IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received September 30, 2017, accepted November 3, 2017, date of publication November 8, 2017,
date of current version December 22, 2017.

Digital Object Identifier 10.1109/ACCESS.2017.2771365

A Data-Driven Clustering Approach for Fault Diagnosis

JIAN HOU, (Member, IEEE), AND BING XIAO
College of Engineering, Bohai Universit y, Jinzhou 121013, China

Corresponding author: Jian Hou (dr.houjian@gmail.com)

This work was supported in part by the National Natural Science Foundation of China under Grant 61473045 and in part by the Natural Science Foundation of Liaoning Province under Grant 20170540013.

ABSTRACT Clustering is an important approach in fault diagnosis. The dominant sets algorithm is a graph-based clustering algorithm, which defines the dominant set as a concept of a cluster. In this paper, we make an in-depth investigation of the dominant sets algorithm. As a result, we find that this algorithm is dependent on the similarity parameter in constructing the pairwise similarity matrix, and has the tendency to generate spherical clusters only. Based on the merits and drawbacks of this algorithm, we apply the histogram equalization transformation to the similarity matrices for the purpose of removing the influence of similarity parameters, and then use a density-based cluster expansion process to improve the clustering results. In experimental validation of the proposed algorithm, we use two criterions to evaluate the clustering results in order to arrive at convincing conclusions. Data clustering experiments on ten data sets and fault detection experiments on the Tennessee Eastman process demonstrate the effectiveness of the proposed algorithm.

INDEX TERMS Clustering, fault diagnosis, dominant set, cluster expansion.

I. INTRODUCTION

Data clustering refers to the process of grouping data into clusters based on their distance or similarity, so that the data in the same cluster are similar, and they are less similar to those in other clusters. As the clusters usually reflect the implicit pattern in a set of data, clustering is widely used in pattern recognition, fault diagnosis, data mining, image analysis and machine learning [1]–[8]. Some of the most commonly used clustering algorithms include K-means, normalized cuts (NCuts) [9] and DBSCAN (Density-Based Spatial Clustering of Applications with Noise) [10]. In recent developments, some algorithms have been presented from different focuses, including the affinity propagation algorithm [11], spectral clustering [12], dominant sets [13], density peak based algorithm [14] and others [15], [16].

Existing algorithms usually face some challenges in accomplishing real-world clustering tasks. The well-known K-means algorithm needs to be fed the number of clusters. In addition, its clustering results are influenced by the initial cluster centers which are usually selected randomly, and it can only generate spherical clusters in general. As a typical spectral clustering approach, the NCuts algorithm also requires to specify the number of clusters and generates clusters of spherical shapes only. Although DBSCAN is able to extract clusters of arbitrary shapes and obtain the number of clusters automatically, it involves a neighborhood radius and the minimum cluster size as the density parameters. In other works, the affinity propagation (AP) algorithm [11] requires as input the preference values of all the data to be clustered, and the density peak based algorithm (DP) [14] involves the cutoff distance and possibly human selection of cluster centers. In summary, the majority of existing clustering algorithms either demand prior knowledge of the dataset for input parameters determination, or are only suitable for datasets with spherical clusters. In the case that the conditions are not satisfied, the performance of these algorithms usually degrades significantly. Some works have been published to solve these problems, e.g., [17], [18] in estimating the number of clusters. However, the problems of parameter estimation and limited cluster shapes are still open in general.

The dominant sets (DSets) algorithm defines dominant set as a non-parametric concept of a cluster, and generates the clusters sequentially. With the pairwise data similarity matrix as the input, the DSets algorithm regards a dominant set as a cluster, and generates the clusters in a sequential fashion. This algorithm requires the pairwise data similarity matrix as the input, instead of the commonly used number of clusters. In addition, it can be used to generate overlapping clusters in a game-theoretic framework [19], [20]. Based on the nice properties, the DSets algorithm has been applied successfully in image segmentation [13], [21], human activity analysis [22], object detection [23] and object classification [24], [25], etc. Some closely related works to the DSets algorithm include [26]–[31].

The DSets algorithm uses as the input the pairwise data similarity matrix, thereby eliminating the requirement of data to be represented as vectors of features. However, in the case that the data for clustering are represented in the form of vectors, we need to calculate the pairwise similarity matrix with the data vectors. Usually we can estimate the similarity between two data x and y by $s(x, y) = exp(-d(x, y)/\sigma)$, with $d(x, y)$ denoting the distance between data vectors and σ as the similarity parameter. Here we see that the variances of σ's lead to the changes of similarity matrices, which are found to result in different clustering results. In this paper we present an approach to solve this problem. Our contributions are as follows. First, we investigate how σ's impact on the clustering results, and apply histogram equalization transformation to similarity matrices for the purpose of eliminating the influences of σ's. Second, we study the reason behind the observation that histogram equalization results in small clusters, and present a simple density based cluster expansion method to solve the problem and improve the clustering results. Third, we demonstrate the effectiveness of the proposed algorithm with data clustering and fault detection experiments.

The remainder of this paper is structured as follows. We firstly present a brief introduction of the dominant set definition and the DSets algorithm in Section II, based on which we investigate the problems of the DSets algorithm and present our solution in Section III. Data clustering and fault detection experiments and comparison with other algorithms are reported in Section IV. The concluding remarks are given in V.

II. DOMINANT SET

In this section we firstly present a brief introduction of the definition of dominant set, based on which the DSets algorithm and its properties are derived. For details we refer the reader to [13].

As aforementioned, existing algorithms usually rely on user-specified parameters to accomplish the clustering process. The K-means-like algorithms partition the data with the given number of clusters, and treat each part as a cluster. In contrast, DBSCAN uses the specified density parameters to determine the cluster borders, and then extracts clusters one by one from the set of data. Different from these two kinds of approaches, [13] presented dominant set as a non-parametric concept of a cluster. Informally, a dominant set is defined as a maximal subset with internal coherency. This definition means that the data in a dominant set are similar to each other, and they are less similar to those outside the dominant set. This property qualifies a dominant set as a cluster. Similar to DBSCAN, by extracting the dominant sets one by one, we can obtain all the clusters and determine the number of clusters automatically from the clustering process.

