Journal of Computer Science and Cybernetics, V.37, N.1 (2021), 71–89
DOI 10.15625/1813-9663/37/1/15773

GRAPH BASED CLUSTERING WITH CONSTRAINTS AND ACTIVE LEARNING

VU-TUAN DANG[1], VIET-VU VU[1,*], HONG-QUAN DO[1], THI KIEU OANH LE[2]

[1] *Information Technology Institute, Vietnam National University, 144 Xuan Thuy Street, Cau Giay District, Ha Noi, Viet Nam*
[2] *University of Economics - Technology for Industry, 456 Minh Khai Street, Hai Ba Trung District, Ha Noi, Viet Nam*

Abstract. During the past few years, semi-supervised clustering has emerged as a new interesting direction in machine learning research. In a semi-supervised clustering algorithm, the clustering results can be significantly improved by using side information, which is available or collected from users. There are two main kinds of side information that can be learned in semi-supervised clustering algorithms including class labels(seeds) or pairwise constraints. In this paper, we propose a semi-supervised graph based clustering algorithm that tries to use seeds and constraints in the clustering process, called MCSSGC. Moreover, we also introduce a simple but efficient active learning method to collect the constraints that can boost the performance of MCSSGC, named KMMFFQS. These obtained results show that the proposed algorithm can significantly improve the clustering process compared to some recent algorithms.

Keywords. Semi-supervised clustering, active learning, constraints.

1. INTRODUCTION

Clustering is the task of partitioning a set of objects into clusters such that objects in the same cluster are similar to each other while objects in different clusters are dissimilar. Clustering is useful in a wide variety of applications, including image segmentation, data/text/web mining, and social science, to mention just a few. The process aims to discover the natural structure of the data, relationships between data and last but not least outliers detection. Over the last decades, the clustering topic has been widely studied with many clustering algorithms proposed in literature to work with different problem domains and scenarios [1]. These include partition-based, density-based, graph-based, distance-based, and probability-based algorithms. Detailed surveys can be found in [1, 2, 3]. Among these researches, graph based clustering has grown more and more popular with many proposals in recent years [4, 5].

In general, clustering is usually performed when there is no variable outcome or nothing known about the relationship between observations in the data set. For this reason, clustering

*Corresponding author.
E-mail addresses: vuvietvietnam@gmail.com (V.T.Dang); vuvietvu@.vnu.edu.vn (V.V. Vu); quandh.iti@gmail.com (H.Q.Do); ltkoanh@uneti.edu.vn (T.K.O. Le)

is traditionally seen as part of unsupervised learning. However, during the past few years, semi-supervised clustering has emerged as a new interesting direction in machine learning research. By using side information, which is available or queried from users, the clustering results can be improved [6]. It is also worth noting that a traditional clustering algorithm, e.g. K-Means, Fuzzy C-Means, DBSCAN, etc., does not works well for data sets with different shapes and sizes or with the presence of background noise. The semi-supervised clustering algorithms, on the other hand, can efficiently tackle with these kinds of data sets [7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18].

As mentioned in [19], side information is data that is neither from the input space nor from the output space of the function, but includes useful information to learn it. Side information is applied in many machine learning models such as Support Vector Machine [20], feature selection [21], Deep learning, etc. [19]. Given a data set $X = \{x_1, x_2, \ldots, x_n\}$, we review some concepts of side information in the context of semi-supervised clustering algorithms as follows:

- Must-link: A Must-link constraint specifies that two instances should be in the same cluster.
- Cannot-link: A Cannot-link constraint specifies that two instances should not be in the same cluster.
- Seed: A set $S \in X$ is called the seed set if for all $x_i \in S$, the user provides the label $\mathcal{C}$ of the cluster to which it belongs. Generally, we assume that for each cluster $\mathcal{C}$ of X, there is at least one seed-point $x_i \in S$.

Figure 1 illustrates the spectrum of different prior knowledge types that can be integrated in the process of classifying data. In the past two decades, many semi-supervised clustering have been proposed. It can be mentioned here the semi-supervised K-means [7, 8, 9, 10, 13], semi-supervised Fuzzy clustering [22, 23, 24, 25], semi-supervised spectral clustering [11, 12, 17], semi-supervised density based clustering [26, 27, 28], semi-supervised hierarchical clustering [29, 30], and semi-supervised graph based clustering [31, 32]. Although there are a lot of researches for semi-supervised clustering, however the problem of integrating both types of side information in the same algorithm is an interesting question [2]. Our previous work in 2019 [33] is probably one of the first work that integrated both kinds of side information in a density-based clustering.

In this work, we propose a semi-supervised graph based clustering algorithm that can efficiently integrate constraints and seeds in the same clustering process. The contributions of our paper are detailed as follows:

- We develop a proper way that tries to combine the use of seeds in a graph partitioning process and the use of Must-link and Cannot-Link constraints in the clustering process in a unified graph-based model. We named our proposed semi-supervised graph-based clustering algorithm as MCSSGC. The preliminary work of this paper is presented in [35].
- We introduce a new active learning method to collect constraints that can further improve the performance of MCSSGC. The short version is introduced in [36].

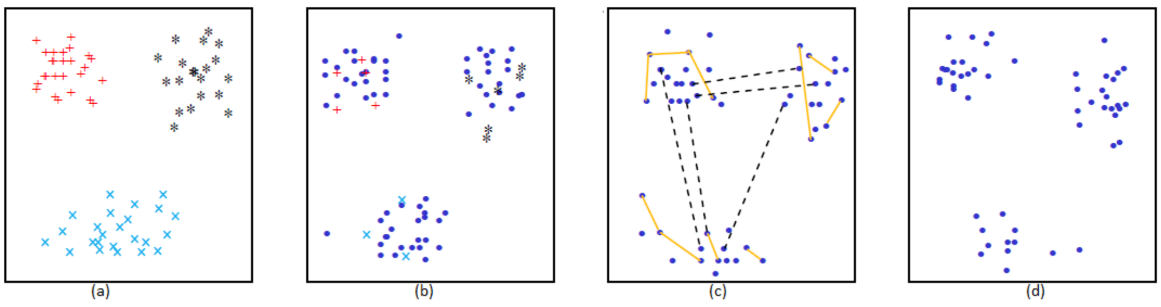


Figure 1: Spectrum of four kinds of machine learning including supervised (a), labelled (b), constrained (c), and unsupervised (d). Dots are denoted for points without any labels; circles, asterisks and crosses are denoted for points with labels. In (c), solid and dashed lines are expressed for Must- and Cannot-link constraints respectively [34].

- We carefully conducted experiments on UCI and real data set to show the effectiveness of the proposed algorithm and method.

The rest of this paper is organized as follows. Section 2 introduces the related work about semi-supervised clustering algorithms. Section 3 proposes our new semi-supervised clustering algorithm embedding both kinds of side information (MCSSGC) and an active learning method to collect constraints for the MCSSGC. Section 4 reports the experiment results. Finally, Section 5 concludes our paper and proposes some future research directions.

2. RELATED WORK

Semi-supervised clustering has attracted a lot of attention in the past two decades. Table 2. summarizes the method name, types of the constraints in current semi-supervised clustering algorithms. In the following sections, we present these methods in details.

