Statistics and Its Interface Volume 16 (2023) 363–377

A semi-supervised density peaks clustering algorithm

Yuanyuan Wang* and Bingyi Jing

Density peaks clustering (DPC) is a density-based unsupervised clustering algorithm with the advantages of fast clustering capacity for arbitrary shape data and easy implementation without iteration. However, in practice, a small amount of label information might be partially available but not sufficient to be used to generate supervised learning. Semi-supervised clustering is often adopted to incorporate such partial information. In this paper, a novel semi-supervised density peaks clustering algorithm (SS-DPC) is proposed to extend the classical density peaks clustering algorithm to the semi-supervised clustering. In contrast to DPC, SS-DPC uses prior information in the form of class labels to guide the learning process for improved clustering. SS-DPC is a semi-supervised clustering that can handle data with a small number of labels. First, SS-DPC identifies possible cluster centers based on labeled and unlabeled data automatically. Then, to incorporate partial information, virtual labels are brought in to integrate the partial information with identified centers in a uniform framework. Moreover, labeled data are used to initialize the semi-supervised clustering process to maintain the correctness of prior information in the clustering procedure. Subsequently, the nearest-point-based method is used to detect the labels of non-center unlabeled data. Finally, a step-by-step mergence strategy is introduced to generate more reasonable results. Experiments on eight UCI datasets illustrate that the proposed semi-supervised clustering algorithm yields promising clustering results.

Keywords and phrases: Semi-supervised clustering, Density peaks clustering, Partial information, Virtual label, Mergence of clusters.

1. INTRODUCTION

Clustering is a popular unsupervised learning method that reveals the underlying structures from the unlabeled data. It aims to partition the dataset of n points into different groups such that the data in the same groups have relatively higher similarities than those in different groups [11]. With the development of big data technology, clustering has attracted increasing attention and wide applications in many fields [6], such in image segmentation [22], community detection [30], information retrieval [4], recommendation systems [37], and bioinformatics [14]. Moreover, various methods have been proposed for data clustering, among which K-means [21], spectral clustering [20], DBSCAN [10], GMM [3], and hierarchical agglomerative clustering [15], are the most popular ones.

In practice, some prior knowledge about data labels may be available. However, this kind of partial information is often lacking and always insufficient to generate supervised learning because of its high cost. For example, in disease diagnosis, the experts often have no time to mark the type of disease for most patients, so it is common in the obtained datasets that only a small number of patients are marked by their disease type. Similarly, in image recognition, it is difficult and expensive to label the categories for most of the mass images, so we might only have a few labeled images. Although the labeled data that can be observed usually is few, it can provide valuable information, which could be helpful in clustering the data. To avoid the information of labeled data wasted, it is necessary to make full use of this partial information to induction and process the data for better clustering, which is also the purpose of this study.

Traditional unsupervised clustering, which only uses unlabeled data for learning, have no way to utilize such partial information. Consequently, semi-supervised clustering, which uses a small amount of labeled data along with unlabeled data to aid clustering, is not only a way often to be considered but also a significant topic in machine learning. In semi-supervised clustering, partial information is used as a small amount of supervision knowledge to guide the learning procedure. Class labels and pairwise constraints (e.g., must-link and cannot-link) are two common forms of partial information [16, 12]. Of which, class labels express prior knowledge with data labels directly, whereas pairwise constraints specify whether the two points should or should not be in the same cluster.

To use partial information for clustering performance improvement effectively, many works have focused on extending traditional unsupervised clustering to account for semi-supervised clustering. Among these works, most of them incorporate the pairwise constraints or prior class labels into the objective function of the clustering algorithm for improved results. For example, COP-Kmeans [32] incorporates pairwise constraints into the clustering process of K-means

*Corresponding author.

to obtain a partition that satisfies all the specified relations of two points in the dataset. SDEC [25] modifies deep embedded clustering based on a joint objective that considers unlabeled data and prior information. CPSSAP [34] extends affinity propagation to semi-supervised clustering, which defines a projection objective function based on pairwise constraints to project all the points in a lower-dimensional space. CCSR [17] develops a regularization framework for constrained spectral clustering, which adapts spectral embedding to accord with pairwise constraints through optimization. Although these methods have been empirically proved to be more practical and effective than their unsupervised baseline counterparts, the optimization of these models is computationally rather expensive [28].

Rodriguez and Laio [26] proposed density peak clustering (DPC), which was a density-based clustering algorithm with the advantages of fast clustering capacity for arbitrary shape data and easy implementation without the need of iteration and optimization. By computing local density and pseudo distance, the algorithm first constructs a decision graph to identify points of higher densities and pseudo distance manually to serve as cluster centers. Subsequently, non-center points are assigned to clusters by using a nearest-point method. The cutoff distance, which is used for local density estimation, is the only parameter that needs to be specified. Due to a simple and efficient clustering algorithm, DPC has attracted considerable attentions and various applications, such as, DPSCD for social network community detection [33], combining dynamic time warping and DPC for intradialytic blood pressure pattern recognition [35], DPNN for intrusion detection [19], and DPC-MD for mixed data [9]. Despite being a promising clustering algorithm for addressing unlabeled data, DPC is a kind of unsupervised clustering method that does not combine the partial information with unlabeled data to bias the clustering process. For the fast clustering ability and the widespread use, extending DPC to a semi-supervised clustering is meaningful for the clustering community.

In this paper, a semi-supervised DPC algorithm (SS-DPC) is proposed to incorporate partial information in the form of class labels into DPC. In comparison with DPC, SS-DPC is a semi-supervised clustering algorithm that can handle data with small amount of labels. The algorithm first identifies cluster centers with labeled and unlabeled data. Generally, prior labels indicate the classes that we have known, but they might not reveal all the underlying clusters because only a small amount of labeled data are available for semi-supervised clustering. Therefore, identifying possible new cluster centers is necessary. In this paper, similar to DPC, SS-DPC identifies cluster centers on the entire data space. A geometric method is adopted to identify possible cluster centers automatically to address the shortcoming of manual selection in DPC. This method tends to detect more cluster centers to avoid missing centers.

To incorporate partial information, this paper constructs a semi-supervised clustering process based on the assumption that prior information is always correct. When two or more identified centers have the same prior label, the cluster possibly consists of two or more distinguished sub-clusters. Therefore, in the semi-supervised clustering construction stage, virtual labels are brought in to integrate partial information and identified centers into a uniform framework. Note that in this study, a virtual label, different from any prior labels, is used to mark the cluster of a center that has the identical prior label with other centers or that has not been labeled previously. Specifically, each identified center is labeled either by its given label (for the center that contained in the labeled dataset) or by a new virtual label (for the center that contained in the unlabeled dataset or share label with other centers). Because all identified centers are assigned different labels to avoid sharing, SS-DPC can find new sub-clusters that are not labeled previously.

Moreover, labeled data are used to initialize the semi-supervised clustering process in SS-DPC. Regardless of the densities they have, all labeled data are assigned to their true labels before the clustering procedure to maintain the correctness of the prior information of labeled data. Subsequently, the nearest-point method is used to generate the labels for the non-center unlabeled points. Finally, based on the result of the semi-supervised procedure, a step-by-step mergence strategy is adopted to possibly merge two resulted clusters to obtain a more reasonable result. To keep the prior class information, the mergence strategy only merges the clusters marked by virtual labels with their nearest ones whenever the distance is small enough.

The main contributions of this study can be summarized as follows.

- Classical DPC algorithm is first extended to a semi-supervised clustering with both labeled and unlabeled data.
- A geometric method is introduced to select the possible cluster number k automatically.
- Virtual labels are brought in to integrate partial information and identified centers into a uniform framework.
- Labeled data are used to initialize the semi-supervised process.
- A step-by-step mergence strategy is adopted for semi-supervised clustering result adjustment.

The rest of this paper is organized as follows: Section 2 describes the details of SS-DPC. Section 3 reports the experimental results on eight UCI datasets. Section 4 draws the conclusion.

2. METHODOLOGY

In this section, we first provide a brief introduction to the traditional DPC algorithm. Second, the proposed SS-DPC algorithm is described in detail.

2.1 Review of unsupervised density peaks clustering

The DPC [26] algorithm is proposed on the basis that: (i) the local densities of the cluster centers are higher than the points surrounding them, and (ii) the cluster centers are at a relatively large distance from any points with a higher local density. This algorithm has been extensively used in practice because it can detect cluster centers and recognize non-spherical clusters and spot outliers [8, 18, 29]. This density-distance-base clustering algorithm can be described as follows.

Suppose that $\mathcal{X} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_n\}$ is the dataset to be clustered, where n is the number of points in the dataset, and $\boldsymbol{x}_i = (x_{i1}, x_{i2}, ..., x_{ip})$ is the ith point expressed by p attributes. Then, for any $\boldsymbol{x}_i \in \mathcal{X}$, the local density ρ_i is measured by

$$(1) \qquad \rho_i = \sum_{\boldsymbol{x}_j \in \mathcal{X}, i \neq j} 1_{(d_{ij} - d_c < 0)},$$

where d_{ij} is the distance between $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$, and d_c is the cutoff distance to be pre-specify. ρ_i is defined as the number of points in the d_c-neighborhood of $\boldsymbol{x}_i$. d_c is the only parameter affecting the values of local densities. As suggested by the authors in reference [26], d_c can be selected so that the average number of neighbors is around 1% to 2% of the total number of points in the dataset. Therefore, d_c is usually set as the 1% to 2% values of all the distance between every two points sorted in descending order.

In Equation (1), ρ_i only describes the number of points in the d_c-neighborhood of $\boldsymbol{x}_i$, which can not measure how closely these points in d_c-neighborhood are around $\boldsymbol{x}_i$. Moreover, for datasets with a small number of points, a reliable estimate of the local densities is difficult because of the large statistical errors. So more accurate measures for local density are significant [26]. Therefore, DPC adopts the Gaussian kernel function to estimate the local densities, defined as

$$(2) \qquad \rho_i = \sum_{\boldsymbol{x}_j \in \mathcal{X}, i \neq j} \exp\left(-\frac{d_{ij}^2}{d_c^2}\right),$$

where d_c is an adjustable parameter which is used to control the weight degradation rate [8].

The pseudo distance δ_i is defined as the minimum distance between $\boldsymbol{x}_i$ and any other points with higher densities than that of $\boldsymbol{x}_i$,

$$(3) \qquad \delta_i = \min_{\rho_j > \rho_i} d_{ij}.$$

Where, if $\boldsymbol{x}_i$ has the highest density, δ_i is defined as the maximum distance between $\boldsymbol{x}_i$ and the other points in dataset $\mathcal{X}$. That is, if $\boldsymbol{x}_i = \max(\rho)$, then $\delta_i = \max_j(d_{ij}), j \neq i$.