In order to define dominant set formally, we firstly use an edge-weighted graph $G = (V, E, w)$ to represent the n data for clustering, where V, E and w denote the data set, the edge set and the edge weight function, respectively. With $A = (a_{ij})$ representing the pairwise similarity matrix, it is evident that $w_{ij} = a(i, j)$ if $(i, j) \in E$ and $w_{i,j} = 0$ otherwise. In our clustering application one data should not be similar to itself, and therefore the similarity values on the main diagonal are all set to zero.

As a dominant set is a maximal subset of data with internal coherency, we can regard the dominant set extraction as a process to maximize the cluster size on condition that the internal coherency is preserved. For this purpose, [13] presented a criterion to evaluate if a subset of data is internal coherent, and proposed to include into a dominant set only the data does not destroy the internal coherency. Specifically, for $i \in S, j \notin S$ and $S \subseteq V$, we firstly define

$$aws_S(i) = \frac{1}{|S|} \sum_{j \in S} a_{ij}, \tag{1}$$

and

$$\phi_S(i, j) = a_{ij} - aws_S(i). \tag{2}$$

Obviously, $aws_S(i)$ measures the average similarity between i and all the data in S, and $\phi_S(i, j)$ measures the relationship of two similarities, namely the similarity of i and j, and the average similarity of i with the data in S. We then define the key variable in the dominant set definition as

$$w_S(i) = \begin{cases} 1, & \text{if} \quad |S| = 1, \\ \sum_{j \in S \setminus \{i\}} \phi_{S \setminus \{i\}}(j, i) w_{S \setminus \{i\}}(j), & \text{otherwise.} \end{cases} \tag{3}$$

Since $w_S(i)$ is defined in a recursive form, its meaning is not straightforward. From Eq. (3) we see that $w_S(i)$ can be roughly regarded as a weighted sum of $\phi_{S \setminus \{i\}}(j, i)$. Considering the definition of $\phi_S(i, j)$ in Eq. (2), we see that $w_S(i)$ actually reflects the relationship of two similarities, i.e., the average similarity between i and the data in $S \setminus \{i\}$, and the overall similarity in $S \setminus \{i\}$. Now we see that $w_S(i) > 0$ means that the former similarity is larger than the latter one, and including i is able to preserve the internal coherency in $S \setminus \{i\}$. In contrast, a negative $w_S(i)$ indicates that including i into $S \setminus \{i\}$ will destroy the internal coherency in $S \setminus \{i\}$.

Now we are ready to define dominant set in a formal way. With $W(S) = \sum_{i \in S} w_S(i)$, a subset S such that $W(T) > 0$ for all non-empty $T \subseteq S$ is called a dominant set if

1) $w_S(i) > 0$, for all $i \in S$.
2) $w_{S \bigcup \{i\}}(i) < 0$, for all $i \notin S$.

In the definition, the first condition indicates that all the data in the subset are able to preserve the internal coherency, and the second one states that one data will be rejected if it destroys the internal coherency. These two conditions together shape a dominant set as a maximal subset of data with internal coherency. This further means that the data in a dominant set have high similarities with each other and low similarities with those outside, and qualifies a dominant set as a cluster.

Pavan and Pelillo [13] proposed to extract a dominant set with the replicator dynamics used in evolutionary game theory. With $x \in R^n$ denoting the weights of all the n data, we calculate the final weights of the data by

$$x_k^{(t+1)} = x_k^{(t)} \frac{(Ax^{(t)})_k}{x^{(t)T}Ax^{(t)}}, \tag{4}$$

where $k = 1, \ldots, n$ and t denotes the iteration number. After the iteration converges, the data with weights above a threshold form a dominant set. In [32] the so-called infection and immunization dynamics are presented, where

$$x^{(t+1)} = \theta_{F(x^{(t)})}(x^{(t)})[F(x^{(t)}) - x^{(t)}] + x^{(t)}. \tag{5}$$

In this dynamics, F is a function used to search the most infective strategy y for x, and θ represents the minimum portion of y to make the new population $(1-\theta)x + \theta y$ immune to y. The details of these denotations can be found in [32] and are skipped here for space reason. Compared with the replicator dynamics, the infection and immunization dynamics involve no weight thresholds and the data with non-zero weights form a dominant set. In this paper we adopt the infection and immunization dynamics.

With the method to extract a dominant set, the DSets clustering can be accomplished with the same sequential manner as DBSCAN, and the number of clusters can be determined in the clustering process.

III. THE PROPOSED ALGORITHM

Since the pairwise similarity matrix is the sole input of the DSets algorithm, if the data to be clustered are given in the form of the pairwise similarity matrix, the DSets algorithm is parameter independent. However, in many tasks, the data are represented in the form of vectors of features. In this case, we usually use $s(x, y) = exp(-d(x, y)/\sigma)$ to calculate the similarity between x and y, and the similarity parameter σ is involved. With a given set of data, the variance of σ's leads to the change of similarity matrices, which are then found to result in different clustering results. One example is shown in Figure 1, where the DSets clustering results on the Jain dataset [33] with different σ's are reported, and $\overline{d}$ is the average of the pairwise Euclidean distances. We observe from Figure 1 that with a small σ, e.g., $\sigma = 0.5\overline{d}$, we obtain many small clusters. With the increase of σ, the clusters become larger and larger, and the number of clusters decreases dramatically. This means that σ has a significant influence on DSets clustering results.

We then apply the DSets algorithm to the other nine datasets, including Aggregation [34], Compound [35], Pathbased [36], R15 [37], Flame [38] and UCI datasets Thyroid, Wine, Iris and Breast. The brief description of all the ten datasets is in Table 1. We experiment with $\sigma = \alpha\overline{d}$, where $\alpha = 0.1, 0.2, 0.5, 1, 2, 5$ and 10 to 100 with the step of 10, and show the clustering results in Figure 2. Here we use F-measure and Jaccard index to evaluate the clustering results.