Enforcing methods

In this method, user constraints will be integrated as support hints in the clustering process. For example, they can be used in the conditional of assigning points to clusters, in initializing centers at the first step of K-Means, or in voting clustering algorithms, etc. In [37], the COP-KMeans is presented, this is one of the newest constraint based clustering algorithm. In the COP-KMeans, constraints are integrated in the clustering assignment loop. The clusters detected must satisfy all the provided user constraints. This algorithm has opened a direction for various researches to follow. In [7], the authors introduced a constraint based clustering algorithm using some labeled data points, named Seed K-Means. The seeds are used to initialize k centers for K-Means at the first step. In [38], the constraint graph based clustering is proposed. The constraints in this case are used to help the process of constructing k-NN graph. In [39], the Constraint Based Selection algorithm (CBS) is presented. In the research, the authors used constraints in the voting process. Three principle algorithms including K-Means, DBSCAN and spectral clustering are used to produce clusters. After that, they select the best solution from a set of clusters that satisfies the largest number of given constraints. In [32], the seed graph based clustering is proposed. Using a k-NN graph to present the data set, the seeds are used in the partitioning process to form the

Table 1: Main semi-supervised clustering algorithms in literature (not exhausted works)

ID	Work	#Side inf. used	#Method
1	COP-KMeans (2001)	ML, CL	Enforcing constraints
2	SKM (2002)	seeds	Enforcing seeds
3	CCL (2002)	ML, CL	Metric learning
4	MPCK-Means (2004)	ML, CL	Metric learning
5	HCC (2004)	ML, CL	Enforcing constraints
6	CVQE (2004)	ML, CL	Penalty based
7	LCVQE (2006)	ML, CL	Penalty based
8	MDCA (2006)	ML, CL	Metric learning
9	AFCC (2008)	ML, CL	Metric learning
10	HISSCLU (2008)	seeds	Enforcing seeds
11	CDBSCAN (2009)	ML, CL	Enforcing constraints
12	SSDBSCAN (2009)	seeds	Enforcing seeds
13	MCLA (2009)	ML, CL	Enforcing constraints
14	SS-Kernel-kmeans (2009)	ML, CL	Enforcing constraints
15	KML (2010)	ML, CL	Metric learning
16	CECM (2012)	ML, CL	Metric learning
17	2SAT (2010)	ML, CL, MD, MS	SAT based method
18	CGC (2011)	ML, CL	Enforcing constraints
19	SECM (2014)	seeds	Enforcing seeds
20	CBS (2017)	ML, CL	Enforcing constraints
21	CPCC (2017)	ML, CL	CP based method
22	ILP-HC (2017)	ML, CL	ILP based method
23	CVQE+ (2018)	ML, CL	Penalty based
24	SSGC (2018)	seeds	Enforcing seeds
25	MCSSDBS (2019)	ML, CL, seeds	Enforcing seeds+constraints

connected components (principle clusters), i.e. each connected component contains at most one kind of seeds. The use of seeds brings benefits in finding the best solution of partitioning a graph into clusters. In [28], the constraint DBSCAN (C-DBSCAN) is introduced. The key idea of C-DBSCAN includes some steps as follows. Firstly, partitioning the data space into denser subspaces with the help of a KD-Tree. Secondly, from such the denser subspaces, we build a set of initial local clusters, which are groups of points within the leaf nodes of the KD-tree, split them finely such that they satisfy those Cannot-Link constraints associated with their contents. Then, density-connected local clusters are merged and enforced using Must-Link constraints. Finally, the adjacent neighborhoods in a bottom-up fashion are also merged and enforced the remaining Cannot-Link constraints. By these steps, the constraints are enforced in the clustering processes. And hence, the clustering results will be improved.

In [33], this is the first semi-supervised density-based clustering that tries to integrate seeds and constraints in the clustering process, named MCSSDBS. MCSSDBS uses a fully weighted graph to present data w.r.t users constraints. The process of detecting clusters is equivalent to the process of finding the Minimum Spanning Tree for each part of clusters. To find a part of a cluster, it starts at a seed point p_1 and detect the MST until there is a seed point p_n having a different label from p, assume that these points of MST are $\{p_1, p_2, ..p_n\}$. At that point, we have to detect the edge (p_i, p_{i+1}) to separate the MST. We can use the constraints set (if existing a Cannot-Link constraints in the MST), an active learning process to get label from users for the pair (p_i, p_{i+1}) or using the largest value (p_i, p_{i+1}) of the MST. Finally, we will obtain a part of cluster that includes the points $\{p_1, p_2, ..p_i\}$. The process of clustering will stop when all seeds have been examined.

Penalty based methods

The idea of these methods is that the clustering results may violate some user constraints. In [10], the CVQE (constrained vector quantization error) has been developed. Based on the objective function of K-Means algorithm, a new differentiable error function called the constrained vector quantization error is developed. In the error function, if a must-link constraint is violated, the cost then is measured by the distance between the cluster centroids containing the two instances. Similarly, in case a cannot-link constraint is violated, the cost is the distance between the cluster centroid that both instances are in and the nearest cluster centroid to one of the instances. Some improved version of the CVQE can be noted here such as LCVQE [13] and CVQE+ [40].

In [8], another version of constraints K-Means algorithm is presented. Similarly to CVQE, the objective function in that work is designed to integrate a set of constraints in the clustering process. It minimize the sum of the distance between every data points to the cluster centroids. The cost of constraint violations and the assignment process will satisfy as many must-links and cannot-links as possible. In [24], a constraint Fuzzy C-Means has been introduced, named AFCC. The constraints are embedded in the membership matrix using in an objective function. In AFCC, for a must-link constraints, the penalty corresponding to the presence of two such points in different clusters is weighted by the corresponding membership values; for a cannot-link constraint, the penalty corresponding to the presence of two such points in a same cluster is weighted by their membership values. In [41], the Constraints Evidential C-Means (CECM) has been proposed. In CECM, the objective function integrates a penalty term using the given constraints. The conducted experiments on UCI datasets showed the efficient of that proposed algorithm.

Metric and hybrid based methods

Given a set of constraints, in metric learning method, constraints are used to train a metric such that after training phase, must-link (similar) instances are close together and cannot-link (different) instances are far apart. Some distances have been used such as Jensen-Shannon divergence trained using gradient descent [42], Euclidean distance modified by a shortest-path algorithm [30], Mahalanobis distances trained using convex optimization (MCO) [43, 44], distance metric learning based on Discriminative Component Analysis (MDCA) [45] and a kernel-based metric learning (KML) method that provides a non-linear transformation for semi-supervised clustering [46].

Declarative methods

In these methods, the clustering model can be transformed in an optimization framework using integer linear programming (ILP), SAT, constraint programming or mathematical programming, etc. These approaches can present the different types of user constraints and the search of an exact solution that is a global optimum and satisfies all the user constraints.

In [47], a general and declarative framework based on constraint programming for constrained clustering has been developed, we refer the algorithm as CPCC for short. The framework allows to model different problems of constrained clustering, by integrating several optimization criteria (diameter, split, within-cluster sum of dissimilarities - WCSD, within-cluster sum of squares - WCSS) and various types of user constraints.

