Self-paced Adaptive Bipartite Graph Learning for Consensus Clustering

PENG ZHOU, School of Computer Science and Technology, Anhui University
XINWANG LIU, College of Computer, National University of Defense Technology
LIANG DU, School of Computer and Information Technology, Shanxi University
XUEJUN LI, School of Computer Science and Technology, Anhui University

Consensus clustering provides an elegant framework to aggregate multiple weak clustering results to learn a consensus one that is more robust and stable than a single result. However, most of the existing methods usually use all data for consensus learning, whereas ignoring the side effects caused by some unreliable or difficult data. To address this issue, in this article, we propose a novel self-paced consensus clustering method with adaptive bipartite graph learning to gradually involve data from more reliable to less reliable ones in consensus learning. At first, we construct an initial bipartite graph from the base results, where the nodes represent the clusters and instances, and the edges indicate that an instance belongs to a cluster. Then, we adaptively learn a structured bipartite graph from this initial one by self-paced learning, i.e., we automatically determine the reliability of each edge with adaptive cluster similarity measuring and involve the edges in bipartite graph learning in order of their reliability. At last, we obtain the final consensus result from the learned structured bipartite graph. We conduct extensive experiments on both toy and benchmark datasets, and the results show the effectiveness and superiority of our method. The codes of this article are released in http://Doctor-Nobody.github.io/codes/code_SCCABG.zip.

CCS Concepts: • **Computing methodologies → Ensemble methods;**

Additional Key Words and Phrases: Consensus clustering, clustering ensemble, bipartite graph learning, self-paced learning

ACM Reference format:
Peng Zhou, Xinwang Liu, Liang Du, and Xuejun Li. 2023. Self-paced Adaptive Bipartite Graph Learning for Consensus Clustering. *ACM Trans. Knowl. Discov. Data.* 17, 5, Article 62 (February 2023), 35 pages.
https://doi.org/10.1145/3564701

Peng Zhou is also with the State Key Laboratory of Computer Science, Institute of Software, Chinese Academic of Sciences, Beijing 100190, China.
This work is supported by the National Natural Science Foundation of China grants 62176001, 61806003, 61976129, 61922088, and 61972001; and Natural Science Foundation of Anhui Province grant 1908085MF187.
Authors' addresses: P. Zhou and X. Li (corresponding author), School of Computer Science and Technology, Anhui University, Hefei 230601, China; emails: {zhoupeng, xjli}@ahu.edu.cn; X. Liu, College of Computer, National University of Defense Technology, Changsha 410073, China; email: xinwangliu@nudt.edu.cn; L. Du, School of Computer and Information Technology, Shanxi University, Taiyuan 410073, China; email: duliang@sxu.edu.cn.

1556-4681/2023/02-ART62 $15.00
https://doi.org/10.1145/3564701

1 INTRODUCTION

Clustering is a fundamental unsupervised learning problem in machine learning and artificial intelligence, and attracts increasingly more attention in recent years. However, it is well known that conventional clustering methods often suffer from stability and robustness problems [49, 50]. To tackle these problems, consensus clustering is proposed.

Consensus clustering, also known as clustering ensemble, was first proposed by Strehl et al. [43], and it aims to ensemble multiple weak base clustering results to obtain a robust and stable consensus clustering result. In recent years, many consensus clustering methods have been proposed [29–31, 44, 52]. For example, Li et al. applied non-negative matrix factorization to ensemble multiple clusterings [29]; Liu et al. applied the de-noising auto-encoder to learn the consensus clustering result [31]; Tao et al. proposed a clustering ensemble method with adversarial loss [44].

Although these methods achieve promising performance on clustering, they may still suffer from robustness problems. We observe that these methods always use *all* data for consensus learning. However, since the base clustering results may be imperfect, it is inappropriate to always apply all data for learning. These methods may be misled by some difficult or unreliable data in the process of the ensemble. Intuitively, in the beginning, we should not use the difficult or unreliable data for ensemble, because this early model may be too weak to handle them. Then, with consensus learning, the model becomes increasingly stronger and gradually obtains the ability to tackle those difficult and unreliable data.

To fulfill this idea, we propose a novel **Self-paced Consensus Clustering with Adaptive Bipartite Graph (SCCABG)** method, which applies instances from more reliable to less reliable ones to learn the consensus result. By observing that the base clustering results can naturally be represented as a bipartite graph, where the nodes represent the instances and clusters and an edge indicates that an instance belongs to a cluster, we develop the consensus clustering method on the bipartite graph. We aim at dynamically learning a structured bipartite graph from an initial one, which contains exact c components, where c is the number of clusters. Note that, since the base results are imperfect, the edges in the initial bipartite graph are also unreliable, and thus the initial bipartite graph may not reveal such a clear clustering structure. Therefore, we plug the self-paced learning into the structured bipartite graph learning, leading to a self-paced adaptive bipartite graph learning. That is, we automatically determine the reliability of each edge, and adopt the edges in the order of their reliability to adaptively learn the structured graph. On one hand, the reliable edges are helpful to structured graph learning. On the other hand, in the process of structured graph learning, increasingly more edges become reliable. To characterize the reliability of the edges, we carefully design a regularized term with an adaptive cluster similarity measuring method. Due to the reliability characterized term, the proposed model is more robust and can even handle incomplete data. At last, we directly obtain the final consensus clustering result from the learned structured bipartite graph by finding its connective components. Therefore, the proposed method is in an end-to-end way without any uncertain postprocessing such as k-means and spectral clustering.

Although the objective function seems complex due to the carefully designed regularized term, we develop an effective block coordinate descent algorithm to optimize it, whose convergence is theoretically guaranteed. The extensive experimental results on benchmark datasets well demonstrate the effectiveness and superiority of the proposed algorithm.

Notice that this work is an extension of our early work [62]. The present work adds to the conference version in some significant ways:

- Firstly, the reliability evaluation in [62] uses the fixed cluster similarity matrix of clusters, which is inappropriate for consensus learning due to the unreliability of the base clusterings. Notice that, in [62], it constructs the similarity matrix according to the base partitions.

However, base partitions are often unreliable as introduced before, and thus the similarity matrix used in [62] is also unreliable and may still mislead the bipartite graph learning. To address this issue, in this article, we design an adaptive cluster similarity measuring mechanism, i.e., the cluster similarity matrix updates with the bipartite graph learning. With the bipartite graph becoming increasingly more reliable, the similarity matrix will also become more accurate. The ablation study in our experiments demonstrates that.

— Secondly, in real applications, it often happens that some data are missing in some base results. In this article, we also apply the proposed method to handle this incomplete consensus clustering setting.
— Thirdly, considerable new theoretical analyses and technical details are provided in this article.
— Lastly, we extend the experiments by adding a toy example to intuitively show the effectiveness of the proposed method and adding the most recent state-of-the-art consensus clustering method for comparison on benchmark datasets. We also add some more experiments to comprehensively show the performance of our method.

The remaining parts of this article are organized as follows. Section 2 provides some related work. Section 3 introduces our SCCABG in detail. Section 4 shows the experimental results. Section 5 concludes this article.

2 RELATED WORK

In this section, we briefly review some related works about consensus clustering and self-paced learning. Firstly, we introduce some notations used in this article. Boldface uppercase and lowercase letters are used to denote matrices and vectors, respectively. For a matrix $\mathbf{A}$, $\mathbf{A}_{i\cdot}$ and $\mathbf{A}_{\cdot i}$ are used to denote the ith row and column vector of matrix $\mathbf{A}$, respectively. A_{ij} denotes the (i, j)th element of $\mathbf{A}$.

2.1 Consensus Clustering

Consensus clustering, also known as clustering ensemble or clustering aggregation, aims to aggregate multiple weak base clustering results to generate a consensus and robust one. More formally, according to [43, 48, 49], given a data $\mathcal{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ with n instances, we first generate m base partitions $C^1, \ldots, C^m$ by some standard clustering method, where C^i contains k_i clusters $\pi_1^i, \ldots, \pi_{k_i}^i$, such that $\mathcal{X} = \bigcup_{j=1}^{k_i} \pi_j^i$ and $\pi_p^i \cap \pi_q^i = \emptyset$ for any $p, q \in \{1, 2, \ldots, k_i\}$ and $p \neq q$. Then consensus clustering learns a consensus partition by integrating $C^1, \ldots, C^m$.

One related task of consensus clustering is multi-view clustering [8, 11, 21, 32, 41, 65, 67]. They integrate multiple views of data feature to generate a consensus clustering result. For example, Cai et al. integrated kmeans and multiple views fusion into a unified objective function, leading to a multi-view kmeans method [8]; Zhou et al. proposed an incremental multi-view spectral clustering to handle streaming views data [67]; Peng et al. developed a parameter-free multi-view clustering method by learning a consensus embedding of multiple views [41]; Liu et al. ensembled incomplete multi-view data to obtain a robust and consensus clustering result [32]. Note that, these multi-view methods usually take multi-view features of data as input, and they often integrate information at the data level or model level. Different from multi-view clustering, consensus clustering often ensembles at the decision level, i.e., it only takes the multiple base clustering results as input without accessing the original data features. Therefore, consensus clustering is a more challenging task. Moreover, since consensus clustering does not access the original data, it can protect the privacy of data to some extent [25].

Since consensus clustering can often provide a more robust and stable clustering result than the single clustering methods and does not need to access original data, it attracts increasingly more attention in recent years. One of the most popular schema to ensemble the multiple base results is using the connective matrices [18, 28, 45, 47, 63, 64, 66]. For each base partition C^i, they construct an $n \times n$ connective matrix $\mathbf{H}^{(i)}$, where $H_{pq}^{(i)} = 1$ if $\mathbf{x}_p$ and $\mathbf{x}_q$ belong to the same cluster in C^i and $H_{pq}^{(i)} = 0$ otherwise. Then, they ensemble $\mathbf{H}^{(1)}, \ldots, \mathbf{H}^{(m)}$ to learn a consensus matrix $\mathbf{H}$ and obtain the final consensus clustering result from $\mathbf{H}$. For example, Li et al. applied symmetric nonnegative matrix factorization to obtain the consensus results from connective matrices [28]; Zhou et al. extracted the noises on connective matrices and recover the clean ones for ensemble [66]; Tao et al. used spectral clustering to learn the consensus partition from the connective matrices [45, 47]; Zhou et al. perform self-paced learning on the connective matrices to obtain the final consensus result [63].

Although connective matrix based methods have demonstrated promising performance in literature, they need to construct m $n \times n$ connective matrices which are inappropriate to handle large scale datasets. Therefore, many methods try to ensemble base clusterings with other data structures. For example, Strehl et al. constructed hyper-graphs from base clustering results and obtain final consensus result by partitioning the hyper-graph [43]; Zhou et al. applied an alignment method to directly combine the base clustering result matrices [69]; Liu et al. proposed a new ensemble method to handle missing values in base results, which also directly uses the result matrices [34]; Huang et al. constructed a factor graph for consensus clustering [14]; Li et al. applied multiple clustering results to measure the stability of each instance and assigned instances in the cluster according to the stability [25]; Huang et al. ensembled multiple clustering results by a fast propagation of cluster-wise similarity [17]; Bai et al. proposed a new graph based consensus clustering methods which used the k-means as the base clustering methods to handle non-linear data [4]; Abbasi et al. and Bagherinia et al. applied quality and diversity of base results to guide the consensus clustering [1, 3].

Besides the above-mentioned works which ensemble all base clustering results, some works aim at selecting some informative and non-redundant base clustering results for consensus clustering. For example, Azimi et al. provided an adaptive consensus clustering selection method to select the informative base results [2]; Parvin et al. developed a weighted locally adaptive clustering method for consensus clustering selection [39, 40]; Yu et al. transferred the base results selection to feature selection and proposed a hybrid strategy to select base clusterings [55]. These methods concentrated on how to select base results. However, in this article, we focus on how to ensemble base clustering results.

In this article, we learn the consensus result on a bipartite graph. Different from conventional methods, which may be misled by unreliable data, our bipartite graph learning method is in a self-paced learning framework, which could alleviate the side effects caused by unreliable data.

2.2 Self-Paced Learning

To mimic the learning process of humans, self-paced learning incrementally involves data in learning, where easy ones are used first and difficult ones are then involved gradually [22]. More formally, given a dataset $\mathcal{X} = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\}$, where $\mathbf{x}_i$ is the feature vector of the ith instance and y_i is its label, in machine learning tasks, we aim at learning a hypothesis $h(\mathbf{x}_i, \boldsymbol{\theta})$ where $\boldsymbol{\theta}$ denotes the model parameters. To learn the parameters $\boldsymbol{\theta}$, we need to minimize some loss function $\mathcal{L}(h(\mathbf{x}_i, \boldsymbol{\theta}), y_i)$ between the hypothesis and the ground truth. According to [56], self-paced learning introduces a weighted loss term on instances and a general regularized term on the weights as follows:

$$\min_{\mathrm{w}, \boldsymbol{\theta}} \quad \sum_{i=1}^{n} w_i \mathcal{L}(h(\mathbf{x}_i, \boldsymbol{\theta}), y_i) + \Omega(w_i, \lambda), \tag{1}$$

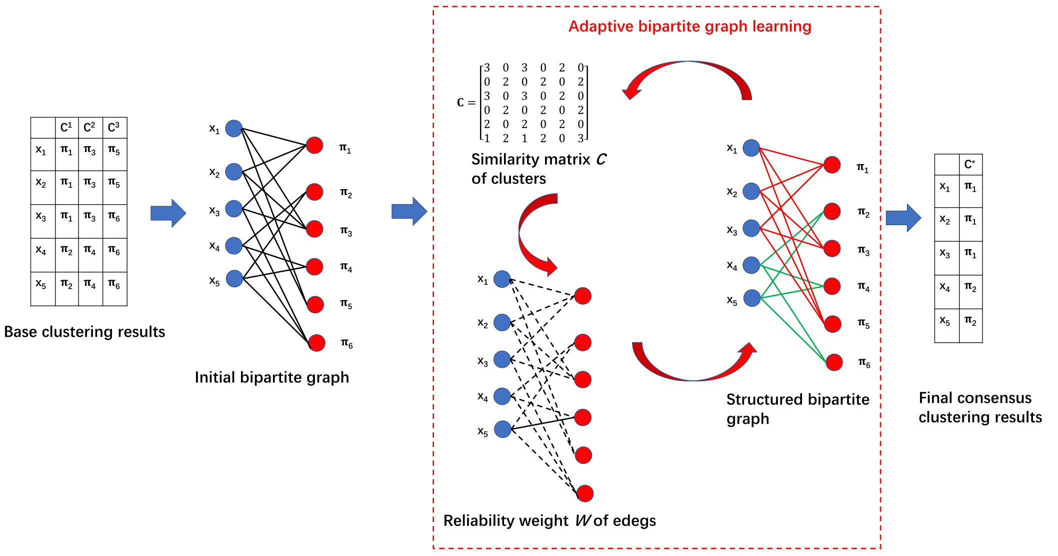

Adaptive bipartite graph learning
Similarity matrix C of clusters
Reliability weight W of edegs
Structured bipartite graph
Base clustering results
Initial bipartite graph
Final consensus clustering results

Fig. 1. The framework of SCCABG. It first generates an initial bipartite graph from base results, and then adaptively learns a structured bipartite graph with self-paced learning and adaptive cluster similarity measuring. At last, it obtains the final consensus clustering result from the learned structured bipartite graph.

where w_i is the weight of the ith instance, λ is an adaptive age parameter that grows in the process of learning, and $\Omega(w_i, \lambda)$ is the self-paced regularized term on the weights.

In Equation (1), when fixing $\boldsymbol{\theta}$, we denote $w_i^*(\lambda, l_i)$ as the optimum weight on $\mathbf{x}_i$ where $l_i = \mathcal{L}(h(\mathbf{x}_i, \boldsymbol{\theta}), y_i)$. Then, according to [19, 35, 58], the regularized term $\Omega(w_i, \lambda)$ should satisfy the following properties: (1) $w_i^*(\lambda, l_i)$ should decrease monotonically with l_i because instances with small loss, i.e., easy instances, should have large weights, so that they could be involved in learning early. (2) $w_i^*(\lambda, l_i)$ should increase monotonically with λ, so that with the process of learning, more and more instances will be involved in learning. Therefore, in self-paced learning, Equation (1) is solved in an iterative way. When fixing $\boldsymbol{\theta}$, it optimizes $\mathbf{w}$. This process is to assign weight to each instance. When fixing $\mathbf{w}$, it learns the model parameter $\boldsymbol{\theta}$. This process is to train the model with easy data.

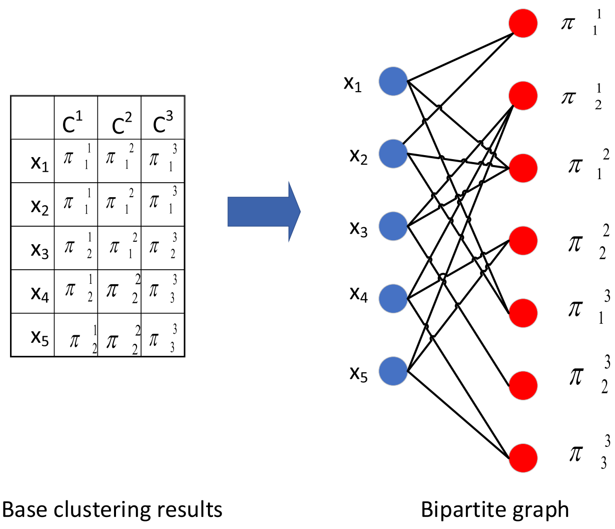
Due to its promising performance, self-paced learning has already been adopted in many applications. For example, in [5, 6], it was used to tackle the local optimum problem in non-convex optimization; in [24, 42], it was plugged in the multi-task learning; Jiang et al. extended it into subspace learning [20]; Guo et al. applied it to deep clustering [13]; Pan et al. proposed a self-paced deep regression forests method [38]. In this article, we will plug self-paced learning into bipartite graph learning for unsupervised consensus learning.

3 SELF-PACED CONSENSUS CLUSTERING WITH ADAPTIVE BIPARTITE GRAPH

In this section, we introduce the proposed SCCABG method. Figure 1 shows the framework of SCCABG. We first construct an initial bipartite graph from the given multiple base results. Then we adaptively learn a structured bipartite graph with self-paced learning and adaptive cluster similarity measuring. At last, we obtain the final clustering result directly from the structured bipartite graph. The details of each step are introduced in the following subsections.