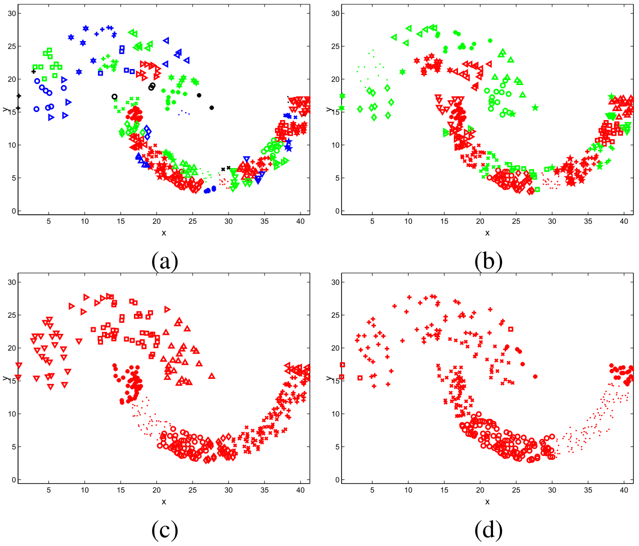


FIGURE 1. The clustering results of the DSets algorithm on the Jain dataset, obtained with different σ's. (a) $\sigma = 0.5\overline{d}$. (b) $\sigma = \overline{d}$. (c) $\sigma = 5\overline{d}$. (d) $\sigma = 20\overline{d}$.

TABLE 1. The characteristics of datasets used in experiments.

	# of points	Data dimension	# of clusters
Aggregation	788	2	7
Compound	399	2	6
Pathbased	300	2	3
R15	600	2	15
Jain	373	2	2
Flame	240	2	2
Thyroid	215	5	2
Wine	178	13	3
Iris	150	4	3
Breast	699	9	2

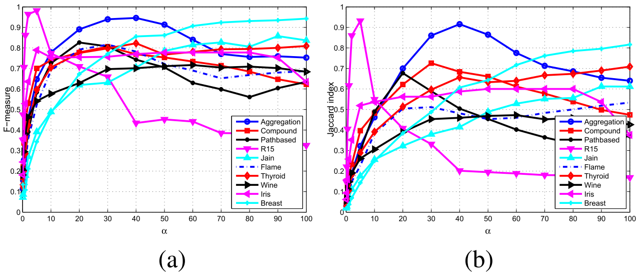


FIGURE 2. The clustering results of the DSets algorithm on ten datasets. (a) F-measure. (b) Jaccard index.

We observe from Figure 2 that on all the ten datasets, the clustering results of the DSets algorithm vary significantly with the variance of σ. In general, we find that both small and large σ's result in the decrease of the clustering quality, and the best results are obtained at limited σ's. However, we find that the best-performing σ's vary widely on the ten datasets, from the smallest $5\overline{d}$ on R15 to the largest $250\overline{d}$ on Breast, and there isn't a fixed σ suitable for different datasets. On one hand, there is still no method available to derive the best-performing σ for a given dataset. On the other hand, we notice that on some datasets, even the best-performing σ's generate unsatisfactory clustering results. This observation implies that it may not be a good option to explore a best-performing σ determination method. Therefore we need to explore a different approach to solve the parameter dependence problem.

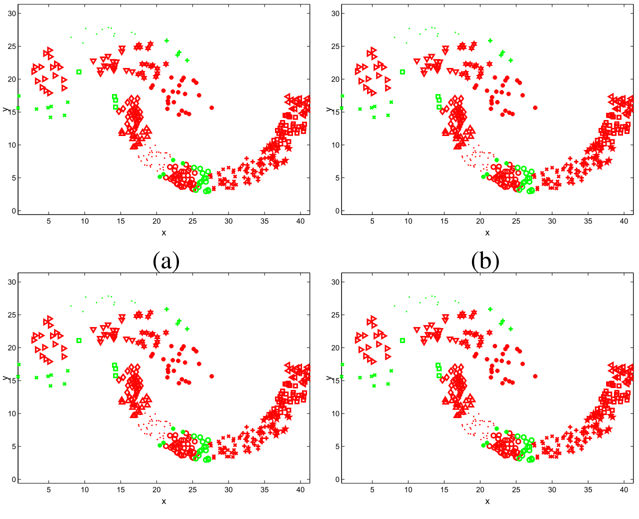
A. HISTOGRAM EQUALIZATION

In order to eliminate the influence of the similarity parameter on the DSets clustering results, we firstly analyze the reason behind the influence. By definition a dominant set includes all data helpful to preserve the internal coherency and excludes all those not helpful for this purpose. In other words, the extraction of a dominant set is a process of maximizing the dominant set size on condition that the internal coherency is preserved. This definition has the following consequences. First, the dominant set definition pays attention only to the high internal similarity in a dominant set, and ignores the importance of low external similarity required by clustering. This leads the DSets clustering results to be very sensitive to the absolute magnitude of similarity values. Specifically, a large σ increases the similarity values and reduces the difference among large similarity values. In this case, more data is likely to be included into a dominant set and we obtain large clusters with the DSets algorithm. In contrast, with a small σ, the DSets algorithm tends to generate small clusters. Since over-small and over-large clusters deviate from the ground truth and degrade the clustering results, we see both very large and very small σ's result in the decrease of clustering quality, as illustrated in Figure 2. Second, the dominant set definition requires each included data to be able to preserve the internal coherency. This strict constraint implies that in a dominant set, each data has to be very similar to all the others in the dominant set. As a result, the obtained dominant sets can only be of spherical shapes and the DSets algorithm is not capably of generating clusters of non-spherical shapes. This explains the observation in Figure 2 that on some datasets, even the best-performing σ's generate unsatisfactory results.