In [48], a ILP based method for constraint hierarchical clustering has been proposed, named ILP-HC. In the work, they formalize hierarchical clustering as an integer linear programming problem with a natural objective function and the dendrogram properties enforced as linear constraints. Formulating the problem as an ILP allows the use of high quality solvers (free and commercial) such as CPLEX and Gurobi (used in all our experiments) to find solutions and automatically take advantage of multi-core architectures.

In [49], a two-cluster problem has been introduced. By transforming instance level constraints (must-link,cannot-link) and cluster-level constraints (maximum-diameter (DS), minimum separation (MS)) to 2SAT problem, they produce an efficient algorithm for clustering task. The obtained results outperform some other constraint based clustering algorithms.

Discussions

From the previous works above, it can be said that semi-supervised clustering is one of the most attractive research directions in the data mining and machine learning task in the past two decades. These works have been effective in integrating side information to detect clusters that the users expect.

3. ACTIVE SEMI-SUPERVISED GRAPH BASED CLUSTERING WITH BACKGROUND KNOWLEDGE

3.1. Graph-based Clustering

Given a data set $X = \{x_1, x_2, \ldots, x_n\}$, the idea of using graph for clustering problem is as follows [50, 4]. Firstly, a weighted undirected graph will be used for presenting the data set X where each vertex represents a point of data, and has at most k edges to its nearest neighbors. An edge is constructed between two points x_i and x_j, if and only if they have each other in their set of k-nearest neighbors. The weight of two vertices x_i and x_j, called $\omega(x_i, x_j)$, is defined as the number of common nearest neighbors that two vertices share, as in the equation (1)

$$\omega(x_i, x_j) = \mid NN(x_i) \cap NN(x_j) \mid \tag{1}$$

where $NN(.)$ is the k-nearest neighbors set for a specified point.

From the graph defined above, in [50], authors find clusters by partitioning the graph in clusters with a threshold θ. The limitation of this algorithm is that it is not easy to find the threshold θ. In [32], a semi-supervised graph based clustering by seeding has been proposed, named SSGC. The SSGC method uses a set of seeds that overcomes the limit in finding parameters for the partitioning step into sub-clusters. Moreover, the algorithm uses only one parameter k that is the number of nearest neighbors. The SSGC has shown its effectiveness compared to the another semi-supervised density based clustering (SSDBSCAN) [27]. In the following section, we will try to extend the SSGC (Semi-supervised Clustering using Seeds) to work with both seeds and constraints in the same clustering process.

3.2. Semi-supervised graph-based clustering with background knowledge

In this subsection, we propose a new semi-supervised graph-based clustering that tries to integrate both constraints and seeds in the process of clustering. We name our algorithm MCSSGC (Semi-Supervised Graph based Clustering with Must-link and Cannot-link constraints). This is probably the second algorithm that can use both kinds of side information after our previous work [33]. The MCSSGC algorithm is presented in Algorithm 1 and is explained as follows:

- Firstly, we use constraints in the construction of k nearest neighbors ($k-NN$) graph. In general, the weight $\omega(x_i, x_j)$ of the edge between two vertices x_i and x_j is calculated as the equation (1). If a Cannot-link constraint exists between x_i and x_j then we will not calculate the weight between x_i and x_j. Must-link constraints are also used in finding nearest neighbors for each data point.

- Secondly, by using a parameter θ initialized by zero, this step aims to partition a graph into some connected components by using threshold θ in a loop: all edges which have weight less than θ will be removed to form connected components at each step. The loop will stop when the following cut condition is satisfied: each connected component has at most one kind of seeds.

- Thirdly, we need to further divide connected components that contains cannot-link constraints (violation of cannot-link constraints). For each connected component containing cannot-links, we repeat the removing process of edges that are equal to the value of α initialized by the current value of θ until the violation of cannot-link constraints is false - there are no cannot-link constraints existing in the connected components. This is the new point in the version of MCSSGC in the paper.

- Finally, the extracted connected components form the principal clusters. In order to construct the final clusters, we use constraints to push sub-clusters or isolated points in clusters.

We also note that the MCSSGC needs only one parameter k - the number of nearest neighbors. The θ (initialized by 0) (also α) is identified automatically in these loops as mentioned above.

Complexity analysis of the proposed method

As mentioned in [4], the complexity of the construction of the $k-NN$ graph is $O(n^2)$ or $O(n \times \log n)$ (applied for low dimension data and an optimized structure is used, e.g. R-Tree). Using Best First Search for finding connected components, the complexity of connected components extraction (step 3-6) is $O(\theta \times k \times n)$ where k is the number of neighbors; the complexity of the steps from 7 to 14 is $O((\alpha_{\max} - \theta) \times k \times n)$. In fact, $\alpha << n$ and $\theta << n$, so the complexity of the algorithm MCSSGC is $O(n^2)$ or $O(n \times \log n)$ (if the dimension of data is low).

Algorithm 1 MCSSGC Algorithm

```
Input: Data set $\mathcal{X}$, number of neighbors $k$, a set of seeds $\mathcal{S}$, a set of constraints $\mathcal{C}$
Output: A partitioning of $\mathcal{X}$
Process:
1: Integrate the constraints set $\mathcal{C}$ in the construction of $k-NN$ graph
2: $\theta = 0$
3: repeat
4:     Extract connected components with threshold $\theta$
5:     $\theta = \theta + 1$
6: until the cut condition is satisfied
7: for all connected components containing at least one Cannot-links do
8:     $\alpha = \theta$, $\alpha_{\max} = \alpha$
9:     repeat
10:        Delete any edge that its weight is equal to $\alpha$ (applied for connected components
           containing at least one CL)
11:        $\alpha = \alpha + 1$
12:    until violation of cannot-links = false
13:    $\alpha_{\max} = \max\{\alpha_{\max}, \alpha\}$
14: end for
15: Use constraints in propagation process of label to form principal clusters
16: Use constraints set $\mathcal{C}$ to construct final clusters
```

3.3. Active learning for MCSSGC

As mentioned in Section 2, the constraints or seeds must be labeled by users before their use in semi-supervised clustering algorithms. In some applications, getting label is time consuming so the question here is how we can minimize the effort required from the user, by only soliciting her (him) for the most useful constraints/seeds. There are some active learning algorithms in literature that we can note here such as min-max method [8, 51] applied for constrained K-Means, the border based method applied for semi-supervised fuzzy C-Means [24], the method based on Ability Separate between Clusters measure, we refer as the ASC method [52], the method based on density of data [53], the IPCM method applied for Fuzzy C-Means algorithms. Among these methods, the ASC method has efficiently shown for all constraint-based clustering algorithms.

In this section we propose a simple but efficient method to collect the constraints that can boost the performance of the MCSSGC algorithm. Our method bases on the Min-Max Farthest First Query Selection (MMFFQS) method [54] and the K-Means algorithm, so we call the KMMFFQS algorithm.

The idea of the Min-Max Approach (MMA) presented is to build a set of points Y from a dataset X such that the points in Y are far from each other and ensures a good coverage of the dataset. The main principles of MMA are described hereafter.