3.1 Initial Bipartite Graph Construction

Given m base clustering results $C = \{C^1, \ldots, C^m\}$ of a dataset with n instances $\mathcal{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$, where $C^i = \{\pi_1^i, \ldots, \pi_{k_i}^i\}$, we first construct an initial bipartite graph $\mathcal{G} = \{\mathcal{V}^1, \mathcal{V}^2, \mathcal{E}\}$ from it.



	C^1	C^2	C^3
x_1	π^1_1	π^2_1	π^3_1
x_2	π^1_1	π^2_1	π^3_1
x_3	π^1_2	π^2_1	π^3_2
x_4	π^1_2	π^2_2	π^3_3
x_5	π^1_2	π^2_2	π^3_3

Fig. 2. An illustration of constructing the bipartite graph. The left side shows the three base clustering results (C^1, C^2, and C^3) of the 5 instances ($\mathbf{x}_1, \ldots, \mathbf{x}_5$). The right side shows the corresponding bipartite graph. The blue nodes represent the instances and the red nodes represent the clusters. If one instance belongs to a cluster, then there exists an edge between the corresponding blue and red nodes.

In more detail, $\mathcal{V}^1$ contains n nodes and each node represents an instance. $\mathcal{V}^2$ contains $k = \sum_{i=1}^{m} k_i$ nodes and each node represents a cluster π^i_j ($i = 1, \ldots, m$, and $j = 1, \ldots, k_i$). $\mathcal{E}$ is a set of edges which link nodes between $\mathcal{V}^1$ and $\mathcal{V}^2$. If instance $\mathbf{x}_i$ belongs to the cluster π^q_p, then there is an edge between $\mathbf{x}_i$ and π^q_p. Figure 2 shows an illustration of constructing the bipartite graph. In this example, we have five instances $\mathbf{x}_1, \ldots, \mathbf{x}_5$ and three base clusterings. For example, in the first clustering C^1, $\mathbf{x}_1$ and $\mathbf{x}_2$ belong to cluster π^1_1, and $\mathbf{x}_3$, $\mathbf{x}_4$ and $\mathbf{x}_5$ belong to the cluster π^1_2. The right side of Figure 2 shows the corresponding bipartite graph $\mathcal{G}$. $\mathcal{V}^1$ contains the blue nodes, $\mathcal{V}^2$ contains the red nodes, and $\mathcal{E}$ denotes the set of edges. In the following, for the simplicity of notations, we use $\pi_1, \ldots, \pi_k$ to denote all base clusters.

After obtaining the bipartite graph $\mathcal{G}$, we can get its adjacent matrix:

$$\mathrm{G} = \begin{bmatrix} \mathbf{0} & \mathrm{Y} \\ \mathrm{Y}^T & \mathbf{0} \end{bmatrix}, \tag{2}$$

where $\mathrm{Y} \in \{0,1\}^{n \times k}$ and $Y_{ij} = 1$ means there is an edge linking the ith instance ($\mathbf{x}_i$) and the jth cluster (π_j), and $Y_{ij} = 0$ means there is no edge between them. Taking Figure 2 as an example, its Y is defined as

$$\mathrm{Y} = \begin{bmatrix} 1 & 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 0 & 1 \end{bmatrix}.$$

3.2 Structured Bipartite Graph Learning

After obtaining the initial bipartite graph $\mathcal{G}$, we need to learn a partition on $\mathcal{G}$ as the consensus clustering result. However, this initial bipartite graph may not have a very clear clustering structure since each base clustering may not be perfect. Taking Figure 2 as an example again, we find

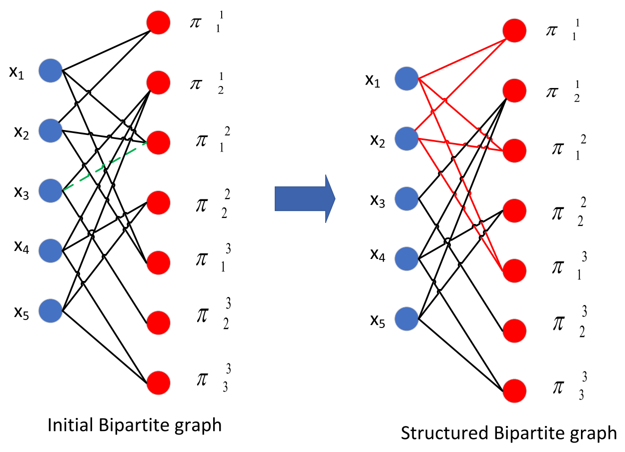


Fig. 3. An illustration of constructing structured hypergraph.

that there is only one connective component in $\mathcal{G}$, since all instances are entangled together. To make it have a clearer clustering structure and obtain the final consensus partition, we need to learn a structured bipartite graph $\mathcal{G}'$ which has exact c connective components where c is the number of clusters. Figure 3 is an illustration of the structured bipartite graph learning. The left side of Figure 3 is the initial bipartite graph $\mathcal{G}$ in Figure 2. The right side of Figure 3 is the learned structured bipartite graph $\mathcal{G}'$, which contains two connective components (i.e., one is denoted in the red lines and the other is denoted in black lines). We find that we just need to adjust one edge in the initial bipartite graph, which is denoted as the green dotted line, and we can obtain the structured one. Then clustering on $\mathcal{G}'$ is trivial because we just need to put instances in the same connective components into the same cluster.

Similar to $\mathbf{G}$, we can write the adjacent matrix of $\mathcal{G}'$ as follows:

$$\mathbf{G}' = \begin{bmatrix} \mathbf{0} & \mathbf{S} \\ \mathbf{S}^T & \mathbf{0} \end{bmatrix}, \tag{3}$$

where $\mathbf{S} \in [0,1]^{n\times k}$. To make $\mathcal{G}'$ preserve $\mathcal{G}$ as well as possible, we should minimize the difference $||\mathbf{S} - \mathbf{Y}||_F^2$. Moreover, we also need to impose some constraints on $\mathbf{S}$ to make sure that $\mathcal{G}'$ has c connective components.

Given $\mathbf{G}'$, we first obtain its normalized Laplacian matrix $\mathbf{L} = \mathbf{I} - \mathbf{D}^{-\frac{1}{2}}\mathbf{G}'\mathbf{D}^{-\frac{1}{2}}$, where $\mathbf{I}$ is an identity matrix and $\mathbf{D}$ is a diagonal matrix whose diagonal element $D_{ii} = \sum_{j=1}^{k+n} G'_{ij}$. Then, according to [36], we have that the number of connected components in $\mathcal{G}'$ is equal to $n+k$ minus the rank of $\mathbf{L}$, i.e., $rank(\mathbf{L}) = n+k-c$. To this end, we obtain the following formula:

$$\begin{aligned} \min_{\mathbf{S}} \quad & ||\mathbf{S} - \mathbf{Y}||_F^2, \\ s.t. \quad & 0 \le S_{ij} \le 1, \quad rank(\mathbf{L}) = n+k-c. \end{aligned} \tag{4}$$

3.3 Self-Paced Bipartite Graph Learning

Equation (4) provides a framework to learn an adaptive structured bipartite graph. However, as introduced before, since the base clusterings may be imperfect, each edge obtained from the base clusterings may also be unreliable. To characterize the reliability of each edge, we introduce a weight matrix $\mathbf{W} \in [0,1]^{n\times k}$ where the larger W_{ij} is, the more reliable the corresponding edge is. With $\mathbf{W}$ we can integrate our consensus clustering task into a self-paced learning framework seamlessly, which involves edges gradually from more reliable ones to less reliable ones. As suggested in [19], we set $\Omega(w_i, \lambda)$ in Equation (1) as $-\lambda\|\mathbf{W}\|_1$, and obtain:

$$\begin{aligned} \min_{\mathbf{S},\mathbf{W}} \quad & \|\mathbf{W} \odot (\mathbf{S}-\mathbf{Y})\|_F^2 - \lambda\|\mathbf{W}\|_1, \\ s.t. \quad & 0 \le S_{ij} \le 1, \quad rank(\mathbf{L}) = n+k-c, \quad 0 \le W_{ij} \le 1, \end{aligned} \tag{5}$$

where $\odot$ is the Hadamard product, which means the element-wise production of two matrices; the second term is the self-paced regularized term, and λ is the age parameter and becomes increasingly larger in the process of optimization.

Unfortunately, it is not enough to characterize the reliability of edges only by the first term in Equation (5). We need to take a closer look at the $\mathbf{W}$. Notice that, if two clusters π_p and π_q are very similar, then for any instance $\mathbf{x}_i$, it is very likely that either $\mathbf{x}_i$ belongs to both clusters or $\mathbf{x}_i$ belongs to neither. Therefore, if $(S_{ip}-S_{iq})^2$ is large, which means $\mathbf{x}_i$ is more likely to belong to one of the clusters, then at least one of S_{ip} and S_{iq} is unreliable, i.e., at least one of W_{ip} and W_{iq} should be small. More formally, we use the following carefully designed regularized term to characterize the reliability of edges:

$$\min_{\mathbf{W}} \quad \sum_{i=1}^{n}\sum_{p,q=1}^{k} C_{pq}(S_{ip}-S_{iq})^2 W_{ip}W_{iq}, \tag{6}$$

where C_{pq} is the (p,q)th element in $\mathbf{C} \in \mathbb{R}^{k\times k}$, which characterizes the similarity of two clusters. In our previous work [62], we simply fix $\mathbf{C}$ as $\mathbf{C} = \mathbf{Y}^T\mathbf{Y}$. We can find that, if C_{pq} is large (i.e., π_p and π_q are similar) and $(S_{ip}-S_{iq})^2$ is large (i.e., $\mathbf{x}_i$ only belongs to one of the clusters and does not belong to the other cluster), then the only chance to minimize Equation (6), which is a production of C_{pq}, $(S_{ip}-S_{iq})^2$, W_{ip} and W_{iq}, is that at least one of W_{ip} and W_{iq} should be small, which means at least one of the relationship S_{ip} and S_{iq} is unreliable. Taking it into our self-paced framework (Equation (5)), we obtain the following objective function:

$$\begin{aligned} \min_{\mathbf{S},\mathbf{W}} \quad & \|\mathbf{W} \odot (\mathbf{S}-\mathbf{Y})\|_F^2 - \lambda\|\mathbf{W}\|_1 + \gamma_1 \sum_{i=1}^{n}\sum_{p,q=1}^{k} C_{pq}(S_{ip}-S_{iq})^2 W_{ip}W_{iq} \\ s.t. \quad & 0 \le S_{ij} \le 1, \quad rank(\mathbf{L}) = n+k-c, \quad 0 \le W_{ij} \le 1, \end{aligned} \tag{7}$$

where γ_1 is a balanced parameter. In practice, γ_1 is small to make the subproblems involving $\mathbf{W}$ and $\mathbf{S}$ convex.

Figure 4 provides a simple toy example to show the effects of the regularized term of reliability. There are five instances $\mathbf{x}_1, \ldots, \mathbf{x}_5$, wherein the first base clustering result, $\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3$ belong to π_1, and $\mathbf{x}_4$ and $\mathbf{x}_5$ belong to π_2. In the second base result, $\mathbf{x}_1$ and $\mathbf{x}_2$ belong to π_3 and the other three instances belong to π_4. Its $\mathbf{Y}$ and initial similarity matrix of clusters $\mathbf{C}$ are shown on the right side of Figure 4. Notice that π_1 and π_3 are similar, because the only difference between them is $\mathbf{x}_3$. π_2 and π_4 are also similar, and the only difference is also $\mathbf{x}_3$. Intuitively, since π_1 and π_3 are similar, and $\mathbf{x}_3$ only belongs to π_1 but does not belong to π_3, at least one of S_{31} and S_{33} is unreliable. Similarly, at least one of S_{32} and S_{34} is unreliable.

Considering the subproblem w.r.t. $\mathbf{W}_{3.}$, which is corresponding to $\mathbf{x}_3$, we minimize the following formula:

$$\min_{\mathbf{W}_{3.}} \|\mathbf{W}_{3.} \odot (\mathbf{S}_{3.} - \mathbf{Y}_{3.})\|_2^2 + \sum_{p,q=1}^{4} C_{pq}(S_{3p}-S_{3q})^2 W_{3p}W_{3q} - \|\mathbf{W}_{3.}\|_1$$

For simplicity, we set $\gamma_1 = 1$ and $\lambda = 1$. In the first iteration, we initialize $\mathbf{S} = \mathbf{Y}$ and initialize $\mathbf{C}$ as shown in Figure 4. Notice that $C_{12}, C_{21}, C_{23}, C_{32}, C_{34}$, and C_{43} are zeros, and $S_{31} - S_{34}$ and $S_{32} - S_{33}$

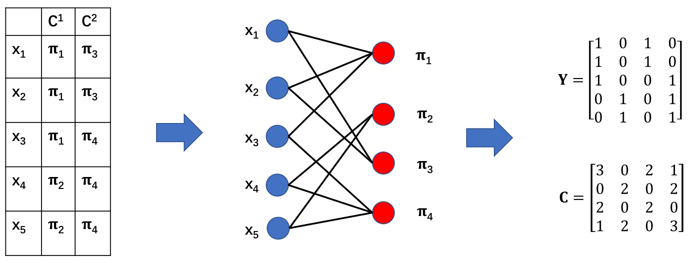

C¹
C²
x1 π1 π3
x2 π1 π3
x3 π1 π4
x4 π2 π4
x5 π2 π4
x1
x2
x3
x4
x5
π1
π2
π3
π4
Y = [1 0 1 0; 1 0 1 0; 1 0 0 1; 0 1 0 1; 0 1 0 1]
C = [3 0 2 1; 0 2 0 2; 2 0 2 0; 1 2 0 3]

Fig. 4. A toy example of the reliability. There are five instances $\mathbf{x}_1, \ldots, \mathbf{x}_5$, wherein the first base clustering result, $\mathbf{x}_1, \ldots, \mathbf{x}_3$ belong to π_1, and $\mathbf{x}_4$ and $\mathbf{x}_5$ belong to π_2. In the second base result, $\mathbf{x}_1$ and $\mathbf{x}_2$ belong to π_3 and the other three instances belong to π_4. The middle side shows the initial bipartite graph, and the right side shows the corresponding $\mathbf{Y}$ and initial similarity matrix of clusters $\mathbf{C}$.

are zeros. When removing the zero terms, we have

$$\begin{aligned}\min_{\mathbf{W}_{3.}} \quad & 4(S_{31} - S_{33})^2 W_{31} W_{33} + 4(S_{32} - S_{34})^2 W_{32} W_{34} - \|\mathbf{W}_{3.}\|_1 \\ & = 4W_{31}W_{33} + 4W_{32}W_{34} - \|\mathbf{W}_{3.}\|_1\end{aligned}$$

Since in the first term, we need to minimize $W_{31}W_{33}$, at least one of W_{31} and W_{33} should be small. Similarly, at least one of W_{32} and W_{34} should be small. In fact, when minimizing it, we obtain $W_{31} = W_{32} = W_{33} = W_{34} = 0.25$ which are all small. It is consistent with our intuition.

3.4 Adaptive Cluster Similarity Measuring

In our previous work [62], we use $\mathbf{Y}^T\mathbf{Y}$ as the cluster similarity matrix $\mathbf{C}$. However, as mentioned before, the base clustering results $\mathbf{Y}$ may be unreliable, and thus $\mathbf{C} = \mathbf{Y}^T\mathbf{Y}$ is also unreliable. Since we evaluate the reliability $\mathbf{W}$ of edges according to $\mathbf{C}$ as shown in Equation (6), unreliable $\mathbf{C}$ may also mislead the evaluation of $\mathbf{W}$. Therefore, to better evaluate the reliability of the edges, we need a more accurate measurement of the cluster similarity.

An ideal cluster similarity measuring method is that the similarity of clusters should be adaptively adjusted during the bipartite graph learning process. On one hand, a more reliable bipartite graph can lead to a more accurate cluster similarity measuring; and on the other hand, a more accurate cluster similarity measuring can guide us to determine the reliability of edges more accurately and further be helpful to learning a better structured graph. To this end, $\mathbf{C}$ in Equation (7) should be learned as a variable instead of being predefined.

To adaptively learn the similarity matrix, we need some prior knowledge on $\mathbf{C}$. Although predefining $\mathbf{C} = \mathbf{Y}^T\mathbf{Y}$ may be inappropriate, $\mathbf{Y}^T\mathbf{Y}$ can still be regarded as a good prior on $\mathbf{C}$. Before graph learning, we have no extra information on $\mathbf{C}$ except $\mathbf{Y}$, and thus we can initialize $\mathbf{C} = \mathbf{Y}^T\mathbf{Y}$ and hope $\mathbf{C}$ should not be too far away from $\mathbf{Y}^T\mathbf{Y}$. Then, in the process of learning, we can adaptively fine-tune $\mathbf{C}$ based on $\mathbf{S}$ and $\mathbf{W}$. More formally, we can obtain the following objective function:

$$\begin{aligned}\min_{\mathbf{S},\mathbf{W},\mathbf{C}} \quad & \|\mathbf{W} \odot (\mathbf{S} - \mathbf{Y})\|_F^2 - \lambda\|\mathbf{W}\|_1 + \gamma_1 \sum_{i=1}^{n} \sum_{p,q=1}^{k} C_{pq}(S_{ip} - S_{iq})^2 W_{ip} W_{iq} + \gamma_2 \|\mathbf{C} - \mathbf{Y}^T\mathbf{Y}\|_F^2, \\ s.t. \quad & 0 \le S_{ij} \le 1, \quad rank(\mathbf{L}) = n + k - c, \quad 0 \le W_{ij} \le 1, \quad C_{pq} \ge 0, \quad \mathbf{C} = \mathbf{C}^T, \end{aligned} \tag{8}$$

where γ_2 is another parameter to control the prior of $\mathbf{C}$. Since $\mathbf{C}$ is a similarity matrix, it should satisfy the nonnegative and symmetric constraints as denoted in Equation (8).

It is worthy to take a close look at the carefully designed regularized term $\sum_{i=1}^{n}\sum_{p,q=1}^{k} C_{pq}(S_{ip} - S_{iq})^2 W_{ip} W_{iq}$ again. Although the motivation of this term is to characterize the reliability of edges, it has three functions in total:

— when fixing **S** and **C** to learn **W**, it evaluates the reliability of edges;
— when fixing **W** and **C** to optimize **S**, it propagates the connection information on the bipartite graph;
— when fixing **W** and **S** to learn **C**, it adaptively learns the similarity of all clusters.