Once the local density ρ_i and the distance δ_i are calculated, the cluster centers can be chosen by $\rho - \delta$ decision graph or by the criterion γ_i where $\gamma_i = \rho_i * \delta_i$, $i = 1, 2, 3, ..., n$. The larger the γ_i, the more likely the $\boldsymbol{x}_i$ is to be the cluster center.

After finding the cluster centers, in decreasing order of density, each of the remaining data points is assigned point-by-point to the cluster of its nearest neighbor with a higher density.

2.2 Our algorithm: SS-DPC

In semi-supervised clustering, a small amount of partial information is available to help clarify the underlying clustering problem. Given that the partial information in the form of class labels is always reliable, SS-DPC algorithm that uses a few number of labeled data to guide clustering is proposed in this paper. The main algorithm of SS-DPC includes the identification of cluster centers and nearest points, construction of semi-supervised clustering, and mergence of clusters.

2.2.1 Identification of cluster centers and nearest points

Let $\mathcal{X} = \mathcal{L} \cup \mathcal{U}$ be the dataset to be clustered, where $\mathcal{L} = \cup_{i=1}^{n_0} \boldsymbol{x}_i$ is the set of labeled data, whereas $\mathcal{U} = \cup_{i=1}^{n_1} \boldsymbol{x}_i$ is the set of unlabeled data, and $n_0 + n_1 = n$, $n_0 \ll n_1$. For $\forall \boldsymbol{x}_i \in \mathcal{X}$, the local density ρ_i can be calculated by Equation (1) or (2), and the pseudo distance δ_i can be calculated by Equation (3). Meanwhile, according to the procedure of computing δ_i, each point corresponds to a unique nearest point with a higher local density than it [36]. Here, we denote the nearest point of $\boldsymbol{x}_i$ by $\boldsymbol{nearest}_i$. That is, $\boldsymbol{nearest}_i$ is the closest point to $\boldsymbol{x}_i$ among the points with higher densities. Particularly, if $\boldsymbol{x}_i$ has the highest local density, $\boldsymbol{nearest}_i$ is defined as itself by default [18]. For $i = 1, 2, ..., n$, $\boldsymbol{nearest}_i$ can be expressed by

$$(4) \quad \boldsymbol{nearest}_i = \begin{cases} \boldsymbol{x}_i & \text{if } \boldsymbol{x}_j \in \mathcal{X}, \rho_i \geq \rho_j \\ \boldsymbol{x}_j, j = \arg\min_j d_{ij} & \text{if } \boldsymbol{x}_j \in \mathcal{X}, \rho_i < \rho_j. \end{cases}$$

Then, the points with relatively high ρ values and high δ values are remarkable choice for the cluster centers. As mentioned in Section 2.1, the centers can be identified by the $\rho - \delta$ decision graph or by the criterion $\gamma = \rho * \delta$. However, it still has drawbacks that the cluster centers need to be selected manually or the number of cluster centers needs to be pre-defined. To address this problem, we propose that the possible cluster number k is selected automatically by

$$(5) \qquad k = sup\{i : \gamma_{(i)} - \gamma_{(i+1)} > \epsilon, i = 1, 2, ..., n\},$$

where $\boldsymbol{\gamma}_{order} = \{\gamma_{(1)}, ..., \gamma_{(n)}\}$ is the γ values sorted in descending order, and $\mathcal{X}_{order} = \{\boldsymbol{x}_{(1)}, ..., \boldsymbol{x}_{(n)}\}$ is the corresponding sample dataset. Inspired by the fact that the γ values on γ descending graph declines sharply and then remains roughly stable (as the example shown in Figure 1(b)), this study introduces a geometric method to select the parameter ϵ adaptively. This method selects ϵ such that the

Algorithm 1 Identification of cluster centers and nearest points

Input:
$\mathcal{X} = \mathcal{L} \cup \mathcal{U}$: $\mathcal{L}$ is labeled dataset, $\mathcal{U}$ is unlabeled dataset;
d_c: the cutoff distance.
Output:
Possible cluster center set: $\mathcal{C} = \{\boldsymbol{c}_1, \boldsymbol{c}_2, ..., \boldsymbol{c}_k\}$;
Nearest points set: $\boldsymbol{nearest} = \{\boldsymbol{nearest}_1, \boldsymbol{nearest}_2, ..., \boldsymbol{nearest}_n\}$.

Algorithm

```
1: For $\forall \boldsymbol{x}_i \in \mathcal{X}$, calculate $\rho_i$ according to Equation (1) or (2);
2: For $\forall \boldsymbol{x}_i \in \mathcal{X}$, calculate $\delta_i$ according to Equation (3);
3: For $\forall \boldsymbol{x}_i \in \mathcal{X}$, calculate the criterion $\gamma_i = \rho_i * \delta_i$;
4: For $\forall \boldsymbol{x}_i \in \mathcal{X}$, calculate $\boldsymbol{nearest}_i$ according to Equation (4), obtaining the nearest points set $\boldsymbol{nearest}$;
5: Sorted $\gamma$ values in descending order, obtaining $\boldsymbol{\gamma}_{order} = \{\gamma_{(1)}, \gamma_{(2)}, ..., \gamma_{(n)}\}$ and $\mathcal{X}_{order} = \{\boldsymbol{x}_{(1)}, \boldsymbol{x}_{(2)} ..., \boldsymbol{x}_{(n)}\}$;
6: Calculate the line $l_{max-min} : (\gamma_{(1)} - \gamma_{(n)})x + (n-1)y + (\gamma_{(n)} - n\gamma_{(1)}) = 0$;
7: Calculate distance from point $(i, \gamma_{(i)})$ to $l_{max-min}$ to obtain $\boldsymbol{d}_\gamma = \{d^\gamma_{(1)}, ..., d^\gamma_{(n)}\}$;
8: Set $\epsilon = \gamma_{(l)} - \gamma_{(l+1)}$, where $l = \arg\max_i d^\gamma_{(i)}$;
9: Determine the number of clusters $k = l$, and select the top $k$ points in $\mathcal{X}_{order}$ as the possible cluster centers, $\mathcal{C} = \{\boldsymbol{x}_{(1)}, ..., \boldsymbol{x}_{(k)}\}$.
```

value of the difference between $\gamma_{(k)}$ and $\gamma_{(k+1)}$ can be regarded as 0. That is, after selecting k using Equation (5), the differences between the adjacent γ values that correspond to non-center points must very close to 0.

Suppose that $\Delta_{(i)} = \gamma_{(i)} - \gamma_{(i+1)}, \gamma_{(i)} \in \boldsymbol{\gamma}_{order}$, then $\exists\ \gamma_{(l)} \in \boldsymbol{\gamma}_{order}$, s.t., the values of series $\{\Delta_{(l)}, \Delta_{(l+1)}, ..., \Delta_{(n-1)}\}$ are roughly stable, whereas the values of series $\{\Delta_{(1)}, \Delta_{(2)}, ..., \Delta_{(l-1)}\}$ are changeable. $\gamma_{(l)}$ is the right value to distinguish relatively higher γ values and lower γ values. Furthermore, the possible centers corresponds to the points that have relatively higher γ values. Therefore, $\Delta_{(l)}$ is the optimal value for ϵ. Accordingly, we can determine the cluster number $k = l$ and the center set as $\mathcal{C} = \{\boldsymbol{x}_{(1)}, ..., \boldsymbol{x}_{(l)}\}$. Consequently, the key issue is how to determine $\gamma_{(l)}$.

To this end, we first calculate the distance from each γ to the line $l_{max-min}$ that passes through $\min(\gamma)$ and $\max(\gamma)$, and then select the one that has a maximum distance to $l_{max-min}$ as the optimal choice for $\gamma_{(l)}$. Figure 1(a) shows an artificial dataset with 5% of data labeled. Figure 1(b) plots the γ values sorted in descending order, as well as the line $l_{max-min}$ passing through $(1, \gamma_{(1)})$ and $(n, \gamma_{(n)})$. The red solid point in Figure 1(c) marks the one in $\gamma_{(order)}$ that has the maximum distance to $l_{max-min}$, which is the $\gamma_{(l)}$ we expect to find. Therefore, the first l points in $\mathcal{X}_{order}$ are collected to serve as the possible cluster centers, as shown by the red points in Figure 1(d). In fact, ϵ can be selected to obtain more clusters that can be corrected by the following cluster mergence strategy.

Algorithm 1 summarizes the details of cluster centers and the nearest point identification process in this paper.

2.2.2 Construction of semi-supervised clustering

Generally, the partial information of data labels is beneficial to the whole clustering result, and the prior labels are always considered correct. In the semi-supervised clustering stage, virtual labels are brought in to integrate identified centers and partial information into a uniform framework. Labeled data are used to initialize the clustering process, and the labels of labeled data are kept unchanged in the label assignment process. Each labeled data is used as a supervisor to guide the label assigning of non-center unlabeled data with lower densities than it, thereby ensuring the effective spreading of prior information.

Firstly, the labeled data are assigned to their true labels regardless of their densities. Then, on the basis of the principle that no centers share the same prior labels, SS-DPC assigns labels to the centers according to the following rules. Let $\mathcal{C} = \{\boldsymbol{c}_1, ..., \boldsymbol{c}_k\}$ be the set of identified cluster centers, $\boldsymbol{l}_{prior}$ is the set of prior labels,