In this paper we eliminate the influence of σ by transforming the similarity matrices with histogram equalization before clustering. Histogram equalization is a popular image enhancement technique used to increase the intensity contrast in an image [39]. This technique transforms the intensity values of image pixels based on the intensity histogram, and after transformation the intensity histogram becomes more flat than the original one. Given an image, we quantize the range of intensity values into N bins and construct the intensity histogram $H = \{h_k\}, k = 1, \cdots, N$, with h_k denoting the amount of pixels falling in the k-th bin. With histogram equalization transformation, the pixels in the k-th bin are assigned a new intensity value as

$$g_k = L \sum_{j=1}^{k} \frac{h_j}{n}, \tag{6}$$

where L is the maximum intensity value and n is the total number of pixels in the image. In transforming the similarity matrices with histogram equalization, the similarity values are used to build the histogram and transformed to new values. As in our application the similarity is evaluated by $s(x, y) = exp(-d(x, y)/\sigma)$, the maximum similarity value is 1. Therefore the similarity values in the k-th bin are assigned the new value as $\sum_{j=1}^{k} \frac{h_j}{n}$. With histogram equalization transformation, the similarity elements in one bin are assigned the same new value, which is determined only by the amount of similarity elements in that bin and in the bins with smaller similarity values. If the similarity range [0,1] are quantized into a sufficient large number of bins such that at most one similarity element exists in each bin, we see that the new value of a similarity element is influenced only by the amount of smaller similarity values. This means that after the transformation by histogram equalization, the new similarity matrices are determined only by the sorting of the similarity values.



(a)

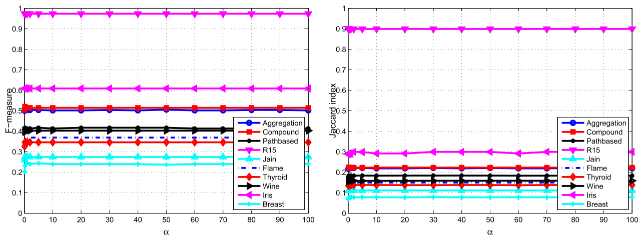
(b)

(c)

(d)

FIGURE 3. The DSets-histeq clustering results on the Jain dataset with different σ. (a) $\sigma = 0.5\overline{d}$. (b) $\sigma = \overline{d}$. (c) $\sigma = 5\overline{d}$. (d) $\sigma = 20\overline{d}$.

Now we are in a position to explain why we transform similarity matrices by histogram equalization to eliminate the influence of σ. With a given set of data, the pairwise distances are fixed and not influenced by σ. In this case, while the variance of σ's changes the absolute similarity values, the relative relationship among the similarity values is unchanged. For example, if $d(x1, y1) > d(x2, y2)$, then $s(x1, y1) < s(x2, y2)$ holds for arbitrary σ's. As a result, the sorting of the similarity values are fixed and not influenced by σ's. Recalling that after histogram equalization transformation, the new similarity matrices are determined only by the sorting of the similarity values, we see that new similarity matrices are invariant to σ's. Consequently, the DSets clustering results are no longer influenced by σ's. For simplicity of expression, in this paper DSets-histeq is used to denote the DSets algorithm with similarity matrices transformed by histogram equalization before clustering. Corresponding to the clustering results in Figure 1 and Figure 2, we use DSets-histeq to do the clustering and show the results in Figure 3 and Figure 4, respectively. We observe



(a) (b)

FIGURE 4. The clustering results of the DSets-histeq algorithm on ten datasets. (a) F-measure. (b) Jaccard index.

that with the help of histogram equalization, the clustering results of DSets-histeq are invariant to σ's almost completely. This confirms that the histogram equalization transformation of similarity matrices is effective in solving the parameter dependence problem of the DSets algorithm. The slight variance observed on some datasets are caused by the quantization process in histogram equalization, and can be reduced by increasing the number of bins.

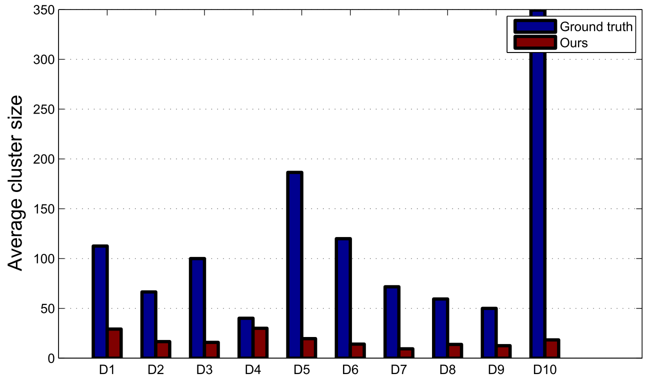


FIGURE 5. Comparison of the average cluster sizes from DSets-histeq and ground truth.

While Figure 3 and Figure 4 show that the influence from σ is eliminated effectively, they also indicate that the clustering results of DSets-histeq are far from being satisfactory. In Figure 3, the clusters are usually quite small and of spherical shapes, and in Figure 4 the clustering results are usually much inferior to the best possible ones shown in Figure 2. We explain this observation as follows. The dominant set definition requires each included data to be able to preserve the internal coherency in the dominant set. This requirement is very strict and only a set of data with high pairwise similarities are able to form a dominant set. After histogram equalization transformation, the contrast of the similarity values is increased and it is difficult to find a large amount of data with high pairwise similarities. As a result, DSets-histeq is likely to generate small clusters, which are usually smaller than the real ones. As a result, the clustering results are not satisfactory. In fact, we report the comparison of the average cluster sizes obtained with DSets-histeq and the true average cluster sizes in Figure 5, and observe that the clusters obtained with DSets-histeq are usually much smaller than the real ones. In Figure 5 D1, D2, $\cdots$, D10 are used to represent the ten datasets in the order of Aggregation, Compound, Pathbased, R15, Jain, Flame, Thyroid, Wine, Iris and Breast,

While the small clusters from DSets-histeq means unsatisfactory clustering results, they also provide an opportunity to solve the second problem and generate clusters of arbitrary shapes. As DSets-histeq groups only very similar data into a cluster, the data in such a cluster usually do belong to one single cluster and should not be partitioned. Considering also that the small clusters are usually much smaller than the real ones, we notice that the generated clusters by DSets-histeq are usually subsets of real clusters. Therefore it is possible for us to expand the small clusters to obtain clusters of arbitrary shapes and improve the clustering results. In this sense, although the histogram equalization transformation doesn't solve the second problem explicitly, it does provide an opportunity to achieve the purpose. In the succeeding subsection we introduce the cluster expansion algorithm used in this paper.

