An Efficient Density-based Clustering Algorithm for Face Identification

Shenfei Pei, Feiping Nie, Rong Wang, and Xuelong Li, *Fellow, IEEE*

***Abstract*—This paper focuses on the following problem: Given a large number of unlabeled face images, group them into the individual clusters where the number of the clusters to construct is not known. The families of hierarchical and density-based clustering methods are almost the most widely used algorithms in such scenes. However, most of them are difficultly scalable to large-scale problems due to their high computational overhead. And their performance is greatly affected by the user-specified parameters that are difficult to tune in practical applications.**

To this end, an Efficient Density-based clustering incorporated with the model of Graph partitioning called EDG is proposed. (1) EDG has a very low computational and memory overhead, is easily scalable and applicable to large-scale practical problems. Specifically, the computational and memory overhead are $O(NK^3)$ and $O(NK)$, respectively. (2) Inspired by the progress of graph partitioning clustering, a novel criterion that can be seen as a variant of the Normalized-cut model is employed to measure the similarity between two samples. Benefiting from the novel criterion, our algorithm shows satisfactory performance on both synthetic and real-world datasets. (3) Only one user-specified parameter (the number of nearest neighbors) is involved in our model. Different from the user-specified parameters involved in other density-based methods, it is an integer and is easy to tune.

In order to assess the performance of EDG on face images, extensive experiments based on a two-stage framework have been conducted on 25 benchmark datasets (20 middle-scale and 5 large-scale) from the literature. The experimental results have shown the effectiveness and robustness of our model, compared with the state-of-the-art methods. [code release]

***Index Terms*—Fast clustering, linkage-based, density-based, Normalized-cut.**

I. Introduction

With the development and popularization of information science, the amount of data, for example, text, image, and video have been increased rapidly. And the large-scale data can be found everywhere in our lives at present, such as photos and reports of celebrities on the internet, the news generated on social platforms, and transaction records generated on e-commerce platforms.

This work was supported in part by the National Key Research and Development Program of China under Grant 2018AAA0101902, in part by the National Natural Science Foundation of China under Grant 61936014, Grant 61772427 and Grant 61751202, and in part by the Fundamental Research Funds for the Central Universities under Grant G2019KY0501.

Corresponding author: Feiping Nie.

S. Pei, F. Nie, and X. Li are with the School of Computer Science and Center for OPTical IMagery Analysis and Learning (OPTIMAL), Northwestern Polytechnical University, Xi'an, 710072, Shaanxi, P. R. China e-mail: (shenfeipei@gmail.com, feipingnie@gmail.com, xuelong_li@nwpu.edu.cn).

R. Wang is with the School of Cybersecurity and Center for OPTical IMagery Analysis and Learning (OPTIMAL), Northwestern Polytechnical University, Xi'an, 710072, Shaanxi, P. R. China e-mail: (wangrong07@tsinghua.org.cn).

Considering social media, 350 million photos are uploaded every day, 20 million friend requests are sent every 20 Minutes, and 5 billion comments are left on Facebook pages monthly, reported by Facebook[1]. These large-scale data are relatively easy to collect but further analysis is required to obtain useful information from these data.

As one of the fundamental techniques in pattern recognition, data mining, machine learning, and many other fields, clustering is getting more and more attention in recent years. A series of algorithms have been proposed for cluster analysis and applied to various areas successfully in recent years, such as document clustering [1], image segmentation [2], load profiling [3], and social networks [4], [5].

Hierarchical and density-based clustering methods are almost the most widely used algorithms for such scenes in which the number of clusters to construct is unknown. However, most of them are arduously scalable to large-scale problems for their high computational overhead. And their performance is greatly affected by the user-specified parameters that are difficult to tune in practical applications.

(1) In most hierarchical clustering methods such FIN [6], PAHC [7], multiple clustering results will be output, each of which has a different number of clusters. Therefore, a user-specified parameter is needed to identify which clustering result will be used ultimately. Since the true number of clusters in the dataset cannot be known in advance, in practical applications, authors usually evaluated their methods at several effective values of the user-specified parameter and reported the best results, in their papers.

(2) In most density-based algorithms such as DBSCAN [8], DP [9], a user-specified parameter γ is required to compute the local density for each sample as follows

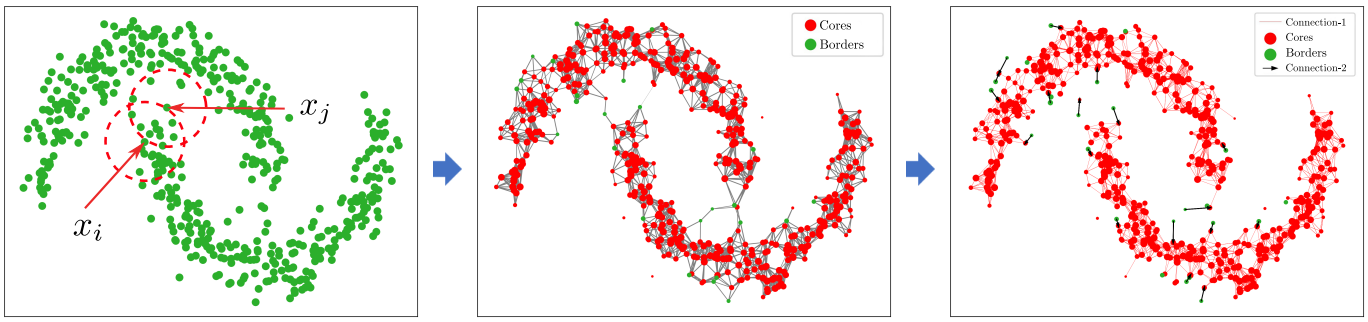
$$\rho_i = \sum_{j=1}^{N} \chi\left(D_{ij} - \gamma\right), \tag{1}$$

where $\chi(\cdot)$ is an indicator function, namely $\chi(x) = 1$ if $x \leq 0$, and $\chi(x) = 0$ otherwise. ρ_i represents the local density of i-th sample. D_{ij} denotes the distance between i-th and j-th sample.

Although many improvements have been made [10], [11], clustering large-scale datasets remains a challenge for these algorithms. A detailed description of these algorithms is given in Section II (Related work).

Next, we briefly introduce the proposed algorithm according to the main concepts involved. To begin with, we denote the i-th sample in the dataset by x_i.

[1]https://www.omnicoreagency.com/facebook-statistics/



(a) A novel criterion (b) Compute the local density (c) Establish connections

Fig. 1. Schematic diagram of EDG: (a) Compute the similarity for each sample pair. (b) Obtain the local density of each sample based on the similarity. Thicker line denotes higher similarity and larger point denotes greater local density. (c) Establish the connections. Connection-1 and connection-2 denote the connection starting from core and border, respectively. Clustering result is presented by the connected components of the graph.

a) A novel criterion: We observe that the objective function value of spectral clustering can reflect the separability of the two clusters, as shown in Fig. 2, i.e. the closer the two clusters are, the greater the objective value of spectral clustering, which is the main proposition of this paper. That is to say, the similarity can be revealed by the separability of the two clusters. With this observation, a local Normalized-cut model is proposed to measure the similarity between samples x_i and x_j. Specifically, EDG converts the similarity of the two samples to the similarity of the two clusters consisting of k-nearest neighbors of x_i and x_j respectively. To the best of our knowledge, our method is the first to use Normalized-cut model in density-based clustering methods. Benefiting from the Normalized-cut model, there is only one parameter K, the number of nearest neighbors, is involved when measuring the similarity between samples.

b) Local density: According to graph theory, nodes with high degrees usually mean something, such as influential nodes in a social network, and important pages on the web. Therefore, we take the degree of node x_i as its local density. That is to say

$$\rho_i = \sum_{x_j \in N(x_i)} Similarity(x_i, x_j) \tag{2}$$

where, $N(x_i)$ represents the set of some neighbors of x_i, and its definition will be described in detail in Section IV. As can be seen from Eq. (2), the local density of each sample can be determined automatically without any parameters, which is the main difference from other density-based clustering methods such as DP, DBSCAN.

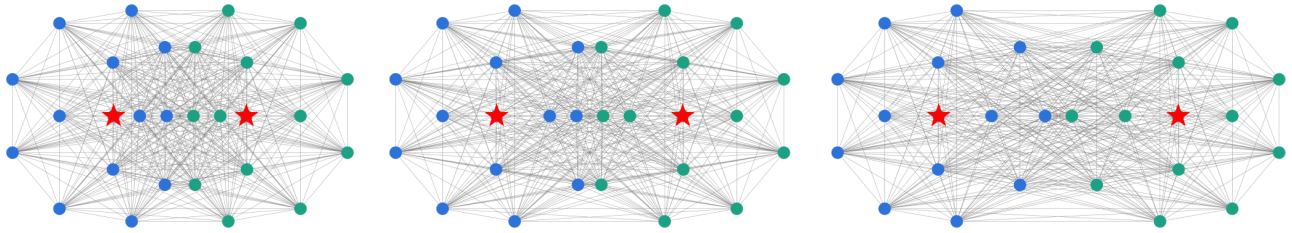
c) Connections: According to the local density of samples, EDG first divides them into two classes: core and border points. Then, the connections between border and core, core and core are established, to obtain the final clustering result. No connection will be established between border and border. The connection between the border and core samples is directed, which ensures that the samples in different clusters will not be connected via borders as shown in Fig. 1(c).