3.5 Optimization

Equation (8) involves the constraint $rank(\mathbf{L}) = n+k-c$, which is hard to optimize. We first handle this constraint. According to [37], by introducing the auxiliary orthogonal matrix $\mathbf{F} \in \mathbb{R}^{(n+k)\times c}$ and a large enough parameter ρ, Equation (7) is equivalent to the following formula:

$$\begin{aligned}\min_{\mathbf{S},\mathbf{W},\mathbf{C},\mathbf{F}} \quad & \|\mathbf{W}\odot(\mathbf{S}-\mathbf{Y})\|_F^2 - \lambda\|\mathbf{W}\|_1 + \gamma_1 \sum_{i=1}^{n}\sum_{p,q=1}^{k} C_{pq}(S_{ip}-S_{iq})^2 W_{ip}W_{iq} + \gamma_2\|\mathbf{C}-\mathbf{Y}^T\mathbf{Y}\|_F^2 \\ & + \rho tr(\mathbf{F}^T\mathbf{L}\mathbf{F}) \\ s.t. \quad & 0 \le S_{ij} \le 1, \quad 0 \le W_{ij} \le 1, \quad \mathbf{F}^T\mathbf{F} = \mathbf{I}, \quad C_{pq} \ge 0, \quad \mathbf{C} = \mathbf{C}^T. \end{aligned} \tag{9}$$

Then, we optimize **W**, **F**, **S**, and **C**, respectively, by fixing the other variables as many other machine learning methods do [36, 37, 61].

3.5.1 Optimizing **W**. When optimizing **W**, we find that Equation (9) can be decoupled into n independent subproblems by rows. Considering the ith subproblem, we have

$$\begin{aligned}\min_{\mathbf{W}_{i\cdot}} \quad & \sum_{p=1}^{k} W_{ip}^2 A_{ip} - \lambda \sum_{p=1}^{k} W_{ip} + \gamma_1 \sum_{p,q=1}^{k} W_{ip} B_{pq} W_{iq}, \\ s.t. \quad & 0 \le W_{ij} \le 1, \end{aligned} \tag{10}$$

where $A_{ip} = (S_{ip} - Y_{ip})^2$ and $B_{pq} = C_{pq}(S_{ip} - S_{iq})^2$.

Note that, Equation (10) is a quadratic programming problem with bounded constraint and can be solved by some standard optimization methods, such as trust-region reflective algorithm [9]. In our implementation, we use *quadprog* function provided in Matlab.

3.5.2 Optimizing **F**. When optimizing **F**, we need to solve the following subproblem:

$$\begin{aligned}\min_{\mathbf{F}} \quad & tr(\mathbf{F}^T\mathbf{L}\mathbf{F}) \\ s.t. \quad & \mathbf{F}^T\mathbf{F} = \mathbf{I}. \end{aligned} \tag{11}$$

According to Ky Fan Theory [10], Equation (11) can be solved by computing the eigen-decomposition of **L**. However, conducting eigen-decomposition on an $(n+k)\times(n+k)$ matrix is often in $O((n+k)^3)$ time and is very time consuming. Fortunately, since **L** is a Laplacian matrix of a bipartite graph, according to [36], Equation (11) can be solved by conducting **Singular Valued Decomposition** (**SVD**) on a small rectangle matrix.

In more detail, define diagonal matrices $\hat{\mathbf{D}} \in \mathbb{R}^{n\times n}$ and $\tilde{\mathbf{D}} \in \mathbb{R}^{k\times k}$ whose diagonal elements are $\hat{D}_{ii} = \sum_{j=1}^{k} S_{ij}$ and $\tilde{D}_{jj} = \sum_{i=1}^{n} S_{ij}$, respectively, and write **F** as the block matrices $\mathbf{F} = [\mathbf{U}^T, \mathbf{V}^T]^T$, where $\mathbf{U} \in \mathbb{R}^{n\times c}$ and $\mathbf{V} \in \mathbb{R}^{k\times c}$. Equation (11) can be rewritten as

$$\begin{aligned}
&\min_{\mathbf{F}^T\mathbf{F}=\mathbf{I}} \ tr(\mathbf{F}^T\mathbf{L}\mathbf{F}) \qquad (12)\\
\Leftrightarrow &\max_{\mathbf{U}^T\mathbf{U}+\mathbf{V}^T\mathbf{V}=\mathbf{I}} tr\left(\begin{bmatrix} \mathbf{U}^T & \mathbf{V}^T \end{bmatrix}\begin{bmatrix} \hat{\mathbf{D}}^{-\frac{1}{2}} & \mathbf{0} \\ \mathbf{0} & \tilde{\mathbf{D}}^{-\frac{1}{2}} \end{bmatrix}\begin{bmatrix} \mathbf{0} & \mathbf{S} \\ \mathbf{S}^T & \mathbf{0} \end{bmatrix}\begin{bmatrix} \hat{\mathbf{D}}^{-\frac{1}{2}} & \mathbf{0} \\ \mathbf{0} & \tilde{\mathbf{D}}^{-\frac{1}{2}} \end{bmatrix}\begin{bmatrix} \mathbf{U} \\ \mathbf{V} \end{bmatrix}\right)\\
\Leftrightarrow &\max_{\mathbf{U}^T\mathbf{U}+\mathbf{V}^T\mathbf{V}=\mathbf{I}} tr\left(\begin{bmatrix} \mathbf{U}^T & \mathbf{V}^T \end{bmatrix}\begin{bmatrix} \mathbf{0} & \hat{\mathbf{D}}^{-\frac{1}{2}}\mathbf{S}\tilde{\mathbf{D}}^{-\frac{1}{2}} \\ \tilde{\mathbf{D}}^{-\frac{1}{2}}\mathbf{S}^T\hat{\mathbf{D}}^{-\frac{1}{2}} & \mathbf{0} \end{bmatrix}\begin{bmatrix} \mathbf{U} \\ \mathbf{V} \end{bmatrix}\right)\\
\Leftrightarrow &\max_{\mathbf{U}^T\mathbf{U}+\mathbf{V}^T\mathbf{V}=\mathbf{I}} 2tr(\mathbf{U}^T\hat{\mathbf{D}}^{-\frac{1}{2}}\mathbf{S}\tilde{\mathbf{D}}^{-\frac{1}{2}}\mathbf{V}).
\end{aligned}$$

Equation (12) can be solved by the following Theorem:

Theorem 1. *[36] The optimal solutions to the problem* (12)*:*

$$\max_{\mathbf{U}^T\mathbf{U}+\mathbf{V}^T\mathbf{V}=\mathbf{I}} \quad tr(\mathbf{U}^T\hat{\mathbf{D}}^{-\frac{1}{2}}\mathbf{S}\tilde{\mathbf{D}}^{-\frac{1}{2}}\mathbf{V})$$

are $\mathbf{U} = \frac{\sqrt{2}}{2}\mathbf{U}'$ *and* $\mathbf{V} = \frac{\sqrt{2}}{2}\mathbf{V}'$*, where* $\mathbf{U}'$ *and* $\mathbf{V}'$ *are the leading c left and right singular vectors of* $\hat{\mathbf{D}}^{-\frac{1}{2}}\mathbf{S}\tilde{\mathbf{D}}^{-\frac{1}{2}}$.

Notice that since $\hat{\mathbf{D}}^{-\frac{1}{2}}\mathbf{S}\tilde{\mathbf{D}}^{-\frac{1}{2}}$ is an $n \times k$ matrix and often has $n \gg k$, the time complexity of computing its SVD is $O(nk^2)$ which is much smaller than $O((n+k)^3)$.

3.5.3 Optimizing $\mathbf{S}$. When optimizing $\mathbf{S}$, notice that $\mathbf{L}$ is relative to $\mathbf{S}$, and thus we should handle $tr(\mathbf{F}^T\mathbf{L}\mathbf{F})$ first. Let $\mathbf{F}' = \mathbf{D}^{-\frac{1}{2}}\mathbf{F}$, and we have

$$\begin{aligned}
tr(\mathbf{F}^T\mathbf{L}\mathbf{F}) &= tr(\mathbf{F}'^T\mathbf{D}\mathbf{F}') - tr(\mathbf{F}'^T\mathbf{G}'\mathbf{F}')\\
&= \frac{1}{2}\left(\sum_{i=1}^{c}\sum_{j,l=1}^{n+k} F_{ji}'^2 G_{jl}' + \sum_{i=1}^{c}\sum_{j,l=1}^{n+k} F_{li}'^2 G_{jl}' - \sum_{i=1}^{c}\sum_{j,l=1}^{n+k} 2F_{ji}'G_{jl}'F_{li}'\right)\\
&= \frac{1}{2}\left(\sum_{j,l=1}^{n+k} G_{jl}' \sum_{i=1}^{c}(F_{ji}' - F_{li}')^2\right)\\
&= \sum_{j=1}^{n}\sum_{l=n+1}^{n+k} S_{j(l-n)} \left\|\frac{\mathbf{F}_{j.}}{\sqrt{d_j}} - \frac{\mathbf{F}_{l.}}{\sqrt{d_l}}\right\|_2^2, \qquad (13)
\end{aligned}$$

where $d_j = \sum_{m=1}^{k} S_{jm}$ and $d_l = \sum_{m=1}^{k} S_{lm}$, respectively.

Taking Equation (13) back into Equation (9), we can find that Equation (9) can also be decoupled into n subproblems by rows. Considering the ith subproblem, we have:

$$\begin{aligned}
\min_{\mathbf{S}_{i.}} \quad & \sum_{p=1}^{k} W_{ip}^2(S_{ip} - Y_{ip})^2 + \gamma_1 \sum_{p,q=1}^{k} E_{pq}(S_{ip} - S_{iq})^2 + \rho \sum_{p=1}^{k} H_{ip}S_{ip}\\
s.t. \quad & 0 \le S_{ip} \le 1, \qquad (14)
\end{aligned}$$

where $E_{pq} = C_{pq}W_{ip}W_{iq}$, and $H_{ip} = \|\frac{\mathbf{F}_{i.}}{\sqrt{d_i}} - \frac{\mathbf{F}_{n+p.}}{\sqrt{d_{n+p}}}\|_2^2$.

Equation (14) is also a quadratic programming problem with bounded constraint and can be solved by the same way as solving Equation (10).

3.5.4 Optimizing **C**. When optimizing **C**, we need to optimize the following problem:

$$\min_{\mathbf{C}} \quad \gamma_1 \sum_{i=1}^{n} \sum_{p,q=1}^{k} C_{pq}(S_{ip} - S_{iq})^2 W_{ip} W_{iq} + \gamma_2 \|\mathbf{C} - \mathbf{Y}^T\mathbf{Y}\|_F^2,$$
$$s.t. \quad C_{pq} \geq 0, \quad \mathbf{C} = \mathbf{C}^T. \tag{15}$$

For simplicity, we first remove the symmetric constraint $\mathbf{C} = \mathbf{C}^T$, and then show that the learned **C** can satisfy the constraint naturally. We decouple Equation (15) into $k \times k$ independent subproblems. Considering the (p, q)th subproblem, we obtain:

$$\min_{C_{pq}} \quad C_{pq}G_{pq} + \tau(C_{pq} - K_{pq})^2,$$
$$s.t. \quad C_{pq} \geq 0, \tag{16}$$

where $G_{pq} = \sum_{i=1}^{n}(S_{ip} - S_{iq})^2 W_{ip}W_{iq}$, $\mathbf{K} = \mathbf{Y}^T\mathbf{Y}$ and $\tau = \frac{\gamma_2}{\gamma_1}$. Setting the derivative of Equation (16) w.r.t. C_{pq} to zero, we obtain:

$$C_{pq} = \frac{2\tau K_{pq} - G_{pq}}{2\tau}. \tag{17}$$

If $2\tau K_{pq} - G_{pq} < 0$, it is easy to verify that Equation (16) increases monotonically in the range $[0, \infty)$, and thus the optima is 0. Therefore, the optima of C_{pq} is

$$C_{pq} = \max\left(\frac{2\tau K_{pq} - G_{pq}}{2\tau}, 0\right). \tag{18}$$

Note that $G_{pq} = G_{qp}$ and $K_{pq} = K_{qp}$, and thus **C** computed by Equation (18) also satisfies that $\mathbf{C} = \mathbf{C}^T$.

3.6 Algorithm and Discussion

Algorithm 1 summarizes the whole process of SCCABG. The following Theorem provides the convergence analysis of SCCABG.

THEOREM 2. *With bounded hyper-parameter γ_1, Algorithm 1 always converges.*

PROOF. Since the self-paced parameter λ always changes, it is difficult to analyze the convergence of Algorithm 1 directly. To address this issue, we should focus on λ first. Notice that λ only directly influences **W**, and thus we review the solution of **W** again. We can rewrite the subproblem w.r.t. $\mathbf{W}_{i.}$ (Equation (10)) as a more concise form:

$$\min_{\mathbf{W}_{i.}} \quad \mathbf{W}_{i.}^T\mathbf{M}\mathbf{W}_{i.} - \lambda\mathbf{1}^T\mathbf{W}_{i.},$$
$$s.t. \quad \mathbf{0} \leq \mathbf{W}_{i.} \leq \mathbf{1}, \tag{19}$$

where $\mathbf{M} = \gamma_1\mathbf{B} + diag(\mathbf{A}_{i.})$ and $diag(\mathbf{A}_{i.})$ denotes the diagonal matrix whose diagonal vector is $\mathbf{A}_{i.}$, **1** and **0** denotes the vectors whose elements are all 1's and 0's, respectively. Note that for any p, q, $0 \leq S_{pq} \leq 1$, and thus $\mathbf{0} \leq \mathbf{A}_{i.} \leq \mathbf{1}$. According to Equation (18), we have $0 \leq C_{pq} \leq K_{pq}$ and $\mathbf{K} = \mathbf{Y}^T\mathbf{Y}$ which is constant, and thus all elements in **C** have lower and upper bounds. Therefore, **B** also has lower and upper bound. With bounded γ_1, **M** also has a lower and upper bound. Obviously, **M** is non-negative, and thus the lower bound of the elements in **M** is zero. Denote u as the upper bound of $tr(\mathbf{M}) + \mathbf{1}^T\mathbf{M}\mathbf{1}$, i.e., $u = \sup(tr(\mathbf{M}) + \mathbf{1}^T\mathbf{M}\mathbf{1})$. We have the following lemma.

LEMMA 1. *When $\lambda > u$, the global optima of Equation (19) is $\mathbf{W}_{i.} = \mathbf{1}$.*

Proof. We use the proof by contradiction. Here, we wish to prove that when $\lambda > u$, all elements in the global optima $\mathbf{W}_{i.}$ of Equation (19) (denoted by $\mathbf{w}^*$) are 1's. To apply the proof by contradiction, we assume the contrary is true, i.e., there exists at least one element in $\mathbf{w}^*$ (denoted by w_j^*), which is t where $t < 1$, and then we try to find a contradiction. To find the contradiction, we construct a new vector $\hat{\mathbf{w}}$ which is the same with $\mathbf{w}^*$, except that the jth element of $\hat{\mathbf{w}}$ (denoted by $\hat{w}_j$) is 1 instead of t.

Now, we compute the difference between $\hat{\mathbf{w}}^T\mathbf{M}\hat{\mathbf{w}} - \lambda\mathbf{1}^T\hat{\mathbf{w}}$ and $\mathbf{w}^{*T}\mathbf{M}\mathbf{w}^* - \lambda\mathbf{1}^T\mathbf{w}^*$:

$$\begin{aligned}
&\hat{\mathbf{w}}^T\mathbf{M}\hat{\mathbf{w}} - \lambda\mathbf{1}^T\hat{\mathbf{w}} - \mathbf{w}^{*T}\mathbf{M}\mathbf{w}^* + \lambda\mathbf{1}^T\mathbf{w}^* \\
&= 2\sum_{k\neq j}(\hat{w}_j - w_j^*)M_{kj}w_k^* + M_{jj}(\hat{w}_j^2 - (w_j^*)^2) - \lambda(\hat{w}_j - w_j^*) \\
&= 2\sum_{k\neq j}(1-t)M_{kj}w_k^* + M_{jj}(1-t^2) - \lambda(1-t) \\
&= (1-t)\left(2\sum_{k\neq j}M_{kj}w_k^* + M_{jj}(1+t) - \lambda\right) \\
&\quad < (1-t)\left(2\sum_{k\neq j}M_{kj} + 2M_{jj} - \lambda\right) \\
&\quad < (1-t)\left(2\sum_{k\neq j}M_{kj} + 2M_{jj} - u\right) \\
&\leq 0,
\end{aligned} \tag{20}$$

where the first inequality is due to that all M_{ij} is non-negative, $t < 1$, and all $w_k^* \leq 1$. The last inequality is due to $u \geq tr(\mathbf{M}) + \mathbf{1}^T\mathbf{M}\mathbf{1}$.

Equation (20) shows that, $\hat{\mathbf{w}}^T\mathbf{M}\hat{\mathbf{w}} - \lambda\mathbf{1}^T\hat{\mathbf{w}} < \mathbf{w}^{*T}\mathbf{M}\mathbf{w}^* - \lambda\mathbf{1}^T\mathbf{w}^*$, which means $\hat{\mathbf{w}}$ leads to a smaller objective value than the optima $\mathbf{w}^*$. It is a contradiction, which means the assumption (i.e., there exists at least one element in $\mathbf{w}^*$ is not 1) is false. Therefore, all elements in $\mathbf{w}^*$ should be 1's. This concludes the proof. □

Now get back to the proof of Theorem 2. Note that, in Algorithm 1, we double λ in each iteration. If Algorithm 1 does not converge before $\lambda > u$, when $\lambda > u$, according to Lemma 1, all elements in $\mathbf{W}_{i.}$ should be 1 to obtain the minimum of Equation (19).

Therefore, after several iterations, either Algorithm 1 converges or all elements of $\mathbf{W}$ become 1. When all elements of $\mathbf{W}$ reach 1, according to Algorithm 1, λ will be fixed, and thus λ and $\mathbf{W}$ will not change. We just need to focus on $\mathbf{F}$, $\mathbf{S}$, and $\mathbf{C}$. When updating $\mathbf{F}$, we find the closed-form solution of Equation (11), which makes the objective function Equation (9) decrease. When updating $\mathbf{S}$, we solve Equation (14) by the trust region reflective method, which can also make the objective function decrease. When updating $\mathbf{C}$, we also obtain its closed-form solution as Equation (18), and thus also makes the objective function decrease. Since Equation (9) has a lower bound, Algorithm 1 always converges. □

In fact, SCCABG often converges very fast. In our experiments, it often converges within 10 iterations.

Then, we analyze the space and time complexity of the proposed method. Since the graph we used is a bipartite graph and $\mathbf{C}$ is a $k \times k$ matrix, the space complexity of our method is $O(nk + k^2)$.