B. CLUSTER EXPANSION

Since the clusters generated by DSets-histeq are usually the subsets of real clusters, we regard them as initial clusters and propose to expand them to obtain clusters of arbitrary shapes and improve the clustering results. Our cluster expansion method is based on the property of dominant set and is derived in details in the following.

The dominant set definition imposes a very strong restriction of the high internal similarity in the dominant set. As a result, the data in a dominant set are very similar to each other. The histogram equalization transformation enlarges the similarity contrast in the similarity matrices and strengthens the requirement on high internal similarity. Considering also the fact that DSets-histeq generates clusters one by one, we find that the first generated cluster corresponds to the densest part in the dataset, and has higher internal similarity than the clusters extracted later. Taking only the first cluster into account, this means that this cluster has higher density than neighboring areas. Therefore we can make use of this observation to expand the initial cluster. The other clusters can be obtained in the remaining unclustered data with the same method. In the following we use the first cluster as an example to show how to accomplish the cluster expansion.

In cluster expansion, we need a criterion to determine which neighboring data should be included into the cluster. The first cluster generated by DSets-histeq has the highest internal similarity in all the clusters, and we make use of this information to expand the cluster. In stating that the first cluster has the highest internal similarity, it should be noted that the internal similarity is evaluated with all the data in the cluster. When it comes to the local density of each data, the data in the cluster not necessarily have larger densities than those outside. Therefore if we find out the minimum local density in the cluster, we can add an outside data into the cluster if its local density is above the minimum local density.

In other words, we intend to maximize the cluster size on condition that the minimum local density in the cluster is unchanged. Denoting the first initial cluster by D, we intend to obtain the final cluster as

$$C = \max D, \quad sb.t. \; \xi_C = \xi_D, \tag{7}$$

where ξ_C and ξ_D denote the minimum local density in C and D, respectively.

We use the following method to obtain the final cluster by cluster expansion [40]. Firstly, for each data i in the initial cluster D, we calculate its average similarity ψ_i with its nearest neighbors in the cluster. The average similarity reflects the local density of the data in the cluster. Then we find the minimum of these average similarities as $th = \min_{i \in D} \psi_i$, which denotes the minimum local density acceptable in the initial cluster. Therefore if an outside data has a larger local density than th, it should be included into the cluster. Note that in calculating the local density of an outside data, only the nearest neighbors in the initial cluster are used. Before cluster expansion, we sort the outside data according to their distance to the initial cluster so that the nearest ones will be considered first.

In summary, the cluster expansion process of an initial cluster D is described as follows.

1) For each data $i \in D$, calculate its local density as

$$\psi_i = \frac{1}{|S_{inn}|} \sum_{k \in S_{inn}} s_{ik}, \tag{8}$$

where S_{inn} is the set of nearest neighbors of i in D.

2) Calculate the density threshold as $th = \min_{i \in D} \psi_i$.

3) Sort the outside data in decreasing order based on their average similarity with the data in the cluster.

4) Starting from the nearest outside data, and for each outside data j, calculate its similarity with the cluster as $\psi_j = \frac{1}{|S_{jnn}|} \sum_{k \in S_{jnn}} s_{jk}$. If $\psi_j > th$, include j into the cluster.

IV. EXPERIMENTS

We firstly compare the proposed algorithm with DSets-histeq to show the effect of the cluster expansion method. As both algorithms are not influenced by σ's, we use $\sigma = \overline{d}$ to generate the results. The comparison of two algorithms is reported in Figure 6. From the comparison we observe that on all the ten datasets, our algorithm generates better results than DSets-histeq, showing the effectiveness of the cluster expansion algorithm.

Since our algorithm is proposed to solve the problems of the DSets algorithm, we then make a comparison between these two algorithms. As the clustering results of the DSets algorithm are influenced by σ's, we set σ as $30\overline{d}$, which is selected from testing values from $0.1\overline{d}$ to $100\overline{d}$ as the one generating the best average results. The clustering results of these two algorithms are shown in Figure 7, where we observe that on seven out of the ten datasets, the results of our algorithm are better than or comparable to those of the DSets algorithm with selected σ's. This shows the effectiveness of our algorithm in solving the problems of DSets. The reason why on D2, D3 and D7 our algorithm is outperformed by DSets, in our opinion, is as follows. The key of our algorithm is to generate small clusters with DSets-histeq and then expand the clusters to improve the clustering results. Due to the imperfection in both DSets-histeq and cluster expansion, the obtained clusters may be larger or smaller than the real ones. Even if the average cluster size from our algorithm is the same as ground truth, the obtained clusters may still be different. All these factors degrade the performance of our algorithm. On the other hand, the DSets algorithm may generate very good results if the major clusters are spherical and σ is selected appropriately. In this sense, it is not surprising that the DSets algorithm performs better than ours on some datasets.