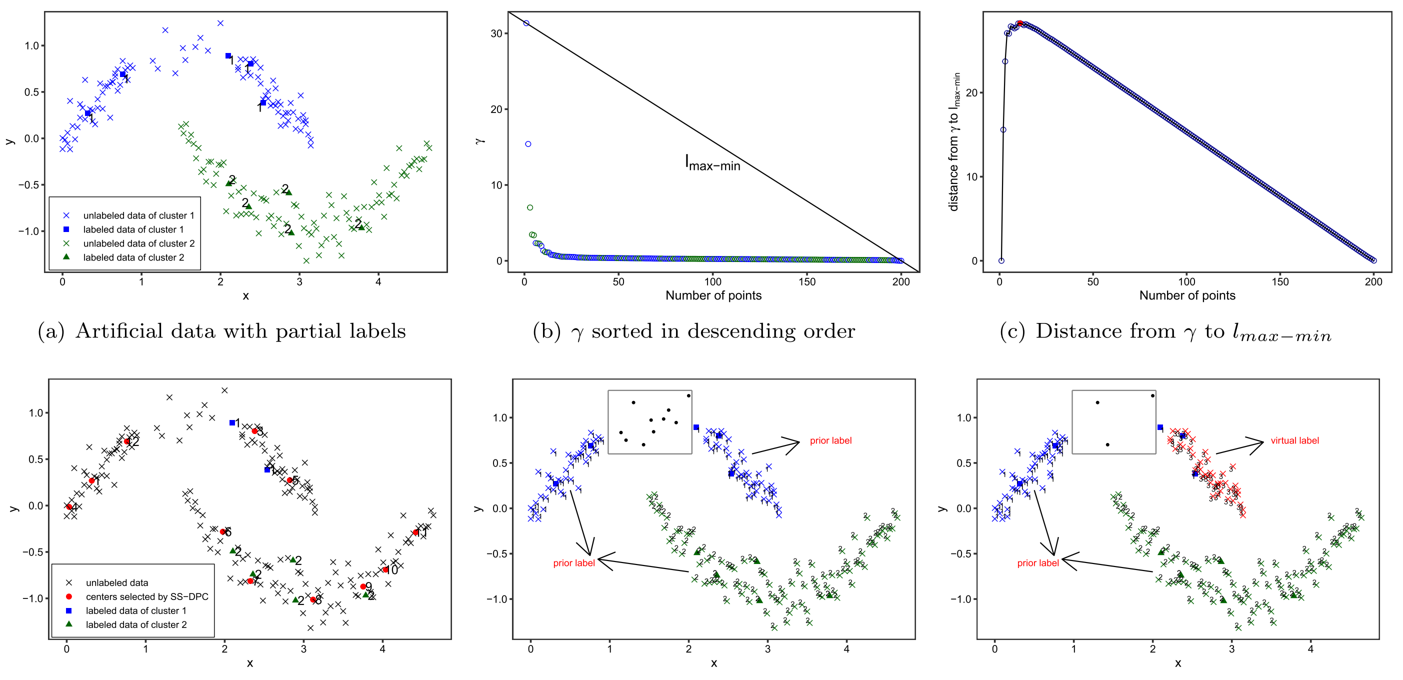
- The identified centers that belong to labeled dataset and not share labels with other ones keep their prior labels. That is,

$$\begin{aligned} &\boldsymbol{c}_i \in \mathcal{L}, \forall \boldsymbol{c}_j \in \mathcal{L}, label(\boldsymbol{c}_i) \neq label(\boldsymbol{c}_j) \\ &\Rightarrow \quad keep\ \boldsymbol{c}_i\ the\ prior\ label. \end{aligned} \tag{6}$$

- If more than one identified centers have the same prior label, then only one of them keeps the prior label, whereas the others are assigned to different virtual labels. That is,

$$\begin{aligned} &\boldsymbol{c}_i \in \mathcal{L}, \exists \boldsymbol{c}_j \in \mathcal{L}, s.t.\ label(\boldsymbol{c}_i) = label(\boldsymbol{c}_j) \\ &\Rightarrow \begin{cases} keep\ \boldsymbol{c}_i\ the\ prior\ label \\ remark\ \boldsymbol{c}_j\ with\ a\ virtual\ label. \end{cases} \end{aligned} \tag{7}$$

In clustering, each identified center represents a possible cluster. Two or more centers with the same prior label mean the cluster may consist of two or more sub-clusters. In this case, virtual labels are adopted to distinguish the centers of identical prior labels, thereby marking the possible new sub-clusters splitting from



(a) Artificial data with partial labels (b) γ sorted in descending order (c) Distance from γ to $l_{max-min}$

(d) Identified centers (e) Mergence result(two clusters) (f) Mergence result(three clusters)

Figure 1. Example of the proposed semi-supervised clustering algorithm (SS-DPC) on artificial dataset.

prior observed clusters. Here, we use $\boldsymbol{V}_{prior}$ to collect these virtual labels. Obviously, for each virtual label i in $\boldsymbol{V}_{prior}$, there exists a corresponding prior label in $\boldsymbol{l}_{prior}$ which is the true label of cluster i, denoted by $prior(i)$. To ensuring the correctness of prior information, clusters marked by virtual labels coming from $\boldsymbol{V}_{prior}$ are merged only with their corresponding prior ones in the following cluster merging stage.

- The identified centers that belong to the unlabeled dataset are assigned to different virtual labels directly. That is,

$$\boldsymbol{c}_i \in \mathcal{U} \Rightarrow assign\ \boldsymbol{c}_i\ a\ virtual\ label. \tag{8}$$

In this case, virtual labels are adopted to label the centers of no prior labels, thereby marking the possible new clusters that have not been prior observed. Here, we use $\boldsymbol{V}_{new}$ to collect these virtual labels.

In this way, SS-DPC integrates the partial information and identified centers as well as assigns each center a unique label.

Finally, similarly to DPC, SS-DPC determines the label of each non-center unlabeled data according to its corresponded nearest point. Let $\boldsymbol{\rho}_{order}^{\mathcal{U}} = \{\rho_{(1)} \geq \rho_{(2)} \geq ... \geq \rho_{(n_1)}\}$ be the ρ values of unlabeled data sorted in descending order, $\mathcal{U}_{order} = \{\boldsymbol{x}_{(1)}, \boldsymbol{x}_{(2)}, ..., \boldsymbol{x}_{(n_1)}\}$ be the corresponding points of $\boldsymbol{\rho}_{order}^{\mathcal{U}}$ with the corresponding nearest points $\boldsymbol{nearest}_{order}^{\mathcal{U}} = \{\boldsymbol{nearest}_{(1)}, \boldsymbol{nearest}_{(2)}, ..., \boldsymbol{nearest}_{(n_1)}\}$, where n_1 is the number of unlabeled data. Each non-center unlabeled data is assigned the same label as its nearest point. That is,

$$\begin{aligned} &\boldsymbol{x}_{(i)} \in \mathcal{U}_{order}, \boldsymbol{x}_{(i)} \notin \mathcal{C} \\ &\Rightarrow \quad label(\boldsymbol{x}_{(i)}) = label(\boldsymbol{nearest}_{(i)}). \end{aligned} \tag{9}$$

Thus, a density peaks based semi-supervised clustering process is constructed by using labeled and unlabeled data. In this way, the labels of all the unlabeled data can be initially assigned successfully. Once an unlabeled data belongs to the center set or two or more centers possess the same prior label, a new cluster may appear.

Figure 1 shows an example of an artificial dataset that illustrates how the proposed algorithm works. Figure 1(a) shows the artificial data with 5% of data labeled, numbers 1 and 2 indicate the prior classes of labeled data. Figure 1(d) presents the possible centers identified by Algorithm 1, which are marked by • in red. In Figure 1(d), the points marked by 1, 3, and 12 are identified centers, whereas they are all prior labeled by 1 in Figure 1(a). These three identified centers share the same prior label, so in the semi-supervised clustering construction stage, only the center marked by 1 keeps the prior label, and the two other labels are re-labeled by new virtual labels 3 and 12. Besides, as shown in Figure 1(d), the identified centers that are not previously labeled are assigned with different virtual labels, and the non-center labeled data keeps their prior class labels unchanged. Subsequently, by assigning each unlabeled point to its nearest point, the artificial data may be separated into several small sub-clusters.

2.2.3 Mergence of clusters

Based on Section 2.2.2, the new labels may appear, and the new clusters may divide an essentially huge cluster into several small clusters. Thus combining the clusters further is necessary. To this end, this study merges the clusters step-by-step according to the following strategy:

- Two clusters with the minimum distance are merged in each mergence. Let $\mathcal{C}_m$ is the set of points that belong to the cluster m, then, $\mathcal{C}_i$ and $\mathcal{C}_j$ are the clusters to be merged in the current mergence, if

$$(i,j) = \arg\min_{i,j} dist(\mathcal{C}_i, \mathcal{C}_j). \tag{10}$$

- Two clusters with different prior labels cannot be merged into one cluster, and only the cluster with a virtual label can be possibly merged with its nearest one. Specifically, let $\mathcal{C}_i$ and $\mathcal{C}_j$ are two pre-merged clusters, then

$$i \in \boldsymbol{l}_{prior}, j \in \boldsymbol{l}_{prior} \Rightarrow not\ merge\ \mathcal{C}_i\ and\ \mathcal{C}_j; \tag{11}$$

$$i \in \boldsymbol{V}_{new}, j \in \boldsymbol{V}_{new} \Rightarrow merge\ \mathcal{C}_i\ and\ \mathcal{C}_j; \tag{12}$$

$$i \in \boldsymbol{V}_{new}, j \in \boldsymbol{V}_{prior} \Rightarrow merge\ \mathcal{C}_i\ to\ \mathcal{C}_j; \tag{13}$$

$$i \in \boldsymbol{V}_{new}, j \in \boldsymbol{l}_{prior} \Rightarrow merge\ \mathcal{C}_i\ to\ \mathcal{C}_j; \tag{14}$$

$$\begin{aligned} & i \in \boldsymbol{V}_{prior}, j \in \boldsymbol{V}_{prior} \\ & \Rightarrow \begin{cases} merge\ \mathcal{C}_i\ and\ \mathcal{C}_j & prior(i) = prior(j), \\ not\ merge & otherwise; \end{cases} \end{aligned} \tag{15}$$

$$\begin{aligned} & i \in \boldsymbol{V}_{prior}, j \in \boldsymbol{l}_{prior} \\ & \Rightarrow \begin{cases} merge\ \mathcal{C}_i\ to\ \mathcal{C}_j & if\ prior(i) = j, \\ not\ merge & otherwise. \end{cases} \end{aligned} \tag{16}$$

- The procedure stops when no cluster can be merged.

This strategy guarantees that the prior class information is unchanged throughout the mergence process while maintaining the correctness of prior information about data labels.

For mergence purposes, the distance between any two clusters is calculated. However, the noise points with relatively lower density in each pre-merged cluster may affect distance. To address this problem, we first calculate each point the density in the pre-merged cluster it belongs to, according to Gaussian kernel function,

$$\rho_i^{\mathcal{C}_m} = \sum_{\boldsymbol{x}_i, \boldsymbol{x}_j \in \mathcal{C}_m, j \neq i} \exp\left(-\frac{d_{ij}^2}{2\sigma^2}\right), \quad m = 1, 2, ..., k. \tag{17}$$

Then, for each pre-merged cluster, the points whose density is lower than a threshold ρ_{thre} are recognized as noises. By removing these noise points, only the points whose density is higher than ρ_{thre} are kept for distance measurement. In SS-DPC, the distance of two clusters is measured in terms of three methods [27]: *min*, *max*, and *average*, which are defined as follows,

$$min(\mathcal{C}_i, \mathcal{C}_j) = \min_{\boldsymbol{x}_i \in \mathcal{C}_i, \boldsymbol{x}_j \in \mathcal{C}_j} d_{ij}, \tag{18}$$

$$max(\mathcal{C}_i, \mathcal{C}_j) = \max_{\boldsymbol{x}_i \in \mathcal{C}_i, \boldsymbol{x}_j \in \mathcal{C}_j} d_{ij}, \tag{19}$$

$$average(\mathcal{C}_i, \mathcal{C}_j) = \frac{1}{|\mathcal{C}_i|\,|\mathcal{C}_j|} \sum_{\boldsymbol{x}_i \in \mathcal{C}_i, \boldsymbol{x}_j \in \mathcal{C}_j} d_{ij}, \tag{20}$$

where $\mathcal{C}_i$ is the set of points that belong to the ith cluster and $|\mathcal{C}_i|$ is the number of points in the ith cluster.