Honestly, parameters are used to identify whether the point is a core or a border. However, the parameters are automatically determined and can even be ignored by users.

In this paper, we consider the problem of grouping a collection of unlabeled face images, where the number of subjects is not known. An Efficient Density-based clustering algorithm incorporated with the model of Graph partitioning, call EDG, is proposed, inspired by the recent progress on density-based and partition-based clustering. Specifically, our method employs a novel criterion to measure the similarity between samples which is a variant of the normalized-cut model. While maintaining low computational complexity, EDG can detect clusters with an arbitrary shape. It is worthwhile to highlight the main contributions of this paper as follows:

- To the best of our knowledge, EDG is the first density-based clustering method incorporated with the graph partition model. Benefiting from the Normalized-cut model, our algorithm shows satisfactory performance on both synthetic and real-world datasets.
- There is only one parameter K, the number of nearest neighbors of each sample, is involved when measuring the similarity between samples. Different from ε, $MinPts$ in DBSCAN, and δ in DP, this parameter is an integer, which is easy to tune.
- We perform only one round of merges of individual images into clusters, and additionally only check for merges on a subset of all possible pairs (the top-K nearest neighbors for each sample), which leads that the computational and memory overhead of EDG are both linear with respect to the number of samples. In addition, the optimization algorithm can be operated in parallel, which is important for large-scale problems.
- Extensive experiments on 25 benchmark datasets and 4 synthetic datasets validate the advantage of EDG compared to the state-of-the-art clustering algorithms.

Notations: Throughout the paper, vectors are written as boldface lowercase letters, and the matrices are represented by uppercase letters. A matrix is called cluster indicator matrix, if where each row consists of one and only one element equal to 1 to indicate the cluster membership, while the rest elements are all 0. The set of all cluster indicator matrices is denoted by Φ. The trace of matrix A is denoted by $Tr(A)$. $X = [x_1, \cdots, x_N]^T \in R^{N \times d}$ denotes the whole dataset, where x_i represents i-th sample. $x_i^{(j)}$ denotes the j-th nearest neighbor of x_i in terms of Euclidean distance.



(a) $Obj = 0.357$ (b) $Obj = 0.182$ (c) $Obj = 0.087$

Fig. 2. The similarity calculated by the objective function of Normalized-cut. The sample pairs we focus on are indicated by red asterisks. The adjacency matrix W is constructed by heat kernel, and t, the parameter involved in heat kernel, is fixed at 10.

II. Related work

Among various clustering methods, partitional, hierarchical, and density-based methods have almost got the most attention. The proposed method EDG can be seen as a combination of density- and partition-based clustering. So an introduction to both is given below.

a) Partition-based clustering: These methods usually aim to discover the structure of clusters in the dataset by optimizing a specific objective function and iteratively improving the quality of the partitions according to the value of the objective function. Generally, the number of clusters to construct is **required** by those methods.

Among various graph partitioning clustering algorithms, spectral clustering methods [12], [13] developed from Normalized-cut gain almost the most popularity since their simplicity and efficacy. However, most of them are difficultly scalable to large-scale problems, for their high time and space complexity. To this end, much effort has been devoted. (1) [14] compute the approximate solution of the eigenvalue decomposition problem by using the classical Nyström algorithm. Based on the randomized low-rank matrix approximation method, [15] proposed a scalable Nyström scheme to further accelerate the algorithm. And Nyström algorithm was revisited in [16] for hypergraph clustering. (2) In recent years, the anchor-based approaches [17], [18] have attracted a lot of attention. These methods transform the problem of eigenvalue decomposition into the problem of singular value decomposition, thereby greatly reducing the computational complexity. However, the performance of these methods is sensitive to how the anchor-based graph is constructed and which samples are selected as anchors. Based on the anchor-based graph [19] introduced two efficient approximation techniques to construct the affinity matrix. (3) In addition, [20] proposed to model the human visual cortex with sparse coding, which has been shown useful for applications in many fields. Many sparse versions of classical methods have been proposed [21], [22].

b) Density-based clustering: The basic idea of those algorithms is that the samples that are in a region with a high density of the data space are considered to belong to the same cluster. Different from the graph partitioning clustering, these methods usually do not need to specify the number of clusters.

(1) DBSCAN [8] is a well-known density-based clustering method. Its basic idea is that for each sample of a cluster, the neighborhood of a given radius (ε) must contain at least a minimum number of points ($MinPts$), where ε and $MinPts$ are two parameters specified by the user. Despite its good performance, it is difficultly scalable to large-scale problems due to its high complexity. And the performance is greatly affected by the two parameters that are very difficult to tune. (1.1) In order to reduce the computational complexity of DBSCAN, [23], [24] provided various fast implementations of it, utilizing various techniques. It is worth noting that both the original DBSCAN and its fast implementations require the user to specify the value of ε and $MinPts$. However, choosing proper thresholds can be non-trivial, in practical applications. (1.2) In order to reduce the sensitivity to parameters, Ricardo et.al [25] proposed DB-H by transforming DBSCAN into a hierarchical clustering. DB-KNN [26] and DB-RNN [10] employ the K-nearest neighbors graph to set the parameters involved in DBSCAN automatically. Both involve only one parameter, the number of nearest neighbors. More density-based clustering methods can be found here [27], [28].

(2) DP [9] is another significant clustering algorithm proposed in Science in 2014. DP based on the idea that cluster centers are characterized by a higher density ρ than their neighbors and by a relatively large distance δ from points with higher densities. Similar to DBSCAN, DP is limited in its applicability to large-scale problems for its high complexity. (2.1) With the help of a fast K-nearest neighbors algorithm, [29] proposed FastDPeak, in which local density was replaced by KNN-density. [30] employed K-means, a classical clustering method, to enhance the scalability of DP. Its basic idea is to reduce the computational complexity involved in the local density through the clustering results generated by K-means. (2.2) DenPEHC [31] transforms DP into a hierarchical clustering. It generates clusters directly on each possible clustering layer and detects all cluster centers automatically. ADPC-KNN [32] is an adaptive clustering algorithm. As the name suggests, the idea of K-nearest neighbors was introduced to compute the global threshold δ and a new approach was proposed to select cluster centers automatically. Some improvements of DP can be found here [33], [34].

In hierarchical clustering methods, the main problem is how to properly measure the similarity of the samples. PAHC [7] measures the similarity between points by evaluating linear SVM margins. [6] found that the first integer neighbor of each sample is all one needs to find the clusters in the dataset and proposed FIN that has a very low computational complexity, and is scalable to large-scale problems easily. More hierarchical methods can be found in [35], [36].

III. Graph Clustering Revisit

Given a set of data points $X = [x_1, \cdots, x_N]$, the undirected graph $G =< V, E >$ is constructed first in Normalized-cut. It usually is represented by the adjacency matrix $W \in R^{N\times N}$, where W_{ij} denotes the weight of the connection between x_i and x_j. In general, the adjacency matrix W can be determined in the following two ways:

(1) Heat kernel. The weight of connection between nodes x_i and x_j can be determined by

$$W_{ij} = \begin{cases} e^{-\frac{\|x_i - x_j\|^2}{t}} & \text{If } x_i \text{ and } x_j \text{ are connected} \\ 0 & \text{otherwise} \end{cases}, \quad (3)$$

where $t \in R^+$ is a parameter.

(2) Simple-mineded. $W_{ij} = 1$ if x_i and x_j are connected, $W_{ij} = 0$ otherwise.

Once the adjacency matrix W is obtained, the Cut, a basic concept in graph theory, can be easily expressed.

a) Definition: Given two subsets $A_1, A_2 \in V$ where $A_1 \cap A_2 = \oslash$, $A_1 \cup A_2 = V$, a Cut can be represented as

$$Cut(A_1, A_2) = \sum_{i\in A_1, j\in A_2} W_{ij}. \quad (4)$$

Based on the definition of Cut, The objective function of Normalized-cut is as follows

$$N - cut(A_1, \cdots, A_c) = \sum_{k=1}^{c} \frac{cut(A_k, \bar{A}_k)}{vol(A_k)}, \quad (5)$$

where $vol(A_k) = \sum_{i\in A_k} \sum_{j=1}^{n} W_{ij}$ denotes the sum of the weights of all edges attached to vertices in A_k.