ALGORITHM 1: SCCABG Algorithm

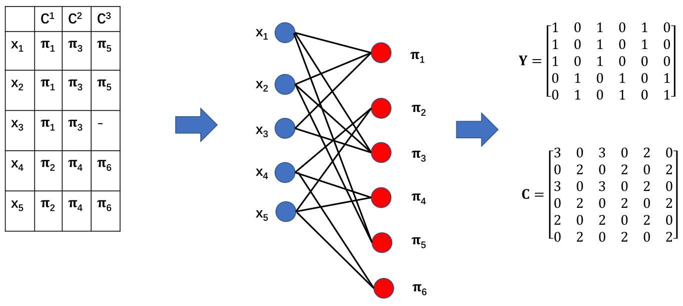
```
Input: $m$ base clustering results, number of clusters $c$, hyper-parameters $\gamma_1$ and $\gamma_2$.
Output: Consensus clustering results
 1: Construct the initial bipartite graph from $m$ based clustering and obtain $\mathbf{Y}$, and initialize the
    age parameter $\lambda = 0.5$, $\mathbf{S} = \mathbf{Y}$, and $\mathbf{C} = \mathbf{Y}^T\mathbf{Y}$.
 2: while not converge do
 3:     Update $\mathbf{W}$ by solving Equation (10).
 4:     Update $\mathbf{F}$ by Theorem 1.
 5:     Update $\mathbf{S}$ by solving Equation (14).
 6:     Update $\mathbf{C}$ by Equation (18)
 7:     Update the age parameter by $\lambda = \lambda * 2$, until all elements in $\mathbf{W}$ reach 1.
 8: end while
 9: Obtain the final bipartite graph $\mathcal{G}'$ from $\mathbf{S}$.
10: Obtain the final clustering result from the $c$ connective component in $\mathcal{G}'$.
```

For the time complexity, we analyze it step by step. In each iteration, when updating **W** and **S**, we need to solve n quadratic programming problem and each problem involves k variables, respectively. Note that, in practice, we set γ_1 a small value to make sure the quadratic programming problem is convex. Therefore, each subproblem costs $O(k^3)$ time and updating **W** and **S** costs $O(nk^3)$ time. Updating **F** needs $O(nk^2)$ as introduced in the previous subsection. When updating **C**, we need to compute **G** whose time complexity is $O(nk^2)$. Supposing the number of iterations is l, the whole time complexity is $O(lnk^3)$. In practice, we often have $n \gg k$, and thus the time complexity is linear with n. Notice that, when optimizing **S** and **W**, the n subproblems are independent, and thus they can be solved in parallel for a further speedup.

At last, we discuss the relationship between our self-paced consensus clustering and robust consensus clustering, which is very related to our self-paced schema. Robust consensus clustering extracts noises from original data or base results and recovers the clean results for consensus learning. For example, Tao et al. proposed spectral based robust consensus clustering methods [46, 47]; Huang et al. adopted probability trajectories to robust consensus clustering [15]; Wang et al. developed an ensemble method to handle incomplete data [52]. Although these methods often provide more robust results than conventional consensus clustering methods, they only pay attention to the outliers or noises without distinguishing between uncontaminated data. However, in our self-paced schema, the contaminated data can be viewed as the most unreliable ones. In addition, the uncontaminated data can also be handled in order of reliability. Therefore, our method provides a more sophisticated framework to handle all instances, no matter contaminated ones or uncontaminated ones. Moreover, in our framework, the reliability of each edge is changing in the process of learning. With the growth of λ, **W** will be increasingly large until it reaches 1, which means the edges become increasingly more reliable with learning. At last, we will obtain a reliable structured bipartite graph $\mathcal{G}'$ for clustering.

3.7 Hanldle Incomplete Data

In many real applications, it often happens that some data are missing in some base results, especially in some federated learning scenarios. For example, in the bank system of a city, most people are the customers of only a few banks in the city. When these banks do the base clustering locally, the data of many people are missing in each base result. Then, they upload the incomplete base results to the cloud server, and learn the consensus result from the incomplete base results in the cloud server. Most existing consensus clustering methods may fail because they need complete



	C¹	C²	C³
x_1	π_1	π_3	π_5
x_2	π_1	π_3	π_5
x_3	π_1	π_3	-
x_4	π_2	π_4	π_6
x_5	π_2	π_4	π_6

$$\mathbf{Y} = \begin{bmatrix} 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 \end{bmatrix}$$

$$\mathbf{C} = \begin{bmatrix} 3 & 0 & 3 & 0 & 2 & 0 \\ 0 & 2 & 0 & 2 & 0 & 2 \\ 3 & 0 & 3 & 0 & 2 & 0 \\ 0 & 2 & 0 & 2 & 0 & 2 \\ 2 & 0 & 2 & 0 & 2 & 0 \\ 0 & 2 & 0 & 2 & 0 & 2 \end{bmatrix}$$

Fig. 5. A toy example of the incomplete consensus clustering. There are five instances $\mathbf{x}_1, \ldots, \mathbf{x}_5$, wherein the first base clustering result, $\mathbf{x}_1, \ldots, \mathbf{x}_3$ belong to π_1, and $\mathbf{x}_4$ and $\mathbf{x}_5$ belong to π_2. The second base clustering result is the same as the first one. In the third base result, $\mathbf{x}_1$ and $\mathbf{x}_2$ belong to π_5, $\mathbf{x}_4$ and $\mathbf{x}_5$ belong to π_4, and $\mathbf{x}_3$ is missing. The middle side shows the initial bipartite graph, and the right side shows the corresponding $\mathbf{Y}$ and initial similarity matrix of clusters $\mathbf{C}$.

data for consensus learning. The problem is also different from the widely studied incomplete multi-view learning [33, 53, 54, 57]. Incomplete multi-view clustering needs to use the original features of data, whereas incomplete consensus clustering does not access the original features of data, which is more challenging and is quite under-explored.

Fortunately, since the proposed method applies self-paced learning to characterize the reliability of each edge and re-learns the bipartite graph according to its reliability, it is robust and can handle the incomplete setting easily. In more detail, when constructing the initial bipartite graph, we construct $\mathbf{Y}$ directly by observed base results. For example, if $\mathbf{x}_i$ is missing in the pth base clustering C^p, then for any clusters in C^p (i.e., $\pi_1^p, \ldots, \pi_{k_i}^p$), we do not set any edges between $\mathbf{x}_i$ and $\pi_1^p, \ldots, \pi_{k_i}^p$ initially. Then, we learn the final $\mathbf{S}$ by optimizing Equation (9). Due to the regularized term $\sum_{i=1}^n \sum_{p,q=1}^k C_{pq}(S_{ip} - S_{iq})^2 W_{ip} W_{iq}$, it can fill the initial bipartite graph automatically. To see this, if $\mathbf{x}_i$ is missing in the base clustering which contains π_q, and π_q is similar to π_p (i.e., C_{pq} is large), and it is reliable that $\mathbf{x}_i$ belongs to π_p (i.e., S_{ip} and W_{ip} are large), then by minimizing Equation (9), S_{iq} will be large with high reliability W_{iq}, which means that in the learned bipartite graph $\mathbf{S}$, it is more likely that there is an edge between $\mathbf{x}_i$ and π_q in the final graph, although $\mathbf{x}_i$ is missing in the initial graph. Since some values are missing in the initial bipartite graph, some values in the initial $\mathbf{W}$ are small, because they are unreliable due to the absence in the initial observation, and the initial similarity matrix $\mathbf{C}$ is often not accurate. With the learning, more and more edges become increasingly more reliable and $\mathbf{C}$ also becomes more accurate. Therefore, the adaptive cluster similarity measuring mechanism is also necessary to handle the incomplete data.

Figure 5 shows a simple toy example. Notice that $\mathbf{x}_3$ is missing in the base clustering C^3. Therefore, in the initial bipartite graph, there are no edges between $\mathbf{x}_3$ and π_5 or π_6. Despite this, the proposed one can automatically learn the relationship between $\mathbf{x}_3$ and π_5 or π_6 by minimizing the carefully designed term $\sum_{i=1}^n \sum_{p,q=1}^k C_{pq}(S_{ip} - S_{iq})^2 W_{ip} W_{iq}$. In the first iteration, we initialize $\mathbf{S} = \mathbf{Y}$, and then we evaluate the reliability $\mathbf{W}_{3.}$. Notice that π_5 is similar to π_1, which leads to a large C_{15}. When minimizing $C_{15}(S_{31} - S_{35})^2 W_{31} W_{35}$ w.r.t. W_{35}, since C_{15} is large and $S_{31} = 1$ and $S_{35} = 0$, W_{35} should be small, which means $S_{35} = 0$ is unreliable and thus there may be an edge between $\mathbf{x}_3$ and π_5. In fact, in the first iteration, after optimizing $\mathbf{W}$, we obtain $\mathbf{W}_{3.} = [1, 1, 1, 1, 0.3976, 1]$, where $W_{35} = 0.3976$ which is small.

Then, we optimize $\mathbf{S}$ by fixing other variables. When minimizing $C_{15}(S_{31} - S_{35})^2 W_{31} W_{35}$ w.r.t. S_{35}, since initial $S_{31} = 1$, it will pull S_{35} away from 0 to 1, i.e., it automatically fills the missing edges. In fact, after optimizing $\mathbf{S}$, we obtain $\mathbf{S}_{3\cdot} = [0.93, 0, 0.93, 0, 0.886, 0]$. Notice that although in the initial bipartite graph, S_{35} is missing and we initialize it as 0, after the first iteration, it can automatically fill it with a large value (i.e., 0.886) which means there should be an edge between $\mathbf{x}_3$ and π_5. It well demonstrates that the proposed method is robust and can handle the incomplete consensus clustering setting.

4 EXPERIMENTS

In this section, we first use a toy example to show the effectiveness of the proposed method, and then we compare it with other state-of-the-art consensus clustering methods on several benchmark datasets.

4.1 Toy Example

Before comparing with other methods on benchmark datasets, we first provide a toy example to show the effectiveness of SCCABG intuitively.

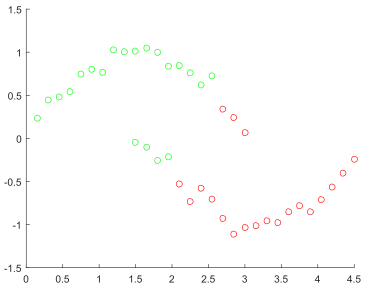
We use the two-moon data as the toy example. We run k-means five times with different numbers of clusters (in the range $2, 4, \ldots, 10$) as the five base clustering results. The five base results are shown in Figure 6(a)–(e). It can be seen that k-means cannot handle this non-linear manifold data well. We use the five base results to construct the initial bipartite graph $\mathcal{G}$ and show it in Figure 6(f). The red points indicate the instances and the black points indicate the clusters. We can find that, in Figure 6(f), all instances and clusters are entangled together because of the unreliable base results. Then, we run our SCCABG on $\mathcal{G}$ to learn the structured bipartite graph $\mathcal{G}'$. Our SCCABG converges in nine iterations. Figure 6(g) and 6(h) show the learned $\mathcal{G}'$ in the 4th and 9th iteration, respectively. Note that in the 9th iteration, our method can already discover the two-moon structure of data.

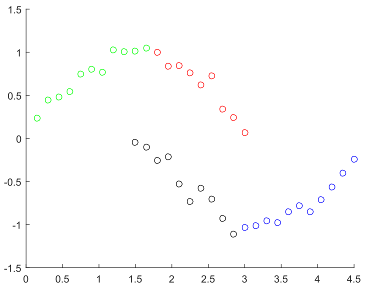
From Figure 6, we have some interesting observations. Firstly, although all base k-means methods fail on the non-linear manifold data, our consensus clustering can handle it well. Secondly, from Figure 6(f) and (g), we find that in the first several iterations, our method prefers to propagate the connective information on the graph, so that it will add some new edges to the graph. From Figure 6(g) and (h), it is shown that in the last several iterations, SCCABG prefers to partition the graph by removing some edges to make sure the learned graph contains just c connective components. The process is reasonable. If we partition the graph before propagating connective information, the graph will be broken into a lot of pieces and the propagation will be more difficult.

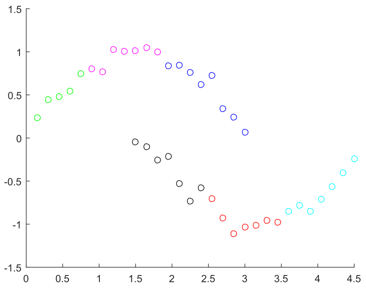
4.2 Benchmark Datasets

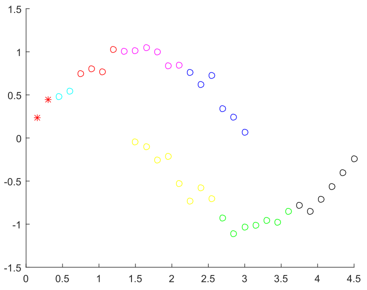
We conduct experiments on the following eight benchmark datasets:

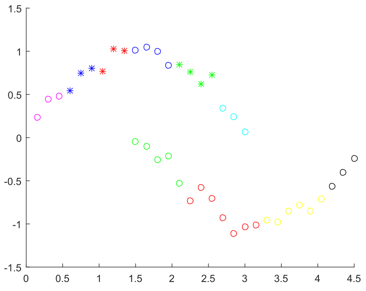
— ALLAML [12]. It is a dataset that consists of 72 bone marrow samples for leukemia diagnosis. It contains 7,129 probes from 6,817 human genes.
— GLIOMA [26]. It is a microarray data for glioma subtypes. It contains 50 samples with 4,434 features in four subtypes of glioma.
— Tr41 [59]. It is a text dataset from **Text Retrieval Conference** (**TREC**). It contains 878 texts with 7,454 features in 10 classes.
— Tdt2 [7]. It is a text dataset containing 10,212 documents with 36,771 features in 96 categories.
— TOX [26]. It is a gene dataset containing 171 instances with 5,748 features in four classes.
— K1b [59]. It is a text dataset from the WebACE project, where each document is a web page. It contains 2,340 documents with 21,839 features in five classes.

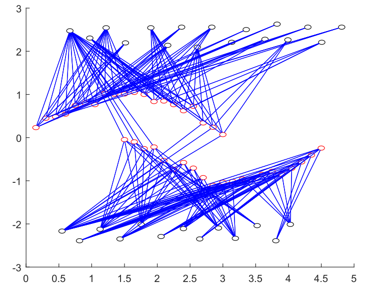

(a) K-means base result

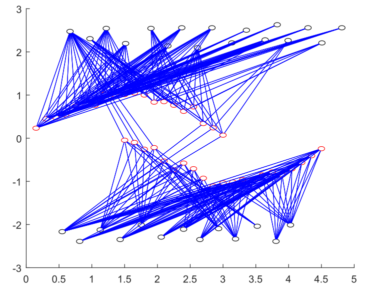

(b) K-means base result


(c) K-means base result


(d) K-means base result


(e) K-means base result


(f) Initial bipartite graph


(g) Learned bipartite graph in the 4th iteration of SCCABG

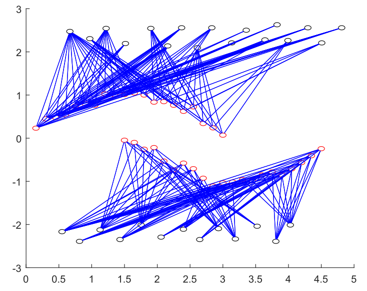

(h) The final learned bipartite graph of SCCABG (in the 9th iteration)

Fig. 6. Toy example results on the two-moon data. (a)–(e) show the 5 k-means base clustering results. (f) shows the initial bipartite graph $\mathcal{G}$ constructed by the five base results. (g) shows the learned bipartite graph $\mathcal{G}'$ in the 4th iteration of SCCABG. (h) shows the final learned bipartite graph $\mathcal{G}'$ of SCCABG (it is obtained in the 9th iteration).

Table 1. Description of the Datasets

	#instances	#features	#classes
ALLAML	72	7,129	2
GLIOMA	50	4,434	4
Tr41	878	7,454	10
Tdt2	10,212	36,771	96
TOX	171	5,748	4
K1b	2,340	21,839	6
Medical	706	1,449	17
USPS	11,000	256	10

— Medical [27]. It is originally a multi-label text dataset. Following [27, 66], we use the 706 instances with single label. It contains 1,449 features and 17 classes.
— USPS.[1] It is an image dataset containing 11,000 16×16 handwritten digit images in 10 classes.

The information of these datasets is summarized in Table 1.

4.3 Experimental Setup

Following the setup in [51, 66], we also apply k-means to generate the base clustering results. In more detail, we run k-means 200 times with different random initializations to obtain 200 base results. Then, we partition them into 10 subsets, with 20 in each one. Next, we run consensus clustering methods on each subset and report the average results on the 10 subsets. We compare the proposed SCCABG with the following algorithms:

— **KM.** It is the mean result of all base k-means clustering.
— **KM-best.** It is the best result among all base k-means results.
— **CSPA** [43]. **Cluster-based Similarity Partitioning Algorithm** (**CSPA**) adopts the relationship between instances to construct a measure of pairwise similarity and applies the similarity to recluster the data to obtain the final consensus clustering result.
— **HGPA** [43]. **HyperGraph Partitioning Algorithm** (**HGPA**) combines the base results to generate a hypergraph and applies a constrained minimum cut objective on the hypergraph for consensus clustering.
— **MCLA**[43]. **Meta-CLustering Algorithm** (**MCLA**) transforms the consensus clustering into a cluster correspondence problem. Then, the groups of clusters, which are called meta-clusters, are identified and combined.
— **NMFC** [28]. **Nonnegative Matrix Factorization based Consensus clustering** (**NMFC**) uses nonnegative matrix factorization to learn the consensus clustering result.
— **RCE** [66]. **Robust Clustering Ensemble** (**RCE**) explicitly extracts the noises on the connective matrices to recover the clean connective matrices, and then it ensembles the clean matrices by minimizing the KL-divergence between the base matrices and consensus matrix.
— **MEC** [45], **Multi-view Ensemble Clustering** (**MEC**) is a robust consensus clustering method on connective matrices, which uses sparse and low-rank decomposition to integrate base clustering and extract the noises. Notice that, although it is proposed to handle multi-view clustering, since it only takes the connective matrices as inputs without access to the original data, it can be used as a baseline in our consensus clustering task.

[1] https://cs.nyu.edu/~roweis/data.html.

