbitrary
Noise handling	Not very well	Not very well	Very well	Very well

5.4 Comparative Study of Methods

In this section, compared the proposed method was compared to DBSCAN and OPTICS and DENCLUE in terms of complexity, noise handling, ability to form clusters with different shapes and their ability to handle big datasets. These algorithms were selected because they were all density-based methods. Table 4 illustrates the comparison results.

6 Conclusion

In this paper, an extension of DBSCAN was proposed which was equipped with a FEM algorithm to diminish the input dimension and project the input samples into more separable subspaces. Afterward, the parameters of DBSCAN were tuned according to the Gap statistics diagram on each subspace. Since each DBSCAN was executed on a portion of data (subspace), these algorithms can be run in parallel and the computational complexity of DBSCAN on each subspace is much lower than executing it on the whole data. The proposed method outperforms to OPTICS and DBSCAN, due to OPTICS clustering methods, because DBSCAN uses certain values for the epsilon (radius) and MinPts, which are not necessarily optimal for the whole space. In addition, although OPTICS can detect clusters with different densities, the clusters cannot be explicitly represented and also suffers a high computational complexity. For the future work, it is possible to extend FEM-DBSCAN to stream data in a real-time manner.

References

Bhardwaj S, Dash SK (2015) VDMR-DBSCAN: varied density MapReduce DBSCAN. In: Kumar N, Bhatnagar V (eds) Big data analytics. BDA 2015, Lecture notes in computer science, vol 9498. Springer, Cham

Birant D, Kut A (2007) ST-DBSCAN: an algorithm for clustering spatial–temporal data. Data Knowl Eng 60(1):208–221

Bouveyron C, Brunet C (2011) On the estimation of the latent discriminative subspace in the Fisher-EM algorithm. J Soc Fr Statistique 152(3):98–115

Cai M, Liang Y (2018) An improved CURE algorithm. In: International conference on intelligence science. Springer, Cham, pp 102–111

Chaudhari Chaitali G (2012) Optimizing clustering technique based on partitioning DBSCAN and ant clustering algorithm. Int J Eng Adv Technol 2(2) ISSN 2249-8958

Chowdhury N, Bhattacharjee P (2014) Using an MST-based value for ε in DBSCAN algorithm for obtaining better result. Int J Inf Technol Comput Sci 6:55–60

Davies DL, Bouldin DW (1979) A cluster separation measure. IEEE Trans Pattern Anal Mach Intell (2):224–227

Duan L, Xu L, Guo F, Lee J, Yan B (2007) A local-density based spatial clustering algorithm with noise. Inf Syst 32(7):978–986

Duda RO, Hart PE (1973) Pattern classification and scene analysis, vol 3. Wiley, New York

Fränti P, Sieranoja S (2018) K-means properties on six clustering benchmark datasets. Appl Intell 48(12):4743–4759

He Y, Tan H, Luo W, Feng S, Fan J (2014) MR-DBSCAN: a scalable MapReduce-based DBSCAN algorithm for heavily skewed data. Front Comput Sci 8(1):83–99

Jiang M, Liang Y, Feng X, Fan X, Pei Z, Xue Y, Guan R (2018) Text classification based on deep belief network and softmax regression. Neural Comput Appl 29(1):61–70

Kazemi U (2018) A survey of big data: challenges and specifications. CiiT Int J Softw Eng Technol 10(5):87–93

Kriegel HP, Kröger P, Sander J, Zimek A (2011) Density-based clustering. Wiley Interdiscip Rev Data Min Knowl Discov 1(3):231–240

Lin W, Wu Z, Lin L, Wen A, Li J (2017) An ensemble random forest algorithm for insurance big data analysis. IEEE Access 5:16568–16575

Liu P, Zhou D, Wu N (2007) VDBSCAN: varied density based spatial clustering of applications with noise. In: 2007 International conference on service systems and service management. IEEE, New York, pp 1–4

Nagpal A, Jatain A, Gaur D (2013) Review based on data clustering algorithms. In: 2013 IEEE conference on information & communication technologies. IEEE, New York, pp 298–303

Noticewala M, Vaghela D (2014) MR-IDBSCAN: efficient parallel incremental DBSCAN algorithm using mapreduce. Int J Comput Appl 93(4):13–18

Pratas D, Pinho AJ, Ferreira PJ (2016) Efficient compression of genomic sequences. In: 2016 Data compression conference (DCC). IEEE, New York, pp 231–240

Rahmah N, Sitanggang IS (2016) Determination of optimal epsilon (eps) value on DBSCAN algorithm to clustering data on peatland hotspots in Sumatra. In: IOP conference series: earth and environmental science, vol 31(1). IOP Publishing, Bristol, p 012012

Rousseeuw PJ (1987) Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. Comput Appl Math 20:53–65. https://doi.org/10.1016/0377-0427(87)90125-7

Sadeghi-Mobarakeh A, Mohsenian-Rad H (2018) Performance accuracy scores in CAISO and MISO regulation markets: a comparison based on real data and mathematical analysis. IEEE Trans Power Syst 33(3):3196–3198

Soni N, Ganatra A (2012) Categorization of several clustering algorithms from different perspective: a review. Int J Adv Res Comput Sci Softw Eng 2(8):63–68

Tibshirani R, Walther G, Hastie T (2001) Estimating the number of clusters in a data set via the gap statistic. J R Stat Soc B Stat Methodol 63(2):411–423

Xiong Z, Chen R, Zhang Y, Zhang X (2012) Multi-density DBSCAN algorithm based on density levels partitioning. J Inf Comput Sci 9(10):2739–2749

Ye X, Sakurai T (2015) Spectral clustering using robust similarity measure based on closeness of shared nearest neighbors. In: 2015 international joint conference on neural networks (IJCNN). IEEE, pp 1–8

Zhou S, Liu X, Li M, Zhu E, Liu L, Zhang C, Yin J (2019) Multiple kernel clustering with neighbor-kernel subspace segmentation. IEEE Trans Neural Netw Learn Syst 31(4):1351–1362