We define the indicator matrix Y by

$$Y_{ij} = \begin{cases} 1 & \text{if } x_i \in A_j \\ 0 & \text{otherwise} \end{cases}, i = 1, \cdots, n, j = 1, \cdots, c.$$

Then the problem Normalized-cut in Eq. (5) can be rewrite in a more convenient form

$$\min_{Y\in\Phi^{N\times c}} Tr((Y^T DY)^{-\frac{1}{2}} Y^T (D - W) Y (Y^T DY)^{-\frac{1}{2}}, \quad (6)$$

where D is a diagonal matrix, denotes the degree matrix of graph G, and is defined as $D_{jj} = \sum_{i=1}^{n} W_{ij}$.

Unfortunately the problem in Eq. (6) is NP-hard, which means "at least as hard as any NP-problem". Traditional spectral clustering methods [12] solve the relaxed problem which obtained by substituting $F = D^{-\frac{1}{2}} Y (Y^T DY)^{-\frac{1}{2}}$ and relaxing the discreteness condition.

$$\max_{F^T F = I_c} Tr(F^T D^{-\frac{1}{2}} W D^{-\frac{1}{2}} F). \quad (7)$$

The problem in Eq. (7) can be solved easily by computing the top c eigenvectors of $D^{-\frac{1}{2}} W D^{-\frac{1}{2}}$ corresponding to the c smallest eigenvalues with the computational complexity of $O(N^3)$, where N denotes the number of samples.

It is not difficult to find that traditional spectral clustering derived from Normalized-cut has a very high computational overhead, which can be seen from stage of eigen-decomposition. Although many fast version of it have been proposed, as mentioned in Section II, it is still a great challenge to deal with large-scale problems efficiency with traditional spectral clustering.

IV. An Efficient Density-based Clustering method

In this section, we introduce the proposed algorithm EDG in detail. Our model runs on the approximate K-nearest neighbor graph represented by matrix I and D, where the (i, k)-th entry of matrix I and D denote the rank of $x_i^{(k)}$ in X and the squared Euclidean distance between samples x_i and $x_i^{(k)}$. As a basic concept involved in density-based clustering, the local density can be determined automatically based on the definition of similarity. Thus, the key of our algorithm is the similarity criterion developed from the Normalized-cut model.

A. The proposed similarity criterion

We take the sample pair (x_i, x_j) as an example to illustrate the meaning of the similarity. The model used to measure the similarity between x_i and x_j runs on a local data set $\tilde{X}$. The local data set consists of K-nearest neighbor samples of x_i and x_j Here we focus on the case of $c = 2$, the objective function of Normalized-cut is as follows, as mentioned above,

$$\min_{Y\in\Phi^{N\times 2}} Tr((Y^T \tilde{D} Y)^{-\frac{1}{2}} Y^T (\tilde{D} - \tilde{W}) Y (Y^T \tilde{D} Y)^{-\frac{1}{2}}. \quad (8)$$

where $\tilde{W} \in R^{2K\times 2K}$ represents the local adjacency matrix and is constructed by Eq. (3), $\tilde{D}$ is the degree matrix, i.e., $D_{jj} = \sum_{i=1}^{2K} \tilde{W}_{ij}$. The main idea of the model is to measure the similarity between samples x_i and x_j by the optimal value of Eq. (8). However, solving Eq. (8) for each sample pair will bring a lot of computational overhead. Thus, we take $x_i^{(1)}, \cdots, x_i^{(K)}$ as one cluster, and $x_j^{(1)}, \cdots, x_j^{(K)}$ as another cluster. The value of Eq. (8) then can be obtained directly.

The procedure employed to measure the similarity is then:

(1) Prepare the local data set $\tilde{X}$ in a specific order, that is

$$\tilde{X} = [x_i^{(1)}, \cdots, x_i^{(K)}, x_j^{(1)}, \cdots, x_j^{(K)}]. \quad (9)$$

(2) Compute the local adjacency matrix

$$\tilde{W} = \begin{bmatrix} \tilde{W}_{ii} & \tilde{W}_{ij} \\ \tilde{W}_{ji} & \tilde{W}_{jj} \end{bmatrix}, \quad (10)$$

where

$$\tilde{W}_{ii}(a, b) = exp(-dist(x_i^{(a)}, x_i^{(b)})/t), \quad (11)$$

$$\tilde{W}_{jj}(a, b) = exp(-dist(x_j^{(a)}, x_j^{(b)})/t), \quad (12)$$

$$\tilde{W}_{ij}(a, b) = \tilde{W}_{ij}(b, a) = exp(-dist(x_i^{(a)}, x_j^{(b)})/t). \quad (13)$$

(2.1) $dist(x_m, x_n)$ denotes the squared Euclidean distance between x_m and x_n. According to the definition of D, it can be obtained by traversing the m-th row of D. If D is stored in the form of a hash table, the operation can be completed in $O(1)$ time. $dist(x_m, x_n) = \gamma$, where γ denotes the maximum value in D, If x_n is not among the K-nearest neighbors of x_m. t is the median of D.

(3) Sets A and B are used to represent the approximate clustering results. That is to say,

$$A = \{x_i^{(1)}, \cdots, x_i^{(K)}\}, \quad B = \{x_j^{(1)}, \cdots, x_j^{(K)}\}. \quad (14)$$

So far, the similarity between x_i and x_j, W_{ij}, can be obtained by

$$W_{ij} = \sum_{x_g, x_h \in A} \tilde{L}_{gh}/vol(A) + \sum_{x_g, x_h \in B} \tilde{L}_{gh}/vol(B), \quad (15)$$

Algorithm 1: The proposed algorithm, EDG.

Data: $X \in R^{N\times d}$, number of nearest neighbors K
Result: Clustering result y

1 Compute the K-nearest neighbor graph by EFANNA[37] and denote it by matrix I and D, where the (i,k)-th entry of matrix I and D denote the index of $x_i^{(k)}$ and the squared Euclidean distance between x_i and $x_i^{(k)}$;
2 Initial an empty sparse matrix W with shape (N, N);
3 **for** $i = 1, \cdots, N$ **do**
4 **for** $k = 1, \cdots, K$ **do**
5 $j = I(i, k)$;
6 **if** x_i *is not among the* K*-nearest neighbors of* x_j **then** break;
7 Find the local dataset $\tilde{X}$ as Eq. (9);
8 Compute the local weight matrix $\tilde{W}$ according to Eq. (10) - Eq. (13);
9 Compute the similarity W_{ij} between samples x_i and x_j by Eq. (15);
10 Compute local density for each sample by Eq. (16);
11 Divide samples into cores and borders by Eq. (17);
12 **for** $i = 1, \cdots, N$ **do**
13 **for** $k = 0, \cdots, K$ **do**
14 $j = I(i, k)$;
15 **if** x_i *is not among the* K*-nearest neighbors of* x_j **then** break;
16 **if** x_i *is a core* **then**
17 **if** x_j *is a core* **then**
18 Let x_i and x_j be connected;
19 **else** break;
20 **else**
21 **if** x_j *is a core* **then**
22 Let x_i and x_j be connected;
23 break;
24 The final clustering result y is obtained according to the connected components of the graph.

where $\tilde{L} = \tilde{D} - \tilde{W}$. $\tilde{D}$, a diagonal matrix, is the local degree matrix, $\tilde{D}_{jj} = \sum_j \tilde{W}_{ij}$.

In order to make the classification between the data with very different density regions easier, we only consider the similarity between such sample pair (x_i, x_j) that x_i and x_j are among K-nearest neighbors of each other, which is reflected in the 6th and 15th lines of Algorithm 1.

B. Local density

We define as the degree of a node in graph G the local density of a sample. G is represented by the similarity matrix W as mentioned in Section III. Therefore, the local density of each sample can be obtained by

$$\rho_i = D_{ii} = \sum_{j=1}^{N} W_{ij}, \tag{16}$$

where W_{ij} denotes the similarity between samples x_i and x_j and is defined in Eq. (15). Only the similarity between the sample and its K-nearest neighbors is considered, which leads to the sparsity of W. In other words, it takes $O(NK)$ time to compute the local density for each sample.

Unlike other density-based clustering methods, we define sample x_i as a core point in a parameter-free way. Specifically we identify x_i as a core if

$$\rho_i \geq \rho_g, \text{or } \rho_i \geq \rho_l, \tag{17}$$

where ρ_g and ρ_l represent the median of the local density of all samples and the K-nearest neighbors of x_i, respectively. The samples that are not cores are identified as borders. Although two thresholds are involved in Eq. (17), the thresholds are automatically determined and can even be ignored by users.

C. Connection

(1) x_i is a border. Let x_i and x_j be connected, where x_j is the core closest to x_i among x_i's K-nearest neighbors, in terms of Euclidean distance.