Table 2. Average ACC and Standard Deviation on All Datasets

Methods	ALLAML	GLIOMA	Tr41	Tdt2	Tox	K1b	Medical	USPS
KM	0.6545 ±0.0644	0.4239 ±0.0347	0.5626 ±0.0717	0.4104 ±0.0188	0.4229 ±0.0322	0.6726 ±0.0980	0.3996 ±0.0364	0.4435 ±0.0289
KM-best	0.7292 ±0.0118	0.4880 ±0.0193	0.6946 ±0.0468	0.4460 ±0.0081	0.4825 ±0.0152	0.8559 ±0.0246	0.4707 ±0.0268	0.5057 ±0.0141
CSPA [43]	0.6583 ±0.0134	0.4100 ±0.0271	0.5213 ±0.0282	0.2850 ±0.0047	0.4246 ±0.0373	0.4531 ±0.0027	0.3500 ±0.0150	0.4475 ±0138
HGPA [43]	0.5444 ±0.0403	0.4180 ±0.0394	0.4894 ±0.0549	0.2959 ±0.0041	0.3854 ±0.0286	0.5326 ±0.0469	0.2950 ±0.0283	0.1004 ±0.0000
MCLA [43]	0.6722 ±0.0149	0.4000 ±0.0133	0.5698 ±0.0392	0.4000 ±0.0088	0.4152 ±0.0242	0.7383 ±0.0913	0.4017 ±0.0197	0.4438 ±0.0239
NMFC [28]	0.6722 ±0.0149	0.4140 ±0.0212	0.6323 ±0.0370	0.3716 ±0.0169	0.4269 ±0.0226	0.5860 ±0.0348	0.3789 ±0.0183	0.4362 ±0.0320
RCE [66]	0.6708 ±0.0161	0.4260 ±0.0097	0.6391 ±0.0227	–	0.4105 ±0.0264	0.6887 ±0.0372	0.3851 ±0.0301	–
MEC [45]	0.6056 ±0.0360	0.3940 ±0.0366	0.6559 ±0.0444	–	0.4304 ±0.0310	0.8190 ±0.0901	0.3627 ±0.0167	–
LWEA [16]	0.6736 ±0.0210	0.4320 ±0.0140	0.6719 ±0.0473	0.5744 ±0.0273	0.4234 ±0.0127	0.8279 ±0.0760	0.4208 ±0.0076	0.4111 ±0.0108
LWGP [16]	0.6750 ±0.0176	0.4320 ±0.0103	0.6483 ±0.0340	0.4288 ±0.0103	0.4193 ±0.0259	0.7172 ±0.0773	0.4047 ±0.0141	0.4477 ±0.0215
RSEC [46]	0.5917 ±0.0908	0.4180 ±0.0503	0.6367 ±0.0435	0.3029 ±0.0772	0.4041 ±0.0243	0.8409 ±0.0511	0.3490 ±0.0303	0.3032 ±0.0624
DREC [60]	0.6819 ±0.0249	0.4280 ±0.0103	0.6243 ±0.0271	0.3657 ±0.0036	0.4205 ±0.0408	0.6462 ±0.0654	0.3926 ±0.0195	0.4354 ±0.0161
SPCE [63]	0.6861 ±0.0238	0.4420 ±0.0199	**0.7346** ±0.0757	0.6653 ±0.0741	**0.4485** ±0.0185	0.8663 ±0.0201	0.4534 ±0.0127	0.3259 ±0.0339
TRCE [64]	0.6917 ±0.0333	0.4400 ±0.0063	0.6812 ±0.0403	0.6273 ±0.0457	**0.4491** ±0.0189	**0.8899** ±0.0134	0.4356 ±0.0227	0.4505 ±0.0168
CESHL [68]	0.6736 ±0.0220	0.4420 ±0.0063	0.6952 ±0.0658	0.5301 ±0.0479	**0.4404** ±0.0247	0.8619 ±0.0501	0.4574 ±0.0205	0.4495 ±0.0237
SCCABG	**0.7139** ±0.0450	**0.4620** ±0.0247	**0.7244** ±0.0505	**0.8350** ±0.0672	**0.4427** ±0.0168	**0.8881** ±0.0207	**0.4720** ±0.0140	**0.4620** ±0.0193

The bold font indicates that the difference is statistically significant (i.e., the p-value of t-test is smaller than 0.05).

— **LWEA** [16]. **Locally Weighted Evidence Accumulation** (**LWEA**) designs a local weighting strategy and applies a hierarchical agglomerative consensus clustering method based on such a local weighting strategy.

— **LWGP** [16]. **Locally Weighted Graph Partitioning** (LWGP) designs a local weighting strategy and applies a graph partition consensus clustering method on such a local weighting strategy.

— **RSEC** [46]. **Robust Spectral Ensemble Clustering** (**RSEC**) is a spectral based robust consensus clustering method on connective matrices, which can reduce the noises on the connective matrices.

— **DREC** [60]. **Dense Representation Ensemble Clustering** (**DREC**) learns a dense representation from base results and applies it to construct a pairwise similarity matrix for the consensus clustering.

Table 3. Average NMI and Standard Deviation on All Datasets

Methods	ALLAML	GLIOMA	Tr41	Tdt2	Tox	K1b	Medical	USPS
KM	0.0882 ±0.0490	0.1629 ±0.0391	0.5843 ±0.0512	0.6111 ±0.0072	0.1374 ±0.0397	0.5493 ±0.0608	0.4209 ±0.0286	0.4406 ±0.0166
KM-best	0.1772 ±0.0547	0.2347 ±0.0227	0.6713 ±0.0253	0.6240 ±0.0055	0.2164 ±0.0269	0.6853 ±0.0302	0.4806 ±0.0194	0.4762 ±0.0047
CSPA [43]	0.0815 ±0.0137	0.1716 ±0.0281	0.5919 ±0.0154	0.5589 ±0.0028	0.1436 ±0.0446	0.4071 ±0.0067	0.3992 ±0.0125	0.4342 ±0.0163
HGPA [43]	0.0110 ±0.0141	0.1509 ±0.0362	0.5084 ±0.0351	0.5385 ±0.0088	0.1083 ±0.0211	0.3917 ±0.0742	0.3613 ±0.0329	0.0000 ±0.0000
MCLA [43]	0.0909 ±0.0117	0.1327 ±0.0291	0.6044 ±0.0242	0.6070 ±0.0044	0.1329 ±0.0165	0.5944 ±0.0695	0.4296 ±0.0185	0.4446 ±0.0149
NMFC [28]	0.0909 ±0.0117	0.1550 ±0.0270	0.6512 ±0.0188	0.5930 ±0.0042	0.1434 ±0.0286	0.4995 ±0.0212	0.4259 ±0.0187	0.4471 ±0.0138
RCE [66]	0.0899 ±0.0125	0.1624 ±0.0163	0.6499 ±0.0183	–	0.1344 ±0.0204	0.6068 ±0.0104	0.4475 ±0.0190	–
MEC [45]	0.0485 ±0.0429	0.1312 ±0.0433	0.6758 ±0.0270	–	0.1313 ±0.0308	0.6818 ±0.0707	0.4089 ±0.0289	–
LWEA [16]	0.0935 ±0.0192	0.1686 ±0.0207	0.6666 ±0.0394	0.7183 ±0.0091	0.1236 ±0.0289	0.6948 ±0.0645	0.4185 ±0.0148	0.4211 ±0.0074
LWGP [16]	0.0932 ±0.0142	0.1682 ±0.0177	0.6535 ±0.0281	0.6266 ±0.0053	0.1333 ±0.0280	0.6115 ±0.0493	0.4266 ±0.0109	0.4452 ±0.0136
RSEC [46]	0.0495 ±0.0491	0.1544 ±0.0455	0.6449 ±0.0483	0.4670 ±0.0342	0.1184 ±0.0137	0.6615 ±0.0528	0.4036 ±0.0487	0.2774 ±0.0809
DREC [60]	0.1006 ±0.0218	0.1641 ±0.0189	0.6514 ±0.0169	0.5985 ±0.0013	0.1394 ±0.0276	0.5774 ±0.0410	0.4510 ±0.0203	0.4358 ±0.0098
SPCE [63]	0.1237 ±0.0126	0.3010 ±0.0152	0.6846 ±0.0589	0.7125 ±0.0427	0.1961 ±0.0139	0.6993 ±0.0599	**0.4549** 0.0130	0.3362 ±0.0494
TRCE [64]	0.1150 ±0.0278	0.2289 ±0.0193	0.6849 ±0.0333	0.7275 ±0.0236	0.1541 ±0.0298	**0.7496** ±0.0218	**0.4622** ±0.0266	0.4506 ±0.0103
CESHL [68]	0.0936 ±0.0197	0.1888 ±0.0118	**0.6968** ±0.0471	0.6091 ±0.0585	0.1439 ±0.0285	0.7303 ±0.0557	0.3772 ±0.0119	0.4532 ±0.0176
SCCABG	**0.1459** ±0.0331	**0.3107** ±0.0315	**0.6920** ±0.0445	**0.8391** ±0.0407	**0.2642** ±0.0063	**0.7559** ±0.0327	0.4205 ±0.0148	**0.4655** ±0.0051

The bold font indicates that the difference is statistically significant (i.e., the p-value of t-test is smaller than 0.05).

— **SPCE** [63]. **Self-Paced Clustering Ensemble** (**SPCE**) learns a consensus matrix from multiple connective matrices with self-paced multiple graph learning.
— **TRCE** [64]. **Tri-level Robust Clustering Ensemble** (**TRCE**), which learns a consensus clustering result by a multiple graph learning method. When fusing the multiple graphs, it considers three levels of robustness, i.e., the base result level, the graph level, and the data level.
— **CESHL** [68]. **Clustering Ensemble with Structured Hypergraph Learning** (**CESHL**) is a consensus clustering method with hypergraph learning. Different from the conventional hypergraph based method, it learns a dynamical structured hypergraph in the process of consensus learning.

Table 4. Average ARI and Standard Deviation on All Datasets

Methods	ALLAML	GLIOMA	Tr41	Tdt2	Tox	K1b	Medical	USPS
KM	0.0997 ±0.0103	0.0776 ±0.0060	0.4309 ±0.0256	0.2110 ±0.0025	0.1021 ±0.0381	0.4758 ±0.1210	0.2368 ±0.0310	0.2911 ±0.0047
KM-best	0.1990 ±0.0213	0.1595 ±0.0306	0.5917 ±0.0577	0.2388 ±0.0103	0.1794 ±0.0307	0.7183 ±0.0432	0.3061 ±0.0336	0.3384 ±0.0128
CSPA [43]	0.0850 ±0.0371	0.0768 ±0.0181	0.4274 ±0.0190	0.1422 ±0.0010	0.1112 ±0.0391	0.2551 ±0.0042	0.1951 ±0.0110	0.2843 ±0.0064
HGPA [43]	0.0066 ±0.0161	0.0571 ±0.0275	0.3412 ±0.0483	0.1461 ±0.0038	0.0752 ±0.0167	0.2896 ±0.0628	0.1518 ±0.0282	-0.0008 ±0.0000
MCLA [43]	0.1101 ±0.0523	0.0795 ±0.0115	0.4511 ±0.0421	0.1908 ±0.0086	0.1048 ±0.0213	0.5521 ±0.1242	0.2303 ±0.0231	0.2934 ±0.0242
NMFC [28]	0.1079 ±0.0209	0.0779 ±0.0247	0.5271 ±0.0262	0.1792 ±0.0058	0.1132 ±0.0258	0.3693 ±0.0301	0.2230 ±0.0211	0.2896 ±0.0203
RCE [66]	0.1070 ±0.0215	0.0918 ±0.0067	0.5360 ±0.0227	–	0.0992 ±0.0221	0.5138 ±0.0300	0.2344 ±0.0231	–
MEC [45]	0.0655 ±0.1012	0.0913 ±0.0219	0.5024 ±0.0760	–	0.0963 ±0.0227	0.6562 ±0.1241	0.1897 ±0.0330	–
LWEA [16]	0.1106 ±0.0302	0.0955 ±0.0112	0.5413 ±0.0528	0.4340 ±0.0020	0.1042 ±0.0181	0.6968 ±0.1142	**0.2510** ±0.0060	0.3085 ±0.0147
LWGP [16]	0.1083 ±0.0268	0.0935 ±0.0067	0.5462 ±0.0338	0.2133 ±0.0073	0.1030 ±0.0252	0.5234 ±0.1120	0.2447 ±0.0101	0.3074 ±0.0155
RSEC [46]	-0.0130 ±0.0602	0.0648 ±0.0452	0.4812 ±0.0711	0.1804 ±0.0791	0.0791 ±0.0218	0.6990 ±0.0829	0.1884 ±0.0338	0.1421 ±0.0787
DREC [60]	0.1231 ±0.0361	0.0923 ±0.0073	0.5216 ±0.0293	0.1812 ±0.0038	0.1091 ±0.0312	0.4633 ±0.0661	0.2430 ±0.0190	0.3052 ±0.0167
SPCE [63]	0.1053 ±0.0623	0.1180 ±0.0160	**0.6361** ±0.0931	0.4201 ±0.0791	**0.1265** ±0.0251	0.7432 ±0.0601	**0.2520** ±0.0088	0.0063 ±0.0200
TRCE [64]	0.1065 ±0.0570	0.0952 ±0.0150	0.5576 ±0.0698	0.2039 ±0.0113	0.0988 ±0.0230	0.6281 ±0.1785	**0.2514** ±0.0182	0.2913 ±0.0186
CESHL [68]	0.0960 ±0.0503	0.0861 ±0.0346	0.4564 ±0.2368	0.2394 ±0.0895	0.0901 ±0.0276	0.4534 ±0.3724	0.2155 ±0.0767	0.2807 ±0.0498
SCCBAG	**0.1917** ±0.0406	**0.1280** ±0.0195	0.6150 ±0.0659	**0.5281** ±0.1262	0.1161 ±0.0245	**0.7914** 0.0457	**0.2562** ±0.0056	**0.3177** ±0.0147

The bold font indicates that the difference is statistically significant (i.e., the p-value of t-test is smaller than 0.05).

For a fair comparison, we set the number of clusters to the true number of classes for all algorithms on all datasets. λ in our method is automatically adjusted as introduced in Algorithm 1. The parameter ρ is also automatically determined. In more detail, it is initialized by $\rho = 1$. Then, if the rank of $\mathbf{L}$ is larger than $n + k - c$, i.e., the rank constraint is not strong enough, we double it. If its rank is smaller than $n + k - c$, i.e., the rank constraint is too strong, we reduce it by half. We tune the hyper-parameter γ_1 in the range $[10^{-5}, 10^0]$, because as discussed before, γ_1 should not be too large to guarantee the convexity of the subproblems. The hyper-parameter γ_2 is tuned in $[10^{-3}, 10^3]$ by grid search. We use **Accuracy** (**ACC**), **Normalized Mutual Information** (**NMI**), **Adjust Rand Index** (**ARI**), and **F1 Score** (**F1**) to evaluate the clustering performance. To validate the statistical significance of the results, we also do t-test on the results.

The experiments are conducted using MATLAB on a PC with Windows 10, 4.2-GHz CPU, and 32-GB memory.

Table 5. Average F1 Score and Standard Deviation on All Datasets

Methods	ALLAML	GLIOMA	Tr41	Tdt2	Tox	K1b	Medical	USPS
KM	0.5779 ±0.0046	0.3812 ±0.0043	0.5217 ±0.0195	0.2361 ±0.0024	0.3516 ±0.0079	0.6373 ±0.0223	0.3298 ±0.0063	0.3657 ±0.0040
KM-best	0.6269 ±0.0117	0.4356 ±0.0254	0.6568 ±0.0488	0.2637 ±0.0105	0.4116 ±0.0296	0.8200 ±0.0300	0.3912 ±0.0261	0.4074 ±0.0114
CSPA [43]	0.5686 ±0.0175	0.3513 ±0.0162	0.4926 ±0.0246	0.1577 ±0.0010	0.3351 ±0.0280	0.4309 ±0.0062	0.2692 ±0.0089	0.3564 ±0.0058
HGPA [43]	0.5315 ±0.0076	0.3483 ±0.0119	0.4031 ±0.0371	0.1502 ±0.0025	0.2905 ±0.0228	0.4825 ±0.0321	0.2291 ±0.0152	0.1045 ±0.0000
MCLA [43]	0.5815 ±0.0247	0.3739 ±0.0145	0.5125 ±0.0426	0.1993 ±0.0114	0.3358 ±0.0203	0.6280 ±0.1019	0.3041 ±0.0136	0.3674 ±0.0205
NMFC [28]	0.5798 ±0.0104	0.3778 ±0.0157	0.5936 ±0.0445	0.1973 ±0.0612	0.3487 ±0.0174	0.5874 ±0.0678	0.2967 ±0.0199	0.3651 ±0.0178
RCE [66]	0.5794 ±0.0107	0.3836 ±0.0062	0.6060 ±0.0187	–	0.3464 ±0.0204	0.6566 ±0.0267	0.3148 ±0.0209	–
MEC [45]	0.6143 ±0.0528	0.3885 ±0.0123	0.6236 ±0.0411	–	0.3573 ±0.0206	0.7627 ±0.1009	0.2892 ±0.0284	–
LWEA [16]	0.5811 ±0.0149	0.3868 ±0.0095	0.6203 ±0.0426	0.4644 ±0.0200	0.3642 ±0.0142	0.8028 ±0.0840	0.3460 ±0.0065	0.3845 ±0.01154
LWGP [16]	0.5801 ±0.0135	0.3859 ±0.0065	0.6149 ±0.0292	0.2340 ±0.0077	0.3473 ±0.0185	0.6778 ±0.0788	0.3340 ±0.0128	0.3800 ±0.0136
RSEC [46]	0.6210 ±0.0405	0.3733 ±0.0322	0.5594 ±0.0847	0.2399 ±0.0763	0.3213 ±0.0233	0.6924 ±0.0795	0.2569 ±0.0367	0.2673 ±0.0525
DREC [60]	0.5875 ±0.0180	0.3843 ±0.0072	0.5916 ±0.0254	0.2022 ±0.0052	0.3501 ±0.0238	0.6113 ±0.0546	0.3161 ±0.0166	0.3779 ±0.0146
SPCE [63]	0.5809 ±0.0270	0.3899 ±0.0147	0.6726 ±0.0885	0.3462 ±0.0442	0.3712 ±0.0216	0.8010 ±0.0433	0.3491 0.0635	0.1859 ±0.0131
TRCE [64]	0.5801 ±0.0268	0.3898 ±0.0095	0.6282 ±0.0603	0.2435 ±0.0131	0.3484 ±0.0205	0.7739 ±0.0965	0.3431 ±0.0150	0.3662 ±0.0163
CESHL [68]	0.5749 ±0.0233	0.3926 ±0.0100	0.5771 ±0.1503	0.3357 ±0.0637	0.3475 ±0.0182	0.7442 ±0.1400	0.3393 ±0.0537	0.3605 ±0.0334
SCCABG	**0.6547** ±0.0349	**0.4372** ±0.0122	**0.6850** ±0.0522	**0.5831** ±0.1033	**0.3967** ±0.0121	**0.8737** ±0.0257	**0.3665** ±0.0053	**0.3911** ±0.0138

The bold font indicates that the difference is statistically significant (i.e., the p-value of t-test is smaller than 0.05).