Figures 1(e) and 1(f) illustrate the mergence results of the artificial data. The data are categorized into two (Figure 1(e)) or three clusters (Figure 1(f)) depending mainly on the sparsity of the data in the rectangular box. The more sparse the data in the rectangular box is, the more likely the cluster is divided into two subgroups and vice versa. Furthermore, these two possible results are reasonable in practice. For example, suppose that cluster 1 and cluster 2 represent disease A and disease B, respectively, and disease A can been further divided into two subtypes. Then, the clustering results in Figure 1(f) are beneficial to further research on disease A, whereas the integration of the two subtypes shown in Figure 1(e) is also reasonable. Thus, alternative results are provided to be analyzed.

To sum up, SS-DPC iteratively merges the two closest ones among the current pre-merged clusters together until the mergence conflicts with the prior class labels. To maintain the correctness of partial information, SS-DPC never merges two clusters of different prior labels throughout the merging processes. Consequently, the merging rule is discussed in the case that at least one of the two pre-merged clusters is marked by a virtual label. Each virtual label contained in V_{prior} is returned back to its corresponding true prior one in each mergence and the cluster marked by such a virtual label is merged only with the one that has the same prior label. This is natural because these virtual labels are raised by distinguishing the centers of identical prior labels. Besides, the cluster marked by a virtual label contained in V_{new} is merged directly to its nearest one because these virtual labels have no corresponding prior labels. From the step-by-step mergence strategy, it can be seen that each mergence in fact corresponds to a possible partition result that has the cluster numbers of no less than the prior class numbers. Hence, SS-DPC comes to an alternative clustering result and it is easy to obtain a required clustering result, especially in the case that all the underlying clusters are observed or the number of clusters is estimated.

It is worth noting that the mergence strategy of SS-DPC is restricted to stop when only the clusters of prior labels remain. This leads to an issue of lacking stop conditions to explore new possible clusters, especially in the case that only partial underlying clusters are observed and the number of clusters is difficult to estimate. A possible method for this issue is to set a distance threshold among clusters as another stop condition of mergence process. Once a proper distance

Algorithm 2 Semi-supervised density peak clustering algorithm (SS-DPC)

```
Input:
$\mathcal{X} = \mathcal{L} \cup \mathcal{U}$: $\mathcal{L} = \cup_{i=1}^{n_0} \boldsymbol{x}_i$ is labeled dataset, $\mathcal{U} = \cup_{i=1}^{n_1} \boldsymbol{x}_i$ is unlabeled dataset, $n_0 + n_1 = n$;
$\boldsymbol{label}_L$: the labels of labeled data; $d_c$: the cutoff distance;
$\boldsymbol{l}_{prior}$: the set of prior labels; $\boldsymbol{V}_{prior}$ = NULL; $\boldsymbol{V}_{new}$ = NULL
Output: The label vector: $\boldsymbol{label}_{SC}$.

Algorithm
Step1: Identification of cluster centers and nearest points
  1: Identify the possible cluster centers $\mathcal{C}$ and the nearest points $\boldsymbol{nearest}$ according to Algorithm 1;
Step2: Construction of semi-supervised clustering
  1: Initialize the label vector $\boldsymbol{label}_{SC}$ by $\boldsymbol{label}_L$;
  2: Assign labels to centers $\mathcal{C} = \{\boldsymbol{c}_1, \boldsymbol{c}_2, ..., \boldsymbol{c}_k\}$;
     If $\boldsymbol{c}_i \in \mathcal{U}$, assign $\boldsymbol{c}_i$ a new virtual label, update $\boldsymbol{V}_{new}$;
     If $\boldsymbol{c}_i \in \mathcal{L}$ and $\forall j \neq i, \boldsymbol{c}_j \in \mathcal{L}, label(\boldsymbol{c}_j) \neq label(\boldsymbol{c}_i)$, keeps $\boldsymbol{c}_i$ the prior label;
     If $\boldsymbol{c}_i \in \mathcal{L}$ and $\exists j \neq i, \boldsymbol{c}_j \in \mathcal{L}, label(\boldsymbol{c}_j) = label(\boldsymbol{c}_i)$, keeps $\boldsymbol{c}_i$ the prior label and assign $\boldsymbol{c}_j$ a new virtual label, update $\boldsymbol{V}_{prior}$.
  3: Update the labeled dataset $\mathcal{L} \leftarrow \mathcal{L} \cup \mathcal{C}$, $n_0 = |\mathcal{L}|$;
  4: Update the unlabeled dataset $\mathcal{U} \leftarrow \mathcal{X} - \mathcal{L}$, $n_1 = |\mathcal{U}|$;
  5: Update the label vector $\boldsymbol{label}_{SC}$;
  6: Calculate $\rho_i$ for $\forall \boldsymbol{x}_i \in \mathcal{U}$ according to Equation (2), obtaining $\boldsymbol{\rho}_{\boldsymbol{\mathcal{U}}} = \{\rho_1, ..., \rho_{n_1}\}$;
  7: Sort $\boldsymbol{\rho}_{\boldsymbol{\mathcal{U}}}$ in descending order to yield $\boldsymbol{\rho}_{order}^{\mathcal{U}}$, $\mathcal{U}_{order}$, and $\boldsymbol{nearest}_{order}^{\mathcal{U}}$;
  8: For each $\boldsymbol{x}_{(i)} \in \mathcal{U}_{order}$, assign $\boldsymbol{x}_{(i)}$ a same label as $\boldsymbol{nearest}_{(i)}$;
  9: Update the label vector $\boldsymbol{label}_{SC}$;
Step3: Mergence of clusters
  Repeat until only clusters with prior labels are keep
  1: For $\forall \boldsymbol{x}_i \in \mathcal{C}_m$, calculate $\rho_i^{\mathcal{C}_m}$ by Equation (17)
     If $\rho_i^{\mathcal{C}_m} < \rho_{thre}$, remove $\boldsymbol{x}_i$ from $\mathcal{C}_m$, and update $\mathcal{C}_m \leftarrow \mathcal{C}_m \backslash \boldsymbol{x}_i$;
  2: Calculate distance between two clusters by Equation (18), (19) or (20);
  3: $\mathcal{C}_i$ and $\mathcal{C}_j$ are two clusters to be merged, if $(i,j) = \arg\min_{i,j} dist(\mathcal{C}_i, \mathcal{C}_j)$;
  4: Merge $\mathcal{C}_i$ and $\mathcal{C}_j$ according to the following criteria:
     If $i \in \boldsymbol{l}_{prior}, j \in \boldsymbol{l}_{prior}$, not merge $\mathcal{C}_i$ and $\mathcal{C}_j$;
     If $i \in \boldsymbol{V}_{new}, j \in \boldsymbol{V}_{new}$, merge $\mathcal{C}_i$ and $\mathcal{C}_j$, update $\mathcal{C}_i \leftarrow \mathcal{C}_i \cup \mathcal{C}_j$;
     If $i \in \boldsymbol{V}_{new}, j \in \boldsymbol{V}_{prior}$, merge $\mathcal{C}_i$ to $\mathcal{C}_j$, update $\mathcal{C}_j \leftarrow \mathcal{C}_j \cup \mathcal{C}_i$;
     If $i \in \boldsymbol{V}_{new}, j \in \boldsymbol{l}_{prior}$, merge $\mathcal{C}_i$ to $\mathcal{C}_j$, update $\mathcal{C}_j \leftarrow \mathcal{C}_j \cup \mathcal{C}_i$;
     If $i \in \boldsymbol{V}_{prior}, j \in \boldsymbol{V}_{prior}$ and $prior(i) = prior(j)$, merge $\mathcal{C}_i$ and $\mathcal{C}_j$, update $\mathcal{C}_i \leftarrow \mathcal{C}_i \cup \mathcal{C}_j$;
     If $i \in \boldsymbol{V}_{prior}, j \in \boldsymbol{V}_{prior}$ and $prior(i) \neq prior(j)$, not merge $\mathcal{C}_i$ and $\mathcal{C}_j$;
     If $i \in \boldsymbol{V}_{prior}, j \in \boldsymbol{l}_{prior}$ and $prior(i) = j$, merge $\mathcal{C}_i$ to $\mathcal{C}_j$, update $\mathcal{C}_j \leftarrow \mathcal{C}_j \cup \mathcal{C}_i$;
     If $i \in \boldsymbol{V}_{prior}, j \in \boldsymbol{l}_{prior}$ and $prior(i) \neq j$, not merge $\mathcal{C}_i$ and $\mathcal{C}_j$.
  5: Update the label vector $\boldsymbol{label}_{SC}$;
return $\boldsymbol{label}_{SC}$;
```

threshold is found, SS-DPC can yield possible partition results without estimating the number of clusters. But this may raise another problem that how to find an optimal distance threshold. This may be another concern for further work and would not be discussed in this paper.

The implementation details of the proposed SS-DPC is summarized in Algorithm 2.

3. EXPERIMENTS

In this section, we conduct several experiments to verify the performance of the proposed SS-DPC (Algorithm 2) based on the dataset from UCI repository. SS-DPC is compared with the other semi-supervised and unsupervised clustering algorithms. The experiments are implemented with R.

3.1 Data sets

In this paper, eight commonly used UCI datasets are utilized to carry out the experiments for clustering performance testing. These benchmark datasets include Iris, Sonar, Ionosphere, Parkinsons, Letter Recognition, Image Segmentation, Libras Movement, and Vehicle Silhouettes. They can be obtained from the UCI Machine Learning Repository, available on http://archive.ics.uci.edu/ml. As listed in Table 1, these datasets have various sizes, dimensions, densities, and

Table 1. The details of eight UCI datasets

Dataset	**No. of Instances**	**No. of Attributes**	**No. of Classes**	**Class distribution (%)**
Letter Recognition*	500	16	5	20 in each
Iris	150	4	3	33.3 in each
Ionosphere	351	34	2	35.9/64.1
Image Segmentation	210	19	7	14.3 in each
Libras Movement	360	90	15	6.7 in each
Parkinsons	195	22	2	24.6/75.4
Sonar	208	60	2	53.4/46.6
Vehicle Silhouettes	846	18	4	25.8/25.1/25.7/23.5

* The dataset used in this paper is the subset of the original Letter Recognition dataset, obtained by selecting the first 100 samples from letters "A", "B", "C", "D", "E", respectively.

number of clusters, and are introduced as follows.