(2) x_i is a core. Suppose $x_i^{(k)}$ is the border closest to x_i among x_i's K-nearest neighbors. Then, let x_i be connected to the sample $x_i^{(a)}, \forall a \in \{1, \cdots, k-1\}$. If such a point $x_i^{(k)}$ does not exist, then let x_i be connected to all samples in its K-nearest neighbors.

The clustering procedure is summarized in Algorithm 1, and here are some tips.

(1) Find the approximate K-nearest neighbors by line 1.

(2) Compute the similarity between x_i and $x_i^{(k)}$, for any $i = 1, \cdots N$, $k = 1, \cdots, K$, by lines 2-9.

(3) Compute the local density of each sample by line 10.

(4) Identify borders and cores by line 11.

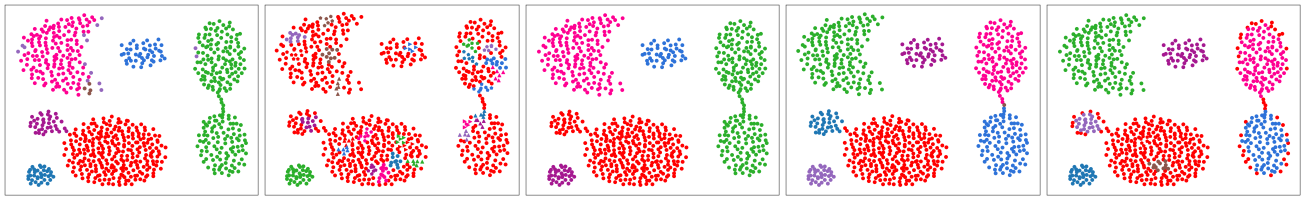
(5) Establish connections by specified rules by lines 12-23.

(6) Obtain the clustering results by line 24.

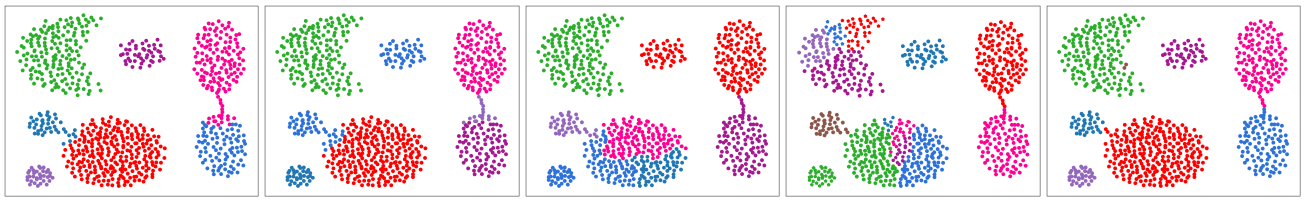
D. Time and space complexity

(1) The memory overhead is mainly occupied by the variables $I \in R^{N\times K}, D \in R^{N\times K}$, and the sparse similarity matrix W with at most NK non-zero elements. So the space complexity of EDG is $O(NK)$. (2) From Eq. (10)-(13), we can see that $O(K^2)$ time is needed to compute the local adjacency matrix $\tilde{W}$. However, we only consider the similarity on a subset of all possible pairs (the top-K nearest neighbors for each sample). Thus it needs $O(NK^3)$ time to compute the sparse matrix W. Compute the degree matrix of G require $O(NK)$ time at most, benefiting from the sparsity of W. Check connectivity for sample pairs considered in EDG require $O(NK)$ as shown in Algorithm 1.

In summary, the computational and memory overhead of EDG are $O(NK^3)$ and $O(NK)$, respectively, where N denotes the number of samples and K denotes the number of nearest neighbors. Both are linear with respect to the number of samples, which means EDG is easily scalable to large-scale problems. In addition, our algorithm can be operated in parallel, which can be seen from Algorithm 1.

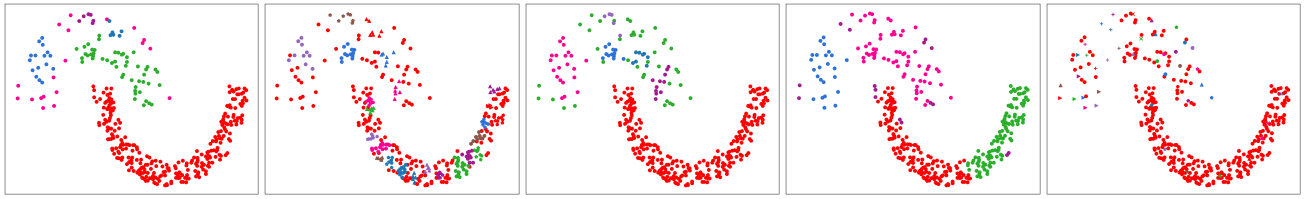


(a) DBSCAN: 0.917 (b) OPTICS: 0.635 (c) DB-H: 0.906 (d) DB-RNN: 0.999 (e) DP: 0.923

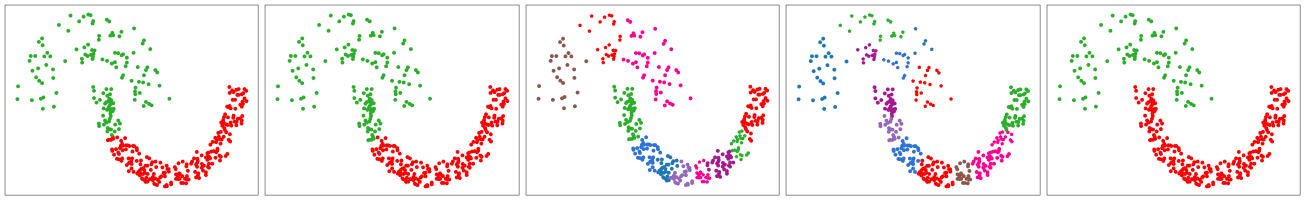


(f) SNNBC: 0.970 (g) AGCI: 0.947 (h) FIN: 0.845 (i) RCC: 0.820 (j) EDG: 0.999

Fig. 3. F_1 score on Aggregation dataset



(a) DBSCAN: 0.933 (b) OPTICS: 0.618 (c) DB-H: 0.917 (d) DB-RNN: 0.745 (e) DP: 0.838



(f) SNNBC: 0.877 (g) AGCI: 0.863 (h) FIN: 0.423 (i) RCC: 0.357 (j) EDG: 1.000

Fig. 4. F_1 score on Jain dataset

V. Experiments

In order to assess the performance of the proposed method (Algorithm 1), extensive experiments have been conducted which are performed on 25 benchmark datasets and 4 synthetic datasets from the literature, based on a two-stage framework. The rest of this section is structured as follows. The datasets used in this paper are introduced firstly. The description of the comparison algorithms and the experimental settings are then represented. An introduction to the two-stage framework is given. Finally, the clustering performance of all algorithms and analysis are shown.

A. Datasets

a) Synthetic: Aggregation, Gestalt, Jain, and Three-circle used in this section are artificial datasets that have been widely studied [6], [10]. These synthetic datasets are available in supplementary materials

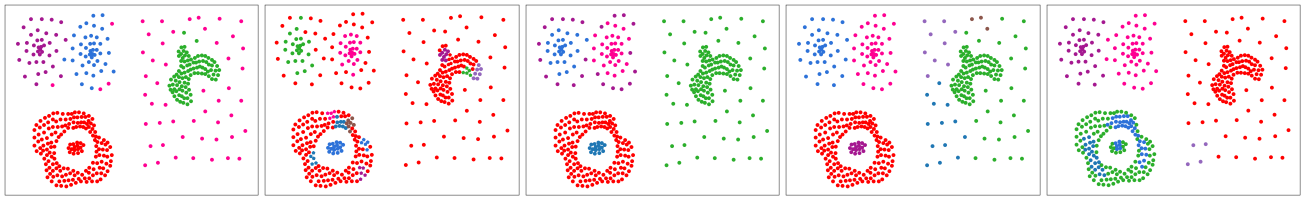
b) Middle-scale datasets: **CFPW** and **Sheffield** have collected face images of celebrities in frontal and profile views. **CNBC**, **Grimace**, **AR**, **JAFFE**, and **RaFD** consisting of face images with different facial expressions are constructed to promote the research of individual facial expressions detection.

Adience [38] is a public dataset aiming at the problem of age and gender recognition, **Face-94**, **Pixraw10P**, and **FERET** [39] were constructed to promote the research of facial recognition system evaluation.