4.4 Experimental Results

Tables 2–5 show the average ACC, NMI, ARI, and F1 results and standard deviation of all consensus clustering methods, respectively. The bold font indicates that the difference is statistically significant (i.e., the p-value of t-test is smaller than 0.05). Note that, because of their high space complexity, RCE and MEC run out of memory on the large dataset Tdt2 and USPS.

From these tables, we find that:

— Many consensus clustering methods, including ours, outperform the KM, which indicates the effectiveness of the consensus clustering, i.e., by integrating multiple weak base clustering results, we can learn a better consensus result.

— Compared with other consensus clustering methods, SCCABG outperforms them on most datasets, which demonstrates its superiority. Especially on the large dataset Tdt2, SCCABG

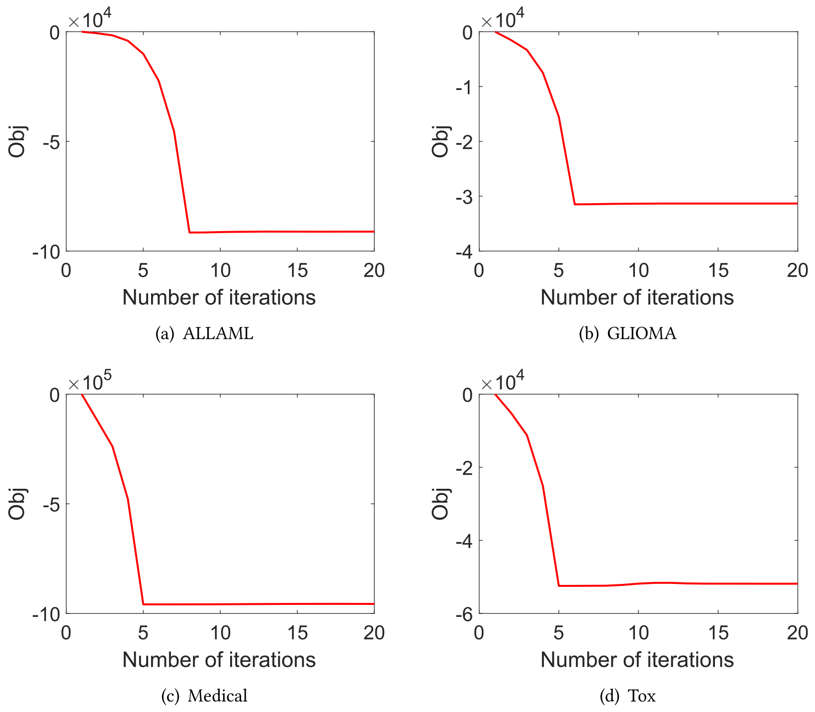
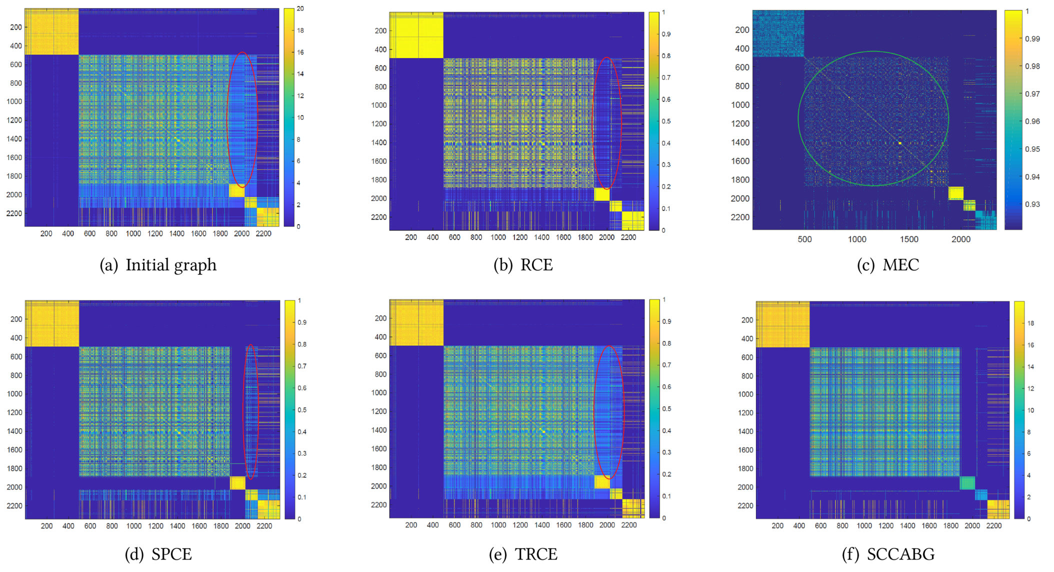


Fig. 7. Convergence curves on ALLAML, GLIOMA, Lung, and Tox datasets.

achieves 25.5% and 15.3% improvements compared with the second best baseline methods on ACC and NMI, respectively. Even compared with the robust methods (RCE, MEC, RSEC, and TRCE), SCCABG also performs better, because it can handle data more sophisticatedly, i.e., the self-paced framework cannot only recognize noises or outliers but also handle those uncontaminated but difficult instances, as discussed in Section 3.6.

— On most datasets, the proposed SCCABG is at least closed to KM-best. It means that SCCABG can provide a stable good clustering result compared with base single clustering. Notice that, SCCABG only takes base clustering results as inputs without accessing original data or labels. On some datasets, SCCABG can even perform better than KM-best, which means SCCABG can alleviate the side effects caused by unreliable results and apply the useful information in the unreliable ones to further improve the performance of the reliable ones. It well demonstrates the effectiveness of our SCCABG.

In Figure 7, we show the convergence curves of SCCABG on ALLAML, GLIOMA, Medical, and Tox datasets. The results on other datasets are similar. From Figure 7, we can find that SCCABG often converges within 10 iterations, which demonstrates the claim in Section 3.6. Notice that the curves do not look as smooth as many other machine learning methods. It is because of the special update of λ in our algorithm. The detailed reason is as follows. At the first several iterations, λ in the term $-\lambda\|\mathbf{W}\|_1$ is updated as $\lambda \leftarrow \lambda * 2$, and thus the objective function decreases faster and faster with iterations. After several iterations (e.g., five iterations on Medical dataset), all values in $\mathbf{W}$ reach 1, and according to our algorithm (i.e., Line 7 in Algorithm 1), λ will not change after the 5th iteration. By the first several iterations, the variables $\mathbf{F}$, $\mathbf{S}$, and $\mathbf{C}$ have almost converged, and thus the objective function hardly changes in the following iterations with a fixed λ.



(a) Initial graph (b) RCE (c) MEC

(d) SPCE (e) TRCE (f) SCCABG

Fig. 8. The visualization of graph matrices learned by RCE, MEC, SPCE, TRCE, and SCCABG. (a) shows the initial graph matrix which is constructed by $\mathbf{YY}^T$. (b)–(e) show the learned graph matrices from RCE, MEC, SPCE, and TRCE, respectively. (f) shows the graph constructed from the learned bipartite graph of SCCABG by $\mathbf{SS}^T$.

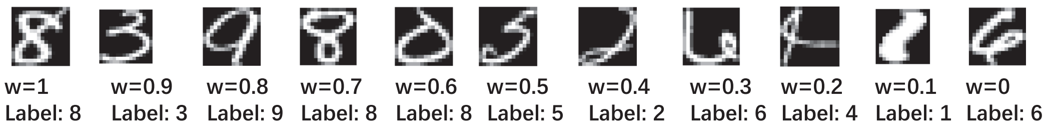


Fig. 9. An example of the data with its weight $\mathbf{w}$ on USPS dataset.

Since our method and many other methods are graph based methods, we also show the visualization of the learned final graph matrices of our SCCABG and other graph based methods (i.e., RCE, MEC, SPCE, TRCE) in Figure 8. The ideal graph matrix should have a clean block diagonal structure. Figure 8(a) shows the initial graph constructed from input base clustering results directly by $\mathbf{YY}^T$. The graph is not clean enough, which can be seen from the zone denoted by the red circle. Figure 8(b)–(e) show the learned graph matrix of RCE, MEC, SPCE, and TRCE, respectively. Figure 8(f) shows the graph constructed from the learned bipartite graph of SCCABG by $\mathbf{SS}^T$. The learned graph of our method is cleaner than other methods. For example, from the zones denoted by the red circles of RCE, SPCE, and TRCE, we can see that in SCCABG, the second, third and fourth clusters can be divided more clearly. From the zone denoted by the green circle of MEC, we can find that the second cluster in MEC is fuzzier than SCCABG.

Since we apply self-paced learning, we involve the weight matrix $\mathbf{W}$ to represent the difficulty or the reliability of data. Here we show an example on USPS dataset, which is a handwritten digit image dataset. Notice that $\mathbf{W}$ is an n-by-k matrix, where each element in $\mathbf{W}$ represents the reliability of an edge in the bipartite graph. To obtain the difficulty or reliability of each data, we need to calculate the weight vector $\mathbf{w} \in [0, 1]^n$ by computing the mean of each row of $\mathbf{W}$. The larger w_i is, the easier or more reliable $\mathbf{x}_i$ is and the earlier $\mathbf{x}_i$ is involved in the consensus learning. Figure 9 shows some example images with different w_i from 1 to 0. From Figure 9, we can find that

Table 6. Clustering Results Compared with Degenerated Versions

Methods	Measures	ALLAML	GLIOMA	Tr41	Tdt2	Tox	K1b	Medical	USPS
SCCBG-W	ACC	0.6681 ±0.0122	0.4080 ±0.0301	0.6136 ±0.1113	0.5011 ±0.0352	0.4053 ±0.0447	0.8405 ±0.0643	0.3980 ±0.0826	0.4496 ±0.0281
	NMI	0.0894 ±0.0104	0.1567 ±0.0401	0.6039 ±0.1461	0.6433 ±0.0116	0.1239 ±0.0433	0.6888 ±0.0667	0.3220 ±0.1076	0.4509 ±0.0231
	ARI	0.1030 ±0.0174	0.0665 ±0.0504	0.4799 ±0.1493	0.2353 ±0.0342	0.0973 ±0.0306	0.6963 ±0.1030	0.2066 ±0.0673	0.3074 ±0.0161
	F1	0.5755 ±0.0084	0.3887 0.0074	0.5791 ±0.1049	0.2918 ±0.0278	0.3568 ±0.0239	0.8098 ±0.0766	0.3275 ±0.0469	0.3806 ±0.0141
SCCBG [62]	ACC	0.6861 ±0.0279	0.4500 ±0.0343	0.6973 ±0.0644	0.7164 ±0.0689	0.4339 ±0.0182	0.8663 ±0.0369	0.4592 ±0.0151	0.4423 ±0.0091
	NMI	0.1252 ±0.0330	0.2163 ±0.0674	**0.6847** ±0.0547	0.7548 ±0.0459	0.2131 ±0.0513	0.7212 ±0.0512	0.3918 ±0.0244	**0.4603** ±0.0117
	ARI	0.1284 ±0.0464	0.1064 ±0.0085	0.5622 ±0.0765	0.4931 ±0.1572	0.1030 ±0.0282	0.7617 ±0.0596	**0.2484** ±0.0074	**0.3081** ±0.0155
	F1	**0.6441** ±0.0488	0.4143 ±0.0131	0.6410 ±0.0613	0.5460 ±0.1304	0.3837 ±0.0160	0.8550 ±0.0387	**0.3609** ±0.0062	0.3813 ±0.0128
SCCABG	ACC	**0.7139** ±0.0450	**0.4620** ±0.0247	**0.7244** ±0.0505	**0.8350** ±0.0672	**0.4427** ±0.0168	**0.8881** ±0.0207	**0.4720** ±0.0140	**0.4620** ±0.0193
	NMI	**0.1459** ±0.0331	**0.3107** ±0.0315	**0.6920** ±0.0445	**0.8391** ±0.0407	**0.2642** ±0.0063	**0.7559** ±0.0327	**0.4205** ±0.0148	**0.4655** ±0.0051
	ARI	**0.1917** ±0.0406	**0.1280** ±0.0195	**0.6150** ±0.0659	**0.5281** ±0.1262	**0.1161** ±0.0245	**0.7914** 0.0457	**0.2562** ±0.0056	**0.3177** ±0.0147
	F1	**0.6547** ±0.0349	**0.4372** ±0.0122	**0.6850** ±0.0522	**0.5831** ±0.1033	**0.3967** ±0.0121	**0.8737** ±0.0257	**0.3665** ±0.0053	**0.3911** ±0.0138

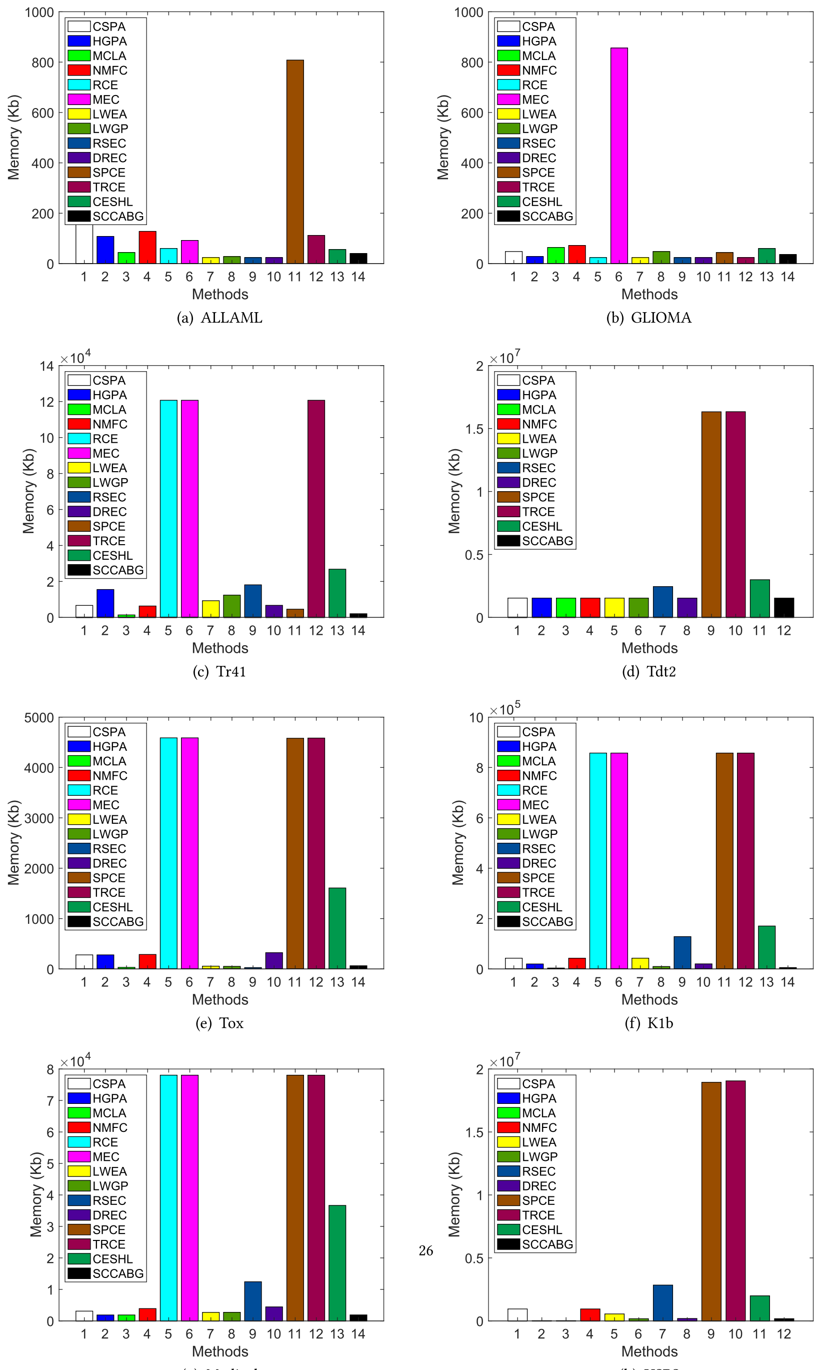
The bold font indicates that the difference is statistically significant (i.e., the p-value of t-test is smaller than 0.05).

easier images have large weights, which are involved in the consensus learning earlier and harder images have small weights. For example, the last one with $w = 0$ whose true label is "6", but it is often assigned to the cluster with label "4" and thus its weight is much lower.