- Iris: The Iris dataset, which consists of 150 instances with 4 attributes, is a popular dataset for clustering analysis in the literatures. The dataset is categorized into three classes of the same sizes, and each class refers to a type of iris plant.
- Sonar: The Sonar dataset is formed by 208 sonar signal records with 60 attributes distributed in 2 classes of sizes 111 and 97. Each record corresponds to a sonar signal bounced off a metal cylinder or bounced off a roughly cylindrical rock.
- Ionosphere: The Ionosphere dataset contains 351 radar returns from the ionosphere with 34 attributes. These radar returns belong to 2 classes of sizes 126 and 225, where the classes represent the "Good" and "Bad" radar returns, respectively.
- Parkinsons: The Parkinsons dataset is composed of 195 voice recordings with 22 attributes, where each recoding describes a biomedical voice measurement from people with Parkinson's disease or healthy. The dataset contains 2 classes of sizes 48 and 147.
- Libras Movement: The Libras Movement dataset contains 360 instances with 90 attributes. These instances are labeled as 15 classes of the same size, and each class refers to a hand movement type in Libras.
- Letter Recognition: The original Letter Recognition dataset consists of 20000 instances with 16 attributes. Each instance is converted from a character image that describes one of the 26 capital letters in the English alphabet. In this paper, we choose respectively the first 100 samples of five letters, "A", "B", "C", "D", "E", for experiments.
- Image Segmentation: The Image Segmentation dataset is composed of 210 instances with 19 attributes and contains 7 classes of the same size. The instances are taken randomly from a database of seven outdoor images, including the brickface, sky, foliage, cement, window, path, and grass. The images are hand-segmented to create a classification for every pixel.
- Vehicle Silhouettes: The Vehicle Silhouettes dataset consists of 846 instances with 18 attributes distributed in 4 classes of different sizes. The attributes are extracted from the silhouettes of four vehicles: a double-decker bus, Chevrolet van, Saab 9000, and an Opel Manta 400.

All the datasets are preprocessed by the min-max normalization in advance. For each original dataset, min-max normalization transforms the value of each attribute into [0,1].

3.2 Algorithms for comparison

SS-DPC proposed a semi-supervised clustering that combines labeled and unlabeled data. To verify the effectiveness of SS-DPC in partial information incorporation, two unsupervised clustering algorithms (DPC [26] and K-means [21]) and four semi-supervised clustering algorithms (Constrained-Kmeans [1], CCLS [13], MPCK-Mean [2], and LCVQE [24]) are used for the comparison. The referred algorithms are introduced as follows.

- DPC [26]: As mentioned in Section 2.1, DPC is a density-based unsupervised clustering algorithm with fast search, based on which the SS-DPC in this paper is proposed.
- K-means [21]: K-means is a classical partition-based unsupervised clustering algorithm.
- Constrained-KMeans [1]: A semi-supervised variant of K-means using labeled data for seeding. Specifically, the labeled data are adopted to initialize K-means, and the cluster labels of the labeled data are not re-computed in the whole algorithm.
- CCLS [13]: Constrained clustering by local search (CCLS) is a pairwise constraint-based semi-supervised clustering algorithm that combines Tabu search, a variation of the local search algorithm, to minimize the objective function for clustering.
- MPCK-Means [2]: Metric pairwise constrained K-Means (MPCK-Means) is a hybrid semi-supervised clustering algorithm that integrates the use of constraints and metric learning.
- LCVQE [24]: Linear-time constrained vector quantization error (LCVQE) algorithm is a pairwise constraint-based

semi-supervised clustering that minimizes a target function composed of the vector quantization error as well as a penalty for violated constraints.

3.3 Evaluation metrics

To evaluate the performance of the proposed semi-supervised clustering algorithm, three common clustering evaluation metrics: *Clustering Accuracy* (ACC) [8], *Rand Index* (RI) [5, 7], and *Normalized Mutual Information* (NMI) [38], are used to compare the generated class labels with respect to the true labels.

Let $\boldsymbol{Y} = \{y_1, y_2, ..., y_K\}$ and $\boldsymbol{C} = \{c_1, c_2, ..., c_{K'}\}$ be the true and predicted labels of the dataset $\mathcal{X}$ with n points, respectively. K and K' are the number of clusters in $\boldsymbol{Y}$ and $\boldsymbol{C}$, respectively; n_{ij} is the number of common points of cluster y_i and c_j; n_i^y is the number of data points in cluster y_i; and n_j^c is the number of data points in cluster c_j. Then, the three cluster evaluation methods can be defined as follows.

- *Clustering Accuracy*(ACC)

$$\text{ACC} = \sum_{\text{i}=1}^{\text{n}} \delta(\text{y}_\text{i}, \text{map}(\text{c}_\text{i}))/\text{n}, \tag{21}$$

where $map(\cdot)$ maps each predicted cluster label into a true label by the Hungarian algorithm [23], and this mapping is optimal. $\delta(y_i, c_i) = 1$ if $y_i = c_i$. The higher the value of ACC is, the better the clustering performance will be.

- *Rand Index*(RI)

$$\text{RI} = \frac{\#\text{correct decision}}{\#\text{total decision}} = \frac{\text{a}+\text{b}}{\text{n}(\text{n}-1)/2}, \tag{22}$$

where # means "*the number of*". The number of total decision means that every partition has C_n^2 pairwise decisions. a denotes the number of decisions that the pairwise points $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ are assigned to the same cluster in $\boldsymbol{Y}$ and $\boldsymbol{C}$, and b is the number of consistence decisions that the two points belongs to different clusters in $\boldsymbol{Y}$ are also partitioned into the different clusters in $\boldsymbol{C}$, then $a+b$ is the number of correct decision. The value of RI is between 0 and 1, the greater RI value means the better performance of cluster algorithm.

- *Normalized Mutual Information*(NMI)

$$\text{NMI} = \frac{\sum_{\text{i}=1}^{\text{K}} \sum_{\text{j}=1}^{\text{K}'} \text{n}_{\text{ij}} \log \frac{\text{n}\cdot\text{n}_{\text{ij}}}{\text{n}_\text{i}^\text{y}\cdot\text{n}_\text{j}^\text{c}}}{\sqrt{\sum_{\text{i}=1}^{\text{K}} \text{n}_\text{i}^\text{y} \log\frac{\text{n}_\text{i}^\text{y}}{\text{n}} \sum_{\text{j}=1}^{\text{K}'} \text{n}_\text{j}^\text{c} \log\frac{\text{n}_\text{j}^\text{c}}{\text{n}}}}, \tag{23}$$

where it is easy to know that the value of NMI ranges from 0 to 1, and NMI = 1 means the two partitions of data set $\mathcal{X}$ are identical. Also, the higher the NMI value is, the closer the clustering result to the true cluster distribution is.

3.4 Experimental setting

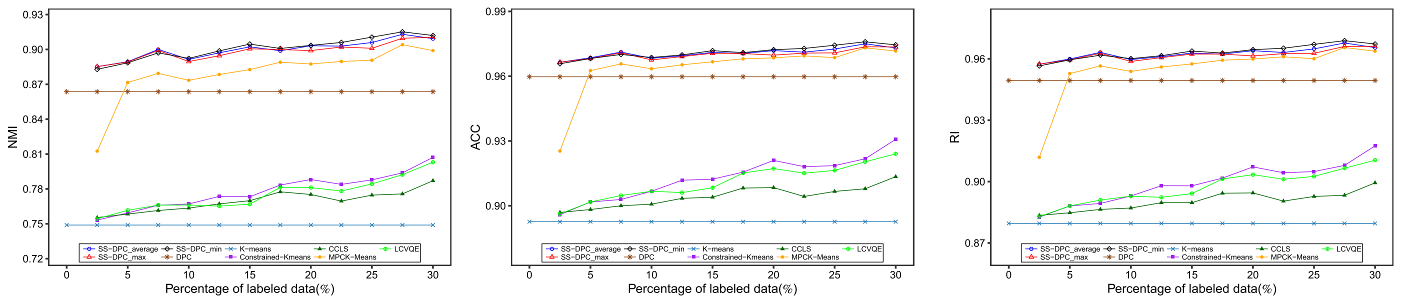
In the experimental stage, the original dataset $\mathcal{X}$ is randomly divided into two parts: labeled dataset $\mathcal{L}$ and unlabeled dataset $\mathcal{U}$. Given that semi-supervised clustering focuses only on the datasets with few labels, this paper designs the percentage of labeled data in each dataset varying from 0 to 30% with an increasing step size of 2.5%. Where 0 corresponds to the particular unsupervised case. The labeled dataset $\mathcal{L}$ consists of the points selected randomly from each class in proportion. For a fixed percentage of labeled data, each clustering procedure is implemented on 20 sampled datasets of differently labeled data to obtain a more stable average result. The three evaluation metrics, namely, NMI, RI, and ACC, are calculated only based on the unlabeled data without considering the labeled data. Note that DPC and K-mean are unsupervised algorithms, so their results begin with labeling 0% of data and then do not change as the percentage of labeled data increases.

In SS-DPC, the cutoff distance d_c must be selected for local density estimation. This paper sets d_c for each dataset as 0.5%, 1%, 1.5%, 2%, 2.5%, 3%, 3.5%, 4%, 4.5%, 5%, 10%, 15%, 20%, 25% and 30% of all the distance between every two points sorted in descending order. Moreover, d_c with the highest NMI, RI and ACC values in each experiment is chosen to be the optimal one. Also, the selection of d_c for DPC is the same as that in SS-DPC. The threshold parameter ρ_{thre} mentioned in Section 2.2.3 is set as $mean(\rho^{\mathcal{C}_m}) - 2sd(\rho^{\mathcal{C}_m})$ for each pre-merged cluster $\mathcal{C}_m$ adaptively, and the bandwidth parameter of Equation (17) is set as $\sigma = 1/\sqrt{2}$ for each dataset.

SS-DPC provides optional results in the clusters merging step, and users can take a particular one according to requirements. The mergence strategy, which is set to not merge two clusters with different prior labels and only merge two clusters with at least one virtual label, remains the clusters with prior labels. That is, the number of clusters is no less than the prior class number after mergence. Furthermore, the new clusters that appeared in the mergence stage refer to the possible partition result rather than a wrong one. Therefore, for convenient comparison, the datasets with the known classes are used in our experiments, and the virtual labels raised by splitting clusters are merged back to the true one when evaluating the clustering performance. Meanwhile, two datasets, namely, Image Segmentation and Vehicle Silhouettes, are used to compare the results on the estimated number of clusters.