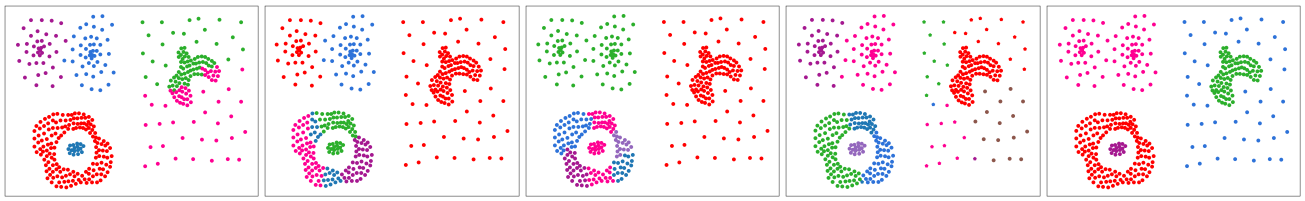
LFW [40] is a public benchmark dataset aiming at the problem of unconstrained face recognition. These images are collected from the web where these faces were detected by the Viola-Jones face detector. **CALFW** [41] and **CPLFW** [42] are two variants of LFW aiming at cross-age and cross-pose face recognition, respectively.

Some non-facial datasets including **Finger**, **MPEG-7**, **MSRA25**, **Palm**, and **GTDB**, are also used to verify our algorithm since our algorithm is not limited to face data. Among them, MPEG-7 and MSRA25 are data sets widely used to evaluate the saliency detection algorithm. Palm and Finger consist of pictures of the palm and fingerprint respectively.

c) Large-scale datasets: **WebFace** [43] and **CACD** [44]

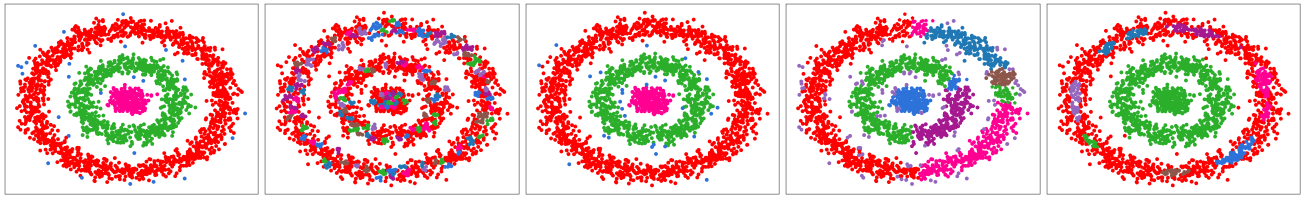


(a) DBSCAN: 0.964 (b) OPTICS: 0.681 (c) DB-H: 0.892 (d) DB-RNN: 0.935 (e) DP: 0.838

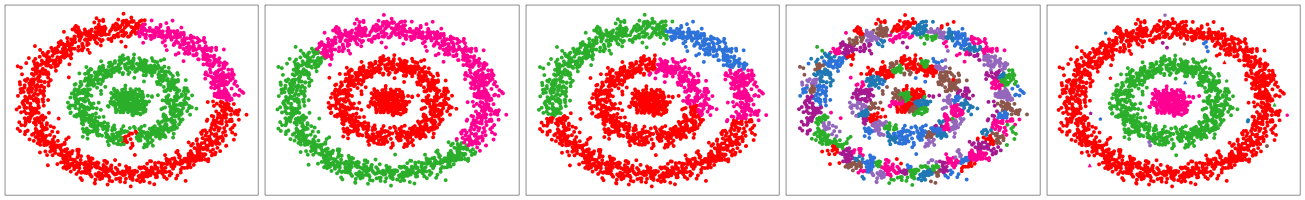


(f) SNNBC: 0.870 (g) AGCI: 0.736 (h) FIN: 0.729 (i) RCC: 0.835 (j) EDG: 0.950

Fig. 5. F_1 score on Gestalt dataset



(a) DBSCAN: 0.993 (b) OPTICS: 0.637 (c) DB-H: 0.993 (d) DB-RNN: 0.711 (e) DP: 0.858



(f) SNNBC: 0.872 (g) AGCI: 0.793 (h) FIN: 0.594 (i) RCC: 0.106 (j) EDG: 0.995

Fig. 6. F_1 score on Three-circle dataset

are two widely used facial datasets for face verification and cross-age face retrieval problems. **CelebA** [45] is a face attributes dataset. The images in this dataset are annotated with 40 attributes and cover large pose variations and background clutter. **IMDB** [46] is a face image dataset of celebrities containing 460,723 images of 20284 celebrities from IMDb. **YouTubeFace** [47] a database of face videos, consists of 621,126 face images of 1595 subjects downloaded from YouTube and is designed for studying the problem of unconstrained face recognition. The cropped version of IMDB and YouTubeFace are used. The statistics of these datasets are summarized in Tab. I.

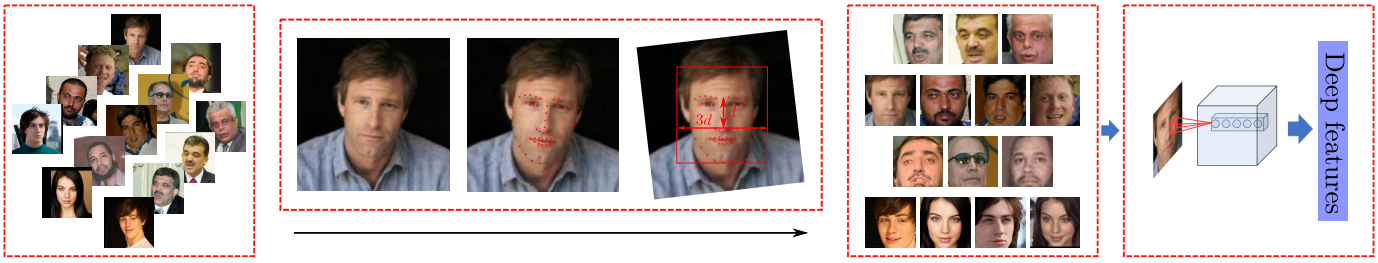
TABLE I
DESCRIPTION OF LARGE-SCALE DATASETS

Datasets	CACD	CelebA	WebFace	IMDB	YouTubeFace
# Samples	163446	202599	494414	460723	621126
# Subjects	2000	10177	10575	20284	1595

B. Clustering result on synthetic datasets

In this subsection, the data is directly used as the input of the algorithm without any preprocessing. Although there are very few samples in those datasets, clustering these datasets is extremely challenging, and performing clustering methods on them can provide a good qualitative analysis of performance. The baselines and experimental settings are given in V-C.

From the results shown in Fig. 3, 4, 5, and 6, it is not difficult to find that: (1) As expected, K-means, and FIN failed to capture the correct clustering structure, since the structure of means obtained by averaging the samples in each cluster is very different from the structure of the original samples. (2) Although RCC has a relatively good performance on the Aggregation and Gestalt, it cannot find the correct clustering structure on Jain and Three-circle, which are more complex in distribution. (3) Our model (EDG) and DBSCAN have achieved very good performance on these synthetic datasets. Unfortunately, users need to fine-tune the parameters $MinPts$ and ε involved in DBSCAN to achieve such performance, which is difficult on the benchmark datasets.



(a) Original (b) Preprocessing (c) Cropped (d) CNN

Fig. 7. Flow chart of deep clustering of face representation: (a) The original facial images; (b) The image processing techniques employed (rotate and crop); (c) Features extracted by a deep convolutional neural network.

C. Baselines and experimental settings

We compare our model with several state-of-the-art clustering algorithms (both recent and classic proposals) including clustering using First Integer Neighbor relations (FIN) [6], Shared-Nearest-Neighbor-Based Clustering (SNNBC) [11], clustering by fast search and find of Density Peaks (DP) [9], K-means (KM) [48], traditional Spectral Clustering (SC) [12], Improved Anchor-based Graph Clustering (I-AGC) [19], Robust Continuous Clustering (RCC) [49], DBSCAN [8], Density-Based clustering based on Hierarchical density estimates (DB-H) [25], Ordering Points to Identify the Clustering Structure (OPTICS) [50], and Density-Based clustering using Reverse Nearest Neighbor (DB-RNN) [10].

Parameters involved in these methods are described below. The adjacency matrix of SC and I-AGC are constructed by the heat kernel and method proposed in [51], respectively. In I-AGC, the anchors are generated in a random way and the number of anchors m is set as $m = min(\frac{n}{2}, 1024)$, where n denotes the number of samples. In SC, I-AGC, DBRNN, SNNBC, and EDG, the number of nearest neighbors K is tuned by a grid-search strategy from $\{5, 6, 7, 8, 9, 10\}$. KM is initialized in a random way and the steps of K-means involved in SC and I-AGC adopt the same configuration as it.

The parameter $MinPts$ involved in DBSCAN and OPTICS is fixed at 5 and ε involved in DBSCAN is tuned from $\{\varepsilon_1, \cdots, \varepsilon_{20}\}$ that is obtained by uniformly sampling in interval $[\varepsilon_{min}, \varepsilon_{max}]$, where ε_{min} and ε_{max} represent the minimum and maximum of set $\{\|x_i - x_i^{(5)}\|_2\}$, respectively.