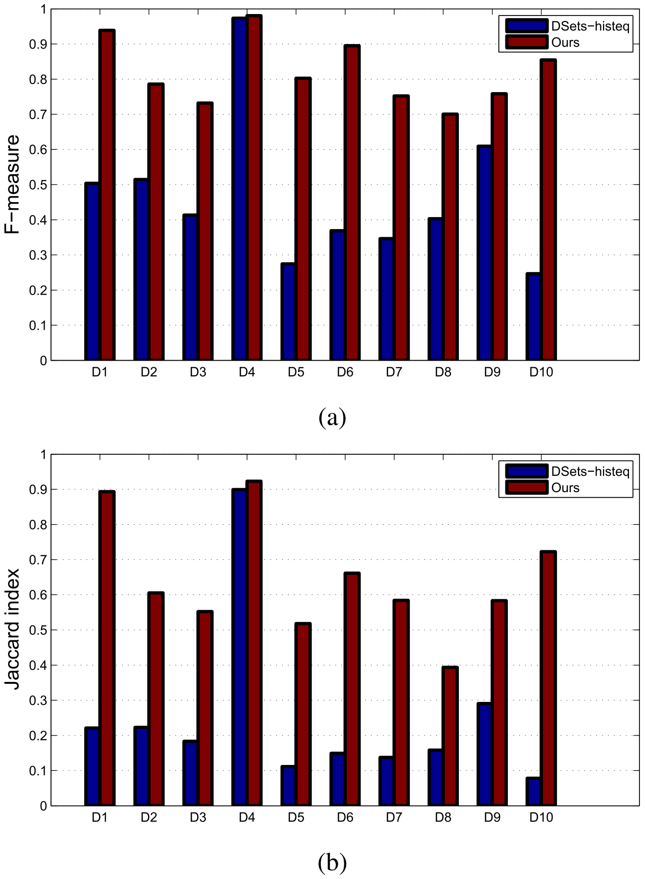
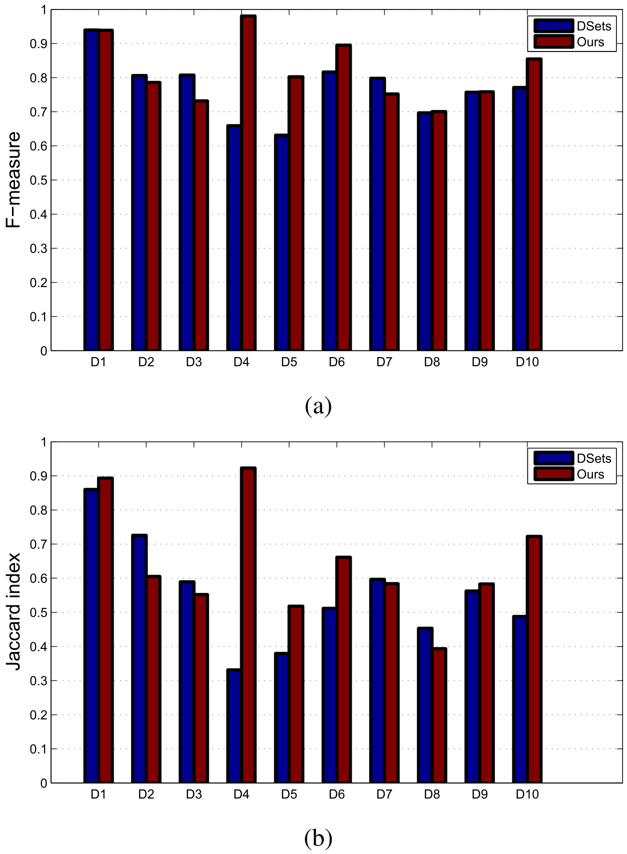


FIGURE 6. The clustering results of DSets-histeq and our algorithm. (a) F-measure. (b) Jaccard index.

We also compare our algorithm with some others, including K-means, NCuts, SPRG [12], DBSCAN, AP and DP. With K-means, NCuts and SPRG, we set the required number of clusters as the ground truth and report the average results of ten runs. With DBSCAN we select the parameter *MinPts* from 1, 2, $\cdots$, 10 and determine *Eps* based on *MinPts* with the method presented in [41]. The AP algorithm requires as input the preference value of each data,



(a)

(b)

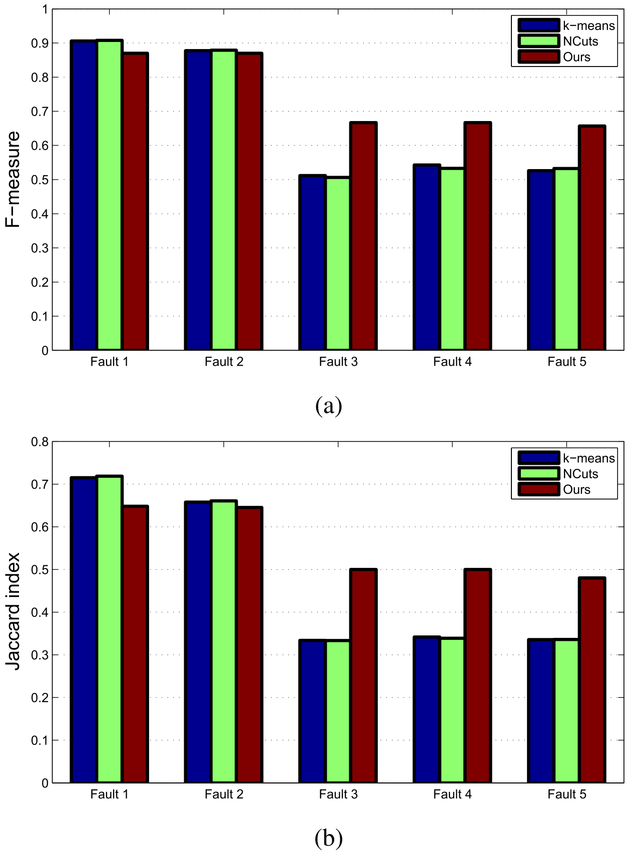
FIGURE 7. The clustering results of DSets with selected σ's and our algorithm. (a) F-measure. (b) Jaccard index.

TABLE 2. Comparison of different algorithms on ten datasets with F-measure.

	A1	A2	D3	A4	A5	A6	Ours
D1	0.83	0.99	0.90	0.82	0.73	0.99	0.94
D2	0.68	0.70	0.92	0.77	0.64	0.82	0.79
D3	0.70	0.96	0.72	0.69	0.56	0.69	0.73
D4	0.82	0.99	0.73	0.54	0.93	0.99	0.98
D5	0.79	0.63	0.85	0.57	0.86	0.90	0.80
D6	0.84	0.99	0.96	0.74	0.60	1.00	0.90
D7	0.83	0.64	0.68	0.52	0.97	0.55	0.75
D8	0.70	0.64	0.51	0.64	0.97	0.72	0.70
D9	0.89	0.93	0.78	0.93	0.87	0.70	0.76
D10	0.96	0.64	0.87	0.82	0.97	0.67	0.85
average	0.80	0.81	0.79	0.70	0.81	0.80	0.82

TABLE 3. Comparison of different algorithms on ten datasets with Jaccard index.