4.5 Experiments on Space and Time Consuming

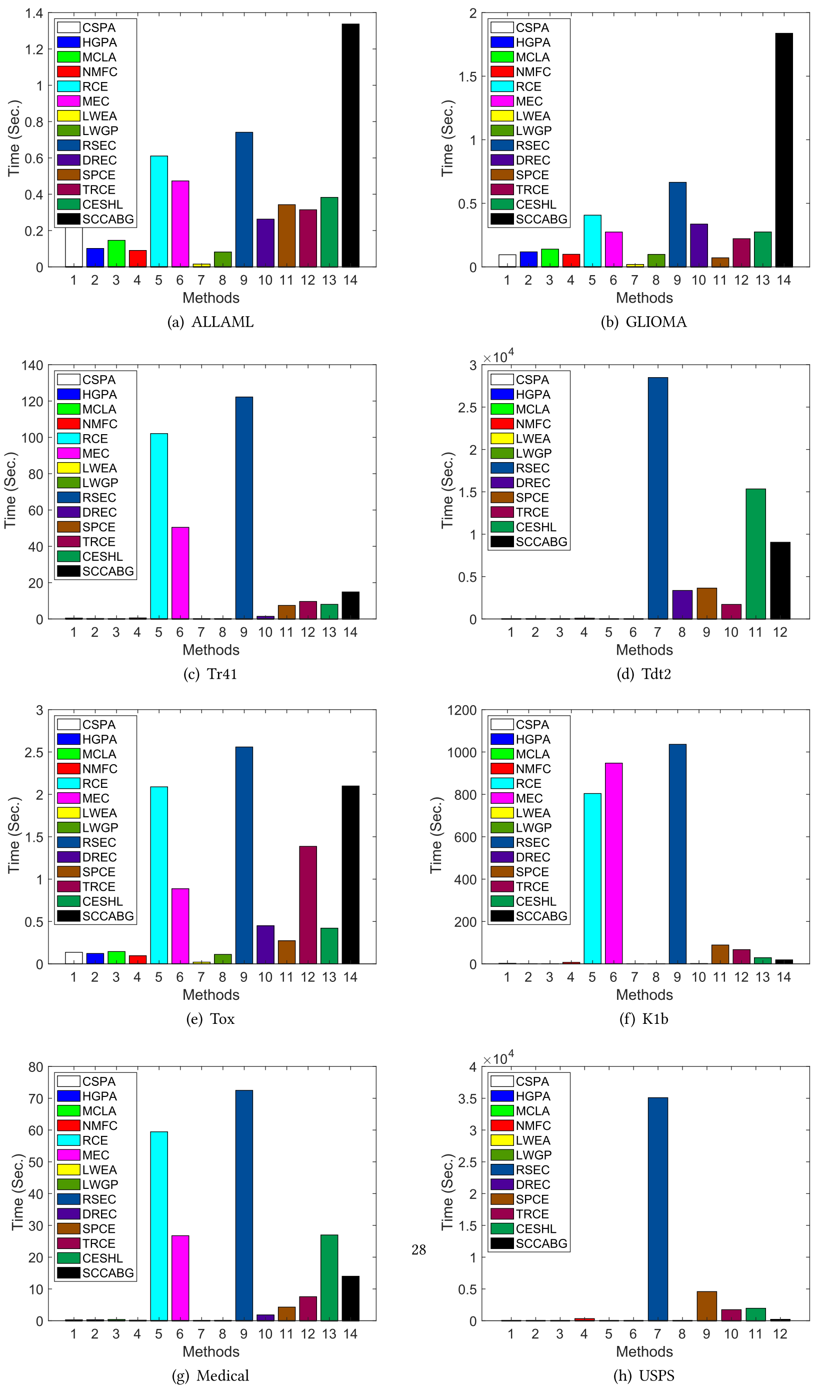
In Figure 10, we show the memory used by all methods on all datasets. From Figure 10, we can find that the memory consumption of the proposed SCCABG is better than most compared methods. Notice that RCE and MEC run out-of-memory on Tdt2 and USPS datasets, and thus have no results. As introduced before, the space complexity of SCCABG is $O(nk + k^2)$, which is linear with the number of instances. However, some other ensemble methods are based on the co-association matrix, whose space complexity is $O(n^2)$. That is why some methods consume much more space than ours.

Figure 11 shows the running time of all methods on all datasets. On the small datasets (i.e., ALLAML and GLIOMA), which only contain no more than 100 instances, SCCABG is slower than other methods. Despite this, SCCABG can obtain the results within 2 seconds. Notice that the time complexity of SCCABG is $O(nk^3)$, where n is the number of instances and k is the total number of clusters. On the small datasets, n is comparable to or even smaller than k. That is why SCCABG is slower than other methods on these two small datasets. However, in practice, n is often much larger than k, like other datasets we used. On these datasets, SCCABG is faster than many other ensemble methods, especially the co-association matrix based methods, whose time complexity is often square or cubic in the number of instances. For example, on the USPS dataset, which contains 11,000 instances, SCCABG only consumes several hundreds of seconds, whereas some compared methods cost several tens of thousands of seconds.



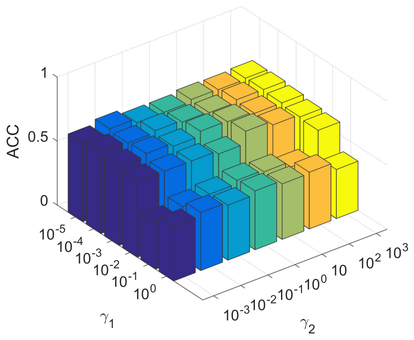
(a) ALLAML (b) GLIOMA (c) Tr41 (d) Tdt2 (e) Tox (f) K1b (g) Medical (h) USPS

Fig. 10. Memory consumption (Kb) of all methods.

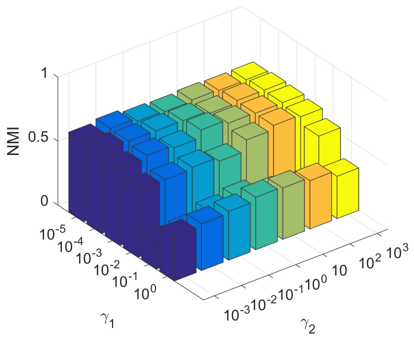


(a) ALLAML (b) GLIOMA
(c) Tr41 (d) Tdt2
(e) Tox (f) K1b
(g) Medical (h) USPS

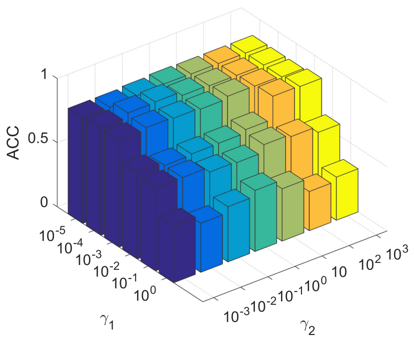
Fig. 11. Running time (Sec.) of all methods.



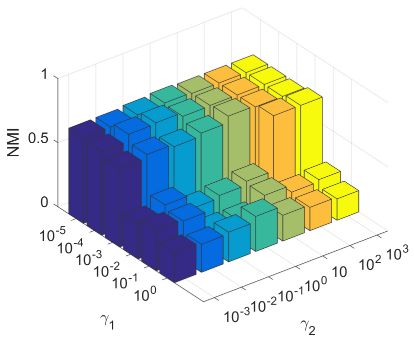
(a) ACC on Tr41



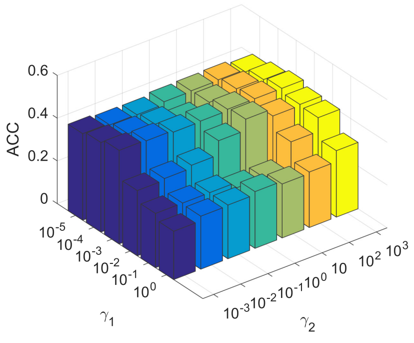
(b) NMI on Tr41



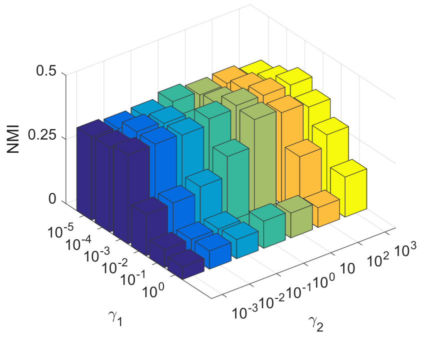
(c) ACC on K1b



(d) NMI on K1b



(e) ACC on Medical



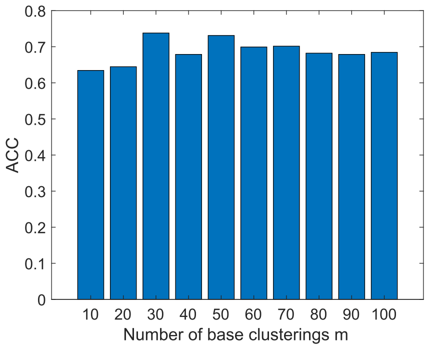
(f) NMI on Medical

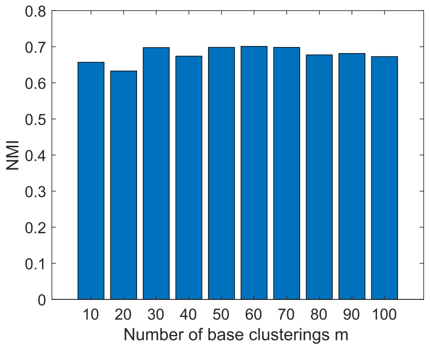
Fig. 12. Clustering results on different values of γ_1 and γ_2.

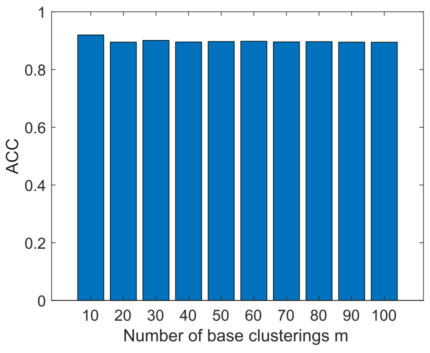
4.6 Ablation Study

To demonstrate the effectiveness of the self-paced learning and the adaptive cluster similarity measuring strategy, we compare our SCCABG with the following two degenerated versions:

- **SCCBG-W**, which is our method without self-paced learning. In more detail, we fix all the elements in the weight matrix **W** as one and do not update them. Since we do not evaluate the reliability of edges, we also remove the self-paced regularized term and adaptive cluster similarity measuring term.


(a) ACC on Tr41


(b) NMI on Tr41


(c) ACC on K1b

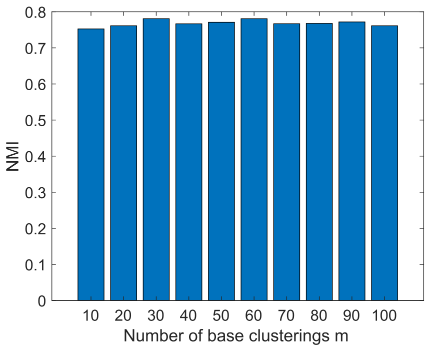

(d) NMI on K1b

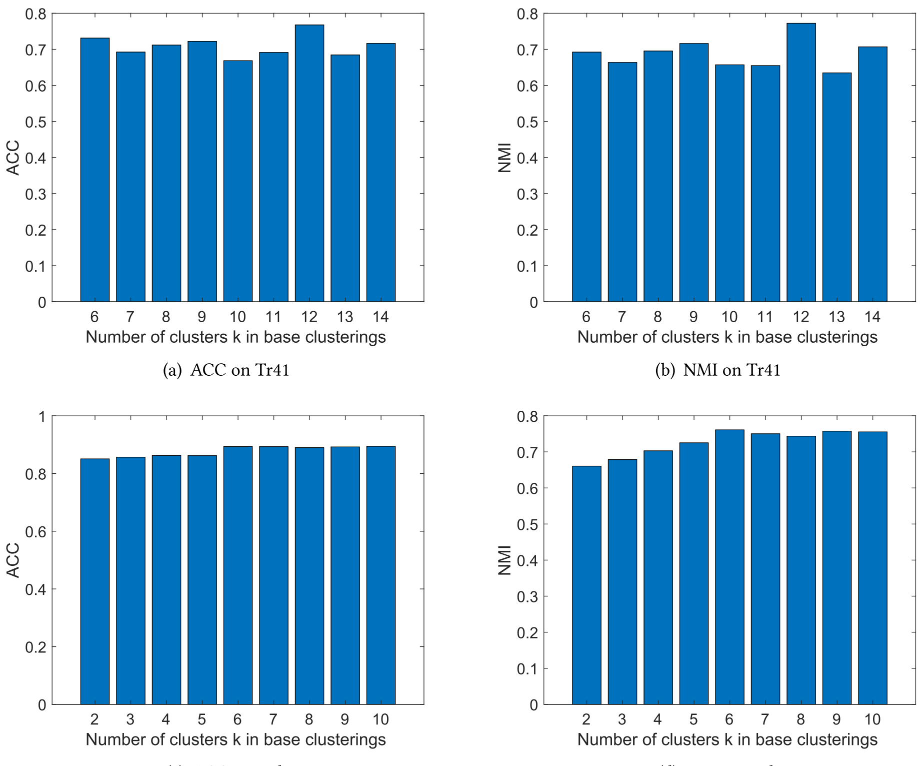
Fig. 13. Clustering results with different numbers of base clusterings *m*.

— **SCCBG** [62], which is our self-paced version with fixed cluster similarity matrix **C**. It is also the method proposed in our previous conference version [62].

Table 6 shows the results compared with SCCBG-W and SCCBG. Compared with SCCBG-W, SCCBG often achieves better performance, which indicates the effectiveness of the self-paced learning framework. With the carefully designed self-paced regularized term, SCCBG can well evaluate the reliability of each edge (i.e., **W**), and alleviate the side effects caused by the unreliability edges in the early model. Compared with SCCBG, SCCABG further improves the performance. It demonstrates the effectiveness of the adaptive cluster similarity measure method, i.e., adaptively updating the cluster similarity will characterize the reliability of edges better than the fixed method.

4.7 Hyper-Parameter Study

In this subsection, we study the effect of the hyper-parameters γ_1 and γ_2. As discussed before, γ_1 should be small to guarantee the convexity of the subproblems. So we tune it in $[10^{-5}, 10^{0}]$. We tune γ_2 in the range $[10^{-3}, 10^{3}]$. Figure 12 shows the ACC and NMI results on Tr41, K1b, and Medical datasets with different hyper-parameters. Results on other datasets are similar. It can be seen that, SCCABG is not sensitive with parameter $\gamma_2 \in [10^{-3}, 10^{3}]$ (and it often achieves the best results when $\gamma_2 \in [10^{0}, 10^{1}]$), and works well when γ_1 is in the range $[10^{-5}, 10^{-3}]$. When γ_1 grows, the performance will deteriorate, which is in line with our previous discussion. Notice that, the conference version SCCBG, which fixes $\mathbf{C} = \mathbf{Y}\mathbf{Y}^T$ and never updates **C**, is equivalent with SCCABG with $\gamma_2 \to +\infty$. Therefore, its performance is somewhat worse than SCCABG, as shown in the ablation study.



(a) ACC on Tr41 (b) NMI on Tr41

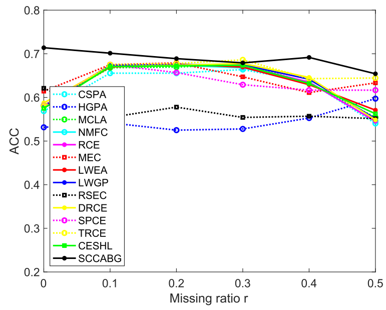
(c) ACC on K1b (d) NMI on K1b

Fig. 14. Clustering results with different numbers of clusters k_i in each base clustering.

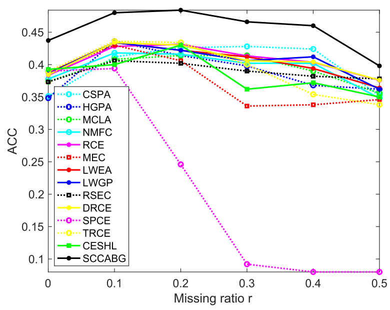
Moreover, we also show the effect of the number of base clusterings m and the number of clusters k_i in each base result. Figure 13 shows the ACC and NMI on Tr41 and K1b with $\{10, 20, \ldots, 100\}$ base clusterings. Figure 14 shows the ACC and NMI on Tr41 and K1b with different numbers of clusters k_i in each base result. If the true number of classes is k, we show different k_i in the range $[k-4, k+4]$. For example, the real number of classes of $Tr41$ is 10, and thus we show k_i in $\{6, 7, 8, \ldots, 14\}$; the real number of classes of K1b is 6, and we show k_i in $\{2, 3, 4, \ldots, 10\}$. The results on other datasets are similar. From Figures 13 and 14, we can see that SCCABG is insensitive with m and k_i.

4.8 Experiments on Incomplete Data

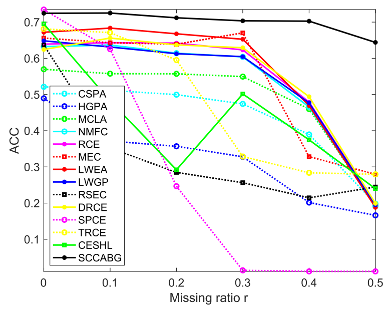
As introduced in Section 3.7, the proposed method can handle incomplete datasets. To show its effectiveness, we also conduct experiments on incomplete datasets. In more detail, for each base result, we set a missing ration $r = \{0, 0.1, \ldots, 0.5\}$, and randomly remove instances according to r. For example, $r = 0.3$ means in each base result, there are 30% instances are missing; $r = 0$ means the base results are complete without any missing data. Notice that the compared methods cannot directly handle the missing values and need the complete data as inputs. Hence, for the compared methods, we apply a random filling method to impute the missing values before doing consensus clustering. Specifically, if x_i is missing in the pth base clustering, we randomly assign x_i to one cluster in the pth base clustering, and then we run the compared consensus clustering methods on the filled data. Figure 15 shows the ACC results of all methods on all datasets with a missing ratio



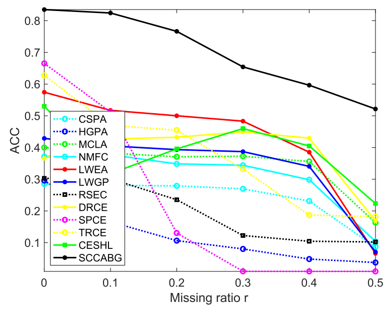
(a) ALLAML



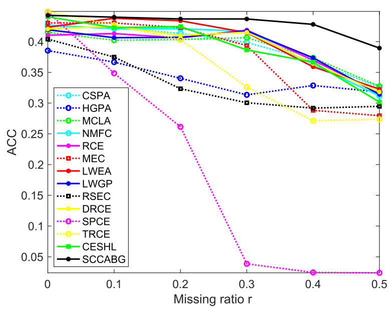
(b) GLIOMA



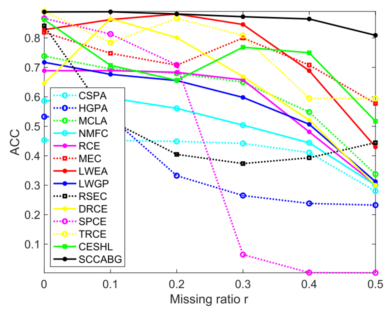
(c) Tr41



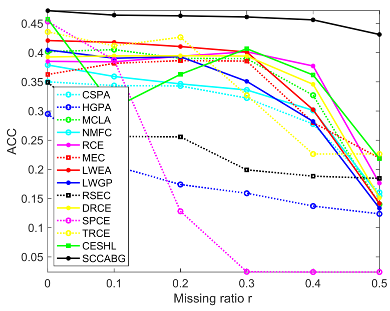
(d) Tdt2



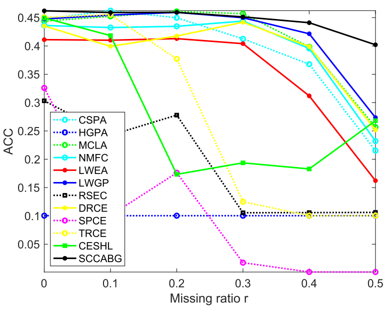
(e) Tox



(f) K1b



32



(g) Medical

(h) USPS

Fig. 15. ACC results on all datasets with missing ratio from 0 to 0.5.