The parameter ρ, δ involved in DP are tuned from evenly spaced values sampling from interval $[0, max_t]$, where max_t denotes the maximum value that ρ or δ can take.

In DB-H, we tune the parameter $MinClus$ from $\{5, 10, 15, 20, 25\}$. In RCC, we fix the number of nearest neighbors K at 10 but tune the parameter t used to assign points together in a cluster. Since the appropriate value of t is very difficult to obtain, we simply tune it by a grid-search strategy from $\{0.1, 0.2, \cdots, 1.0\}$. In the hierarchical clustering method, FIN, all clustering results it outputs are used to verify its performance. In SNNBC, we set the number of clusters to the ground truth class number in the data. The best results in terms of F_1 score are reported.

We compare EDG with all baselines on the middle-scale datasets. Only FIN and K-means are compared with our model on large-scale datasets.

D. Face image processing

Clustering unconstrained Faces is one of the important research topics in computer vision research and many other fields. In generally, given a set of input patterns, the purpose of clustering is to group the data into a certain number of clusters so that the samples in the same cluster are similar to each other, and the samples in different clusters are not.

However, to achieve such a purpose is very difficult for unconstrained face images, since those samples have extreme intra-class variations caused by nuisance factors such as illumination of venue, pose, and expressions of the subject. Therefore, these traditional clustering methods may not be applicable directly.

To this end, recent studies employ a two-stage clustering framework [52], [53], [6] to alleviate the intra-class variations of unconstrained facial images.

a) Face representation: Some facial image processing techniques are employed firstly, and then the processed images are used as the input of the trained deep neural network to extract features.

b) Clustering: The features extracted in the first stage instead of the images are directly used as the input of the clustering algorithm to obtain the clustering results.

The framework is easily scalable to large-scale problems since there is no need to train deep neural networks repeatedly. Therefore, we employ this framework to verify the performance of the proposed algorithm. For a fair comparison, all comparison algorithms share the same deep features as input.

c) Face image preprocessing: As shown in Fig. 7, given a collection of unconstrained face images, 68 facial landmarks are detected using the method proposed in [54] firstly for each face. Then we rotate and crop these images based on these landmarks so that 20-th and 25-th facial landmarks are located on the same horizontal line, and cut out a square with a length of $3d$ centered on 31-th landmark (d is the vertical distance from points 31 to 25). All cropped images are converted to gray-scale images and normalized to 128×128. Finally, a deep convolutional neural network [55] is used with the normalized images as input to learn the feature vectors. Preprocessing plays an important role in the pipeline, which eliminates the variations caused by posture and background. Therefore, the trained CNN model can be used on different datasets, meaning we do **not need** to train a deep model for each data.

TABLE II
PERFORMANCE ON 20 BENCHMARK DATASETS (MIDDLE-SCALE)

Datasets	Metrics	KM	SC	I-AGC	DBSCAN	OPTICS	DB-H	DB-RNN	DP	SNNBC	RCC	FIN	EDG
AR	Precision	0.135	0.151	0.143	0.049	0.042	0.020	0.032	0.099	0.122	0.020	0.171	**0.356**
	F_1 score	0.160	0.184	0.179	0.092	0.081	0.039	0.059	0.179	0.165	0.038	0.117	**0.282**
	FMI	0.052	0.068	0.049	0.080	0.095	0.070	0.050	**0.090**	0.043	0.035	0.062	0.045
Face-94	Precision	0.786	0.636	0.654	0.959	0.824	0.981	0.910	0.333	0.902	**0.998**	0.554	0.990
	F_1 score	0.840	0.715	0.756	0.944	0.712	0.937	0.899	0.495	0.924	0.956	0.456	**0.962**
	FMI	0.731	0.400	0.223	0.877	0.239	0.899	0.611	0.118	0.835	**0.944**	0.445	0.937
Grimace	Precision	0.792	0.668	0.726	0.942	0.814	0.992	0.894	0.925	0.944	1.000	0.708	**1.000**
	F_1 score	0.844	0.735	0.806	0.963	0.693	0.963	0.885	0.952	0.966	0.936	0.588	**0.967**
	FMI	0.739	0.583	0.576	0.933	0.426	0.946	0.774	0.915	0.936	0.910	0.582	**0.951**
JAFFE	Precision	0.766	0.846	0.698	0.906	0.746	0.887	0.864	0.761	0.915	0.577	0.637	**0.995**
	F_1 score	0.807	0.886	0.781	0.843	0.704	0.843	0.814	0.831	0.915	0.723	0.532	**0.925**
	FMI	0.694	0.816	0.645	0.767	0.454	0.763	0.704	0.752	0.864	0.535	0.506	**0.877**
MSRA-25	Precision	0.547	0.580	0.525	0.885	0.379	0.921	0.933	0.251	0.485	0.709	0.723	**1.000**
	F_1 score	0.562	0.610	0.582	0.674	0.491	0.677	0.675	0.361	0.529	0.638	0.394	**0.703**
	FMI	0.426	0.484	0.400	0.573	0.213	0.595	0.617	0.282	0.329	0.555	0.352	**0.673**
Palm	Precision	0.744	0.705	0.633	0.853	0.872	0.950	0.890	0.032	0.792	0.330	0.531	**0.962**
	F_1 score	0.765	0.777	0.722	0.863	0.668	0.838	0.872	0.062	0.827	0.491	0.436	**0.883**
	FMI	0.652	0.495	0.253	0.593	0.303	0.704	0.795	0.098	0.677	0.389	0.404	**0.838**
Pixraw10P	Precision	0.778	0.846	0.772	0.830	0.920	0.940	0.830	**0.990**	0.940	0.300	0.933	0.980
	F_1 score	0.806	0.877	0.804	0.877	0.920	0.925	0.873	0.937	0.940	0.462	0.785	**0.960**
	FMI	0.686	0.797	0.673	0.729	0.853	0.883	0.774	0.913	0.899	0.447	0.689	**0.932**
Sheffield	Precision	0.494	0.761	0.570	0.741	0.579	0.767	0.751	0.311	0.657	0.412	0.547	**0.807**
	F_1 score	0.497	0.792	0.614	0.716	0.555	0.720	0.704	0.448	0.722	0.580	0.401	**0.844**
	FMI	0.360	0.668	0.380	0.510	0.191	0.517	0.509	0.213	0.535	0.528	0.348	**0.732**
Finger	Precision	0.367	0.545	0.408	0.137	0.095	0.119	0.065	0.625	0.167	0.048	-	**0.827**
	F_1 score	0.455	0.553	0.490	0.240	0.174	0.212	0.123	0.541	0.280	0.091	-	**0.559**
	FMI	0.226	**0.348**	0.266	0.218	0.215	0.212	0.199	0.161	0.180	0.205	-	0.243
GTDB	Precision	0.483	0.586	0.508	0.135	0.159	0.072	0.173	0.288	0.453	0.020	-	**0.697**
	F_1 score	0.513	0.604	0.570	0.232	0.271	0.134	0.290	0.413	0.550	0.039	-	**0.646**
	FMI	0.325	**0.428**	0.284	0.158	0.143	0.136	0.152	0.111	0.231	0.137	-	0.340
Mpeg-7	Precision	0.494	0.550	0.484	0.377	0.364	0.458	0.359	0.191	0.064	**0.856**	0.482	0.825
	F_1 score	0.540	0.568	0.541	0.521	0.505	0.587	0.477	0.304	0.119	0.679	0.412	**0.686**
	FMI	0.325	0.383	0.271	0.163	0.137	0.168	0.162	0.107	0.112	**0.551**	0.223	0.491
Umist	Precision	0.480	0.709	0.549	0.678	0.577	0.751	0.730	0.268	0.673	0.143	-	**0.810**
	F_1 score	0.482	0.753	0.581	0.694	0.562	0.706	0.717	0.403	0.743	0.250	-	**0.853**
	FMI	0.346	0.606	0.394	0.452	0.200	0.433	0.513	0.209	0.584	0.300	-	**0.749**
Adience	Precision	0.790	0.791	0.641	0.286	0.254	0.436	0.327	0.163	0.688	0.781	0.496	**0.885**
	F_1 score	0.615	0.585	0.472	0.426	0.393	0.561	0.470	0.274	0.646	**0.817**	0.481	0.810
	FMI	0.253	0.170	0.182	0.096	0.066	0.063	0.083	0.061	0.101	**0.711**	0.085	0.289
CALFW	Precision	0.693	0.780	0.453	0.004	0.003	0.001	0.005	0.014	0.441	0.600	0.274	**0.938**
	F_1 score	0.720	0.797	0.455	0.008	0.007	0.002	0.009	0.027	0.558	0.724	0.342	**0.896**
	FMI	0.253	0.464	0.066	0.014	0.014	0.013	0.016	0.014	0.014	0.488	0.037	**0.752**
CFPW	Precision	0.581	0.621	0.523	0.492	0.545	0.580	0.648	0.476	0.482	0.817	0.461	**0.960**
	F_1 score	0.635	0.693	0.593	0.559	0.615	0.609	0.684	0.543	0.588	0.781	0.427	**0.822**
	FMI	0.255	0.379	0.073	0.056	0.056	0.068	0.085	0.053	0.054	0.568	0.067	**0.671**
FERET	Precision	0.622	0.603	0.515	0.443	0.401	0.496	0.395	0.701	0.610	0.697	0.402	**0.881**
	F_1 score	0.620	0.617	0.510	0.546	0.533	0.575	0.520	0.619	0.624	0.713	0.367	**0.776**
	FMI	**0.359**	0.300	0.231	0.058	0.041	0.059	0.051	0.056	0.063	0.091	0.088	0.308
LFW	Precision	0.771	0.764	0.665	0.252	0.239	0.343	0.180	0.066	0.592	0.596	0.334	**0.928**
	F_1 score	0.694	0.683	0.594	0.396	0.383	0.503	0.304	0.123	0.545	0.742	0.439	**0.927**
	FMI	0.093	0.121	0.085	0.060	0.064	0.058	0.068	0.054	0.011	**0.856**	0.037	0.698
RaFD	Precision	0.486	0.441	0.452	0.507	0.565	0.861	0.845	0.393	0.502	0.984	0.680	**0.986**
	F_1 score	0.532	0.509	0.520	0.552	0.500	**0.618**	0.580	0.508	0.546	0.449	0.316	0.560
	FMI	0.312	0.304	0.229	0.137	0.071	0.255	0.241	0.133	0.152	0.437	0.258	**0.496**
CPLFW	Precision	0.399	0.449	0.363	0.001	0.001	0.001	0.001	0.534	0.359	0.468	0.029	**0.671**
	F_1 score	0.477	0.452	0.439	0.001	0.002	0.002	0.002	0.552	0.404	0.506	0.050	**0.633**
	FMI	0.017	0.110	0.015	0.017	0.014	0.014	0.014	0.009	0.009	**0.020**	0.016	0.017
CNBC	Precision	0.577	0.550	0.495	0.525	0.452	0.572	0.570	0.683	0.544	0.718	0.498	**0.770**
	F_1 score	0.603	0.606	0.546	0.574	0.519	0.558	0.590	0.559	0.620	0.689	0.380	**0.697**
	FMI	0.326	**0.343**	0.238	0.086	0.059	0.087	0.089	0.069	0.092	0.139	0.156	0.246