	A1	A2	A3	A4	A5	A6	Ours
D1	0.64	0.98	0.81	0.71	0.49	0.98	0.89
D2	0.46	0.46	0.87	0.69	0.42	0.71	0.61
D3	0.50	0.85	0.53	0.49	0.34	0.49	0.55
D4	0.65	0.99	0.40	0.25	0.83	0.96	0.92
D5	0.53	0.42	0.91	0.29	0.63	0.71	0.52
D6	0.59	0.97	0.90	0.47	0.41	1.00	0.66
D7	0.64	0.40	0.57	0.29	0.90	0.29	0.58
D8	0.42	0.43	0.34	0.36	0.87	0.44	0.39
D9	0.69	0.79	0.60	0.77	0.66	0.51	0.58
D10	0.87	0.39	0.78	0.56	0.89	0.48	0.72
average	0.60	0.67	0.67	0.49	0.65	0.66	0.64



(a)

(b)

FIGURE 8. The clustering results of three algorithms in fault detection. (a) F-measure. (b) Jaccard index.

and Brendan and Delbert [11] presented a method to calculate the range $[p_{min}, p_{max}]$ of this parameter. We test $p = p_{min} + step * k$ with $step = (p_{max} - p_{min})/10$ and $k = 1, 2, \cdots, 9, 9.1, 9.2, \cdots, 9.9$ and select the one which generates the best average results. With DP we adopts the cutoff kernel and the cutoff distance is determined by including 1.2 percentage of data in the neighborhood after parameter tuning. The comparison of these algorithms is shown in Table 2 and Table 3. In these two table A1, A2, $\cdots$, A6 represent the K-means, NCuts, DBSCAN, AP, SPRG and DP, respectively. From the comparison we observe that while some algorithms generates very good results on some datasets, their results on the other datasets are much worse. As a result, the average results of our algorithm are comparable to the other algorithms.

Finally, we apply our algorithm to fault detection and compare with the K-means and NCuts algorithms. The data are from the widely used Tennessee Eastman process simulator [42]. We make use of the first six testing datasets, including one non-fault dataset and five fault datasets. Each dataset consists of 960 samples with 52 variables, in which we select the first 22 and the last 11 variables in our experiment.

We pool the non-fault dataset and each fault dataset together and do clustering to differentiate between fault and non-fault samples. With F-measure and Jaccard index as the evaluation criterions, the clustering results of the three algorithms are shown in Figure 8. From the comparison we observe that our algorithm performs comparably to the other two widely used algorithms. Considering that these algorithms for comparison benefit from ground truth or manually selected parameters, we believe the comparison shows the effectiveness of our algorithm.

V. CONCLUSION

In this paper we present a data driven clustering algorithm to solve the problems afflicting the dominant sets algorithm. We firstly study the dominant sets algorithm in depth and find its problems lie in the sensitiveness to the similarity parameter and the tendency to generate spherical clusters only. We analyze the reason behind the two problems and present our solution. Firstly, we transform the input similarity matrices with histogram equalization to eliminate the influence of similarity parameters on clustering results. Then the initial clusters are expanded based on the density information captured in the initial clusters. We perform experiments on various datasets and showed that our algorithm improves the clustering results of the dominant sets algorithm significantly. Our algorithm is also shown to perform better than or comparably to other algorithms with parameter tuning. We finally apply our algorithm to fault detection and show its effectiveness by comparison with commonly used clustering algorithms.

REFERENCES

[1] C. Couprie, L. Grady, L. Najman, and H. Talbot, "Power watersheds: A new image segmentation framework extending graph cuts, random walker and optimal spanning forest," in *Proc. IEEE Int. Conf. Comput. Vis.*, Sep./Oct. 2009, pp. 731–738.

[2] S. Yin, L. Liu, and J. Hou, "A multivariate statistical combination forecasting method for product quality evaluation," *Inf. Sci.*, vols. 355–356, pp. 229–236, Aug. 2016.

[3] M. Gong, Y. Liang, J. Shi, W. Ma, and J. Ma, "Fuzzy C-means clustering with local information and kernel metric for image segmentation," *IEEE Trans. Image Process.*, vol. 22, no. 2, pp. 573–584, Feb. 2013.

[4] S. Yin and Z. Huang, "Performance monitoring for vehicle suspension system via fuzzy positivistic C-means clustering based on accelerometer measurements," *IEEE/ASME Trans. Mechatronics*, vol. 20, no. 5, pp. 2613–2620, Oct. 2015.

[5] N. Sharet and I. Shimshoni, "Analyzing data changes using mean shift clustering," *Int. J. Pattern Recognit. Artif. Intell.*, vol. 30, no. 7, p. 1650016, 2016.

[6] S. Yin, H. Gao, J. Qiu, and O. Kaynak, "Descriptor reduced-order sliding mode observers design for switched systems with sensor and actuator faults," *Automatica*, vol. 76, pp. 282–292, Feb. 2017.

[7] X. Xie, W. Sun, and K. C. Cheung, "An advanced PLS approach for key performance indicator-related prediction and diagnosis in case of outliers," *IEEE Trans. Ind. Electron.*, vol. 63, no. 4, pp. 2587–2594, Apr. 2016.

[8] S. Yin, X. Xie, and W. Sun, "A nonlinear process monitoring approach with locally weighted learning of available data," *IEEE Trans. Ind. Electron.*, vol. 64, no. 2, pp. 1507–1516, Feb. 2017.

[9] J. Shi and J. Malik, "Normalized cuts and image segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 8, pp. 888–905, Aug. 2000.

[10] M. Ester, H. P. Kriegel, J. Sander, and X. W. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *Proc. Int. Conf. Knowl. Discovery Data Mining*, 1996, pp. 226–231.

[11] B. J. Frey and D. Dueck, "Clustering by passing messages between data points," *Science*, vol. 315, no. 5814, pp. 972–976, Feb. 2007.

[12] X. Zhu, C. C. Loy, and S. Gong, "Constructing robust affinity graphs for spectral clustering," in *Proc. IEEE Int. Conf. Comput. Vis. Pattern Recognit.*, Jun. 2014, pp. 1450–1457.

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

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

[15] C. Panagiotakis, H. Papadakis, E. Grinias, N. Komodakis, P. Fragopoulou, and G. Tziritas, "Interactive image segmentation based on synthetic graph coordinates," *Pattern Recognit.*, vol. 46, no. 11, pp. 2940–2952, 2013.