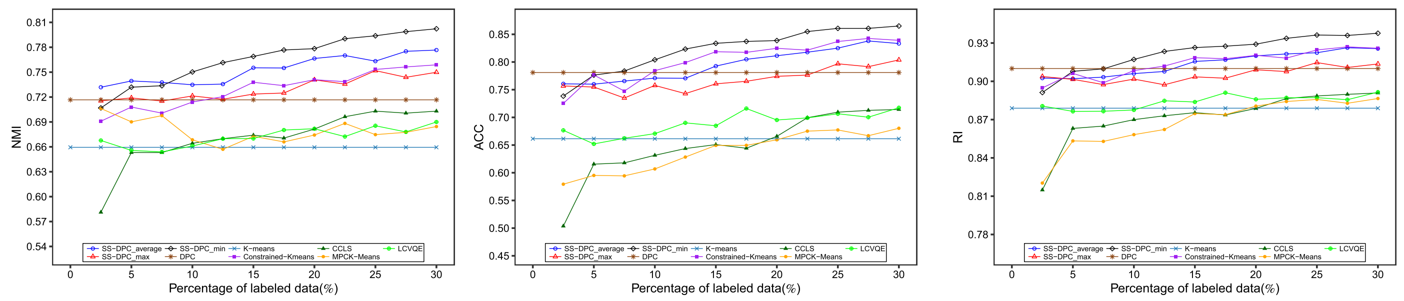
Furthermore, the pairwise constraints, including must-link and cannot-link constraints in the compared algorithms, are derived from the labeled dataset, in which must-link constraints consist of the pairs of points in the same clusters, whereas cannot-link constraints consist of the pairs of points that belong to the different ones.

The proposed semi-supervised clustering algorithm using Equations (18), (19), and (20) to measure the distance between two clusters in the mergence stage are denoted as



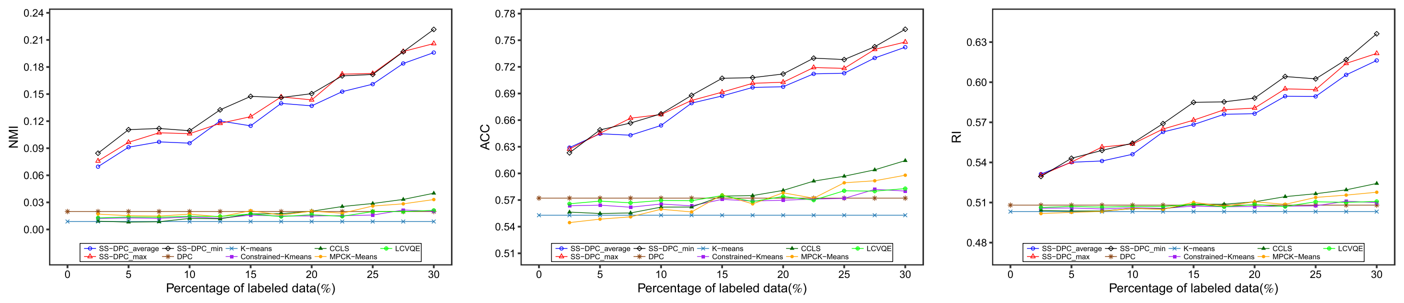
(a) NMI (b) ACC (c) RI

Figure 2. Evaluations on Iris with true class numbers (k=3).



(a) NMI (b) ACC (c) RI

Figure 3. Evaluations on Image Segmentation with true class numbers (k=7).



(a) NMI (b) ACC (c) RI

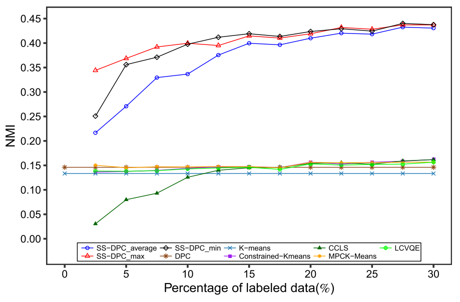
Figure 4. Evaluations on Sonar with true class numbers (k=2).

SS-DPC_min, SS-DPC_max, and SS-DPC_average, respectively.

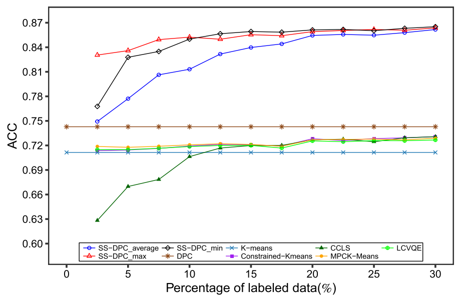
3.5 Results analysis

Figures 2 - 9 illustrate the experiment results of SS-DPC and the compared algorithms on the eight UCI datasets under the true class numbers. The results are compared in terms of evaluation metrics NMI, ACC, and RI over different percentages of labeled data. It can be observed from those figures that SS-DPC often achieve better clustering performance than the compared algorithms. Specifically, all the three SS-DPC-based algorithms outperform other ones on Iris, Sonar, Ionosphere, and Vehicle Silhouettes all the time as the labeled data increases. SS-DPC_min performs best on Letter Recognition according to NMI, ACC, and RI. SS-DPC_average and SS-DPC_min perform best on Parkinson's and Image Segmentation, respectively, according to NMI. The success of SS-DPC on these datasets also demonstrates that SS-DPC keeps the advantages of DPC of detecting clusters of various sizes, arbitrary shapes, and varying densities.

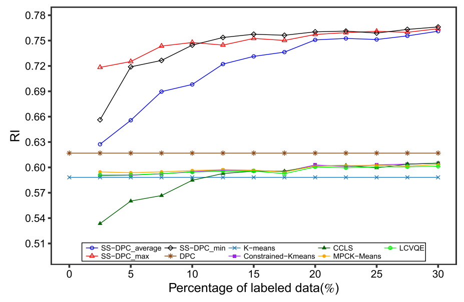
Compared with DPC, SS-DPC has the ability to make use of partial information for better data partition results. This is illustrated by the gap between the plotted results of



(a) NMI

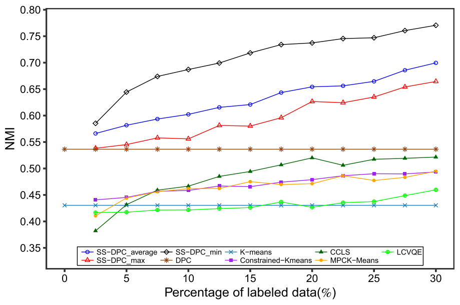


(b) ACC

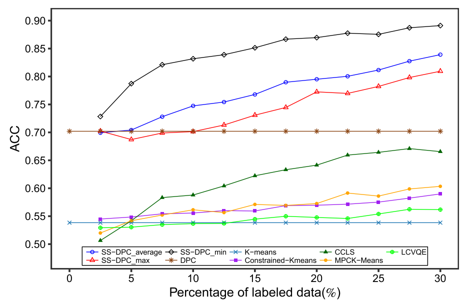


(c) RI

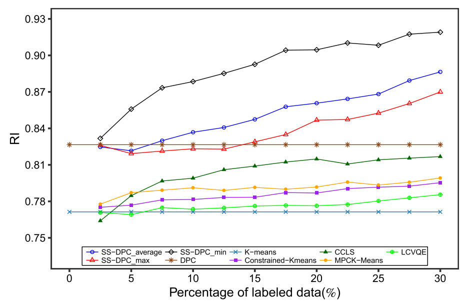
Figure 5. Evaluations on Ionosphere with true class numbers (k=2).



(a) NMI

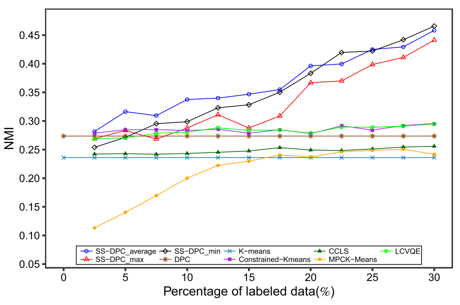


(b) ACC

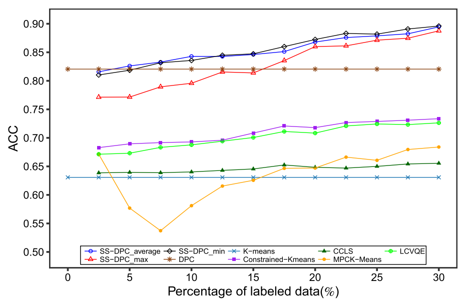


(c) RI

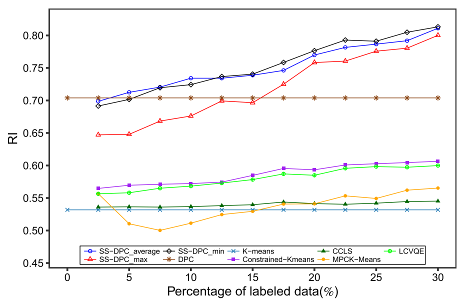
Figure 6. Evaluations on Letter Recognition with true class numbers (k=5).



(a) NMI

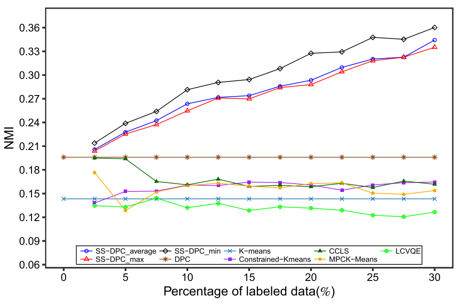


(b) ACC

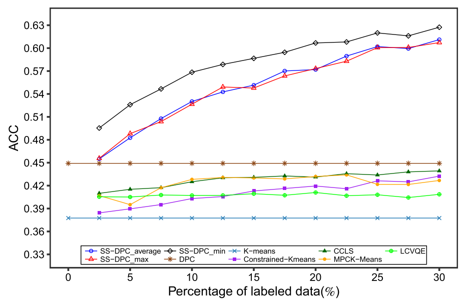


(c) RI

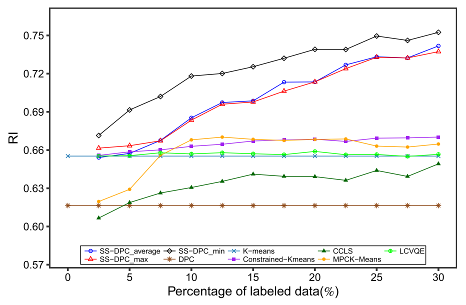
Figure 7. Evaluations on Parkinsons with true class numbers (k=2).