Table 7. Clustering Results on MNIST

Measures	KM	KM-best	HGPA	MCLA	LWGP	SCCABG
ACC	0.5468 ± 0.0084	0.5998 ± 0.0166	0.3042 ± 0.0369	0.5497 ± 0.0125	0.5413 ± 0.0164	**0.5583** ± 0.0235
NMI	0.4933 ± 0.0041	0.5204 ± 0.0139	0.2065 ± 0.0539	0.4882 ± 0.0128	0.4899 ± 0.0030	**0.4943** ± 0.0055

The bold font indicates that the difference is statistically significant (i.e., the p-value of t-test is smaller than 0.05).

from 0 to 0.5. The results on other metrics are similar. The black solid line represents our SCCABG. From Figure 15, we find that the performance of many compared methods deteriorates rapidly with the increase of missing data. However, the performance of our method is relatively more stable on most datasets, which demonstrates that the proposed method can handle incomplete data better than the conventional consensus clustering methods.

4.9 Experiments on Large Dataset

To show the effectiveness of the proposed method on the large scale dataset, we also conducted experiments on MNIST dataset [23]. MNIST is an image dataset, which contains 70,000 handwritten images in 10 categories. The size of each image is 28×28. Following the previous experimental setup, we first also run k-means to generate 200 base results, and partition them into 10 subsets, with 20 in each one. Then, we run consensus clustering methods on each subset. Since the dataset is too large for most compared methods, only HGPA, MCLA, and LWGP have a result and other methods run out-of-memory. Table 7 shows the results. From Table 7, we find that SCCABG also outperforms HGPA, MCLA, and LWGP methods on this large dataset.

5 CONCLUSION

In this paper, we proposed a novel self-paced consensus clustering method with adaptive bipartite graph learning. We constructed an initial bipartite graph with the base results, and then learned a structured bipartite graph from it adaptively. In the process of bipartite graph learning, we adopted the idea of self-paced learning, which automatically determined the reliability of each edge and involved them in bipartite graph learning in the order of reliability. At last, we directly obtained the final result by finding the connective components of the learned bipartite graph without any uncertain postprocessing. We conducted extensive experiments on toy and benchmark datasets. Compared with other state-of-the-art consensus clustering methods, our SCCABG often achieved better performance, which well demonstrated its superiority and effectiveness.

REFERENCES

[1] Sadr-olah Abbasi, Samad Nejatian, Hamid Parvin, Vahideh Rezaie, and Karamolah Bagherifard. 2019. Clustering ensemble selection considering quality and diversity. *Artificial Intelligence Review* 52, 2 (2019), 1311–1340.

[2] Javad Azimi and Xiaoli Z. Fern. 2009. Adaptive cluster ensemble selection. In *Proceedings of the 21st International Joint Conference on Artificial Intelligence, IJCAI*, Craig Boutilier (Ed.). 992–997.

[3] Ali Bagherinia, Behrooz Minaei-Bidgoli, Mehdi Hossinzadeh, and Hamid Parvin. 2019. Elite fuzzy clustering ensemble based on clustering diversity and quality measures. *Applied Intelligence* 49, 5 (2019), 1724–1747.

[4] Liang Bai, Jiye Liang, and Fuyuan Cao. 2020. A multiple k-means clustering ensemble algorithm to find nonlinearly separable clusters. *Information Fusion* 61 (2020), 36–47.

[5] Sumit Basu and Janara Christensen. 2013. Teaching classification boundaries to humans. In *Proceedings of the AAAI Conference on Artificial Intelligence*. 109–115.

[6] Yoshua Bengio, Jerome Louradour, Ronan Collobert, and Jason Weston. 2009. Curriculum learning. In *Proceedings of the 26th Annual International Conference on Machine Learning*. 41–48.

[7] Deng Cai, Xiaofei He, and Jiawei Han. 2011. Locally consistent concept factorization for document clustering. *IEEE Transactions on Knowledge and Data Engineering* 23, 6 (2011), 902–913.

[8] Xiao Cai, Feiping Nie, and Heng Huang. 2013. Multi-view K-means clustering on big data. In *Proceedings of the 23rd International Joint Conference on Artificial Intelligence, IJCAI*. 2598–2604.

[9] Andrew R. Conn, Nicholas I. M. Gould, and Philippe L. Toint. 2000. *Trust Region Methods*. SIAM.

[10] Ky Fan. 1949. On a theorem of Weyl concerning eigenvalues of linear transformations: II*. *Proceedings of the National Academy of Sciences of the United States of America* 36, 1 (1949), 31–35.

[11] Jing Gao, Jiawei Han, Jialu Liu, and Chi Wang. 2013. Multi-view clustering via joint nonnegative matrix factorization. In *Proceedings of the 13th SIAM International Conference on Data Mining, 2013*. SIAM, 252–260.

[12] Todd R. Golub, Donna K. Slonim, Pablo Tamayo, Christine Huard, Michelle Gaasenbeek, Jill P. Mesirov, Hilary Coller, Mignon L. Loh, James R. Downing, Mark A. Caligiuri, Clara D. Bloomfield, and Eric S. Lander. 1999. Molecular classification of cancer: Class discovery and class prediction by gene expression monitoring. *Science* 286, 5439 (1999), 531–537.

[13] Xifeng Guo, Xinwang Liu, En Zhu, Xinzhong Zhu, Miaomiao Li, Xin Xu, and Jianping Yin. 2020. Adaptive self-paced deep clustering with data augmentation. *IEEE Transactions on Knowledge and Data Engineering* 32, 9 (2020), 1680–1693.

[14] Dong Huang, Jianhuang Lai, and Changdong Wang. 2016. Ensemble clustering using factor graph. *Pattern Recognition* 50 (2016), 131–142.

[15] Dong Huang, Jianhuang Lai, and Changdong Wang. 2016. Robust ensemble clustering using probability trajectories. *IEEE Transactions on Knowledge and Data Engineering* 28, 5 (2016), 1312–1326.

[16] Dong Huang, Chang-Dong Wang, and Jian-Huang Lai. 2018. Locally weighted ensemble clustering. *IEEE Transactions on Cybernetics* 48, 5 (2018), 1460–1473. DOI: https://doi.org/10.1109/TCYB.2017.2702343

[17] Dong Huang, Chang-Dong Wang, Hongxing Peng, Jianhuang Lai, and Chee-Keong Kwoh. 2021. Enhanced ensemble clustering via fast propagation of cluster-wise similarities. *IEEE Transactions on Systems, Man, and Cybernetics: Systems* 51, 1 (2021), 508–520. DOI: https://doi.org/10.1109/TSMC.2018.2876202

[18] Dong Huang, Chang-Dong Wang, Jian-Sheng Wu, Jian-Huang Lai, and Chee-Keong Kwoh. 2020. Ultra-scalable spectral clustering and ensemble clustering. *IEEE Transactions on Knowledge and Data Engineering* 32, 6 (2020), 1212–1226. DOI: https://doi.org/10.1109/TKDE.2019.2903410

[19] Lu Jiang, Deyu Meng, Qian Zhao, Shiguang Shan, and Alexander G. Hauptmann. 2015. Self-paced curriculum learning. In *Proceedings of the 29th AAAI Conference on Artificial Intelligence*. 2694–2700.

[20] Yangbangyan Jiang, Zhiyong Yang, Qianqian Xu, Xiaochun Cao, and Qingming Huang. 2018. When to learn what: Deep cognitive subspace clustering. In *Proceedings of the 26th ACM International Conference on Multimedia*. ACM, 718–726.

[21] Zhao Kang, Wangtao Zhou, Zhitong Zhao, Junming Shao, Meng Han, and Zenglin Xu. 2020. Large-scale multi-view subspace clustering in linear time. In *Proceedings of the 34th AAAI Conference on Artificial Intelligence, AAAI*. 4412–4419.

[22] M. P. Kumar, Benjamin Packer, and Daphne Koller. 2010. Self-paced learning for latent variable models. In *Proceedings of the Advances in Neural Information Processing Systems*. 1189–1197.

[23] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. 1998. Gradient-based learning applied to document recognition. *Proceedings of the IEEE* 86, 11 (1998), 2278–2324.

[24] Changsheng Li, Junchi Yan, Fan Wei, Weishan Dong, Qingshan Liu, and Hongyuan Zha. 2017. Self-paced multi-task learning. In *Proceedings of the 31st AAAI Conference on Artificial Intelligence, AAAI*. 2175–2181.

[25] Feijiang Li, Yuhua Qian, Jieting Wang, Chuangyin Dang, and Liping Jing. 2019. Clustering ensemble based on sample's stability. *Artificial Intelligence* 273 (2019), 37–55.

[26] Jundong Li, Kewei Cheng, Suhang Wang, Fred Morstatter, Robert P. Trevino, Jiliang Tang, and Huan Liu. 2018. Feature selection: A data perspective. *ACM Computing Surveys (CSUR)* 50, 6 (2018), 94.

[27] Ping Li, Chun Chen, and Jiajun Bu. 2012. Clustering analysis using manifold kernel concept factorization. *Neurocomputing* 87 (2012), 120–131.

[28] Tao Li and Chris H. Q. Ding. 2008. Weighted consensus clustering. In *Proceedings of the 2008 SIAM International Conference on Data Mining*. 798–809.

[29] Tao Li, Chris H. Q. Ding, and Michael I. Jordan. 2007. Solving consensus and semi-supervised clustering problems using nonnegative matrix factorization. In *Proceedings of the 7th IEEE International Conference on Data Mining*. 577–582.

[30] Hongfu Liu, Tongliang Liu, Junjie Wu, Dacheng Tao, and Yun Fu. 2015. Spectral ensemble clustering. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 715–724.

[31] Hongfu Liu, Ming Shao, Sheng Li, and Yun Fu. 2018. Infinite ensemble clustering. *Data Mining and Knowledge Discovery* 32, 2 (2018), 385–416.

[32] Xinwang Liu, Miaomiao Li, Chang Tang, Jingyuan Xia, Jian Xiong, Li Liu, Marius Kloft, and En Zhu. 2021. Efficient and effective regularized incomplete multi-view clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 43, 8 (2021), 2634–2646.

[33] Xinwang Liu, Miaomiao Li, Chang Tang, Jingyuan Xia, Jian Xiong, Li Liu, Marius Kloft, and En Zhu. 2021. Efficient and effective regularized incomplete multi-view clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 43, 8 (2021), 2634–2646.

[34] Xinwang Liu, Xinzhong Zhu, Miaomiao Li, Lei Wang, Chang Tang, Jianping Yin, Dinggang Shen, Huaimin Wang, and Wen Gao. 2019. Late fusion incomplete multi-view clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 41, 10 (2019), 2410–2423.

[35] Deyu Meng, Qian Zhao, and Lu Jiang. 2017. A theoretical understanding of self-paced learning. *Information Sciences* 414 (2017), 319–328.

[36] Feiping Nie, Xiaoqian Wang, Cheng Deng, and Heng Huang. 2017. Learning a structured optimal bipartite graph for co-clustering. In *Proceedings of the Advances in Neural Information Processing Systems*. 4129–4138.

[37] Feiping Nie, Xiaoqian Wang, and Heng Huang. 2014. Clustering and projected clustering with adaptive neighbors. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 977–986.

[38] Lili Pan, Shijie Ai, Yazhou Ren, and Zenglin Xu. 2020. Self-paced deep regression forests with consideration on underrepresented examples. In *Proceedings of the European Conference on Computer Vision*, Vol. 12375. 271–287.

[39] Hamid Parvin and Behrouz Minaei-Bidgoli. 2013. A clustering ensemble framework based on elite selection of weighted clusters. *Advances in Data Analysis and Classification* 7, 2 (2013), 181–208.

[40] Hamid Parvin and Behrouz Minaei-Bidgoli. 2015. A clustering ensemble framework based on selection of fuzzy weighted clusters in a locally adaptive clustering algorithm. *Pattern Analysis and Applications* 18, 1 (2015), 87–112.

[41] Xi Peng, Zhenyu Huang, Jiancheng Lv, Hongyuan Zhu, and Joey Tianyi Zhou. 2019. COMIC: Multi-view clustering without parameter selection. In *Proceedings of the 36th International Conference on Machine Learning*, Vol. 97. 5092–5101.

[42] Yazhou Ren, Xiaofan Que, Dezhong Yao, and Zenglin Xu. 2019. Self-paced multi-task clustering. *Neurocomputing* 350 (2019), 212–220.

[43] Alexander Strehl and Joydeep Ghosh. 2003. Cluster ensembles—a knowledge reuse framework for combining multiple partitions. *Journal of Machine Learning Research* 3, 3 (2003), 583–617.

[44] Zhiqiang Tao, Hongfu Liu, Jun Li, Zhaowen Wang, and Yun Fu. 2019. Adversarial graph embedding for ensemble clustering. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence, IJCAI*. 3562–3568.

[45] Zhiqiang Tao, Hongfu Liu, Sheng Li, Zhengming Ding, and Yun Fu. 2017. From ensemble clustering to multi-view clustering. In *Proceedings of the International Joint Conference on Artificial Intelligence*. 2843–2849.

[46] Zhiqiang Tao, Hongfu Liu, Sheng Li, Zhengming Ding, and Yun Fu. 2019. Robust spectral ensemble clustering via rank minimization. *ACM Transactions on Knowledge Discovery From Data* 13, 1 (2019), 1–25.

[47] Zhiqiang Tao, Hongfu Liu, Sheng Li, and Yun Fu. 2016. Robust spectral ensemble clustering. In *CIKM*. 367–376.

[48] Alexander Topchy, Anil K. Jain, and William F. Punch. 2003. Combining multiple weak clusterings. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*. 331–338.

[49] Alexander Topchy, Anil K. Jain, and William F. Punch. 2004. A mixture model for clustering ensembles. In *Proceedings of the 2004 SIAM International Conference on Data Mining*. 379–390.

[50] Fei Wang, Xin Wang, and Tao Li. 2009. Generalized cluster aggregation. In *Proceedings of the International Joint Conference on Artificial Intelligence*. 1279–1284.

[51] Hongjun Wang, Hanhuai Shan, and Arindam Banerjee. 2009. Bayesian cluster ensembles. In *Proceedings of the SIAM International Conference on Data Mining*. 211–222.

[52] Siwei Wang, Xinwang Liu, En Zhu, Chang Tang, Jiyuan Liu, Jingtao Hu, Jingyuan Xia, and Jianping Yin. 2019. Multiview clustering via late fusion alignment maximization. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence, IJCAI*, Sarit Kraus (Ed.). 3778–3784.

[53] Chang Xu, Dacheng Tao, and Chao Xu. 2015. Multi-view learning with incomplete views. *IEEE Transactions on Image Processing* 24, 12 (2015), 5812–5825.

[54] Qiyue Yin, Shu Wu, and Liang Wang. 2015. Incomplete multi-view clustering via subspace learning. In *Proceedings of the 24th ACM International Conference on Information and Knowledge Management, CIKM 2015*. ACM, 383–392.

[55] Zhiwen Yu, Le Li, Yunjun Gao, Jane You, Jiming Liu, Hausan Wong, and Guoqiang Han. 2014. Hybrid clustering solution selection strategy. *Pattern Recognition* 47, 10 (2014), 3362–3375.

[56] Dingwen Zhang, Deyu Meng, and Junwei Han. 2017. Co-saliency detection via a self-paced multiple-instance learning framework. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 39, 5 (2017), 865–878.

[57] Yi Zhang, Xinwang Liu, Siwei Wang, Jiyuan Liu, Sisi Dai, and En Zhu. 2021. One-stage incomplete multi-view clustering via late fusion. In *Proceedings of the 29th ACM International Conference on Multimedia*. ACM, 2717–2725.

[58] Qian Zhao, Deyu Meng, Lu Jiang, Qi Xie, Zongben Xu, and Alexander G. Hauptmann. 2015. Self-paced learning for matrix factorization. In *Proceedings of the 29th AAAI Conference on Artificial Intelligence*. 3196–3202.

[59] Ying Zhao and George Karypis. 2004. Empirical and theoretical comparisons of selected criterion functions for document clustering. *Machine Learning* 55, 3 (2004), 311–331.

[60] Jie Zhou, Hongchan Zheng, and Lulu Pan. 2019. Ensemble clustering based on dense representation. *Neurocomputing* 357 (2019), 66–76.

[61] Peng Zhou, Jiangyong Chen, Liang Du, and Xuejun Li. 2022. Balanced spectral feature selection. *IEEE Transactions on Cybernetics* (2022), 1–13.
[62] Peng Zhou, Liang Du, and Xuejun Li. 2020. Self-paced consensus clustering with bipartite graph. In *Proceedings of the 29th International Joint Conference on Artificial Intelligence, IJCAI 2020.* 2133–2139.
[63] P. Zhou, L. Du, X. Liu, Y. Shen, M. Fan, and X. Li. 2021. Self-paced clustering ensemble. *IEEE Transactions on Neural Networks and Learning Systems* 32, 4 (2021), 1497–1511.
[64] Peng Zhou, Liang Du, Yi-Dong Shen, and Xuejun Li. 2021. Tri-level robust clustering ensemble with multiple graph learning. In *Proceedings of the 35th AAAI Conference on Artificial Intelligence, AAAI 2021.* AAAI Press, 11125–11133.
[65] Peng Zhou, Liang Du, Lei Shi, Hanmo Wang, and Yi-Dong Shen. 2015. Recovery of corrupted multiple kernels for clustering. In *Proceedings of the 24th International Joint Conference on Artificial Intelligence, IJCAI.* 4105–4111.
[66] Peng Zhou, Liang Du, Hanmo Wang, Lei Shi, and Yidong Shen. 2015. Learning a robust consensus matrix for clustering ensemble via Kullback–Leibler divergence minimization. In *Proceedings of the 24th International Joint Conference on Artificial Intelligence.* 4112–4118.
[67] Peng Zhou, Yi-Dong Shen, Liang Du, Fan Ye, and Xuejun Li. 2019. Incremental multi-view spectral clustering. *Knowledge-Based Systems* 174 (2019), 73–86.
[68] Peng Zhou, Xia Wang, Liang Du, and Xuejun Li. 2022. Clustering ensemble via structured hypergraph learning. *Information Fusion* 78 (2022), 171–179.
[69] Zhihua Zhou and Wei Tang. 2006. Clusterer ensemble. *Knowledge Based Systems* 19, 1 (2006), 77–83.

Received 28 January 2022; revised 31 August 2022; accepted 19 September 2022