First a starting point y_1 is randomly chosen from the dataset X. Then, all the other points in Y are chosen among the points of X that maximize their minimal distance from the points already in Y. Thus, when t points already belong to Y, the process that selects the point y_{t+1} from X can be formalized as shown

$$y_{t+1} = \arg\max_{x \in X} \left(\min_{y \in Y} d(x, y) \right) \tag{2}$$

where $d(.,.)$ denotes the distance defined in the space of the objects.

The underlying idea of the MMA in the context of active learning, is to select the point that is the farthest from the points that have already been used to formulate a query to the user. In other words, at each iteration, this method selects the point that exhibits the largest label according to the previous answers of the user.

Based on the min-max method, in [8, 51], the authors proposed an active learning method to collect constraints, however, the constraints collected by these methods were only adapted for semi-supervised partitioning algorithms such as K-Means, and Fuzzy C-Means. Therefore, in our new algorithm, we try to develop an algorithm that can collect pairwise constraints and independent from the shape of clusters. In the KMMFFQS, we chose a skeleton points as follows: using K-Means algorithm to divide the given data set in to U partitions (U is chosen large enough), in each partition, we chose a point nearest the center of the partition; from that we have U points forming a skeleton. From the skeleton, we apply the min-max method to chose points for generating the candidate constraint queries.

Algorithm 2 KMMFFQS Algorithm

```
Input: Data set $\mathcal{X}$, $U$
Output: A set of collected constraints
Process:
1: Using K-Means to divide $\mathcal{X}$ in to $U$ partitions
2: Constructing the skeleton
3: Applying the Min-Max method to collect constraints from users
```

Figure 2 shows some examples of the KMMFFQS algorithm for collecting constraints.

4. EXPERIMENT RESULTS

4.1. Datasets

We use some well-known data sets from the Machine Learning Repository [55], and a document data set to evaluate our algorithm. The characteristics of these data sets are presented in Table 2. Notice that these data sets have been chosen because they have already been used in semi-supervised clustering articles [6, 3, 27, 1].

4.2. Evaluation method

We use the rand index (RI) measure [56] to evaluate the clustering results. It is one of the most widely measures used for clustering evaluation. It measures the agreement between the true partition (P_1) of a data set and a generated partition (P_2) by a clustering algorithm. Given a data set $X = x_1, x_2, \ldots, x_n$, and two partitions P_1 and P_2 as mentioned above, let a the total number of pairs of points located in the same cluster in P_1 and P_2 and let b be

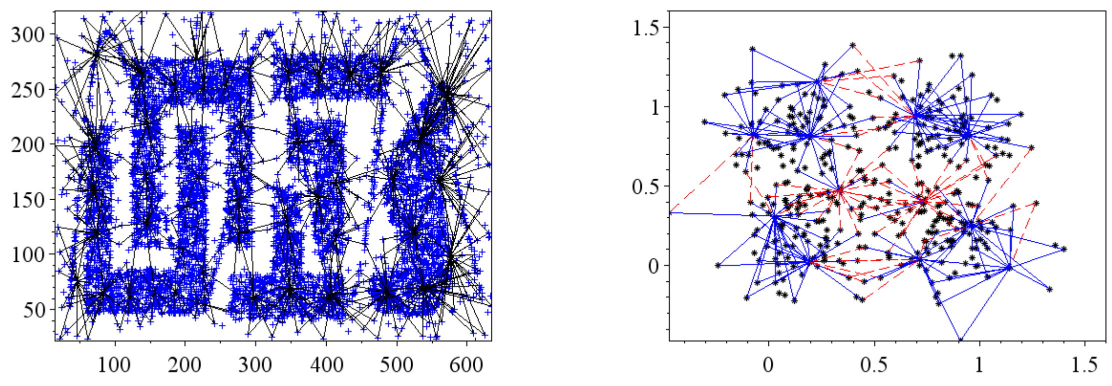


Figure 2: An example of constraints collected by the KMMFFQS for the t4.8k.dat and art1.dat data sets: (left) solid lines are expressed candidate queries; (right) solid and dashed lines are expressed for Must- and Cannot-link constraints respectively.

Table 2: Details of the data sets used in experiments

ID	Data	#Objects	#Attributes	#Clusters
1	Ecoli	336	7	8
2	Iris	150	4	3
3	Protein	115	20	6
4	Soybean	47	35	4
5	Zoo	101	16	7
6	Haberman	306	3	2
7	Yeast	1484	8	10
8	D1	4000	30	10

the total number of pairs of points such that the two points are placed in different clusters in both partitions. The RI is calculated as follows

$$RI(P_1, P_2) = \frac{2(a+b)}{n(n-1)}. \tag{3}$$

The value of RI is in the interval [0, 1], in our experiments, we calculate the RI in percentage. A higher RI value indicates a better performance of the clustering algorithm.

4.3. Comparative results

To show the effectiveness of our proposed algorithm, we compare MCSSGC with MCSS-DBS and SSGC. These algorithms are the recent works in semi-supervised clustering topic. The seeds for all semi-supervised methods are randomly generated for each experiment. Similarly, the Must-link (ML) and Cannot link (CL) sets are also randomly chosen from the data sets. The results are averaged over 20 runs.

The quality of clustering

Figure 3 shows the RI varying with the number of constraints obtained by these algorithms. A glance at the graphs reveals in most cases the MCSSGC obtained comparable and better results with MCSSDBS - the semi-supervised clustering using both seeds and constraints. When comparing with SSGC, MCSSGC significantly improves the results, indicating the benefit of using both seeds and constraints to build the clusters in the proposed algorithm. Notice that all the results in the experiments are calculated on the unlabeled data set, not including the data points (seeds) for which labels were provided to the clustering algorithm.

For more details, the improvement of MCSSGC is pronounced more in Ecoli, Soybean and Haberman data sets, that gives them three biggest improvements over MCSSDBS and SSGC. Especially for Haberman, the performance of MCSSGC significantly improves (about 10%) compared with MCSSDBS and SSGC.

Another observation can be noted is the result on Ecoli. Ecoli is a 7-dimensional data set consisting of 336 objects belonging to 8 clusters and these clusters are highly overlap. Moreover, this is also an unbalanced data set. The details of Ecoli cluster distribution is shown in Table 3. The performance obtained 94.1% when using 90 seeds and 500 constraints.

Similarly, the Yeast dataset is another challenging dataset for classification/clustering tasks since it is a highly unbalance data set, and the size of clusters are very different, as shown in Table 4. However, our new algorithm can produce good results as the MCSSDBS does. The use of constraints in this case has also been proved to be useful. They can significantly help the partitioning process to obtain the principal clusters (small clusters).

In the Haberman dataset, which is unbalanced and has highly overlapped clusters - Figure 2, if we only use seeds, it is very difficult to boost the performance of clustering process, however, the result obtained by MCSSGC is significantly boosted by further using constraints. This can be explained by the fact that when dealing with overlapping clusters, the use of constraints will efficiently help the algorithm separate points between clusters.

Table 3: Ecoli class distribution

#Class	#Objects
cp(cytoplasm)	143
im(inner membrane without signal sequence)	77
pp(perisplasm)	52
imU(inner membrane, uncleavable signal sequence)	35
om(outer membrane)	20
omL(outer membrane lipoprotein)	5
imL(inner membrane lipoprotein)	2
imS(inner membrane, cleavable signal sequence	2

Contribution of Must-Link and Cannot-Link Constraints in MCSSGC

To answer the question which Must-Link or Cannot-Link constraints are more beneficial in MCSSGC clustering process, we test the MCSSGC using only must-link constraints to compare with the MCSSGC results obtained using both Must- and Cannot-link constraints. These results are illustrated in Figure 5.