(a) NMI

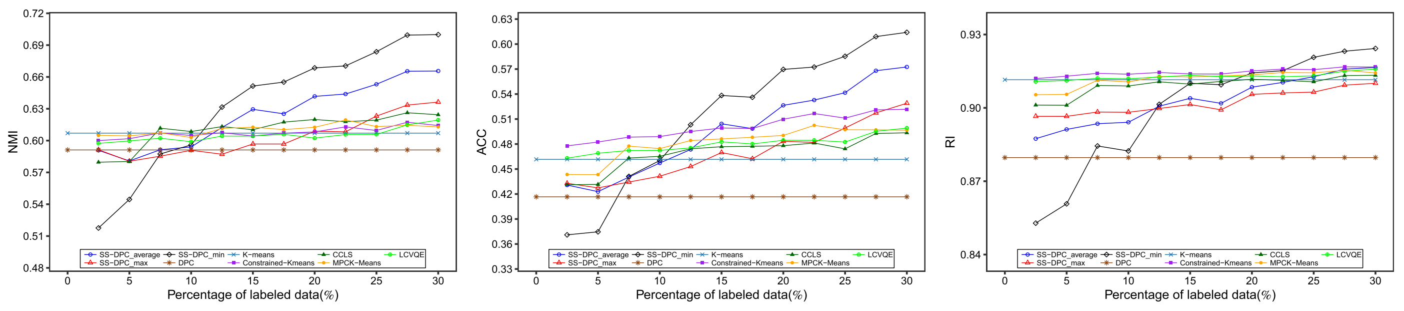


(b) ACC



(c) RI

Figure 8. Evaluations on Vehicle Silhouettes with true class numbers (k=4).



(a) NMI (b) ACC (c) RI

Figure 9. Evaluations on Libras Movement with true class numbers (k=15).

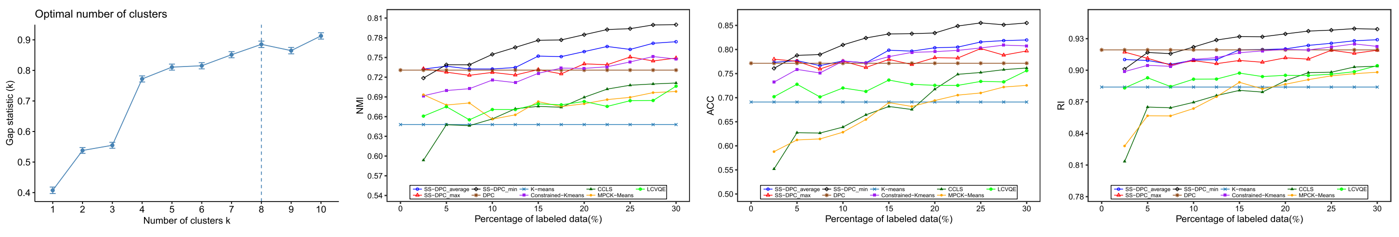
Table 2. Clustering results comparison between SS-DPC and DPC on Vehicle Silhouettes dataset. The value in parentheses indicates the improvement percentage of SS-DPC over DPC.

Labeled data	**Metrics**	**DPC**	**SS-DPC_ave**	**SS-DPC_max**	**SS-DPC_min**
	ACC	0.4492	0.4826(7.44%)	0.4880(8.64%)	0.5260(17.10%)
5%	NMI	0.1960	0.2277(16.17%)	0.2253(14.95%)	0.2388(21.84%)
	RI	0.6164	0.6574(6.65%)	0.6634(7.62%)	0.6915(12.18%)
	ACC	0.4492	0.5302(18.03%)	0.5266(17.23%)	0.5685(26.56%)
10%	NMI	0.1960	0.2636(34.49%)	0.2547(29.95%)	0.2814(43.57%)
	RI	0.6164	0.6853(11.18%)	0.6835(10.89%)	0.7181(16.50%)
	ACC	0.4492	0.5515(22.77%)	0.5478(21.95%)	0.5866(30.59%)
15%	NMI	0.1960	0.2739(39.74%)	0.2699 (37.70%)	0.2943(50.15%)
	RI	0.6164	0.6986(13.34%)	0.6977(13.19%)	0.7254(17.68%)
	ACC	0.4492	0.5719(27.32%)	0.5733(27.63%)	0.6068(35.08%)
20%	NMI	0.1960	0.2934(49.69%)	0.2879(46.89%)	0.3275(67.09%)
	RI	0.6164	0.7135(15.75%)	0.7135 (15.75%)	0.7391(19.91%)
	ACC	0.4492	0.6020(34.02%)	0.6006(33.70%)	0.6199 (38.00%)
25%	NMI	0.1960	0.3202(63.37%)	0.3181(62.30%)	0.3477(77.40%)
	RI	0.6164	0.7333(18.96%)	0.7328(18.88%)	0.7495(21.59%)
	ACC	0.4492	0.6111(36.04%)	0.6073(35.20%)	0.6274(39.67%)
30%	NMI	0.1960	0.3443(75.66%)	0.3352(71.02%)	0.3602(83.78%)
	RI	0.6164	0.7418(20.34%)	0.7373(19.61%)	0.7524(22.06%)

SS-DPC and DPC for five datasets (Iris, Sonar, Ionosphere, Letter RecognitionIris, Vehicle Silhouettes). On Parkinson, evaluated by ACC and RI, though SS-DPC_average and SS-DPC_min perform slightly worse than DPC when the percentage of labeled data is 2.5%, their performance improves as the percentage of labeled data increases and then are significantly better than that of DPC when the percentage is no less than 7.5%. Meanwhile, on Image Segmentation and Libras Movement when respectively labeling no less than 10% and 7.5% of data, SS-DPC_min shows similar results. These results also prove that SS-DPC can effectively incorporate the partial information to improve the performance of DPC.

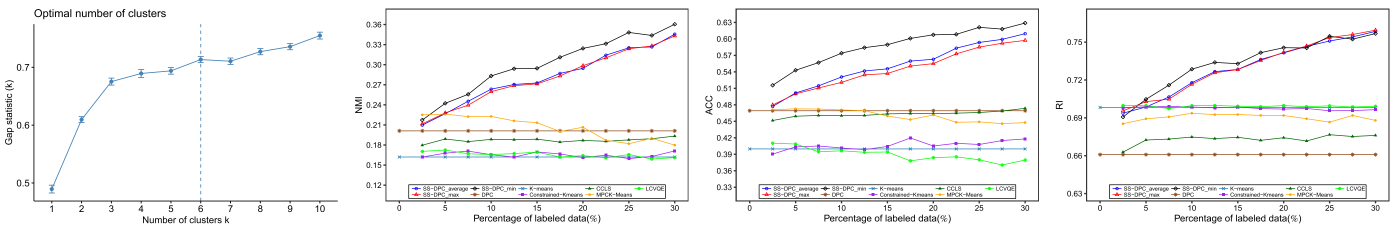
To further compare the improvement of SS-DPC over DPC, we take the Vehicle Silhouettes dataset for example and list the detailed comparison in Table 2. As shown in Table 2, the ACC, NMI, and RI values of DPC are 0.4492, 0.1960, and 0.6164, respectively. For SS-DPC_min, these values, respectively, are 0.5260, 0.2388, and 0.6915 when labeling 5% of data, improving the performance by 17.10%, 21.84%, and 12.18% compared to DPC. And these values, respectively, reach 0.5866, 0.2943, and 0.7254 when labeling 15% of data, improving the performance by 30.59%, 50.15%, and 17.68%. Whereas, these values, respectively, reach 0.6274, 0.3602, and 0.7524 when labeling 30% of data, improving the performance by 39.67%, 83.78%, and 22.06%. These results again demonstrate that SS-DPC can dramatically improve the performance of DPC by leveraging the partial information.

Compared with the referred semi-supervised algorithms, SS-DPC is more sensitive to the amount of supervision information in the form of class labels. With the increas-



(a) Gap statistic (b) NMI (c) ACC (d) RI

Figure 10. Evaluations on Image Segmentation with estimated number of clusters (k=8).



(a) Gap statistic (b) NMI (c) ACC (d) RI

Figure 11. Evaluations on Vehicle Silhouettes with estimated number of clusters (k=6).

ing percentage of labeled data, the performance of SS-DPC improves notably than the other algorithms. Especially, a small amount of labeled data for SS-DPC may bring more attractive clustering results. As shown in Figure 7(a) for Parkinsons, when labeling 2.5% of data, Constrained-Kmeans, LCVQE, and SS-DP_average yield nearly the same NMI values, all slightly higher than that of SS-DP_min. Given that the percentage of labeled data vary from 5% to 30%, the NMI values of SS-DP_average and SS-DP_min improve rapidly and are significantly higher than that of the other compared algorithms. By contrast, the NMI results of Constrained-Kmeans, CCLS, and LCVQE show no remarkable changes. The results on Vehicle Silhouettes also confirm that SS-DPC is more powerful in partial information incorporation compared with the competitors.

Three versions of SS-DPC, that is, SS-DPC_average, SS-DPC_max and SS-DPC_min, are provided in this paper. Experimental results on eight UCI datasets demonstrate that different methods used to measure the distance between two clusters in the merging stage might lead to different SS-DPC results. Evaluated by NMI, ACC, and RI, SS-DPC_min performs the best on Letter Recognition, and SS-DPC_average ranks second, both superior to SS-DPC_max. But on Iris, the three SS-DPC-based algorithms achieve almost the same results. Moreover, on Vehicle Silhouettes, SS-DPC_max and SS-DPC_average show almost the same performance, worse than SS-DPC_min. Whereas on Ionosphere, SS-DPC_min and SS-DPC_max yield nearly identical results, superior to SS-DPC_average.

Considering the fact that the number of clusters is usually unknown in reality, two datasets, Image segmentation and Vehicle Silhouettes, are used to conduct the experiments on the estimated number of clusters for comparison (Figures 10 and 11, respectively). The number of clusters are estimated by Gap statistic [31], which is implemented with R package "factoextra". Figures 10 and 11 confirm that SS-DPC can incorporate the partial information effectively and yield better clustering results based on the estimated number of clusters.