[16] X. Wang, Y. Tang, S. Masnou, and L. Chen, "A global/local affinity graph for image segmentation," *IEEE Trans. Image Process.*, vol. 24, no. 4, pp. 1399–1411, Apr. 2015.

[17] C. Fraley and A. E. Raftery, "How many clusters? Which clustering method? Answers via model-based cluster analysis," *Comput. J.*, vol. 41, no. 8, pp. 578–588, 1998.

[18] G. Evanno, S. Regnaut, and J. Goudet, "Detecting the number of clusters of individuals using the software structure: A simulation study," *Molecular Ecol.*, vol. 14, no. 8, pp. 2611–2620, 2005.

[19] A. Torsello, S. R. Bulo, and M. Pelillo, "Beyond partitions: Allowing overlapping groups in pairwise clustering," in *Proc. Int. Conf. Pattern Recognit.*, Dec. 2008, pp. 1–4.

[20] S. R. Bulò, A. Torsello, and M. Pelillo, "A game–theoretic approach to partial clique enumeration," *Image Vis. Comput.*, vol. 27, no. 7, pp. 911–922, 2009.

[21] J. Hou, H. Gao, and X. Li, "DSets-DBSCAN: A parameter-free clustering algorithm," *IEEE Trans. Image Process.*, vol. 25, no. 7, pp. 3182–3193, Jul. 2016.

[22] R. Hamid, S. Maddi, A. Johnson, A. Bobick, I. Essa, and C. Isbell, "A novel sequence representation for unsupervised analysis of human activities," *Artif. Intell.*, vol. 173, no. 14, pp. 1221–1244, 2009.

[23] X. Yang, H. Liu, and L. J. Latecki, "Contour-based object detection as dominant set computation," *Pattern Recognit.*, vol. 45, no. 5, pp. 1927–1936, 2012.

[24] J. Hou and M. Pelillo, "A simple feature combination method based on dominant sets," *Pattern Recognit.*, vol. 46, no. 11, pp. 3129–3139, 2013.

[25] J. Hou, H. Gao, and X. Li, "Feature combination via clustering," *IEEE Trans. Neural Netw. Learn. Syst.*, to be published, doi: 10.1109/TNNLS.2016.2645883.

[26] M. Pavan and M. Pelillo, "Efficient out-of-sample extension of dominant-set clusters," in *Proc. Adv. Neural Inf. Process. Syst.*, 2005, pp. 1057–1064.

[27] J. Hou and B. Zhang, "Cluster merging based on a decision threshold," *Neural Comput. Appl.*, to be published, doi: 10.1007/s00521-016-2699-4.

[28] S. Vascon, E. Z. Mequanint, M. Cristani, H. Hung, M. Pelillo, and V. Murino, "Detecting conversational groups in images and sequences: A robust game-theoretic approach," *Comput. Vis. Image Understand.*, vol. 143, pp. 11–24, Feb. 2016.

[29] E. Zemene and M. Pelillo, "Interactive image segmentation using constrained dominant sets," in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 278–294.

[30] R. Tripodi and M. Pelillo, "A game-theoretic approach to word sense disambiguation," *Comput. Linguistics*, vol. 43, no. 1, pp. 31–70, 2017.

[31] J. Hou and W. Liu, "A parameter-independent clustering framework," *IEEE Trans. Ind. Informat.*, vol. 13, no. 4, pp. 1825–1832, Aug. 2017.

[32] S. R. Bulò, M. Pelillo, and I. M. Bomze, "Graph-based quadratic optimization: A fast evolutionary approach," *Comput. Vis. Image Understand.*, vol. 115, no. 7, pp. 984–995, 2011.

[33] A. K. Jain and M. H. C. Law, "Data clustering: A user's dilemma," in *Proc. Int. Conf. Pattern Recognit. Mach. Intell.*, 2005, pp. 1–10.

[34] A. Gionis, H. Mannila, and P. Tsaparas, "Clustering aggregation," *ACM Trans. Knowl. Discovery Data*, vol. 1, no. 1, pp. 1–30, 2007.

[35] C. T. Zahn, "Graph-theoretical methods for detecting and describing gestalt clusters," *IEEE Trans. Comput.*, vol. C-20, no. 1, pp. 68–86, Jan. 1971.

[36] H. Chang and D.-Y. Yeung, "Robust path-based spectral clustering," *Pattern Recognit.*, vol. 41, no. 1, pp. 191–203, 2008.

[37] C. J. Veenman, M. J. T. Reinders, and E. Backer, "A maximum variance cluster algorithm," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 9, pp. 1273–1280, Sep. 2002.

[38] L. Fu and E. Medico, ''Flame, a novel fuzzy clustering method for the analysis of DNA microarray data,'' *BMC Bioinform.*, vol. 8, no. 1, pp. 1–17, 2007.

[39] T. Acharya and A. K. Ray, *Image Processing: Principles and Applications*. Hoboken, NJ, USA: Wiley, 2005.

[40] J. Hou, C. Sha, L. Chi, and H. Cui, ''A density based cluster extension method,'' in *Proc. Int. Conf. Ind. Technol.*, Mar. 2016, pp. 932–937.

[41] M. Daszykowski, B. Walczak, and D. L. Massart, ''Looking for natural patterns in data: Part 1. density-based approach,'' *Chemometrics Intell. Lab. Syst.*, vol. 56, no. 2, pp. 83–92, 2001.

[42] J. J. Downs and E. F. Vogel, ''A plant-wide industrial process control problem,'' *Comput. Chem. Eng.*, vol. 17, no. 3, pp. 245–255, Mar. 1993.

JIAN HOU (M'12) received the Ph.D. degree from the Harbin Institute of Technology, Harbin, China, in 2007. He is currently an Associate Professor with the College of Engineering, Bohai University, Jinzhou, China. His research interests include pattern recognition, machine learning, computer vision, and image processing.

BING XIAO is currently an Associate Professor with the College of Engineering, Bohai University, China. His research interests lie in the field of machine learning and visual inspection.