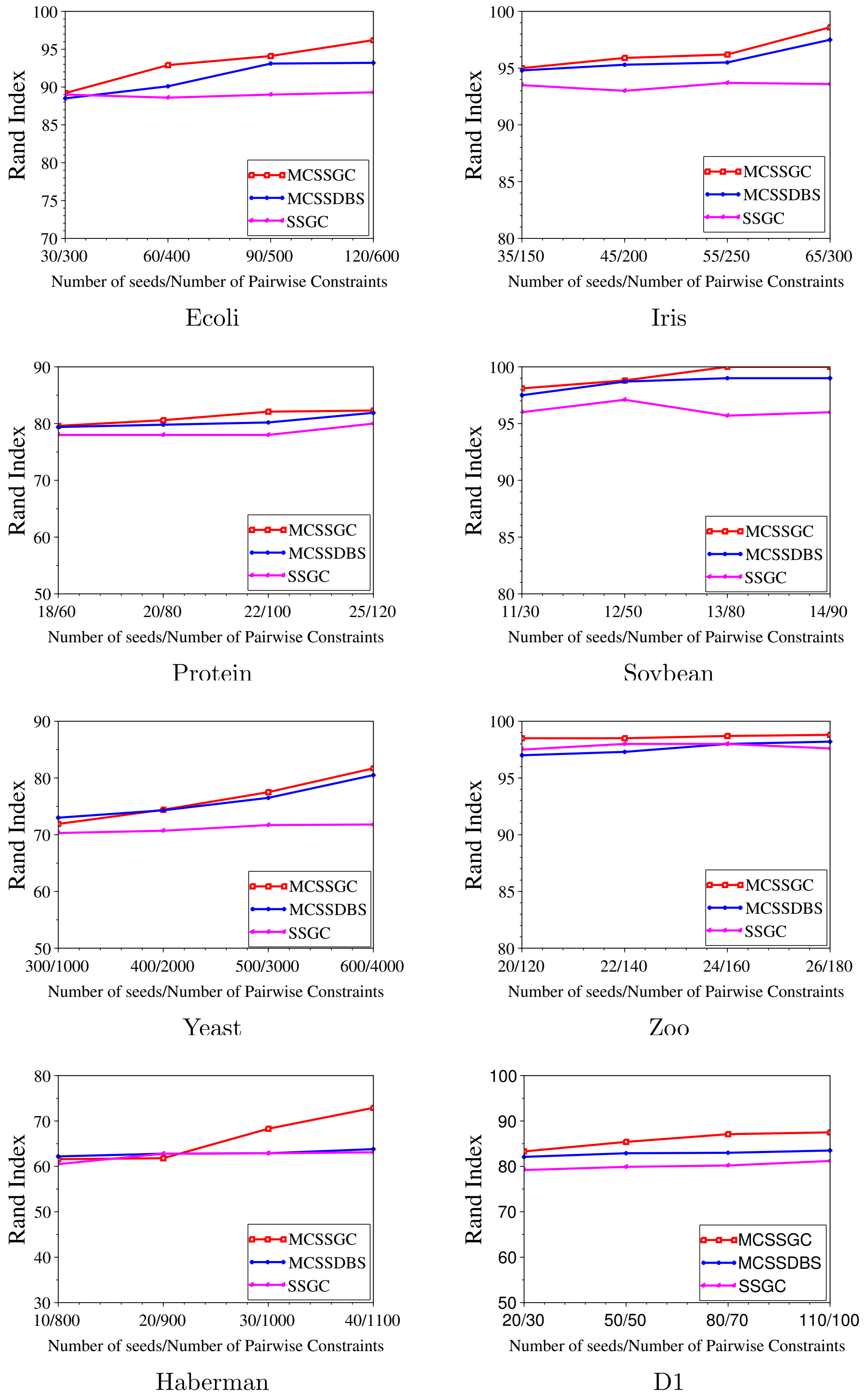


Figure 3: Comparison results between MCSSGC (the version in Algorithm 1), MCSSDBS, and SSGC.

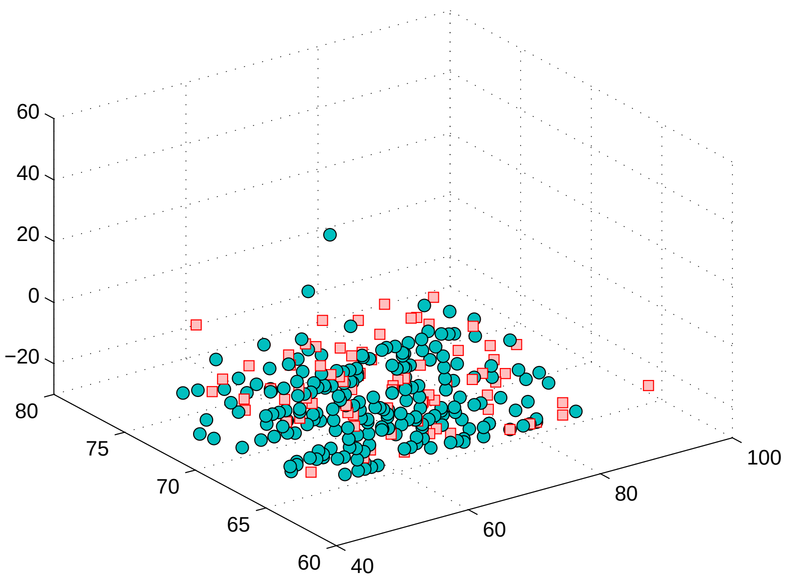
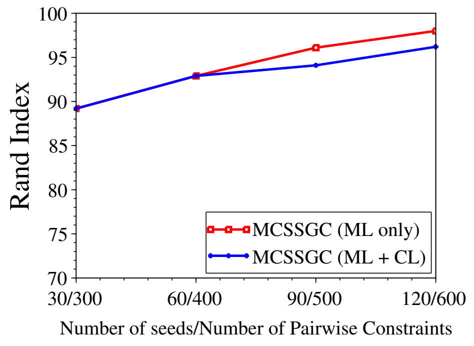


Figure 4: Haberman data set visualization

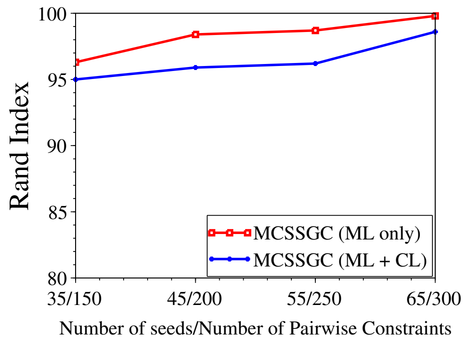
Experimental results in these figures depict that MCSSGC with only Must-Link constraints outperforms in all experiments. Remarkably, the performance on Protein data set reached 86.5% clustering accuracy, an improvement of nearly 6% compared to MCSSGC (ML+CL); for the difficult clustering data set like Ecoli, the accuracy is 97%. Moreover, we can see that the more number of must-link constraints we have, the more benefits of accuracy we obtain. These results have shown the important contribution of must-link constraints in MCSSGC, especially, when the clusters are overlap, e.g. in Haberman, the use of Must-link can significantly help the algorithm to separate those clusters.