Though showing excellent performance on most of the eight UCI datasets, it is worth noting that the proposed algorithm SS-DPC is not always successful over all the 12 percentages for some datasets, such as Libras Movement, Image Segmentation and Parkinsons. Firstly, on Libras Movement, both SS-DPC and DPC show worse performance than the compared algorithms. This is because Libras is a relatively high dimension (90) dataset with small samples (360). In this case, a reliable estimation of local densities is difficult, thereby leading to poor capacity in underlying data structure identification. Secondly, on these three datasets, the performance of SS-DPC is worse than DPC when the percentage of labeled data is relatively lower. The reason for this refers to two aspects. On the one hand, it is difficult to properly merge identified clusters in boundary region with the expected one, especially when the boundary of clusters is not clear. Specifically, the centers identification method of SS-DPC has advantages of detecting more cluster centers to avoid missing centers but brings another problem that a small dense region appeared in the cluster boundary may be recognized as a new cluster. In this case, the performance of SS-DPC is affected by the merging result of this kind of clusters. On the other hand, when the percentage of labeled data is relatively lower, almost non of partial information is provided to aid the clustering of these datasets. Specifically, both Libras Movement (24 samples in each class) and Image Segmentation (30 samples in each class) consist of clusters of small size. Parkinsons is an imbalanced dataset with two classes of size 48 and 147, where one of them has a small number of samples. When the percentage of labeled data is smaller, such as 2.5%, 5%, there are only one or two labeled data in these small size classes. In this case, the partial information is too limited to improve the clustering, so the performance of SS-DPC depends on the merging results of the identified clusters in the boundary region, but it is not easy. Overall, it is these two aspects that cause the worse performance of SS-DPC than DPC on small percentages. However, it is also observed that the performance of SS-DPC on these datasets enhances and then exceeds DPC as well as the other competitors as the percentage of labeled data raising. Then, it can be concluded that SS-DPC indeed can effectively incorporate the partial information to improve the performance and more partial information contributes to better clustering performance.

4. CONCLUSION

In this paper, a semi-supervised clustering algorithm called SS-DPC is proposed to extend classical DPC to semi-supervised clustering. Different from DPC, the proposed SS-DPC can make use of a small amount of labeled data to improve the clustering performance. As a semi-supervised version of DPC, SS-DPC is a density-based clustering algorithm and has the capacity to detect clusters of various sizes, arbitrary shapes and varying densities. The proposed algorithm consists of three main steps. First, the number of clusters is automatically selected on the whole data space, and the nearest point for each data is calculated on the basis of DPC. Second, the labeled dataset is expanded by identified centers first, and then the nearest points are used to assign unique labels to their corresponding unlabeled data. Finally, two clusters with at least one virtual label are merged into one if their distance is small enough. Experiments on eight UCI datasets illustrate the success of SS-DPC in partial information incorporation and DPC performance improvement.

In future research, an interesting work on SS-DPC is to explore more reliable density estimation methods for high-dimension data to improve the clustering performance. Another work is to explore optimal stop conditions for mergence strategy to enhance the ability of SS-DPC in finding new clusters.

ACKNOWLEDGMENTS

The project was supported by the National Natural Science Foundation of China (No. 11971214, 81960309) and Cooperation Project of Chunhui Plan of the Ministry of Education of China-2018 and sponsored by the Scientific Research Foundation for the Returned Overseas Chinese Scholars, Ministry of Education of China, and Natural Science Foundation of Anhui Province (2108085QA14). The authors would also like to thank editor-in-chief and the referees for their suggestions to improve the paper.

Received 19 February 2021

REFERENCES

[1] Basu, S., Banerjee, A. and Mooney, R. (2002). Semi-supervised clustering by seeding. *Proceedings of the 19th International Conference on Machine Learning(ICML-2002)* 19-26.

[2] Bilenko, M., Basu, S. and Mooney, R. J. (2004). Integrating constraints and metric learning in semi-supervised clustering. *ICML '04: Proceedings of the twenty-first international conference on Machine learning.*

[3] Bock, R. D. and Aitkin, M. (1981). Marginal maximum likelihood estimation of item parameters: Application of an EM algorithm. *Psychometrika* **46** 443-459. MR0668311

[4] Bordogna, G. and Pasi, G. (2012). A quality driven Hierarchical Data Divisive Soft Clustering for information retrieval. *Knowledge-Based Systems* **26** 9–19.

[5] Chen, W. and Feng, G. (2012). Spectral clustering: A semi-supervised approach. *Neurocomputing* **77** 229-242.

[6] Chen, J., Lin, X., Zheng, H. and Bao, X. (2017). A novel cluster center fast determination clustering algorithm. *Applied Soft Computing* **57** 539–555.

[7] Ding, S., Jia, H., Du, M. and Xue, Y. (2018). A semi-supervised approximate spectral clustering algorithm based on HMRF model. *Information Sciences* **429** 215-228. MR3739642

[8] Du, M., Ding, S. and Jia, H. (2016). Study on density peaks clustering based on k-nearest neighbors and principal component analysis. *Knowledge-Based Systems* **99** 135-145.

[9] Du, M., Ding, S. and Yu, X. (2017). A novel density peaks clustering algorithm for mixed data. *Pattern Recognition Letters* **97** 46–53.

[10] Ester, M., Kriegel, H., Sander, J. and Xu, X. (1996). A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. *in: Proceedings of 2nd International Conference on Knowledge Discovery and Data Mining (KDD-96)* **96** 226-231.

[11] Friedman, J. H. and Meulman, J. J. (2004). Clustering objects on subsets of attributes. *J. R. Statist. Soc. B* **66** 815–849. MR2102467

[12] Gan, H., Fan, Y., Luo, Z. and Zhang, Q. (2018). Local homogeneous consistent safe semi-supervised clustering. *Expert Systems With Applications* **97** 384-393.

[13] Hiep, T. K., Duc, N. M. and Trung, B. Q. Local search approach for the pairwise constrained Clustering Problem. *SoICT '16: Proceedings of the Seventh Symposium on Information and Communication Technology.*

[14] Higham, D. J., Kalna, G. and Kibble, M. (2007). Spectral clustering and its use in bioinformatics. *Journal of Computational and Applied Mathematics* **204** 25–37. MR2320333

[15] Jain, A. K., Murty, M. N. and Flynn, P. J. (1999). Data clustering: a review. *ACM Computing Surveys* **31** 264-232.

[16] Lai, H. P., Visani, M., Boucher, A. and Ogier, J.-M. (2014). A new interactive semi-supervised clustering model for large image database indexing. *Pattern Recognition Letters* **37** 94–106.

[17] Li, Z., Liu, J. and Tang, X. (2009). Constrained clustering via spectral regularization. *In: IEEE Conference on CVPR 2009 (IEEE 2009)* 421-428.

[18] Li, Z. and Tang, Y. (2018). Comparative Density Peaks Clustering. *Expert Systems with Applications* **95** 236-247.

[19] Li, L., Zhang, H., Peng, H. and Yang, Y. (2018). Nearest neighbors based density peaks approach to intrusion detection. *Chaos, Solitons and Fractals* **110** 33–40. MR3790381

[20] Luxburg, U. V. (2007). A tutorial on spectral clustering. *Statistics and Computing* **17** 395–416. MR2409803

[21] MacQueen, J. (1967). Some methods for classification and analysis of multivariate observations. *in: Proceedings of the 5th Berkeley Symposium on Mathematical Statistics and Probability: University of California Press, Berkeley* **1** 281-297. MR0214227

[22] Miao, J., Zhou, X. and Huang, T. (2020). Local segmentation of images using an improved fuzzy C-means clustering algorithm based on self-adaptive dictionary learning. *Applied Soft Computing Journal* **91** 1–15.

[23] Papadimitriou, C. H. and Steiglitz, K. (1998). *Combinatorial Optimization: Algorithms and Complexity*. Courier Dover Publications, Mineola, New York, USA. MR1637890

[24] Pelleg, D. and Dorit, B.

[25] Ren, Y., Hu, K., Dai, X., Pan, L., Hoi, S. C. H. and Xu, Z. (2019). Semi-supervised deep embedded clustering. *Neurocomputing* **325** 121-130.

[26] Rodriguez, A. and Laio, A. (2014). Clustering by fast search and find of density peaks. *Science* **344** 1492-1496.

[27] Saxena, A., Prasad, M., Gupta, A., Bharill, N., Patel, O. P., Tiwari, A., Joo, E. M., Ding, W. and Lin, C. (2017). A review of clustering techniques and developments. *Neurocomputing* **267** 664–681.

[28] Śmieja, M., Myronov, O. and Tabor, J. (2018). Semi-supervised discriminative clustering with graph regularization. *Knowledge-Based Systems* **151** 24-36.

[29] Su, J., Li, Y. and Zhao, X. (2018). Data stream clustering by fast density-peak-search. *Statistics and Its Interface* **11** 183-189. MR3690808

[30] Tang, F., Wang, C., Su, J. and Wang, Y. (2020). Spectral clustering-based community detection using graph distance and node attributes. *Computational Statistics* **35** 69-94. MR4066276

[31] Tibshirani, R., Walther, G. and Hastie, T. (2001). Estimating the number of clusters in a data set via the gap statistics. *J. R. Statist. Soc. B* **63, part 2** 411–423. MR1841503

[32] Wagstaff, K., Cardie, C., Rogers, S. and Schroedl, S. (2001). Constrained K-means Clustering with Background Knowledge. *in: Proceedings of the Eighteenth International Conference on Machine Learning* 577–584.

[33] Wang, M., Zuo, W. and Wang, Y. (2016). An improved density peaks-based clustering method for social circle discovery in social networks. *Neurocomputing* **179** 219-227.

[34] Wang, H., Nie, R., Liu, X. and Li, T. (2012). Constraint projections for semi-supervised affinity propagation. *Knowledge-Based Systems* **36** 315–321.

[35] Wang, F., Zhou, J., Tian, Y., Wang, Y., Zhang, P., Chen, J. and Li, J. (2018). Intradialytic blood pressure pattern recognition based on density peak clustering. *Journal of Biomedical Informatics* **83** 33–39.

[36] Wu, D., Shang, M., Luo, X., Xu, J., Yan, H., Weihui, D. and Wang, G. (2018). Self-training semi-supervised classification based on density peaks of data. *Neurocomputing* **275** 180–191.

[37] Zahra, S., Ghazanfar, M. A., Khalid, A., Azam, M. A., Naeem, U. and Prugel-Bennett, A. (2015). Novel centroid selection approaches for KMeans-clustering based recommender systems. *Information Sciences* **320** 156–189. MR3367981

[38] Zhao, X., Liang, J. and Dang, C. (2017). Clustering ensemble selection for categorical data based on internal validity indices. *Pattern Recognition* **69** 150-168.

Yuanyuan Wang
School of Mathematics and Statistics, Lanzhou University, Lanzhou
China
E-mail address: wangyuanyuan1021@hotmail.com

Bingyi Jing
Department of Mathematics, Hong Kong University of Science and Technology, Hong Kong
China
E-mail address: majing@ust.hk