TABLE III
CLUSTERING PERFORMANCE ON 5 LARGE-SCALE DATASETS

	Precision			F_1 score			FMI			Time(s)		
	KM	FIN	EDG	KM	FIN	EDG	KM	FIN	EDG	KM	FIN	EDG
CACD	0.648	0.579	**0.914**	0.696	0.435	**0.807**	0.291	0.077	**0.626**	59.8	52.6	**19.4**
CelebA	0.247	0.227	**0.622**	0.318	0.256	**0.559**	0.009	0.009	**0.034**	191.2	51.0	**16.6**
WebFace	0.508	0.384	**0.876**	0.506	0.348	**0.769**	0.108	0.019	**0.243**	4693.4	203.2	**29.9**
IMDB	0.346	0.202	**0.579**	0.317	0.162	**0.444**	0.073	0.020	**0.032**	2154.4	183.4	**21.4**
YouTubeFace	0.782	0.697	**0.994**	0.707	0.348	**0.751**	0.370	0.125	**0.710**	238.2	320.8	**78.4**

E. Clustering results on benchmark datasets

In Tab. II, the results represented by "-" denotes the outlier clustering results in which almost each sample is a cluster by itself, which is not the phenomenon we want to see. From the results shown in Tab. II and Tab. III, we can clearly see that: (1) EDG achieves the highest performance in most cases, compared to the other clustering methods showing the effectiveness of our model. Specifically, EDG achieves 18.5%, 11.0%, 9.9%, 9.8%, 8.1%, and 6.3% gain on LFW, Umist, CALFW, AR, CPLFW, and FERET over the second-best method, respectively, in terms of F_1 score. Similar results can be obtained with the metrics of precision and Fowlkes-Mallows index (FMI). (2) The performance of DBSCAN and DP are extremely unstable, which may be caused by the fact that the appropriate values of these parameters have not been selected. (3) The computational and memory overhead of EDG are both linear with respect to the number of samples. Thus, it can be easily run on large-scale data. From the experimental results shown in Tab. III, our algorithm still achieves promising performs on these large-scale datasets. And the running time of EDG is very short.

VI. CONCLUSIONS

How to cluster large-scale face datasets effectively is a challenge and receiving more and more attention. To address this problem, an Efficient Density-based clustering algorithm incorporated with the model of Graph partitioning called EDG is proposed. (1) Normalized-cut as an effective graph partitioning model verified by many studies is employed in our algorithm to evaluate the similarity between two samples. To the best of our knowledge, EDG is the first density-based algorithm incorporated with the graph partition model. (2) Only one user-specified parameter is involved in EDG, i.e., the number of nearest neighbors. Different from other density-based methods, the parameter is an integer and is easy to tune. (3) We only consider the similarities and connections on a subset of all possible pairs, i.e. the top-K nearest neighbors for each sample. Therefore, the computational and memory overhead of our algorithm are both linear with respect to the number of samples. Specifically, the computational and memory overhead are $O(NK^3)$ and $O(NK)$ respectively, and the algorithm can be operated in parallel, indicating that EDG is easily scalable and applicable to large practical problems. (4) Extensive experiments have been conducted on 4 synthetic datasets and 25 real-world benchmark datasets (20 middle-scale and 5 large-scale) to demonstrate the superior performance of our model.

REFERENCES

[1] R. Janani and S. Vijayarani, "Text document clustering using spectral clustering algorithm with particle swarm optimization," *Expert Systems with Applications*, vol. 134, pp. 192–200, 2019.

[2] R. Jain and R. S. Sharma, "Image segmentation through fuzzy clustering: A survey," in *Harmony Search and Nature Inspired Optimization Algorithms*. Springer, 2019, pp. 497–508.

[3] G. Le Ray and P. Pinson, "Online adaptive clustering algorithm for load profiling," *Sustainable Energy, Grids and Networks*, vol. 17, p. 100181, 2019.

[4] H.-W. Lee, N. Malik, F. Shi, and P. J. Mucha, "Social clustering in epidemic spread on coevolving networks," *Physical Review E*, vol. 99, no. 6, p. 062301, 2019.

[5] M. Xu, Y. Li, R. Li, F. Zou, and X. Gu, "Eadp: An extended adaptive density peaks clustering for overlapping community detection in social networks," *Neurocomputing*, vol. 337, pp. 287–302, 2019.

[6] S. Sarfraz, V. Sharma, and R. Stiefelhagen, "Efficient parameter-free clustering using first neighbor relations," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 8934–8943.

[7] W.-A. Lin, J.-C. Chen, and R. Chellappa, "A proximity-aware hierarchical clustering of faces," in *2017 12th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2017)*. IEEE, 2017, pp. 294–301.

[8] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*. AAAI Press, 1996, p. 226–231.

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

[10] A. Bryant and K. Cios, "Rnn-dbscan: A density-based clustering algorithm using reverse nearest neighbor density estimates," *IEEE Transactions on Knowledge and Data Engineering*, vol. 30, no. 6, pp. 1109–1121, 2018.

[11] R. Liu, H. Wang, and X. Yu, "Shared-nearest-neighbor-based clustering by fast search and find of density peaks," *Information Sciences*, vol. 450, pp. 200 – 226, 2018.

[12] A. Y. Ng, M. I. Jordan, and Y. Weiss, "On spectral clustering: Analysis and an algorithm," in *Advances in neural information processing systems*, 2002, pp. 849–856.

[13] J. Shi and J. Malik, "Normalized cuts and image segmentation," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 22, no. 8, pp. 888–905, 2000.

[14] C. Fowlkes, S. Belongie, F. Chung, and J. Malik, "Spectral grouping using the nystrom method," *IEEE transactions on pattern analysis and machine intelligence*, vol. 26, no. 2, pp. 214–225, 2004.

[15] M. Li, J. T.-Y. Kwok, and B. Lü, "Making large-scale nyström approximation possible," in *ICML 2010-Proceedings, 27th International Conference on Machine Learning*, 2010, p. 631.

[16] G. Zhong and C.-M. Pun, "Revisiting nyström extension for hypergraph clustering," *Neurocomputing*, 2020.

[17] R. Wang, F. Nie, Z. Wang, F. He, and X. Li, "Scalable graph-based clustering with nonnegative relaxation for large hyperspectral image," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 57, no. 10, pp. 7352–7364, 2019.