4.4. KMMFFQS experiments

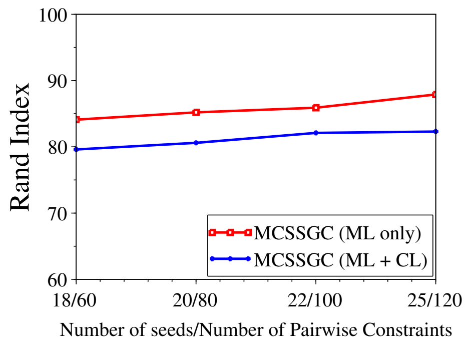
This section presents the results of clustering obtained by using constraints generated by the KMMFFQS, the ASC method, and the Random methods. The Rand Index plotted against the constraints for 8 data sets is shown in the Figure 2. As it can be observed, the constraints collected by the KMMFFQS are generally more beneficial for MCSSGC than those provided by the ASC approach and Random method. It can be explained by the fact that (1) when we partition a data set to a large number of clusters, the process of collecting constraints does not depend on the shape of clusters and (2) by using the min-max method, we can always find the good candidate to get label from users/experts. One more advantage of the KMMFFQS compared with the ASC method is that the KMMFFQS needs only one parameter U that is the number of partitions for K-Means at first step and this value can be easily chosen. In this experiments, the value of U is chosen in the interval of $[\sqrt{n} - 3, \sqrt{n}]$ [57].



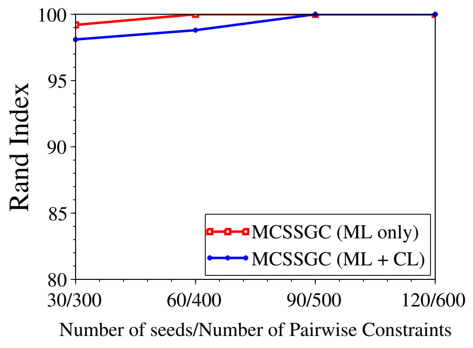
Ecoli



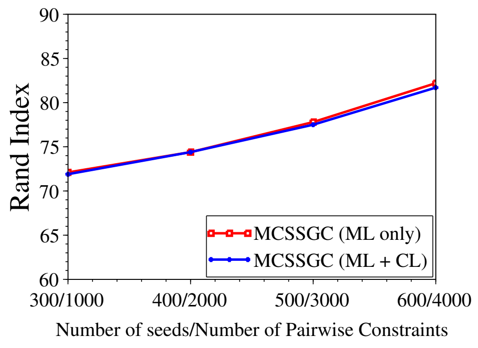
Iris



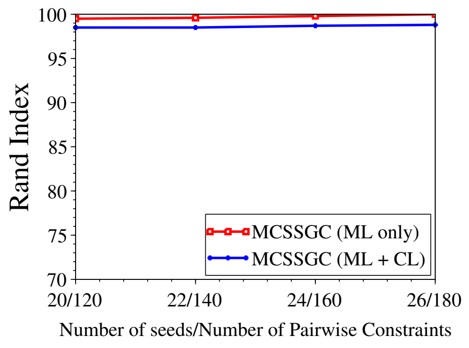
Protein



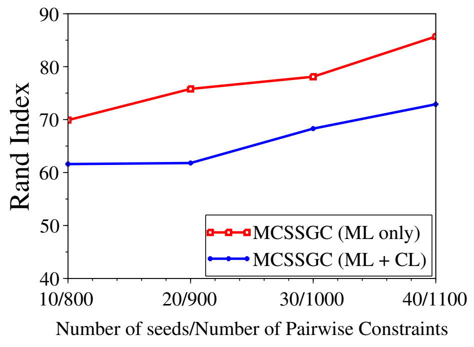
Sovbean



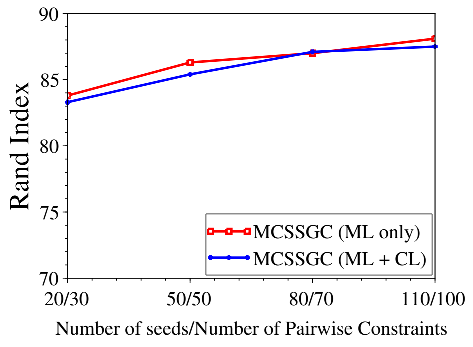
Yeast



Zoo



Haberman



D1

Table 5: Comparison results between MCSSGC + Must-Link constraints only and MCSSGC + Must-Link and Cannot-link constraints combined.

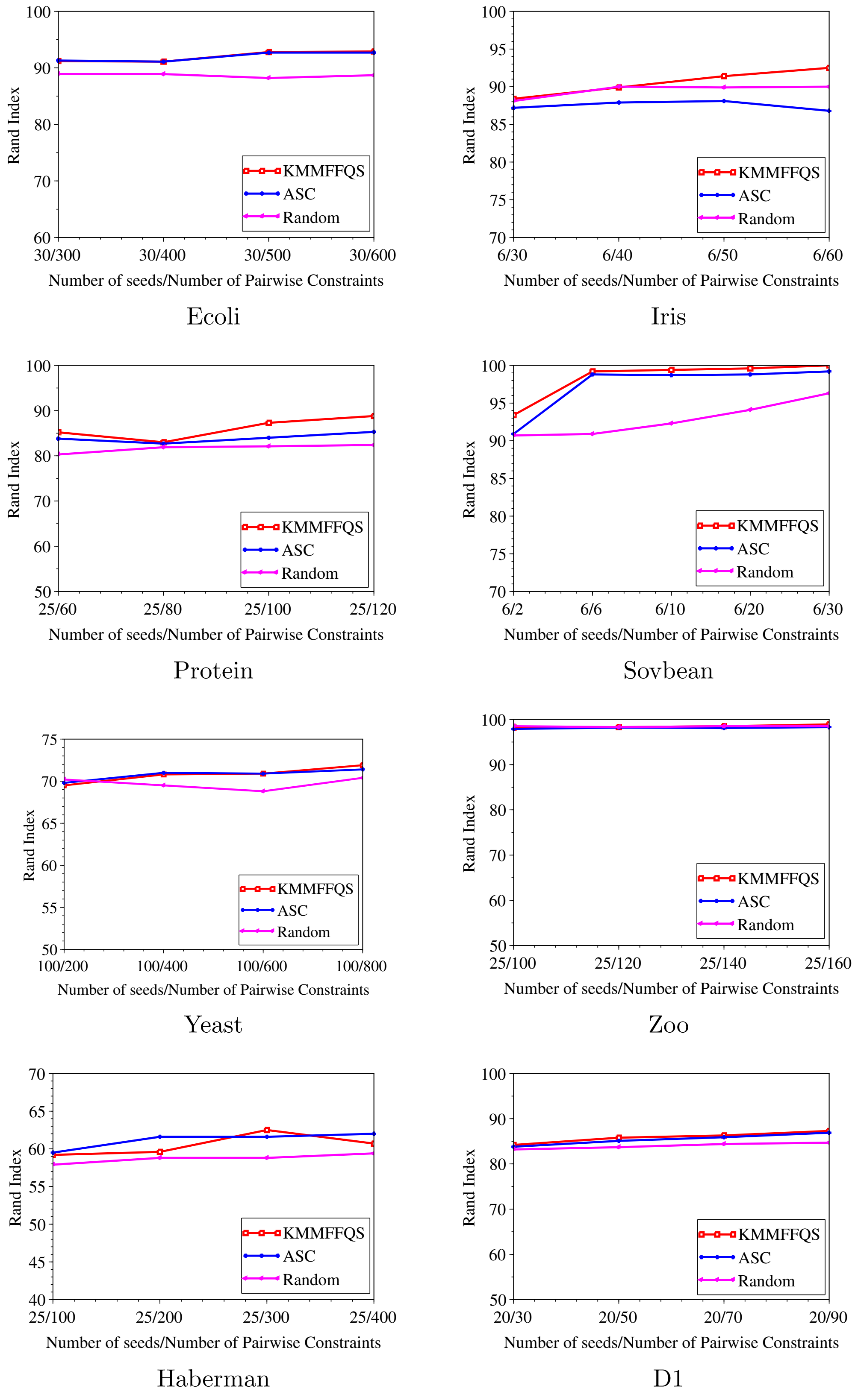


Figure 6: Clustering results by MCSSGC using constraints generated by the KMMFFQS, ASC, and Random methods.

Table 4: Yeast class distribution

#Class	#Objects
CYT (cytosolic or cytoskeletal)	463
NUC (nuclear)	429
MIT (mitochondrial)	244
ME3 (membrane protein, no N-terminal signal)	163
ME2 (membrane protein, uncleaved signal)	51
ME1 (membrane protein, cleaved signal)	44
EXC (extracellular)	37
VAC (vacuolar)	30
POX (peroxisomal)	20
ERL (endoplasmic reticulum lumen)	5

5. CONCLUSIONS

This paper proposed a new semi-supervised graph-based clustering algorithm, named MCSSGC that efficiently integrates both kinds of side information including seeds and constraints. To boost the performance of MCSSGC, we introduce a simple but efficient active learning method for collecting the constraints. Experiments carried out on varied kinds of data sets demonstrated that the proposed method significantly outperforms recent semi-supervised clustering algorithms in literature such as MCSSDBC and SSGC. In the experiments, the benefit of using both seeds and constraints to detect clusters in some challenging data sets, e.g. Ecoli and Yeast was shown. Besides, studying the effect of Must-Link and Cannot-Link constraints claimed the contribution of Must-Link in the new proposed algorithm. In future works, we continue to study other models of semi-supervised clustering as well as its applications. Finally, the study of relationship between semi-supervised clustering and semi-supervised classification is a good research direction [58].

REFERENCES

[1] R. Xu and D. W. II, “Survey of clustering algorithms,” *IEEE Transaction on Neural Networks*, vol. 16, no. 3, pp. 654–678, 2005.

[2] A. K. Jain, “Data clustering: 50 years beyond k-means,” *Pattern Recognit. Lett.*, vol. 31, no. 8, pp. 651–666, 2010.

[3] A. K. Jain and R. C. Dubes, *Algorithms for Clustering Dat.* Prentice-Hall, 1988.

[4] L. Ertoez, M. Steinbach, and V. Kumar, “Finding clusters of different sizes, shapes, and densities in noisy, high dimensional data,” in *Proceeding of the SIAM International Conference on Data Mining*, 2003, pp. 47–58.

[5] S. E. Schaeffer, “Graph clustering,” *Computer Science Review*, vol. 1, pp. 27–64, 2017.

[6] S. Basu, I. Davidson, and W. K.L., *Constrained Clustering: Advances in Algorithms, Theory, and Applications.* Chapman and Hall/CRC Data Mining and Knowledge Discovery Series, 2008.

[7] S. Basu, A. Banerjee, and R. J. Mooney, "Semi-supervised clustering by seeding," in *Proceeding of 19th International Conference on Machine Learning*, 2002, pp. 281–304.

[8] S. Basu, A. Banerjee, and R. Mooney, "Active semi-supervision for pairwise constrained clustering," in *Proceedings of the 2004 SIAM International Conference on Data Mining*, 2004, pp. 333–344.

[9] M. Bilenko, S. Basu, and R. J. Mooney, "Integrating constraints and metric learning in semi-supervised clustering," in *Proceeding of International Conference on Machine Learning*, 2004.

[10] I. Davidson and S. S. Ravi, "Clustering with constraints: Feasibility issues and the k-means algorithm," in *In SDM*, 2005, pp. 138–149.

[11] D. Mavroeidis, "Accelerating spectral clustering with partial supervision," *Data Min. Knowl. Discov.*, vol. 21, no. 2, pp. 241–258, 2010.

[12] D. Mavroeidis and E. Bingham, "Enhancing the stability and efficiency of spectral ordering with partial supervision and feature selection," *Knowl. Inf. Syst.*, vol. 3, no. 2, pp. 43–265, 2010.

[13] D. Pelleg and D. Baras, "K-means with large and noisy constraint sets," in *ECML*, 2007, pp. 674–682.

[14] Y. Shi, C. Otto, and A. K. Jain, "Face clustering: Representation and pairwise constraints," *IEEE Trans. Inf. Forensics Secur.*, vol. 13, no. 7, pp. 1626–1640, 2018.

[15] V.-V. Vu and N. Labroche, "Active seed selection for constrained clustering," *Intelligent Data Analysis.*, vol. 21, no. 3, pp. 537–552, 2017.

[16] V.-V. Vu, N. Labroche, and B. Bouchon-Meunier, "Leader ant clustering with constraints," in *Proceeding of the conference on Research, Innovation and Vision for the Future (RIVF)*, 2009, pp. 1–8.

[17] X. Wang, B. Qian, and I. Davidson, "On constrained spectral clustering and its applications," *Data Min. Knowl. Discov.*, vol. 28, no. 1, pp. 1–30, 2014.

[18] H. Zhang, S. Basu, and I. Davidson, "A framework for deep constrained clustering - algorithms and advances," in *In ECML/PKDD*, 2019, pp. 57–72.

[19] R. Jonschkowski, S. Hofer, and O. Brock, "Patterns for learning with side information," arXiv:arXiv, Tech. Rep., 2015.

[20] N. Nguyen and R. Caruana, "Improving classification with pairwise constraints: A margin-based approach," in *Proceedings of the ECML/PKDD*, 2008, pp. 113–124.

[21] D. Zhang, S. Chen, and Z.-H. Zhou, "Constraint score: A new filter method for feature selection with pairwise constraints," *Pattern Recognition*, vol. 41, no. 5, pp. 1440–1451, 2008.

[22] A. A. Abin, "Clustering with side information: Further efforts to improve efficiency," *Pattern Recognition Letters*, vol. 84, pp. 252–258, 2016.