[18] N. Tremblay and A. Loukas, "Approximating spectral clustering via sampling: a review," in *Sampling Techniques for Supervised or Unsupervised Tasks*. Springer, 2020, pp. 129–183.

[19] Y. Zhao, Y. Yuan, and Q. Wang, "Fast spectral clustering for unsupervised hyperspectral image classification," *Remote Sensing*, vol. 11, no. 4, p. 399, 2019.

[20] D. L. Donoho and M. Elad, "Optimally sparse representation in general (nonorthogonal) dictionaries via l1 minimization," *Proceedings of the National Academy of Sciences*, vol. 100, no. 5, pp. 2197–2202, 2003.
[21] W.-W. Wang, X. Li, X.-C. Feng, and S.-Q. Wang, "A survey on sparse subspace clustering," *Acta automatica sinica*, vol. 41, no. 8, pp. 1373–1384, 2015.
[22] V. Vitelli, "A novel framework for joint sparse clustering and alignment of functional data," *arXiv preprint arXiv:1912.00687*, 2019.
[23] S.-S. Li, "An improved dbscan algorithm based on the neighbor similarity and fast nearest neighbor query," *IEEE Access*, vol. 8, pp. 47 468–47 476, 2020.
[24] D. Han, A. Agrawal, W.-k. Liao, and A. Choudhary, "A fast dbscan algorithm with spark implementation," in *Big Data in Engineering Applications*. Springer, 2018, pp. 173–192.
[25] R. J. G. B. Campello, D. Moulavi, and J. Sander, "Density-based clustering based on hierarchical density estimates," in *Advances in Knowledge Discovery and Data Mining*. Springer, 2013, pp. 160–172.
[26] A. Sharma and A. Sharma, "Knn-dbscan: Using k-nearest neighbor information for parameter-free density based clustering," in *2017 International Conference on Intelligent Computing, Instrumentation and Control Technologies (ICICICT)*, 2017, pp. 787–792.
[27] W. Lai, M. Zhou, F. Hu, K. Bian, and Q. Song, "A new dbscan parameters determination method based on improved mvo," *IEEE Access*, vol. 7, pp. 104 085–104 095, 2019.
[28] M. Hahsler, M. Piekenbrock, and D. Doran, "dbscan: Fast density-based clustering with r," *Journal of Statistical Software*, vol. 25, pp. 409–416, 2019.
[29] Y. Chen, X. Hu, W. Fan, L. Shen, Z. Zhang, X. Liu, J. Du, H. Li, Y. Chen, and H. Li, "Fast density peak clustering for large scale data based on knn," *Knowledge-Based Systems*, vol. 187, p. 104824, 2020.
[30] L. Bai, X. Cheng, J. Liang, H. Shen, and Y. Guo, "Fast density clustering strategies based on the k-means algorithm," *Pattern Recognition*, vol. 71, pp. 375–386, 2017.
[31] J. Xu, G. Wang, and W. Deng, "Denpehc: Density peak based efficient hierarchical clustering," *Information Sciences*, vol. 373, pp. 200 – 218, 2016.
[32] L. Yaohui, M. Zhengming, and Y. Fang, "Adaptive density peak clustering based on k-nearest neighbors with aggregating strategy," *Knowledge-Based Systems*, vol. 133, pp. 208 – 220, 2017.
[33] Z. Guo, T. Huang, Z. Cai, and W. Zhu, "A new local density for density peak clustering," in *Advances in Knowledge Discovery and Data Mining*. Springer International Publishing, 2018, pp. 426–438.
[34] B. Tu, X. Zhang, X. Kang, G. Zhang, and S. Li, "Density peak-based noisy label detection for hyperspectral image classification," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 57, no. 3, pp. 1573–1584, 2019.
[35] C. K. Reddy and B. Vinzamuri, "A survey of partitional and hierarchical clustering algorithms," in *Data Clustering*. Chapman and Hall/CRC, 2018, pp. 87–110.
[36] M. K. Gupta and P. Chandra, "A comparative study of clustering algorithms," in *2019 6th International Conference on Computing for Sustainable Global Development (INDIACom)*. IEEE, 2019, pp. 801–805.
[37] C. Fu and D. Cai, "Efanna: An extremely fast approximate nearest neighbor search algorithm based on knn graph," *arXiv preprint arXiv:1609.07228*, 2016.
[38] T. Hassner, S. Harel, E. Paz, and R. Enbar, "Effective face frontalization in unconstrained images," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 4295–4304.
[39] P. J. Phillips, H. Wechsler, J. Huang, and P. J. Rauss, "The feret database and evaluation procedure for face-recognition algorithms," *Image and vision computing*, vol. 16, no. 5, pp. 295–306, 1998.
[40] E. Learned-Miller, G. B. Huang, A. RoyChowdhury, H. Li, and G. Hua, "Labeled faces in the wild: A survey," in *Advances in face detection and facial image analysis*. Springer, 2016, pp. 189–248.
[41] T. Zheng, W. Deng, and J. Hu, "Cross-age LFW: A database for studying cross-age face recognition in unconstrained environments," *CoRR*, vol. abs/1708.08197, 2017.
[42] T. Zheng and W. Deng, "Cross-pose lfw: A database for studying cross-pose face recognition in unconstrained environments," Beijing University of Posts and Telecommunications, Tech. Rep. 18-01, February 2018.
[43] D. Yi, Z. Lei, S. Liao, and S. Z. Li, "Learning face representation from scratch," *arXiv preprint arXiv:1411.7923*, 2014.
[44] B.-C. Chen, C.-S. Chen, and W. H. Hsu, "Face recognition and retrieval using cross-age reference coding with cross-age celebrity dataset," *IEEE Transactions on Multimedia*, vol. 17, no. 6, pp. 804–815, 2015.
[45] Z. Liu, P. Luo, X. Wang, and X. Tang, "Deep learning face attributes in the wild," in *Proceedings of International Conference on Computer Vision (ICCV)*, 2015, pp. 3730–3738.
[46] R. Rothe, R. Timofte, and L. V. Gool, "Deep expectation of real and apparent age from a single image without facial landmarks," *International Journal of Computer Vision*, vol. 126, no. 2-4, pp. 144–157, 2018.
[47] L. Wolf, T. Hassner, and I. Maoz, "Face recognition in unconstrained videos with matched background similarity," in *CVPR 2011*, 2011, pp. 529–534.
[48] A. David, "Vassilvitskii s.: K-means++: The advantages of careful seeding," in *18th annual ACM-SIAM symposium on Discrete algorithms (SODA), New Orleans, Louisiana*, 2007, pp. 1027–1035.
[49] S. A. Shah and V. Koltun, "Robust continuous clustering," *Proceedings of the National Academy of Sciences*, vol. 114, no. 37, pp. 9814–9819, 2017.
[50] M. Ankerst, M. M. Breunig, H.-P. Kriegel, and J. Sander, "Optics: Ordering points to identify the clustering structure," in *Proceedings of the 1999 ACM SIGMOD International Conference on Management of Data*, ser. SIGMOD '99. New York, NY, USA: Association for Computing Machinery, 1999, p. 49–60.
[51] F. Nie, W. Zhu, and X. Li, "Unsupervised large graph embedding," in *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, ser. AAAI'17. AAAI Press, 2017, p. 2422–2428.
[52] C. Otto, D. Wang, and A. K. Jain, "Clustering millions of faces by identity," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, no. 2, pp. 289–303, 2018.
[53] W.-A. Lin, J.-C. Chen, C. D. Castillo, and R. Chellappa, "Deep density clustering of unconstrained faces," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 8128–8137.
[54] V. Kazemi and J. Sullivan, "One millisecond face alignment with an ensemble of regression trees," *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1867–1874, 2014.
[55] X. Wu, R. He, Z. Sun, and T. Tan, "A light cnn for deep face representation with noisy labels," *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 11, pp. 2884–2896, 2018.

Shenfei Pei is now working toward his PhD degree in Northwestern Polytechnical University, Xi'an, China. His research interests include machine learning and its application fields, such as data mining, computer vision and Clustering.

Feiping Nie is full professor in Northwestern Polytechnical University, China. His research interests are machine learning and its applications. He has published more than 100 papers in the following journals and conferences: TPAMI, IJCV, TIP, ICML, NIPS, ICCV, CVPR, ACM MM. He is now serving as Associate Editor or PC member for several prestigious journals and conferences in the related fields.

Rong Wang is currently an associate professor at the School of Cybersecurity and Center for OPTical IMagery Analysis and Learning (OPTIMAL), Northwestern Polytechnical University, China. His research interests focus on machine learning and its applications.

Xuelong Li (Fellow, IEEE) is a Full Professor with the Center for OPTical IMagery Analysis and Learning (OPTIMAL), Northwestern Polytechnical University, Xi'an, China