[23] A. M. Bensaid, L. O. Hall, J. C. Bezdek, and L. P. Clarke, "Partially supervised clustering for image segmentation," *Pattern Recognition*, vol. 29, no. 5, pp. 859–871, 1996.

[24] N. Grira, M. Crucianu, and N. Boujemaa, “Active semi-supervised fuzzy clustering,” *Data Min. Knowl. Discov.*, vol. 41, no. 5, pp. 1834–1844, 2008.

[25] I. A. Maraziotis, “A semi-supervised fuzzy clustering algorithm applied to gene expression data,” *Pattern Recognition*, vol. 45, no. 1, pp. 637–648, 2012.

[26] C. Böhm and C. Plant, “Hissclu: a hierarchical density-based method for semi-supervised clustering,” in *Proceedings of the 11th international conference on Extending database technology: Advances in database technology*, 2008, pp. 440–451.

[27] L. Lelis and J. Sander, “Semi-supervised density-based clustering,” in *Proceeding of the IEEE International Conference on Data Mining*, 2009, pp. 842–847.

[28] C. Ruiz, M. Spiliopoulou, and E. M. Ruiz, “Density-based semi-supervised clustering,” *Data Min. Knowl. Discov.*, vol. 21, no. 3, pp. 345–370, 2010.

[29] I. Davidson and S. S. Ravi, “Using instance-level constraints in agglomerative hierarchical clustering: theoretical and empirical results,” *Data Min. Knowl. Discov.*, vol. 18, no. 2, pp. 257–282, 2009.

[30] D. Klein, S. D. Kamvar, and C. D. Manning, “From instance-level constraints to space-level constraints: Making the most of prior knowledge in data clustering,” in *ICML*, 2002, pp. 307–314.

[31] B. Kulis, S. Basu, I. S. Dhillon, and R. J. Mooney, “Semi-supervised graph clustering: a kernel approach,” *Machine Learning*, vol. 74, no. 1, pp. 1–22, 2009.

[32] V.-V. Vu, “An efficient semi-supervised graph based clustering,” *Intelligent Data Analysis*, vol. 2, no. 2, pp. 297–307, 2018.

[33] V.-V. Vu, H.-Q. Do, V.-T. Dang, and N.-T. Do, “An efficient density-based clustering with side information and active learning: A case study for facial expression task,” *Intelligent Data Analysis*, vol. 23, no. 1, pp. 227–240, 2019.

[34] T. Lange, M. H. C. Law, A. K. Jain, and J. M. Buhmann, “Learning with constrained and unlabelled data,” in *Proceeding of the SIAM International Conference on Data Mining*, 2005, pp. 731–738.

[35] V.-V. Vu and H.-Q. Do, “Graph-based clustering with background knowledge,” in *The 8th International Symposium on Information and Communication Technology*, 2017, pp. 167–172.

[36] V.-V. Vu, A.-T. Nguyen, and T. K. O. Le, “Selecting constraints for semi-supervised clustering,” in *In Fundamental and Applied Information Technology Conference*, 2018.

[37] K. Wagstaff, C. Cardie, S. Rogers, and S. Schrodl, “Constrained k-means clustering with background knowledge,” in *Proceedings of the International Conference on Machine Learning*, 2001, pp. 577–584.

[38] R. Anand and C. K. Reddy, “Graph-based clustering with constraints,” in *Proceeding of the Pacific-Asia Conference on Knowledge Discovery and Data Mining*, 2011, pp. 51–62.

[39] T. van Craenendonck and H. Blockeel, “Constraint-based clustering selection,” *Machine Learning*, vol. 106, no. 9-10, pp. 1497–1521, 2017.

[40] S. T. Mai, S. Amer-Yahia, and A. D. Chouakria, “K-means with large and noisy constraint sets,” in *ECML*, 2007, pp. 674–682.

[41] V. Antoine, B. Quost, M.-H. Masson, and T. Denoeux, "Cecm: Constrained evidential c-means algorithm," *Computational Statistics and Data Analysis*, vol. 56, no. 4, pp. 894–914, 2012.

[42] D. Cohn, R. Caruana, and A. McCallum, "Semi-supervised clustering with user feedback," Cornell University, Tech. Rep., 2003.

[43] A. Bar-Hillel, T. Hertz, N. Shental, and D. Weinshall, "Learning distance functions using equivalence relations," in *Proceedings of 20th International Conference on Machine Learning*, 2003, pp. 11–18.

[44] E. P. Xing, A. Y. Ng, M. I. Jordan, and S. J. Russell, "Distance metric learning with application to clustering with side-information," in *Proceeding of the NIPS*, 2002, pp. 505–512.

[45] S. C. H. Hoi, W. Liu, M. R. Lyu, and W.-Y. Ma, "Learning distance metrics with contextual constraints for image retrieval," in *Proceeding of the CVPR*, 2006, pp. 2072–2078.

[46] M. S. Baghshah and S. B. Shouraki, "Kernel-based metric learning for semi-supervised clustering," *Neurocomputing*, vol. 73, no. 9, pp. 1352–1361, 2010.

[47] T.-B.-H. Dao, K.-C. Duong, and C. Vrain, "Constrained clustering by constraint programming," *Artif. Intell.*, vol. 244, pp. 70–94, 2017.

[48] S. Gilpin and I. Davidson, "A flexible ilp formulation for hierarchical clustering," *Artif. Intell.*, vol. 244, pp. 95–109, 2017.

[49] I. Davidson, S. S. Ravi, and L. Shamis, "A sat-based framework for efficient constrained clustering," in *SDM*, 2010, pp. 94–105.

[50] R. A. Jarvis and E. A. Patrick, "Clustering using a similarity measure based on shared near neighbors," *IEEE Transactions on Computer*, vol. 22, no. 11, pp. 1025–1034, 1973.

[51] P. K. Mallapragada, R. Jin, and A. K. Jain, "Active query selection for semi-supervised clustering," in *ICPR*, 2008, pp. 1–4.

[52] V.-V. Vu, N. Labroche, and B. Bouchon-Meunier, "Improving constrained clustering with active query selection," *Pattern Recognition*, vol. 45, no. 4, pp. 1749–1758, 2012.

[53] A. A. Abin and V.-V. Vu, "A density-based approach for querying informative constraints for clustering," *Expert System with Application*, vol. 161, 2020.

[54] P. K. Mallapragada, R. Jin, and A. K. Jain, "Active query selection for semi-supervised clustering," in *Proceedings of the International conference on pattern recognition*, 2008, pp. 1–4.

[55] A. Asuncion and D. Newman, "Uci machine learning repository, http://archive.ics.uci.edu/ml/index.php," in *American Statistical Association*, 2015.

[56] W. Rand, "Objective criteria for the evaluation of clustering methods," *American Statistical Association*, vol. 66, no. 336, pp. 846–850, 1971.

[57] C. Zhong, M. I. Malinen, D. Miao, and P. Franti, "A fast minimum spanning tree algorithm based on k-means," *Inf. Sci.*, vol. 295, pp. 1–17, 2015.

[58] J. C. Gertrudes, A. Zimek, J. Sander, and R. J. G. B. Campello, "A unified framework of density-based clustering for semi-supervised classification," in *In SSDBM*, 2018, pp. 11:1–11:12.

Received 21 December 2020
Accepted 02 February 2021